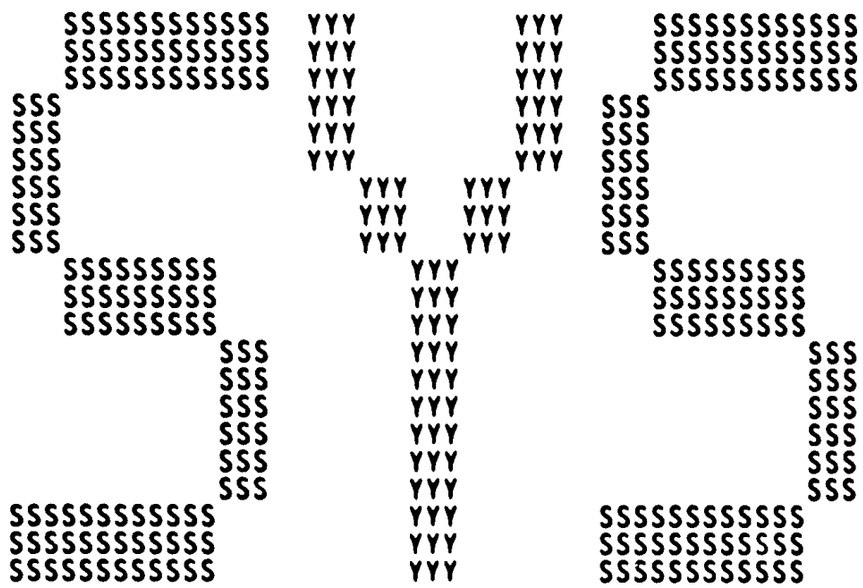

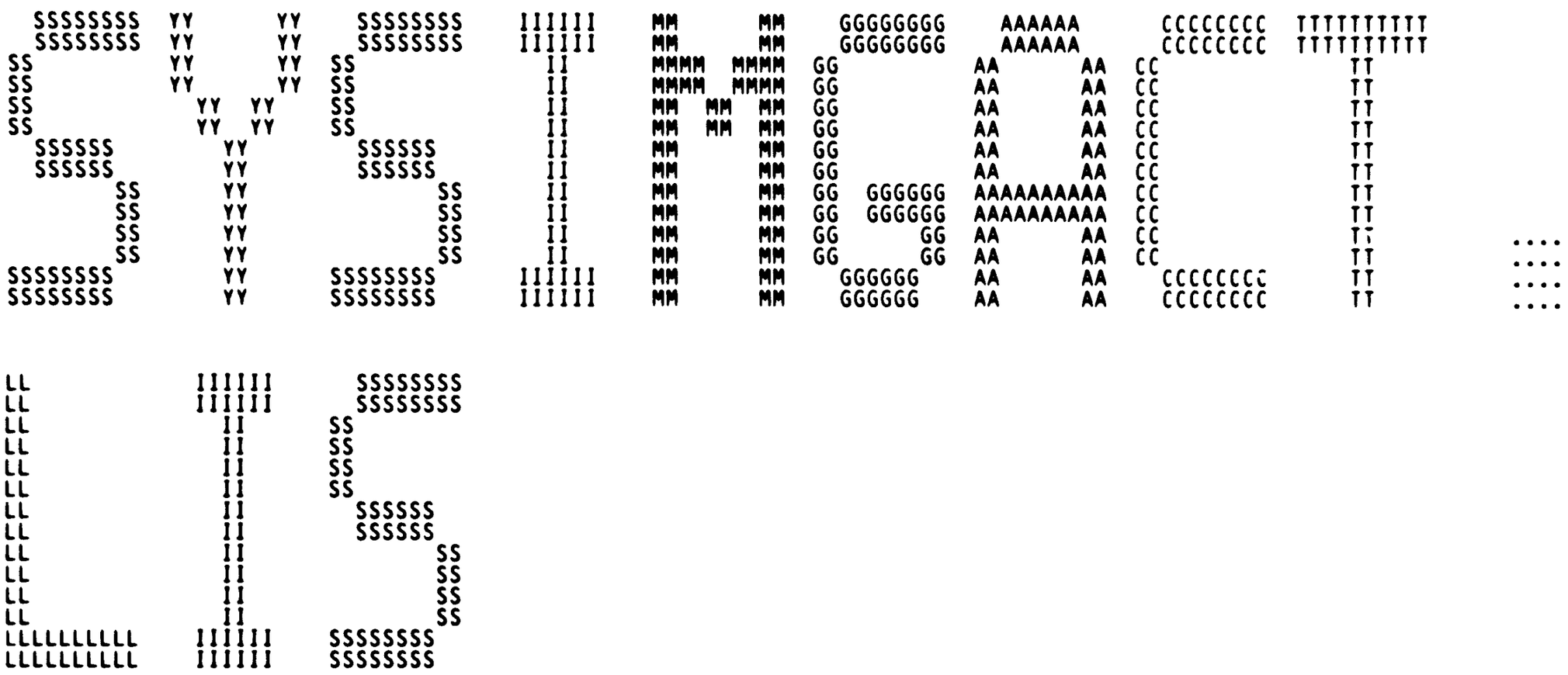
SSSSSSSS YY YY SSSSSSSS IIIIII MM MM GGGGGGGG AAAAAA CCCCCCCC TTTTTTTTTT
SSSSSSSS YY YY SSSSSSSS IIIIII MM MM GGGGGGGG AAAAAA CCCCCCCC TTTTTTTTTT
SS YY YY SS II MMMM MMMM GG AA AA CC TT
SS YY YY SS II MMMM MMMM GG AA AA CC TT
SS YY YY SS II MM MM MM GG AA AA CC TT
SS YY YY SS II MM MM MM GG AA AA CC TT
SSSSSS YY SSSSSS II MM MM GG AA AA CC TT
SSSSSS YY SSSSSS II MM MM GG AA AA CC TT
SS YY SS II MM MM GG GGGGGG AAAAAAAAAA CC TT
SS YY SS II MM MM GG GGGGGG AAAAAAAAAA CC TT
SS YY SS II MM MM GG GG AA AA CC TT
SS YY SS II MM MM GG GG AA AA CC TT
SSSSSSSS YY SSSSSSSS IIIIII MM MM GGGGGG AA AA CCCCCCCC TT
SSSSSSSS YY SSSSSSSS IIIIII MM MM GGGGGG AA AA CCCCCCCC TT
LL IIIIII SSSSSSSS
LL IIIIII SSSSSSSS
LL II SS
LL II SS
LL II SS
LL II SS
LL II SSSSSS
LL II SSSSSS
LL II SS
LL II SS
LL II SS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

```
;  1     0001  0 %TITLE 'SYSIMGACT - Image Activator System Service'
;  2     0002  0 MODULE SYS$IMGACT (
;  3     0003  0                    IDENT = 'V04-001'                ! File: SRC$:SYSIMGACT.B32
;  4     0004  0                    ) =
;  5     0005  1 BEGIN
;  6     0006  1
;  7     0007  1 !*****************************************************************************
;  8     0008  1 !*                                                                           *
;  9     0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
; 10     0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 11     0011  1 !*  ALL RIGHTS RESERVED.                                                     *
; 12     0012  1 !*                                                                           *
; 13     0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 14     0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 15     0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
; 16     0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
; 17     0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
; 18     0018  1 !*  TRANSFERRED.                                                             *
; 19     0019  1 !*                                                                           *
; 20     0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
; 21     0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
; 22     0022  1 !*  CORPORATION.                                                             *
; 23     0023  1 !*                                                                           *
; 24     0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
; 25     0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 26     0026  1 !*                                                                           *
; 27     0027  1 !*                                                                           *
; 28     0028  1 !*****************************************************************************
; 29     0029  1
; 30     0030  1 !++
; 31     0031  1 ! Facility:
; 32     0032  1 !
; 33     0033  1 !       Executive, System Service
; 34     0034  1 !
; 35     0035  1 ! Abstract:
; 36     0036  1 !
; 37     0037  1 !       This module contains the code necessary to map a portion of process
; 38     0038  1 !       address space to a particular image file.
; 39     0039  1 !
; 40     0040  1 ! Environment:
; 41     0041  1 !
; 42     0042  1 !       The bulk of the code in this module executes in executive mode, in
; 43     0043  1 !       a layer outside RMS. One routine executes a small amount of code in
; 44     0044  1 !       kernel mode.
; 45     0045  1 !
; 46     0046  1 !       Note that the image activator is not reentrant.
; 47     0047  1 !
