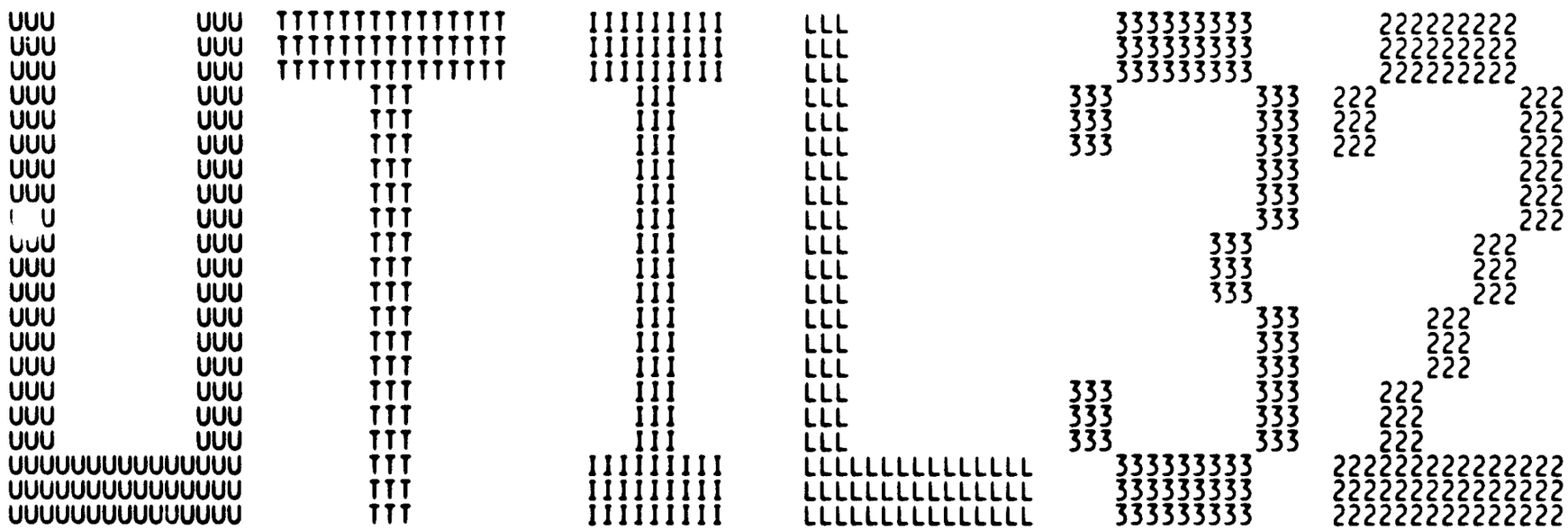
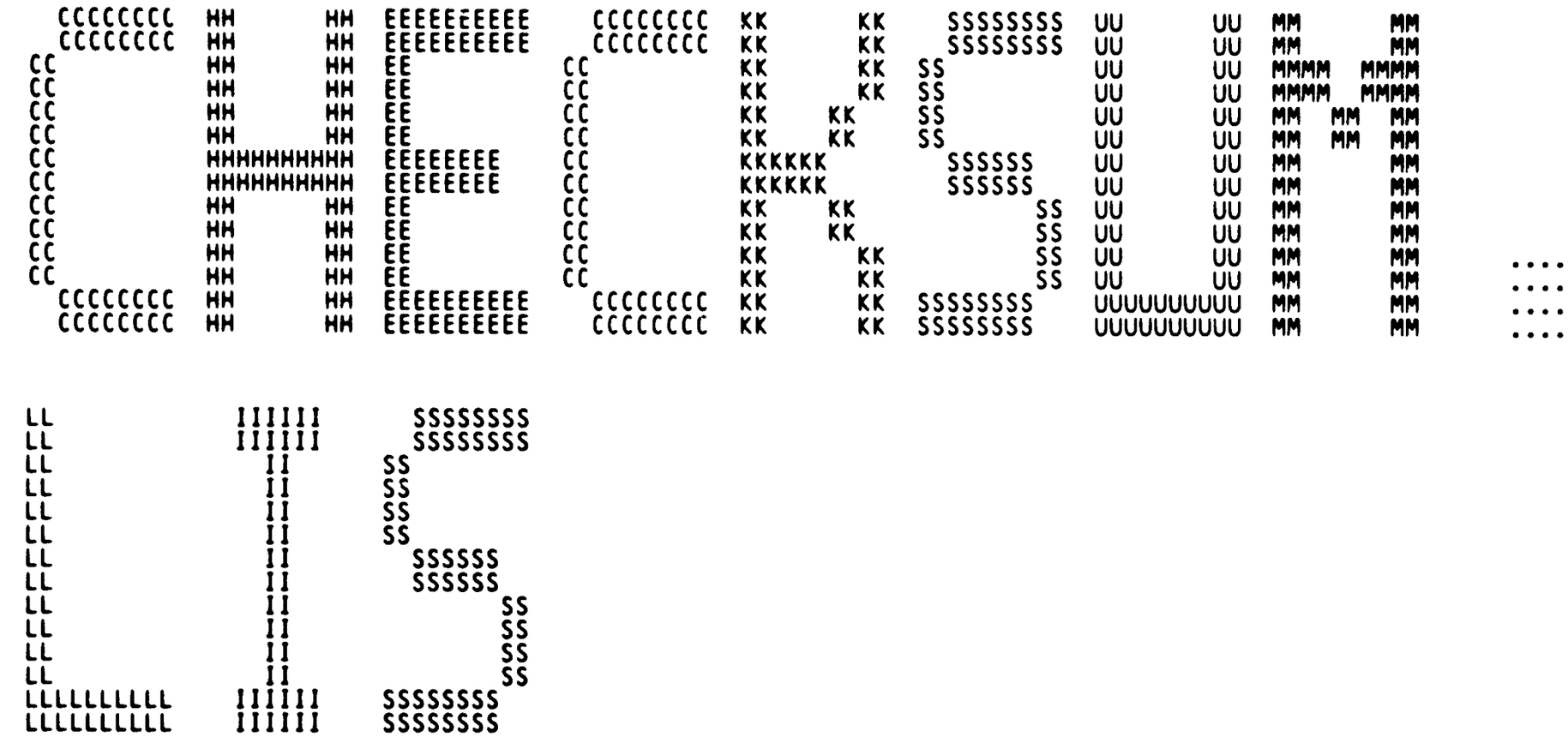

UUU UUU TTTTTTTTTTTTTTT IIIIIIIII LLL 333333333 222222222
UUU UUU TTTTTTTTTTTTTTT IIIIIIIII LLL 333333333 222222222
UUU UUU TTTTTTTTTTTTTTT IIIIIIIII LLL 333333333 222222222
UUU UUU TTT III LLL 333 333 222 222
UUU UUU TTT III LLL 333 333 222 222
UUU UUU TTT III LLL 333 333 222 222
UUU UUU TTT III LLL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

FILEID**CHECKSUM

N 2



```
;     1          0001  0 MODULE checksum_image (
;     2          0002  0                         IDENT='V04-000',
;     3          0003  0                         MAIN=checksum$main
;     4          0004  0                         %TITLE  'Checksum the contents of a file'
;     5          0005  0                         ) =
;     6          0006  0
;     7          0007  1 BEGIN
;     8          0008  1 !
;     9          0009  1 !
;    10          0010  1 !*****************************************************************************
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    13          0013  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    14          0014  1 !*  ALL RIGHTS RESERVED.                                                      *
;    15          0015  1 !*                                                                           *
;    16          0016  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    17          0017  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    18          0018  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    19          0019  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    20          0020  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    21          0021  1 !*  TRANSFERRED.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    24          0024  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    25          0025  1 !*  CORPORATION.                                                              *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    28          0028  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    29          0029  1 !*                                                                           *
;    30          0030  1 !*                                                                           *
;    31          0031  1 !*****************************************************************************
;    32          0032  1 !
;    33          0033  1 !
;    34          0034  1 !++
;    35          0035  1 !
;    36          0036  1 ! FACILITY:  Checksum VAX/VMS image file
;    37          0037  1 !
;    38          0038  1 ! ABSTRACT:
;    39          0039  1 !               Computes the checksum of a file or an image.  The
;    40          0040  1 !               qualifiers on command are /OUTPUT, /FILE and /IMAGE.
;    41          0041  1 !               The /FILE and /IMAGE are conflicting qualifiers.  If
;    42          0042  1 !               specify /IMAGE, then the parts of the image header
;    43          0043  1 !               that can change (Date/Time, etc) are not checksummed.
;    44          0044  1 !               The parameter is the input file(s) to checksum.
;    45          0045  1 !
;    46          0046  1 ! ENVIRONMENT:
;    47          0047  1 !
;    48          0048  1 !       VAX native, user mode.
;    49          0049  1 !
;    50          0050  1 !--
;    51          0051  1 !
;    52          0052  1 !
;    53          0053  1 ! AUTHOR:  Benn Schreiber,       CREATION DATE:  17-Feb-1981
;    54          0054  1 !
;    55          0055  1 ! MODIFIED BY:
;    56          0056  1 !
;    57          0057  1 !       V03-002 LJA0111 Laurie J. Anderson      16-Feb-1984
```

```
;    58          0058  1 !                Change the image checksum to use the new EXEC image
;    59          0059  1 !                header decode routines.  Change the parameters to
;    60          0060  1 !                LIB$FIND_FILE to use the new related file parsing
;    61          0061  1 !                arguments.  Also, add relevant comments so the
;    62          0062  1 !                next person can figure out this mess.
;    63          0063  1 !
;    64          0064  1 !        V03-001 BLS0180 Benn Schreiber          13-Aug-1982
;    65          0065  1 !                Support record by record checksum
;    66          0066  1 !
;    67          0067  1 !--
```

```
;    69          0068  1 LIBRARY
;    70          0069  1         'SYS$LIBRARY:LIB.L32';          !System macro definitions
;    71          0070  1 REQUIRE
;    72          0071  1         'SHRLIB$:IMGMSGDEF.R32';        !Image decode routine's messge literals
;    73          0157  1
;    74          0158  1 MACRO
;    75          0159  1 !
;    76          0160  1 ! Macro to initialize a dynamic descriptor
;    77          0161  1 !
;    78        M 0162  1     $init_ddesc(d) =                    ! Init dynamic string descriptor
;    79        M 0163  1                 begin
;    80        M 0164  1                 d[dsc$w_length] = 0;
;    81        M 0165  1                 d[dsc$b_class] = dsc$k_class_d;
;    82        M 0166  1                 d[dsc$b_dtype] = dsc$k_dtype_t;
;    83        M 0167  1                 d[dsc$a_pointer] = 0;
;    84          0168  1                 end%,
;    85          0169  1 !
;    86          0170  1 ! Macro to generate a pointer to an ascic string
;    87          0171  1 !
;    88          0172  1     cstring (string) = UPLIT(%ASCIC string)%,
;    89          0173  1 !
;    90          0174  1 ! Macro to fao and output a string
;    91          0175  1 !
;    92        M 0176  1     write (string) =
;    93        M 0177  1         cksm_output (fao_buffer (cstring(string)
;    94          0178  1         %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI))%;
;    95          0179  1
;    96          0180  1 !
;    97          0181  1 ! Message codes from shared message definitions
;    98          0182  1 !
;    99          0183  1
;   100        P 0184  1 $SHR_MSGDEF(cksm, 3 ,LOCAL,              !Facility code = 3 (CLI)
;   101        P 0185  1                 (confqual, severe),     ! Conflicting qualifiers
;   102        P 0186  1                 (openin, error),        ! error opening "x" as input
;   103        P 0187  1                 (openout, severe),      ! error opening "x" as output
;   104        P 0188  1                 (closein, warning),     ! error closing "x" as input
;   105        P 0189  1                 (closeout, warning),    ! error closing "x" as output
;   106        P 0190  1                 (readerr, error),       ! error reading "x"
;   107        P 0191  1                 (writeerr, error),      ! error writing "x"
;   108          0192  1                 (parsefail, warning));  ! error parsing "x"
;   109          0193  1
;   110          0194  1 FORWARD ROUTINE
;   111          0195  1     read_header,                                !Read and process image header
;   112          0196  1     checksum_section,                           !Checksum an image section
;   113          0197  1     cksm_foreign,                               !Checksum non-image file
;   114          0198  1     cksm_header,                                !Checksum image header data
;   115          0199  1     read_blocks,                                !Read next block of header
;   116          0200  1     fao_buffer,                                 !Call fao and return descriptor
;   117          0201  1     cksm_output,                                !Output line to output file
;   118          0202  1     getfilename;                                !Get string descriptor for file
;   119          0203  1
;   120          0204  1 EXTERNAL ROUTINE
;   121          0205  1     cli$get_value : ADDRESSING_MODE(GENERAL),   !Get qualifier value
;   122          0206  1     cli$present : ADDRESSING_MODE(GENERAL),     !Test if qualifier present
;   123          0207  1     img$decode_ihd : ADDRESSING_MODE(GENERAL),  !Decode image header
;   124          0208  1     img$get_next_isd : ADDRESSING_MODE(GENERAL), !Get all ISD's from image
;   125          0209  1     lib$get_vm : ADDRESSING_MODE(GENERAL),      !Allocate virtual memory
```

```
;  126          0210  1      lib$free_vm : ADDRESSING_MODE(GENERAL),      !Free virtual memory
;  127          0211  1      lib$find_file : ADDRESSING_MODE(GENERAL),    !Search for wild card files
;  128          0212  1      lib$set_symbol : ADDRESSING_MODE(GENERAL),   !Define local DCL symbol
;  129          0213  1      str$copy_dx : ADDRESSING_MODE(GENERAL),      !Copy a string