; 48     0048  1 ! Author:
; 49     0049  1 !
; 50     0050  1 !       Lawrence J. Kenah
; 51     0051  1 !
; 52     0052  1 !       The original version of the image activator was written by Peter Lipman.
; 53     0053  1 !       During Version 2 of VMS, extensive enhancements were made by Kathy
; 54     0054  1 !       Morse.
; 55     0055  1 !
; 56     0056  1 ! Creation Date:
; 57     0057  1 !
```

```
;    58          0058  1 !       15 April 1983
;    59          0059  1 !
;    60          0060  1 ! Modified By:
;    61          0061  1 !
;    62          0062  1 !    V04-001    LJK0289         Lawrence J. Kenah       7-Sep-1984
;    63          0063  1 !               Make SET_VECTORS loop into a zero-pass loop if there is no
;    64          0064  1 !               work to do.
;    65          0065  1 !
;    66          0066  1 !    V03B-021   MSH0056         Michael S. Harvey       2-Jul-1984
;    67          0067  1 !               Rearrange new code for previous fix so the merged
;    68          0068  1 !               activation path for a CLI doesn't get screwed up.
;    69          0069  1 !
;    70          0070  1 !    V03B-020   MSH0056         Michael S. Harvey       21-Jun-1984
;    71          0071  1 !               When a secondary image must be substituted (such as
;    72          0072  1 !               with an AME or CLI) for a primary image, make sure to
;    73          0073  1 !               reinitialize the proper context variables, thus preventing
;    74          0074  1 !               Executive mode bugchecks.
;    75          0075  1 !
;    76          0076  1 !    V03B-019   LJK0286         Lawrence J. Kenah       5-Jun-1984
;    77          0077  1 !               Set flag bit in IMAGCTX that indicates to $IMGFIX that
;    78          0078  1 !               shareable images contain initialization code.
;    79          0079  1 !
;    80          0080  1 !    V03B-018   LJK0283         Lawrence J. Kenah       11-May-1984
;    81          0081  1 !               Set the DONE bit in all ICBs that result from a successful
;    82          0082  1 !               activation so that they do not disappear in a later
;    83          0083  1 !               activation that fails.
;    84          0084  1 !
;    85          0085  1 !    V03B-017   LJK0276         Lawrence J. Kenah       8-May-1984
;    86          0086  1 !               Make sure that ICBs are not left dangling when an image
;    87          0087  1 !               has already been activated or along error paths.
;    88          0088  1 !
;    89          0089  1 !    V03B-016   LJK0274         Lawrence J. Kenah       16-Apr-1984
;    90          0090  1 !               Fix ERROR_CLEAN_UP routine.
;    91          0091  1 !
;    92          0092  1 !    V03B-015   LJK0269         Lawrence J. Kenah       31-Mar-1984
;    93          0093  1 !               Miscellaneous small changes.
;    94          0094  1 !               Add code that cleans up if an error is detected after
;    95          0095  1 !               some pages have been successfully mapped.
;    96          0096  1 !               Set internal status bit that indicates that the P0 half
;    97          0097  1 !               of a P1 merge operation is taking place.
;    98          0098  1 !               Change the way that privileged vector context is stored.
;    99          0099  1 !               Return correct context if image is already mapped.
;   100          0100  1 !               Add state flags passed from $IMGACT to $IMGFIX.
;   101          0101  1 !               Add error routine that performs a complete cleanup if
;   102          0102  1 !               an error occurs after an image has been successfully mapped.
;   103          0103  1 !               Defer addition of privileged vectors until fixups are done.
;   104          0104  1 !
;   105          0105  1 !    V03B-014   LJK0268         Lawrence J. Kenah       29-Mar-1984
;   106          0106  1 !               Turn on code that records SHRCNT and USECNT in KFE.
;   107          0107  1 !
;   108          0108  1 !    V03B-013   LJK0267         Lawrence J. Kenah       28-Mar-1984
;   109          0109  1 !               Use name stored in KFE as global section name. This allows
;   110          0110  1 !               correct redirection of shareable images installed /SHARED.
;   111          0111  1 !
;   112          0112  1 !    V03B-013   LJK0266         Lawrence J. Kenah       27-Mar-1984
;   113          0113  1 !               Add major and minor ID consistency checks on the
;   114          0114  1 !               image header.