;  130          0214  1      str$upcase : ADDRESSING_MODE(GENERAL);       !Uppercase a string
;  131          0215  1
;  132          0216  1  EXTERNAL LITERAL
;  133          0217  1      cksm$_allischk,                              !Checksum of all image sections
;  134          0218  1      cksm$_badimghdr,                             !Bad image header format
;  135          0219  1      cksm$_file,                                  !Print the filename
;  136          0220  1      cksm$_ihdchksum,                             !Image header checksum
;  137          0221  1      cksm$_isectchk,                              !Checksum of one image section
;  138          0222  1      cksm$_nonimgchk;                             !Non-image file checksum
;  139          0223  1
;  140          0224  1  OWN
;  141          0225  1      chan,                                        !Channel to input file if image
;  142          0226  1      find_context : REF $BBLOCK,                  !Context for lib$find_file
;  143          0227  1      irab : $RAB_DECL,                            !RAB for input
;  144          0228  1      orab : $RAB_DECL,                            ! and output
;  145          0229  1      ixabfhc : $XABFHC_DECL,                      !XAB to get file header characteristics to get file size
;  146          0230  1      outfiledesc : $BBLOCK[dsc$c_s_bln],          !Output file descriptor
;  147          0231  1      find_result : $BBLOCK[dsc$c_s_bln],          !Current input file
;  148          0232  1      headerbuffer : REF $BBLOCK,                  !Pointer to user record buffer
;  149          0233  1      databuffer : REF VECTOR[,LONG],              !and data buffer
;  150          0234  1      exitstatus : $BBLOCK[4];                     !Most severe error status
;  151          0235  1
;  152          0236  1  LITERAL
;  153          0237  1      ihd$k_activoff = ihd$k_length,               !Offset to activation area
;  154          0238  1      ihd$k_symdbgoff = ihd$k_activoff + iha$k_length,
;  155          0239  1      ihd$k_imgidoff = ihd$k_symdbgoff + ihs$k_length,
;  156          0240  1      ihd$k_patchoff = ihd$k_imgidoff + ihi$k_length,
;  157          0241  1      ihd$k_maxlength = ihd$k_patchoff,
;  158          0242  1      maxbytes = 10*512,                           !Maximum record to read
;  159          0243  1      false = 0,                                   !Logical 0
;  160          0244  1      true = 1;                                    ! and 1
;  161          0245  1
;  162          0246  1  GLOBAL
;  163          0247  1      cksm$gl_ihdcksm,                             !Image header checksum
;  164          0248  1      cksm$gl_chksum,                              !Isect checksum
;  165          0249  1      cksm$gl_acucksm,                             !Accumulated checksum
;  166          0250  1      cksm$gl_maxbyt;                              !Maximum record to read
```

```
;  168          0251  1 %sbttl 'allocate_ubf'
;  169          0252  1 ROUTINE allocate_ubf (xab,rab) : NOVALUE =
;  170          0253  2 BEGIN
;  171          0254  2 !
;  172          0255  2 ! Allocate a buffer for records that cross block boundaries
;  173          0256  2 !
;  174          0257  2 MAP
;  175          0258  2     xab : REF $BBLOCK,
;  176          0259  2     rab : REF $BBLOCK;
;  177          0260  2
;  178          0261  2 LOCAL
;  179          0262  2     status,
;  180          0263  2     blocksize;
;  181          0264  2
;  182          0265  2 OWN
;  183          0266  2     ubf_size,           !Size of last UBF
;  184          0267  2     ubf_addr;           !Address of last UBF
;  185          0268  2
;  186          0269  2 !
;  187          0270  2 ! Calculate the max record size based on the longest record in the file.
;  188          0271  2 ! If there is a user buffer allocated and it is not big enough, deallocate
;  189          0272  2 ! it and get a new one.  If it's bigger than needed, then that's fine
;  190          0273  2 !
;  191          0274  2 blocksize = MAXU(MAXU(.xab[xab$w_lrl],.xab[xab$w_mrz]),512);
;  192          0275  2 IF .blocksize GTRU .ubf_size
;  193          0276  3 THEN BEGIN
;  194          0277  3     IF .ubf_size NEQ 0
;  195          0278  3         THEN lib$free_vm(ubf_size,ubf_addr);
;  196          0279  3     ubf_size = .blocksize;
;  197          0280  4     IF NOT (status = lib$get_vm(ubf_size,ubf_addr))
;  198          0281  3         THEN SIGNAL_STOP(.status);
;  199          0282  2     END;
;  200          0283  2 !
;  201          0284  2 ! Set up rab with size and address of the buffer
;  202          0285  2 !
;  203          0286  2 rab[rab$w_usz] = .blocksize;
;  204          0287  2 rab[rab$l_ubf] = .ubf_addr;
;  205          0288  2
;  206          0289  2 RETURN;
;  207          0290  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
                50        04    AC  D0 00007    MOVL    XAB, R0         ; 0274
                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
                                    01  FB 00051          CALLS   #1, LIB$STOP                      ;
                                    50  DO 00058 4$:      MOVL    RAB, R0                           ; 0286
                                    52  B0 0005C          MOVW    BLOCKSIZE, 32(R0)                 ;
                                    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
```

```
;  226          0307  1 %sbttl 'Process CHECKSUM/FILE'
;  227          0308  1 ROUTINE checksum_file : NOVALUE =
;  228          0309  2 BEGIN
;  229          0310  2 !
;  230          0311  2 ! Checksum the contents of a file.  Open the file using RMS and simply read
;  231          0312  2 ! it with $GET's and checksumming the records.  Close the file.
;  232          0313  2 !
;  233          0314  2 LOCAL
;  234          0315  2     checksum,
;  235          0316  2     n_longwords,
;  236          0317  2     n_bytes,
;  237          0318  2     status;
;  238          0319  2
;  239          0320  2 !
;  240          0321  2 ! Open the input file to checksum using RMS.
;  241          0322  2 !
;  242          0323  2 $XABFHC_INIT(xab=ixabfhc);                        !Initialize XAB
;  243        P 0324  2 $RAB_INIT(rab=irab,                               !Initialize input rab
;  244          0325  2         fab=.find_context);
;  245          0326  2
;  246          0327  3 IF NOT (status = $open(fab=.find_context))        !Open the input file
;  247          0328  2 THEN SIGNAL(cksm$_openin, 1,
;  248          0329  2                 find_result, .status,
;  249          0330  2                 .find_context[fab$l_stv])
;  250          0331  2
;  251          0332  3 ELSE IF NOT (status=$connect(rab=irab))           !Connect record stream
;  252          0333  3 THEN BEGIN
;  253          0334  3     SIGNAL(cksm$_openin, 1,
;  254          0335  3                 find_result, .status,
;  255          0336  3                 .irab[rab$l_stv]);
;  256          0337  3     return;
;  257          0338  2     END;
;  258          0339  2
;  259          0340  2 allocate_ubf(ixabfhc,irab);                       !Allocate the buffer to read into
;  260          0341  2
;  261          0342  2 !
;  262          0343  2 ! Checksum the contents.
;  263          0344  2 !
;  264          0345  2 checksum = 0;
;  265          0346  2 WHILE (status = $GET(rab=irab)) NEQ rms$_eof
;  266          0347  3 DO BEGIN
;  267          0348  3     BIND
;  268          0349  3         longbuffer = .irab[rab$l_rbf] : VECTOR[,LONG];
;  269          0350  3
;  270          0351  3     IF NOT .status
;  271          0352  3         THEN SIGNAL(cksm$_readerr,1,find_result,
;  272          0353  3                     .status,.irab[rab$l_stv]);
;  273          0354  3     !
;  274          0355  3     ! Compute number of full longwords and dangling bytes in record
;  275          0356  3     !
;  276          0357  3     n_longwords = (.irab[rab$w_rsz]/4);
;  277          0358  3     n_bytes = .irab[rab$w_rsz] AND 3;
;  278          0359  3     !
;  279          0360  3     ! Checksum the longwords by exclusive or.  Checksum the dangling
;  280          0361  3     ! bytes by simple addition
;  281          0362  3     !
;  282          0363  3     IF .n_longwords NEQ 0
```

```
;   283           0364  3          THEN INCRU i FROM 0 to .n_longwords - 1
;   284           0365  3              DO checksum = .checksum XOR .longbuffer[.i];
;   285           0366  3      IF .n_bytes NEQ 0
;   286           0367  4      THEN BEGIN
;   287           0368  4          BIND
;   288           0369  4              bytebuffer = longbuffer[.n_longwords] : VECTOR[,BYTE];
;   289           0370  4
;   290           0371  4          INCRU i FROM 0 TO .n_bytes - 1
;   291           0372  4              DO checksum = .checksum + .bytebuffer[.i];
;   292           0373  3          END;
;   293           0374  2      END;
;   294           0375  2 !
;   295           0376  2 ! Define a DCL string symbol with the decimal value of the checksum
;   296           0377  2 !
;   297           0378  2 define_sym(.checksum);
;   298           0379  2
;   299           0380  2 !
;   300           0381  2 ! Disconnect the stream and close the input checksummed file.
;   301           0382  2 !
;   302           0383  3 IF NOT (status=$DISCONNECT(rab=irab))   !Disconnect and close output file
;   303           0384  2     THEN SIGNAL(cksm$_closein, 1,
;   304           0385  2                 find_result, .status,
;   305           0386  2                 .irab[rab$l_stv]);
;   306           0387  2
;   307           0388  3 IF NOT (status=$CLOSE(fab=.find_context))
;   308           0389  2     THEN SIGNAL(cksm$_closein, 1,
;   309           0390  2                 find_result, .status,
;   310           0391  2                 .find_context[fab$l_stv]);
;   311           0392  2
;   312           0393  2 RETURN;
;   313           0394  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, $RMS_PTR+60
                                        A8      FC  A8  DD 0002F         PUSHL   FIND_CONTEXT                    ;0327
                                00000000G   00          01  FB 00032         CALLS   #1, SYS$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  DD 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$GET                           ;
                                        54  50  DD 00092          MOVL    R0, STATUS                            ;
                            0001827A    8F  54  D1 00095          CMPL    STATUS, #98938                        ;
                                        5D  13 0009C          BEQL    10$                                   ;
                                      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                                    ;
                                00031082  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
```

```
;  315          0395  1 %sbttl 'signal handler'
;  316          0396  1 ROUTINE signal_handler (sigargs, mechargs) =
;  317          0397  2 BEGIN
;  318          0398  2 !
;  319          0399  2 ! This routine is a condition handler called when ever a
;  320          0400  2 ! SIGNAL is done by cksm.  It merely remembers the
;  321          0401  2 ! most severe error for an exit status.
;  322          0402  2 !
;  323          0403  2 MAP
;  324          0404  2     sigargs : REF $BBLOCK,
;  325          0405  2     mechargs : REF $BBLOCK;
;  326          0406  2
;  327          0407  2 BIND
;  328          0408  2     signame = sigargs [chf$l_sig_name] : $BBLOCK;       !Name of signal
;  329          0409  2
;  330          0410  2 IF NOT .signame                                       !If its an error signal
;  331          0411  4     AND ((.signame [sts$v_severity]                    ! and severity is worse than it was
;  332          0412  4         GEQU .exitstatus [sts$v_severity])
;  333          0413  3         OR .exitstatus [sts$v_severity])               ! or we haven't had any errors
;  334          0414  2     THEN exitstatus = .signame;                        ! then remember it for exiting
;  335          0415  2
;  336          0416  2 RETURN ss$_resignal;                                  !Resignal to get the error printed
;  337          0417  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
```

```
;  339          0418  1 %sbttl 'main routine of checksum'
;  340          0419  1 ROUTINE checksum$main =
;  341          0420  2 BEGIN
;  342          0421  2 !
;  343          0422  2 ! This is the main program.  It gathers all the command inputs and calls
;  344          0423  2 ! appropiate routine to checksum either the file or image.
;  345          0424  2 !
;  346          0425  2 LOCAL
;  347          0426  2     status,
;  348          0427  2     file_sum,
;  349          0428  2     ofab : $BBLOCK[fab$c_bln],
;  350          0429  2     onam : $BBLOCK[nam$c_bln],
;  351          0430  2     orss : $BBLOCK[nam$c_maxrss],
;  352          0431  2     oess : $BBLOCK[nam$c_maxrss],
;  353          0432  2     indesc : $BBLOCK [dsc$c_s_bln];             !User's input file spec
;  354          0433  2
;  355          0434  2 ENABLE signal_handler;                        !Enable the condition handler
;  356          0435  2 exitstatus = ss$_normal;                     !Preset success exit status
;  357          0436  2
;  358          0437  2 cksm$gl_maxbyt = maxbytes;
;  359          0438  2 $init_ddesc(indesc);                         !Initialize dynamic descriptor
;  360          0439  2 CH$MOVE(dsc$c_s_bln,indesc,outfiledesc);
;  361          0440  2 CH$MOVE(dsc$c_s_bln,indesc,find_result);
;  362          0441  2
;  363          0442  2 cli$get_value($descriptor('INPUT'),indesc);      .Get input file spec.
;  364          0443  2 cli$get_value($descriptor('OUTPUT'),outfiledesc);!Get user output file spec
;  365          0444  2
;  366          0445  2 file_sum = cli$present($descriptor('FILE'));    !See if /FILE
;  367          0446  2 status = cli$present($descriptor('IMAGE'));     !See if /IMAGE
;  368          0447  3 IF NOT (.file_sum OR .status)                  !If neither were specified
;  369          0448  2     THEN file_sum = true;                      ! then default to /FILE
;  370          0449  2 IF .file_sum                                   !If both were specified
;  371          0450  2     AND .status
;  372          0451  2     THEN SIGNAL_STOP(cksm$_confqual);          ! that's an error
;  373          0452  2 !
;  374          0453  2 ! Create the output file
;  375          0454  2 !