```

```
;  115     0115  1 ;
;  116     0116  1 ;    V03B-012   MSH0022         Michael S. Harvey       25-Mar-1984
;  117     0117  1 ;               Unconceal known file lookups.
;  118     0118  1 ;
;  119     0119  1 ;    V03B-011   WMC0003         Wayne Cardoza           24-Mar-1984
;  120     0120  1 ;               Call RM$SET to set up image I/O area.
;  121     0121  1 ;
;  122     0122  1 ;    V03B-010   WMC0002         Wayne Cardoza           23-Jan-1984
;  123     0123  1 ;               Misc small fixes.
;  124     0124  1 ;               Add sequential loading of images.
;  125     0125  1 ;
;  126     0126  1 ;    V03B-009   LJK0244         Lawrence J. Kenah       23-Aug-1983
;  127     0127  1 ;               Turn on image accounting. Set default stack size.
;  128     0128  1 ;
;  129     0129  1 ;    V03B-008   WMC0001         Wayne Cardoza           05-Aug-1983
;  130     0130  1 ;               Remove code for passing back FAB on failure.
;  131     0131  1 ;
;  132     0132  1 ;    V03B-007   LJK0242         Lawrence J. Kenah       2-Aug-1983
;  133     0133  1 ;               Add support for writable global sections.
;  134     0134  1 ;
;  135     0135  1 ;    V03B-006   LJK0235         Lawrence J. Kenah       26-Jul-1983
;  136     0136  1 ;               Add concept of image base address, different from starting
;  137     0137  1 ;               address. Continue fixing bugs and cleaning up loose ends.
;  138     0138  1 ;
;  139     0139  1 ;    V03B-005   LJK0232         Lawrence J. Kenah       21-Jul-1983
;  140     0140  1 ;               The starting address of the image I/O segment should
;  141     0141  1 ;               be page aligned.
;  142     0142  1 ;
;  143     0143  1 ;    V03B-004   LJK0230         Lawrence J. Kenah       18-Jul-1983
;  144     0144  1 ;               Add system version check. Add _001 suffix to main image name.
;  145     0145  1 ;
;  146     0146  1 ;    V03B-003   LJK0228         Lawrence J. Kenah       12-Jul-1983
;  147     0147  1 ;               Propogate setting of EXPREG flag into the ICB.
;  148     0148  1 ;
;  149     0149  1 ;    V03B-002   LJK0219         Lawrence J. Kenah       29-Jun-1983
;  150     0150  1 ;               Add support for compatibility mode and other alias images.
;  151     0151  1 ;               Fix the many bugs that were discovered during debugging.
;  152     0152  1 ;
;  153     0153  1 ;    V03B-001   LJK0200         Lawrence J. Kenah       15-Apr-1983
;  154     0154  1 ;               Being a complete rewrite of the original image activator
;  155     0155  1 ;               system service.
;  156     0156  1 ;--
;  157     0157  1
```

```
;  159          0158  1 %SBTTL 'Declarations'
;  160          0159  1
;  161          0160  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;  162          0161  1
;  163          0162  1 PSECT
;  164          0163  1         CODE    = YF$$SYSIMGACT (WRITE),
;  165          0164  1         PLIT    = YF$$SYSIMGACT (WRITE, EXECUTE);
;  166          0165  1
;  167          0166  1 LIBRARY 'SYS$LIBRARY:LIB.L32';          ! Define system data structures
;  168          0167  1
;  169          0168  1 REQUIRE 'LIB$:IMGMSGDEF.R32';          ! Get status code definitions
;  170          0254  1 REQUIRE 'LIB$:IMGACTCTX.R32';          ! Define internal structures
;  171          0401  1
;  172          0402  1 ! Machine dependent features
;  173          0403  1
;  174          0404  1 BUILTIN
;  175          0405  1     MOVC5,
;  176          0406  1     MOVPSL,