;  376        P 0455  2 $FAB_INIT(fab=ofab,                           !Initialize output fab
;  377        P 0456  2           nam=onam,
;  378        P 0457  2           fns=.outfiledesc[dsc$w_length],
;  379        P 0458  2           fna=.outfiledesc[dsc$a_pointer],
;  380        P 0459  2           rat=(cr),                           !With carriage control
;  381        P 0460  2           fop=(ofp),                          !Output file parse
;  382          0461  2           fac=(put));                         !Will do put's to file
;  383          0462  2
;  384          0463  2 CH$FILL(0,nam$c_maxrss,orss);                 !Zero resultant name string
;  385          0464  2 CH$FILL(0,nam$c_maxrss,oess);                 ! and expanded name string
;  386        P 0465  2 $NAM_INIT(nam=onam,                           !Initialize output file nam block
;  387        P 0466  2           rss=nam$c_maxrss,
;  388        P 0467  2           rsa=orss,
;  389        P 0468  2           ess=nam$c_maxrss,
;  390          0469  2           esa=oess);
;  391          0470  2
;  392        P 0471  2 $RAB_INIT(rab=orab,                           !Initialize output file rab
;  393          0472  2           fab=ofab);
;  394          0473  2
;  395          0474  2 status = $CREATE(fab=ofab);
```

```
;  396     0475  2 CH$MOVE(dsc$c_s_bln,getfilename(ofab),          !Get the resulting filename
;  397     0476  2                                 outfiledesc);
;  398     0477  2 IF NOT .status                                  !If error opening output file
;  399     0478  2     THEN SIGNAL_STOP(cksm$_openout,1,outfiledesc, ! then give up now
;  400     0479  2                      .status,.ofab[fab$l_stv]);
;  401     0480  2
;  402     0481  3 IF NOT (status=$CONNECT(rab=orab))              !Connect the record stream
;  403     0482  2     THEN SIGNAL_STOP(cksm$_openout,1,outfiledesc,
;  404     0483  2                      .status,.orab[rab$l_stv]);
;  405     0484  2 !
;  406     0485  2 ! Loop. calling lib$find_file to get files matching the output spec
;  407     0486  2 !
;  408     0487  2 find_context = 0;                               !Initial context is 0
;  409     0488  2
;  410     0489  3 WHILE (status = lib$find_file(indesc,find_result,find_context,
;  411     0490  4                               (IF .file_sum
;  412     0491  4                                   THEN $descriptor('.DAT')
;  413     0492  3                                   ELSE $descriptor('.EXE')),
;  414     0493  2                               0, 0, %REF(2)) ) NEQ rms$_nmf
;  415     0494  2 DO
;  416     0495  2 IF NOT .status                                  !Report error if find_file failure
;  417     0496  2         THEN SIGNAL(cksm$_parsefail,1,find_result,
;  418     0497  2                     .find_context[fab$l_sts],.find_context[fab$l_stv])
;  419     0498  3 ELSE BEGIN
;  420     0499  3     find_context[fab$v_get] = true;             !Initialize for GET's
;  421     0500  3     IF NOT .file_sum
;  422     0501  3         THEN find_context[fab$v_bio] = true;    ! and block i/o
;  423     0502  3
;  424     0503  3 !
;  425     0504  3 ! Call the appropiate routine to do the checksumming of the file contents
;  426     0505  3 ! or image.
;  427     0506  3 !
;  428     0507  3     IF .file_sum                                !If /FILE then checksum the file
;  429     0508  4         THEN BEGIN                              ! by simply reading it
;  430     0509  4             find_context[fab$l_xab] = ixabfhc;  !Set XAB address
;  431     0510  4             checksum_file();                    ! Output is DCL symbol checksum$checksum
;  432     0511  4             END
;  433     0512  4         ELSE BEGIN
;  434     0513  5             IF NOT (status = lib$get_vm(cksm$gl_maxbyt,
;  435     0514  5                                         databuffer))
;  436     0515  4                 THEN SIGNAL_STOP(.status);
;  437     0516  4             find_context[fab$l_xab] = 0; !No need to use XAB
;  438     0517  4             read_header();                      !Process CHECKSUM/IMAGE
;  439     0518  3             END;
;  440     0519  3
;  441     0520  2     END;                                        !Of loop calling lib$find_file
;  442     0521  2 !
;  443     0522  2 ! Close the output file
;  444     0523  2 !
;  445     0524  3 IF NOT (status=$DISCONNECT(rab=orab))
;  446     0525  2     THEN SIGNAL(cksm$_closeout, 1,
;  447     0526  2                 outfiledesc, .status,
;  448     0527  2                 .orab[rab$l_stv]);
;  449     0528  2
;  450     0529  3 IF NOT (status=$CLOSE(fab=ofab))
;  451     0530  2     THEN SIGNAL(cksm$_closeout, 1,
;  452     0531  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

                                                00FFC 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$GL_MAXBYT      ; 0437
                                 04   AE 020E0000    8F    D0 0002F        MOVL    #34471936, INDESC          ; 0438
                                           08   AE    D4 00037        CLRL    INDESC+4                   ;
                             68    04   AE          08    28 0003A        MOVC3   #8, INDESC, OUTFILEDESC    ; 0439
                         08  A8    04   AE          08    28 0003F        MOVC3   #8, INDESC, FIND_RESULT    ; 0440    ; R
                                           04   AE    9F 00045        PUSHAB  INDESC                     ; 0442
                                                5A    DD 00048        PUSHL   R10                        ;
                                 00000000G   00       02    FB 0004A        CALLS   #2, CLI$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$PRESENT                                  ;
                                    57              50  D0 00067          MOVL    R0, FILE_SUM                                    ;
                                                    9F  AA 0006A          PUSHAB  P.AAJ                                           ; 0446
                        00000000G   00              01  FB 0006D          CALLS   #1, CLI$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.AAC, R0                    ; 0491
                               04   11 00170          BRB     8$                           ;
              50          44   AA   9E 00172 7$:      MOVAB   P.AAN, R0                    ; C492
                               50   DD 00176 8$:      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
```

```
;  458          0536  1 %sbttl 'Catch signals'
;  459          0537  1 ROUTINE catch_signals (sigargs,mechargs) =
;  460          0538  2 BEGIN
;  461          0539  2 !
;  462          0540  2 ! This condition handler is enabled when the image is being
;  463          0541  2 ! processed.  It catches info messages, and diverts them to
;  464          0542  2 ! the output file.  Any other errors are resignaled.
;  465          0543  2 !
;  466          0544  2 MAP
;  467          0545  2     sigargs : REF $BBLOCK,
;  468          0546  2     mechargs : REF $BBLOCK;
;  469          0547  2
;  470          0548  2 BIND
;  471          0549  2     signame = sigargs[chf$l_sig_name] : $BBLOCK;
;  472          0550  2
;  473          0551  2 LOCAL
;  474          0552  2     status,
;  475          0553  2     outadr : $BBLOCK[4],
;  476          0554  2     outbufdesc : $BBLOCK[dsc$c_s_bln],
;  477          0555  2     msgbufdesc : $BBLOCK[dsc$c_s_bln],
;  478          0556  2     msgbuf : $BBLOCK[256],
;  479          0557  2     outbuf : $BBLOCK[512];
;  480          0558  2
;  481          0559  2 IF NOT .signame                                  !If this is not an informational message
;  482          0560  2     THEN RETURN ss$_resignal;                   ! then we don't want it to go to output file
;  483          0561  2 msgbufdesc[dsc$w_length] = 256;                 !Initialize the descriptors
;  484          0562  2 msgbufdesc[dsc$a_pointer] = msgbuf;
;  485          0563  2 outbufdesc[dsc$w_length] = 512;
;  486          0564  2 outbufdesc[dsc$a_pointer] = outbuf;
;  487          0565  2 !