;  177          0407  1     MTPR,
;  178          0408  1     PROBER,
;  179          0409  1     PROBEW,
;  180          0410  1     INSQUE,
;  181          0411  1     REMQUE;
;  182          0412  1
;  183          0413  1 ! Miscellaneous internal symbols
;  184          0414  1
;  185          0415  1 LITERAL
;  186          0416  1     TRUE = 1,
;  187          0417  1     FALSE = 0,
;  188          0418  1     BYTES_PER_PAGE = 512,
;  189          0419  1     RETURN_BUFFER_SIZE = 512,
;  190          0420  1     EXTRA_USER_STACK = 2,
;  191          0421  1     END_OF_P0_SPACE = %X'3FFFFFFF',
;  192          0422  1     EXEC_PROT = (PRT$C_UREW ^ 8) OR PSL$C_EXEC;
;  193          0423  1
;  194          0424  1 ! Linkage declaration for procedures invoked with the $CMKRNL system service
;  195          0425  1
;  196          0426  1 LINKAGE
;  197          0427  1     SYS_CMKRNL = CALL : GLOBAL (PCB = 4);
;  198          0428  1
;  199          0429  1 ! Internal references
;  200          0430  1
;  201          0431  1 FORWARD ROUTINE
;  202          0432  1     CHECK_PARAMS,
;  203          0433  1     INIT_WINDOW         : SYS_CMKRNL,
;  204          0434  1     CHECK_MATCH_CONTROL,
;  205          0435  1     GET_OTHER_IMAGE,
;  206          0436  1     END_PROCESSING,
;  207          0437  1     SET_CONTROL_REGION  : SYS_CMKRNL,
;  208          0438  1     GET_LOCK,
;  209          0439  1     RELEASE_LOCK,
;  210          0440  1     ERROR_CLEAN_UP      : NOVALUE,
;  211          0441  1     SET_VECTORS;
;  212          0442  1
;  213          0443  1 ! Routines that will be referenced by the ISD mapping routines as well
;  214          0444  1 ! as internally
;  215          0445  1
```

```
;  216     0446  1 FORWARD ROUTINE
;  217     0447  1     IMG$GET_HEADER,
;  218     0448  1     IMG$OPEN_IMAGE,
;  219     0449  1     IMG$ALLOCATE_ICB,
;  220     0450  1     IMG$DEALLOCATE_ICB  : NOVALUE;
;  221     0451  1
;  222     0452  1 ! Linkage declarations to JSB routines in exec
;  223     0453  1
;  224     0454  1 LINKAGE
;  225     0455  1     RM_RESET          = JSB : NOPRESERVE (0,1,2)
;  226     0456  1                       NOTUSED (3,4,5,6,7,8,9,10,11),
;  227     0457  1     RM_SET            = JSB (REGISTER = 0, REGISTER = 1) :
;  228     0458  1                       NOPRESERVE (0,1,2)
;  229     0459  1                       NOTUSED (3,4,5,6,7,8,9,10,11),
;  230     0460  1     EXE_ALOP1PROC     = JSB (REGISTER = 1; REGISTER = 1,REGISTER = 2) :
;  231     0461  1                       NOPRESERVE (3)
;  232     0462  1                       NOTUSED (4,5,6,7,8,9,10,11),
;  233     0463  1     EXE_DEAP1         = JSB (REGISTER = 0, REGISTER = 1) :
;  234     0464  1                       NOPRESERVE (2,3)
;  235     0465  1                       NOTUSED (4,5,6,7,8,9,10,11),
;  236     0466  1     EXE_MAXACMODE     = JSB (REGISTER = 0; REGISTER = 0) :
;  237     0467  1                       NOPRESERVE (1)
;  238     0468  1                       NOTUSED (2,3,4,5,6,7,8,9,10,11),
;  239     0469  1     EXE_PROBE_DSC     = JSB (REGISTER = 1; REGISTER = 1, REGISTER = 2) :
;  240     0470  1                       NOPRESERVE (3)
;  241     0471  1                       NOTUSED (4,5,6,7,8,9,10,11),
;  242     0472  1     IMG_IS_IT_MAPPED  = JSB (REGISTER = 0; REGISTER = 1) :
;  243     0473  1                       PRESERVE (2,3,4,5,6,7)
;  244     0474  1                       NOTUSED (8,9,10,11),
;  245     0475  1     IOC_VERIFYCHAN    = JSB (REGISTER = 0; REGISTER = 1, REGISTER = 2) :
;  246     0476  1                       NOPRESERVE (3)
;  247     0477  1                       NOTUSED (4,5,6,7,8,9,10,11),
;  248     0478  1     FIL_INIWCB        = JSB (REGISTER = 1, REGISTER = 2, REGISTER = 3;
;  249     0479  1                       REGISTER = 2) :
;  250     0480  1                       PRESERVE (3,4,5)
;  251     0481  1                       NOTUSED (6,7,8,9,10,11);
;  252     0482  1
;  253     0483  1 !  External references with explicit linkage mechanisms
;  254     0484  1
;  255     0485  1 EXTERNAL ROUTINE
;  256     0486  1     EXE$ALOP1PROC     : EXE_ALOP1PROC,
;  257     0487  1     EXE$DEAP1         : EXE_DEAP1,
;  258     0488  1     EXE$MAXACMODE     : EXE_MAXACMODE,
;  259     0489  1     EXE$PROBER_DSC    : EXE_PROBE_DSC,
;  260     0490  1     EXE$PROBEW_DSC    : EXE_PROBE_DSC,
;  261     0491  1     IMG$IS_IT_MAPPED  : IMG_IS_IT_MAPPED,
;  262     0492  1     IOC$VERIFYCHAN    : IOC_VERIFYCHAN,
;  263     0493  1     FIL$INIWCB        : FIL_INIWCB,
;  264     0494  1     RM$RESET          : RM_RESET,
;  265     0495  1     RM$SET            : RM_SET;
;  266     0496  1
;  267     0497  1 ! External procedure references
;  268     0498  1
;  269     0499  1 EXTERNAL ROUTINE
;  270     0500  1     FIL$OPENFILE,
;  271     0501  1     IMG$DECODE_IHD,
;  272     0502  1     IMG$DO_WORK_LIST,
```

```
;  273     0503  1     MMG$CRETVA                      : SYS_CMKRNL;
;  274     0504  1
;  275     0505  1 ! External data cells in P1 space
;  276     0506  1
;  277     0507  1 EXTERNAL
;  278     0508  1     CTL$AG_CMEDATA,
;  279     0509  1     CTL$AL_STACK                    : VECTOR [4],
;  280     0510  1     CTL$A_DISPVEC,
;  281     0511  1     CTL$GL_CTLBASVA,
;  282     0512  1     CTL$GL_FIXUPLNK,
;  283     0513  1     CTL$GL_IMGHDRBF,
;  284     0514  1     CTL$GL_PHD                      : REF $BBLOCK,
;  285     0515  1     CTL$GL_RMSBASE,
;  286     0516  1     CTL$GL_VOLUMES,
;  287     0517  1     CTL$GQ_PROCPRIV                 : VECTOR [2],
;  288     0518  1     IAC$AW_VECSET                   : VECTOR [4,WORD],
;  289     0519  1     IAC$GL_ICBFL                    : VECTOR [2],
;  290     0520  1     IAC$AL_IMGACTBUF,
;  291     0521  1     IAC$GL_FIRST_ICB,
;  292     0522  1     IAC$GL_IMAGCTX                  : $BBLOCK,
;  293     0523  1     IAC$GL_MAIN_ICB,
;  294     0524  1     IAC$GL_WORK_LIST                : VECTOR [2],
;  295     0525  1     IAC$GL_IMAGE_LIST               : VECTOR [2],
;  296     0526  1     IAC$GL_STACK_SIZE,
;  297     0527  1     PIO$GW_IIOIMPA                  : $BBLOCK;
;  298     0528  1
;  299     0529  1 ! The following cells in P1 space are used exclusively for image accounting
;  300     0530  1
;  301     0531  1 EXTERNAL
;  302     0532  1     CTL$GL_ICPUTIM,
;  303     0533  1     CTL$GL_IFAULTS,
;  304     0534  1     CTL$GL_IFAULTIO,
;  305     0535  1     CTL$GL_IWSPEAK,
;  306     0536  1     CTL$GL_IPAGEFL,
;  307     0537  1     CTL$GL_IDIOCNT,
;  308     0538  1     CTL$GL_IBIOCNT,
;  309     0539  1     CTL$GL_IVOLUMES,
;  310     0540  1     CTL$GQ_ISTART                   : VECTOR [2];
;  311     0541  1
;  312     0542  1 ! External data cells in system space
;  313     0543  1
;  314     0544  1 EXTERNAL
;  315     0545  1     EXE$GL_ACMFLAGS                 : $BBLOCK,
;  316     0546  1     EXE$GL_FLAGS                    : $BBLOCK,
;  317     0547  1     EXE$GL_KNOWN_FILES,
;  318     0548  1     EXE$GL_SYSID_LOCK,
;  319     0549  1     EXE$GQ_KFE_LCKNAM,
;  320     0550  1     EXE$GQ_SYSTIME                  : VECTOR [2],