;  488          0566  2 ! Get the message and format the string
;  489          0567  2 !
;  490        P 0568  3 IF NOT (status = $GETMSG(msgid=.signame,msglen=msgbufdesc[dsc$w_length],
;  491          0569  3                        bufadr=msgbufdesc,flags=1,outadr=outadr))
;  492          0570  2     THEN RETURN ss$_resignal                    !Not found, so resignal
;  493        P 0571  3 ELSE IF NOT (status =$faol(ctrstr=msgbufdesc,outlen=outbufdesc,   !Format the string
;  494          0572  3                  outbuf=outbufdesc,prmlst=sigargs[chf$l_sig_arg1]+4))    ! (skip the signal count longword)
;  495          0573  2     THEN RETURN ss$_resignal                    !FAO error, so resignal
;  496          0574  2     ELSE cksm_output(outbufdesc);               !All is well, write string to output        ; F
;  497          0575  2
;  498          0576  2 signame[sts$v_inhib_msg] = true;                !Inhibit further message printing
;  499          0577  2 mechargs[chf$l_mch_savr0] = .signame[sts$v_inhib_msg];
;  500          0578  2
;  501          0579  2 RETURN ss$_continue
;  502          0580  2
;  503          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        EF 0006A          EXTZV   #28, #1, (R2), 12(R0)
                                          50              01   D0 0007U          MOVL    #1, R0                          ; 0579
                                                          04   00072          RET                                     ; 0581

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

```
;  505          0582  1 %sbttl 'Process CHECKSUM/IMAGE'
;  506          0583  1 ROUTINE read_header =
;  507          0584  2 BEGIN
;  508          0585  2 !
;  509          0586  2 ! Open the file using RMS with the UFO (User File Open) bit set.  The
;  510          0587  2 ! read the image header using the EXEC routines which do $QIO's given
;  511          0588  2 ! the channel from the RMS open., and checksum the image sections.  Ignore
;  512          0589  2 ! the parts of the image header that can always be different, (ex: date/time).
;  513          0590  2 !
;  514          0591  2 LOCAL
;  515          0592  2         isdnum,                                 !isd number
;  516          0593  2         header : $BBLOCK[512],                  !Buffer containing decoded header
;  517          0594  2         curhdrisd : $BBLOCK[512],               !current isd in header
;  518          0595  2         hdr_blk : $BBLOCK[512],                 !buffer for image header -undecoded
;  519          0596  2         status,
;  520          0597  2         vbn,                                    !Used by header decode routines
;  521          0598  2         offset,                                 !Used by header decode routines
;  522          0599  2         hdrver,                                 !Used by header decode routines
;  523          0600  2         alias ;                                 !Used by header decode routines
;  524          0601  2
;  525          0602  2 ENABLE catch_signals;                           !We want output to go to output file also
;  526          0603  2 cksm$gl_ihdcksm = 0;                           !Zero image header checksum
;  527          0604  2 isdnum = 0;                                     !first isect is 1
;  528          0605  2 cksm$gl_acucksm = 0;                           !Zero accumulated checksum
;  529          0606  2 SIGNAL(cksm$_file,1,find_result);               !Print the filename
;  530          0607  2
;  531          0608  2 !
;  532          0609  2 ! First, open the file using RMS with an UFO option.
;  533          0610  2 !
;  534          0611  2 find_context[fab$v_ufo] = true;                 !User file open option
;  535          0612  3 IF NOT (status = $open(fab=.find_context))      !Open the input file
;  536          0613  2 THEN SIGNAL(cksm$_openin, 1,
;  537          0614  2                 find_result, .status,
;  538          0615  2                 .find_context[fab$l_stv]);
;  539          0616  2 find_context[fab$v_ufo] = false;                !clear the bit after open
;  540          0617  2 chan = .find_context[fab$l_stv];                !Keep around the assigned channel
;  541          0618  2
;  542          0619  2 !
;  543          0620  2 ! Read the image header into a buffer decoded by the special EXEC routines
;  544          0621  2 !
;  545          0622  2 status = img$decode_ihd( .chan,hdr_blk,header,vbn,offset,hdrver,alias);
;  546          0623  2 IF NOT .status
;  547          0624  2 THEN
;  548          0625  2     SIGNAL( .status );                         !signal any errors in image file
;  549          0626  2
;  550          0627  2 !
;  551          0628  2 ! The following checks are made on the image header fixed portion to see if legal
;  552          0629  2 ! Major neader id must match.  The header fixed part must be less than a block and must
;  553          0630  2 ! contain patch area.  header block count is > 0.
;  554          0631  2 !
;  555          0632  2 IF .header[ihd$w_majorid] NEQ ihd$k_majorid
;  556          0633  4     OR (.(header[ihd$w_minorid])<0,8,0> EQL (ihd$k_minorid and %x'ff')
;  557          0634  3         AND .(header[ihd$w_minorid])<8,8,0> GTRU (ihd$k_minorid and %x'ff00')/256)
;  558          0635  3     OR .header[ihd$w_size] GTRU MAXU((.header[ihd$w_patchoff]
;  559          0636  2                             + ihp$k_length),ihd$k_maxlength)
;  560          0637  3     OR ( .header[ihd$b_hdrblkcnt] LEQ 0)
;  561          0638  3     OR (header + .header[ihd$w_symdbgoff])
```

```
;  562          0639  3                          GEQU (header + .header[ihd$w_size])
;  563          0640  3      THEN BEGIN
;  564          0641  3          SIGNAL(cksm$_badimghdr,1,find_result);  !any above not true, fatal image header error
;  565          0642  3          status = $dassgn(chan=.chan);          !deassign channel
;  566          0643  3          cksm_foreign();                        !Checksum non-image file
;  567          0644  3          RETURN true
;  568          0645  2          END;
;  569          0646  2
;  570          0647  2  ! Otherwise just checksum the contents of the fixed portion of the image
;  571          0648  2  ! header
;  572          0649  2  !
;  573          0650  2  cksm_header(header);
;  574          0651  2
;  575          0652  2  !
;  576          0653  2  ! Now, checksum all the ISD's (Image Section Descriptors)
;  577          0654  2  !
;  578          0655  2  WHILE 1
;  579          0656  3  DO BEGIN
;  580          0657  3      status = img$get_next_isd(.chan,hdr_blk,header,vbn,offset,curhdrisd);
;  581          0658  3      IF .status EQLU IMG$_ENDOFHDR              ! Until no more ISD's....