;  321     0551  1     SGN$GW_IMGIOCNT                 : WORD;
;  322     0552  1
;  323     0553  1 ! Miscellaneous constants defined elsewhere
;  324     0554  1
;  325     0555  1 EXTERNAL LITERAL
;  326     0556  1     EXE$C_SYSEFN        : UNSIGNED (6),
;  327     0557  1     EXE$V_INIT          : UNSIGNED (6),
;  328     0558  1     SYS$K_VERSION       : UNSIGNED (31);
;  329     0559  1
```

```
;  330          0560  1 ! Make the bit position of the INIT flag available to BLISS
;  331          0561  1
;  332          0562  1 MACRO EXE_V_INIT = 0, EXE$V_INIT, 1, 0 %;
;  333          0563  1
;  334          0564  1 ! Some miscellaneous address definitions
;  335          0565  1
;  336          0566  1 ! The first part of the image activator scratch area is divided up into two
;  337          0567  1 ! large pieces, each of which is further subdivided into a FAB, a NAM block,
;  338          0568  1 ! a 512-byte block into which each succeeding block of the image header will
;  339          0569  1 ! be read, and a buffer that will receive the decoded image header. The
;  340          0570  1 ! decoded image header is assumed to be smaller than a page. The area that
;  341          0571  1 ! follows these buffers is used as OWN storage by the image activator.
;  342          0572  1
;  343          0573  1 LITERAL
;  344          0574  1     INPUT_BUFFER_SIZE = BYTES_PER_PAGE,
;  345          0575  1     IHD_BUFFER_SIZE   = BYTES_PER_PAGE;
;  346          0576  1
;  347          0577  1 BIND
;  348          0578  1     INPUT_BUFFER = IAC$AL_IMGACTBUF,
;  349          0579  1     PRIMARY_IHD  = INPUT_BUFFER + 512,
;  350          0580  1     AUX_BUFFER   = PRIMARY_IHD  + 512,
;  351          0581  1     AUX_IHD      = AUX_BUFFER   + 512,
;  352          0582  1     PRIMARY_FAB  = AUX_IHD      + 512,
;  353          0583  1     PRIMARY_NAM  = PRIMARY_FAB  + FAB$K_BLN,
;  354          0584  1     AUX_FAB      = PRIMARY_NAM  + NAM$K_BLN,
;  355          0585  1     AUX_NAM      = AUX_FAB      + FAB$K_BLN,
;  356          0586  1     RESULT_NAME  = AUX_NAM      + NAM$K_BLN,
;  357          0587  1     OWN_STORAGE  = RESULT_NAME  + NAM$C_MAXRSS  : $BBLOCK;
;  358          0588  1
;  359          0589  1 ! There are eight pages set aside in P1 space (in module SHELL) for
;  360          0590  1 ! the image activator scratch area. The following assumption guarantees
;  361          0591  1 ! that the scratch area that is defined here fits into eight pages.
;  362          0592  1
;  363        P 0593  1     $ASSUME ( OWN_STORAGE_SIZE + (2 * (INPUT_BUFFER_SIZE +
;  364        P 0594  1                                        IHD_BUFFER_SIZE +
;  365        P 0595  1                                        FAB$K_BLN +
;  366        P 0596  1                                        NAM$K_BLN) ),
;  367        P 0597  1                 LEQU,
;  368          0598  1                 8 * BYTES_PER_PAGE );
```

```
;  370          0599  1 %SBTTL 'EXE$$IMGACT - Image Activator System Service Routine'
;  371          0600  1
;  372          0601  1 GLOBAL ROUTINE EXE$$IMGACT
;  373          0602  1     (IMAGE_NAME_ADDR , DEFAULT_NAME_ADDR , BUFFER , CONTROL_FLAGS ,
;  374          0603  1     INADR , RETADR , IDENT , MODE ) =
;  375          0604  1
;  376          0605  1 !++
;  377          0606  1 ! FUNCTIONAL DESCRIPTION:
;  378          0607  1 !
;  379          0608  1 !       This routine receives control from the system service dispatcher
;  380          0609  1 !       to perform the actual work of activating an image.