;  582          0659  3      THEN
;  583          0660  3          EXITLOOP;
;  584          0661  3      IF NOT .status
;  585          0662  4      THEN BEGIN
;  586          0663  4          SIGNAL(cksm$_badimghdr,1,find_result);  !signal error
;  587          0664  4          status = $dassgn(chan=.chan);          !deassign channel
;  588          0665  4          cksm_foreign();                        !and checksum as foreign image
;  589          0666  4          RETURN true;
;  590          0667  4          END
;  591          0668  3      ELSE
;  592          0669  4          BEGIN
;  593          0670  4          isdnum = .isdnum + 1;
;  594          0671  4          checksum_section(curhdrisd);           !compute isect checksum
;  595          0672  4          IF .curhdrisd[isd$l_vbn] NEQ 0         !If section has disk file space
;  596          0673  4              AND NOT .curhdrisd[isd$v_dzro]     ! and is not demand zero
;  597          0674  5          THEN BEGIN
;  598          0675  5              SIGNAL(cksm$_isectchk,2,.isdnum,.cksm$gl_chksum);   !Print isect checksum
;  599          0676  5              cksm$gl_acucksm = .cksm$gl_acucksm XOR .cksm$gl_chksum;!Accumulate new improved checksum
;  600          0677  4              END;
;  601          0678  3          END;
;  602          0679  2      END;                                       !end of isection loop
;  603          0680  2
;  604          0681  2  !
;  605          0682  2  ! Print checksum of image header and total checksum, and define a DCL
;  606          0683  2  ! symbol with the checksum value
;  607          0684  2  !
;  608          0685  2  SIGNAL(cksm$_ihdchksum,1,.cksm$gl_ihdcksm);
;  609          0686  2  SIGNAL(cksm$_allischk,1,.cksm$gl_acucksm);
;  610          0687  2  define_sym(.cksm$gl_acucksm);
;  611          0688  2
;  612          0689  3  IF NOT (status=$DASSGN(chan=.chan))
;  613          0690  2      THEN SIGNAL(cksm$_closein, 1,
;  614          0691  2                  find_result, .status);
;  615          0692  2
;  616          0693  2  RETURN true
;  617          0694  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$GL_ACUCKSM                     : 0686
                                        01   DD 00185        PUSHL   #1                                  :
                             00000000G  8F   DD 00187        PUSHL   #CKSM$_ALLISC                       :
                            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  DO 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$SIGNAL                  ;
                                50  01  DO 001B3 10$:      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
```

```
;  619          0695  1 %sbttl 'Checksum image header contents'
;  620          0696  1 ROUTINE cksm_header (imageheader) =
;  621          0697  2 BEGIN
;  622          0698  2 !
;  623          0699  2 ! Checksums the non-variable contents of the image header
;  624          0700  2 !
;  625          0701  2 ! INPUTS:
;  626          0702  2 !
;  627          0703  2 !       Imageheader     points to the first block of image header in memory
;  628          0704  2 !
;  629          0705  2 ! NOTE:
;  630          0706  2 !
;  631          0707  2 !       This routine expects the fixed part of the image header to be contained
;  632          0708  2 !       in the first block of the image.
;  633          0709  2 !
;  634          0710  2 MAP
;  635          0711  2     imageheader : REF $BBLOCK;
;  636          0712  2
;  637          0713  2 BIND
;  638          0714  2     cksm = cksm$gl_ihdcksm,
;  639          0715  2     iha = .imageheader + .imageheader[ihd$w_activoff] : VECTOR[,LONG],
;  640          0716  2     ihs = .imageheader + .imageheader[ihd$w_symdbgoff] : VECTOR[,LONG],
;  641          0717  2     ihi = .imageheader + .imageheader[ihd$w_imgidoff] : $BBLOCK,
;  642          0718  2     ihp = .imageheader + .imageheader[ihd$w_patchoff] : VECTOR[,LONG];
;  643          0719  2
;  644          0720  2 cksm = .cksm XOR .imageheader[ihd$l_lnkflags];  !Include the link flags
;  645          0721  2 !
;  646          0722  2 ! Checksum the activation data
;  647          0723  2 !
;  648          0724  2 INCRU i FROM 0 TO 2
;  649          0725  2 DO IF .iha[.i] EQL 0
;  650          0726  2     THEN EXITLOOP
;  651          0727  2     ELSE cksm = .cksm XOR .iha[.i];
;  652          0728  2 !
;  653          0729  2 ! Checksum the symbol table and debug data
;  654          0730  2 !
;  655          0731  2 IF .imageheader[ihd$w_symdbgoff] NEQ 0
;  656          0732  2 THEN INCRU i FROM 0 TO ihs$c_length/4
;  657          0733  2     DO cksm = .cksm XOR .ihs[.i];
;  658          0734  2 !
;  659          0735  2 ! Checksum the image header ident section data
;  660          0736  2 !
;  661          0737  2 ! Not done, since it's all variable data
;  662          0738  2 !
;  663          0739  2
;  664          0740  2 !
;  665          0741  2 ! Checksum the patch data
;  666          0742  2 !
;  667          0743  2 IF .imageheader[ihd$w_patchoff] NEQ 0
;  668          0744  2 THEN INCRU i FROM 0 to (ihp$c_length-1)/4
;  669          0745  2     DO cksm = .cksm XOR .ihp[.i];
;  670          0746  2
;  671          0747  2 RETURN true
;  672          0748  2
;  673          0749  1 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                 SUBL2   #8, SP                          ;
                                7E  7C 00002    CLRQ    -(SP)                           ; 0766
                                7E  D4 00007    CLRL    -(SP)                           ;
                                08  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
```

```
;  597          0771  1 %sbttl 'Checksum image section descriptor'
;  698          0772  1 ROUTINE checksum_section (isd) =
;  699          0773  2 BEGIN
;  700          0774  2 !
;  701          0775  2 ! Checksums the contents of an image section
;  702          0776  2 !
;  703          0777  2 ! INPUTS:
;  704          0778  2 !
;  705          0779  2 !       isd     pointer to the image section descriptor for
;  706          0780  2 !               an image section.
;  707          0781  2 !
;  708          0782  2 MAP
;  709          0783  2     isd : REF $BBLOCK;
;  710          0784  2
;  711          0785  2 BIND
;  712          0786  2     isdvecbyt = .isd : VECTOR[,LONG];
;  713          0787  2
;  714          0788  2 LOCAL
;  715          0789  2     maxblocks,
;  716          0790  2     tblocks,
;  717          0791  2     blockoff,
;  718          0792  2     iblocks;
;  719          0793  2
;  720          0794  2 !
;  721          0795  2 ! Add isd into image header checksum
;  722          0796  2 !
;  723          0797  2 INCRU i FROM 0 TO .isd[isd$w_size]-1
;  724          0798  2     DO cksm$gl_ihdcksm = .cksm$gl_ihdcksm XOR isdvecbyt[.i];
;  725          0799  2
;  726          0800  2 IF .isd[isd$v_dzro]                        !Skip section if demand zero
;  727          0801  2     OR .isd[isd$l_vbn] EQL 0                ! or no disk address assigned
;  728          0802  2     OR .isd[isd$w_pagcnt] EQL 0             ! or no pages (should never happen)
;  729          0803  2 THEN RETURN true;
;  730          0804  2
;  731          0805  2 cksm$gl_chksum = 0;                       !Zero checksum for section
;  732          0806  2 maxblocks = .cksm$gl_maxbyt/512;          !Figure max number of pages
;  733          0807  2 iblocks = .isd[isd$w_pagcnt];             !Get size of image section
;  734          0808  2 blockoff = 0;
;  735          0809  2 WHILE .iblocks GTRU 0                     !Do all blocks in the image section
;  736          0810  3 DO BEGIN
;  737          0811  3     tblocks = MINU(.iblocks,.maxblocks);   !Compute size of this read
;  738          0812  3     IF read_blocks(.tblocks,.isd[isd$l_vbn]+.blockoff)
;  739          0813  3     THEN
;  740          0814  3         !
;  741          0815  3         ! Checksum what we have read
;  742          0816  3         !
;  743          0817  3         INCRU i FROM 0 TO .tblocks*127
;  744          0818  3             DO cksm$gl_chksum = .cksm$gl_chksum XOR .databuffer[.i];
;  745          0819  3     blockoff = .blockoff + .tblocks;       !Update block offset
;  746          0820  3     iblocks = .iblocks - .tblocks;         ! and blocks to go
;  747          0821  2     END;
;  748          0822  2
;  749          0823  2 RETURN true
;  750          0824  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
```

```
;  752          0825  1 %sbttl 'checksum foreign image files'
;  753          0826  1 ROUTINE cksm_foreign =
;  754          0827  2 BEGIN
;  755          0828  2 !
;  756          0829  2 ! Open the file using RMS with a FHC XAB attached to get the End of File
;  757          0830  2 ! information.  Do a checksum of the non-image file.
;  758          0831  2 !
;  759          0832  2 LOCAL
;  760          0833  2     isectdesc : $BBLOCK[isd$k_lenpriv],
;  761          0834  2     status;
;  762          0835  2
;  763          0836  2 find_context[fab$l_xab] = ixabfhc;                !Set XAB address
;  764          0837  2 $XABFHC_INIT(xab=ixabfhc);                        !Initialize XAB
;  765          0838  3 IF NOT (status = $open(fab=.find_context))        !Open the input file
;  766          0839  3 THEN BEGIN
;  767          0840  3     SIGNAL(cksm$_openin, 1,
;  768          0841  3         find_result, .status,
;  769          0842  3         .find_context[fab$l_stv]);
;  770          0843  3     RETURN false;
;  771          0844  2     END;
;  772          0845  2
;  773          0846  2 CH$FILL(0,isd$k_lenpriv,isectdesc);
;  774          0847  2 isectdesc[isd$w_size] = isd$k_lenpriv;             !Make it look like an isd
;  775          0848  2 isectdesc[isd$l_vbn] = 1;                         !File starts with first block
;  776          0849  2 isectdesc[isd$w_pagcnt] = .ixabfhc[xab$l_ebk] +   !Set length of file
;  777          0850  2                         1*(.ixabfhc[xab$w_ffb] NEQ 0);
;  778          0851  2 checksum_section(isectdesc);                      !Checksum it
;  779          0852  2 SIGNAL(cksm$_nonimgchk,1,.cksm$gl_chksum);
;  780          0853  2
;  781          0854  3 IF NOT (status=$CLOSE(fab=.find_context))
;  782          0855  2     THEN SIGNAL(cksm$_closein, T,
;  783          0856  2                 find_result, .status,
;  784          0857  2                 .find_context[fab$l_stv]);
;  785          0858  2
;  786          0859  2 RETURN true
;  787          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$:     MOVC5   #0, (SP), #0, #16, ISECTDESC                    ; 0846
                                                          6E     00056                                                                ;
                                                          6E  10 BO 00057       MOVW    #16, ISECTDESC                                  ; 0847
                                                    0C    AE  01 DO 0005A       MOVL    #1, ISECTDESC+12                                ; 0848
                                                          50  D4 0005E          CLRL    R0                                              ; 0850
                                                    00A0  C7  B5 00060         TSTW    IXABFHC+20                                      ;
                                                          02  13 00064          BEQL    2$                                              ;
                                                          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
```

```
;  789          0861  1 %sbttl 'Call FAOL'
;  790          0862  1 ROUTINE fao_buffer (ctrstr,args) =
;  791          0863  2 BEGIN
;  792          0864  2 !
;  793          0865  2 ! Call FAOL
;  794          0866  2 !
;  795          0867  2 OWN
;  796          0868  2     desc :      $BBLOCK [dsc$c_s_bln],  !Result descriptor
;  797          0869  2     buf :       VECTOR [132, BYTE];     !Output buffer
;  798          0870  2
;  799          0871  2 MAP
;  800          0872  2     ctrstr :    REF VECTOR [,BYTE],
;  801          0873  2     args :      VECTOR [4];
;  802          0874  2
;  803          0875  2 LOCAL
;  804          0876  2     faodesc : $BBLOCK [dsc$c_s_bln];
;  805          0877  2
;  806          0878  2 faodesc [dsc$w_length] = .ctrstr [0];          !Set up descriptor for fao control string
;  807          0879  2 faodesc [dsc$a_pointer] = ctrstr [1];
;  808          0880  2 desc [dsc$w_length] = 132;                    !Set up result descriptor
;  809          0881  2 desc [dsc$a_pointer] = buf;
;  810        P 0882  2 $faol (ctrstr=faodesc,outlen=desc,            !Format the string
;  811          0883  2                 outbuf=desc,prmlst=args);
;  812          0884  2
;  813          0885  2 RETURN desc;                                  !Return value is address of resulting descriptor
;  814          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 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
```

```
;  838          0908  1 %sbttl 'Get file name'
;  839          0909  1 GLOBAL ROUTINE getfilename (fab) =
;  840          0910  2 BEGIN
;  841          0911  2 !++
;  842          0912  2 ! FUNCTIONAL DESCRIPTION:
;  843          0913  2 !
;  844          0914  2 !       This routine returns a string descriptor for a file.
;  845          0915  2 !
;  846          0916  2 ! Inputs:
;  847          0917  2 !
;  848          0918  2 !       fab             Address of the fab
;  849          0919  2 !
;  850          0920  2 ! Outputs:
;  851          0921  2 !
;  852          0922  2 !       Routine value is address of string descriptor for file name
;  853          0923  2 !
;  854          0924  2 !--
;  855          0925  2
;  856          0926  2 MAP
;  857          0927  2     fab : REF $BBLOCK;
;  858          0928  2
;  859          0929  2 BIND
;  860          0930  2     nam = .fab [fab$l_nam] : $BBLOCK;
;  861          0931  2
;  862          0932  2 OWN
;  863          0933  2     filedesc : $BBLOCK[dsc$c_s_bln];
;  864          0934  2
;  865          0935  2 IF (filedesc [dsc$w_length] = .nam [nam$b_rsl]) NEQ 0   !If resultant name present
;  866          0936  2 THEN filedesc [dsc$a_pointer] = .nam [nam$l_rsa]
;  867          0937  2 ELSE IF (filedesc [dsc$w_length] = .nam [nam$b_esl]) NEQ 0 !If expanded name present
;  868          0938  2     THEN filedesc [dsc$a_pointer] = .nam [nam$l_esa]
;  869          0939  3     ELSE BEGIN
;  870          0940  3         filedesc [dsc$w_length] = .fab [fab$b_fns];     !Use filename string
;  871          0941  3         filedesc [dsc$a_pointer] = .fab [fab$l_fna];    ! if all else fails
;  872          0942  2         END;
;  873          0943  2
;  874          0944  2 RETURN filedesc
;  875          0945  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
```

```
; Compilation Complete
```