;  381          0610  1 !
;  382          0611  1 ! CALLING SEQUENCE:
;  383          0612  1 !
;  384          0613  1 !    CALLx G^SYS$IMGACT
;  385          0614  1 !
;  386          0615  1 ! FORMAL PARAMETERS:
;  387          0616  1 !
;  388          0617  1 !       TBS
;  389          0618  1 !
;  390          0619  1 ! STATUS CODES:
;  391          0620  1 !
;  392          0621  1 !       TBS
;  393          0622  1 !--
;  394          0623  1
;  395          0624  2 BEGIN
;  396          0625  2
;  397          0626  2 BUILTIN
;  398          0627  2     AP,
;  399          0628  2     CALLG;
;  400          0629  2
;  401          0630  2 BIND FLAGS = OWN_STORAGE [INPUT_FLAGS] : $BBLOCK;
;  402          0631  2
;  403          0632  2 LOCAL
;  404          0633  2     ICB_ADR     : REF $BBLOCK,
;  405          0634  2     IHD_CTX     : $BBLOCK [CTX_K_LENGTH],
;  406          0635  2     STATUS;
;  407          0636  2
;  408          0637  2 ! The portion of the impure area that is used as OWN storage is filled with
;  409          0638  2 ! zeros. This initializes system service parameters, copies of input
;  410          0639  2 ! parameters, and the like.
;  411          0640  2
;  412          0641  2 CH$FILL (0, OWN_STORAGE_SIZE, OWN_STORAGE);
;  413          0642  2 OWN_STORAGE [FINAL_STATUS] = SS$_NORMAL;                ! Assume successful completion
;  414          0643  2 OWN_STORAGE [USER_STACK_SIZE] = EXTRA_USER_STACK;       ! Assume a minimal user stack
;  415          0644  2
;  416          0645  2 ! No access check is required for the activation flags, which are present in
;  417          0646  2 ! the argument list itself. A safe copy must be made, however, to insure that
;  418          0647  2 ! the flags are not destroyed by a mapping request issued by the image
;  419          0648  2 ! activator. To avoid this, the flags are stored away in a safe place.
;  420          0649  2
;  421          0650  2 OWN_STORAGE [INPUT_FLAGS] = .CONTROL_FLAGS;
;  422          0651  2
;  423          0652  2 IF .FLAGS [IAC$V_SETVECTOR]
;  424          0653  2 THEN RETURN SET_VECTORS();
;  425          0654  2
;  426          0655  2 ! Save the callers mode for use in various checks
```

```
;  427          0656  2
;  428          0657  2 MOVPSL (OWN_STORAGE [CALL_MODE] );
;  429          0658  2
;  430          0659  2 ! Set up the starting points for the various privileged vectors
;  431          0660  2
;  432          0661  2 INCRU I FROM 0 TO 3 DO
;  433          0662  3     BEGIN
;  434          0663  3
;  435          0664  3     BIND
;  436          0665  3         DISPVEC = CTL$A_DISPVEC + (.I * 256) : LONG;
;  437          0666  3
;  438          0667  3     DISPVEC = .IAC$AW_VECSET [.I];
;  439          0668  3
;  440          0669  2     END;
;  441          0670  2
;  442          0671  2 ! The input parameter list is checked for accessibility and the parameters are
;  443          0672  2 ! stored in the impure area for later use.
;  444          0673  2
;  445          0674  2 STATUS = CALLG (.AP, CHECK_PARAMS);
;  446          0675  2 IF NOT .STATUS THEN RETURN .STATUS;
;  447          0676  2
;  448          0677  2 ! Several other miscellaneous areas need to be initialized if this is
;  449          0678  2 ! not a activation that merges an additional image into existing address
;  450          0679  2 ! space. Note that the fixup vector listhead is unconditionally cleared.
;  451          0680  2
;  452          0681  2 CTL$GL_FIXUPLNK = 0;                    ! Set the fixup vector list to empty
;  453          0682  2 IF NOT .FLAGS [IAC$V_MERGE]
;  454          0683  2 THEN
;  455          0684  3     BEGIN
;  456          0685  3     RMS$RESET ();                       ! Clear the image I/O segment
;  457          0686  3     IAC$GL_IMAGCTX = 0;                 ! Start with a clean context slate
;  458          0687  3     OWN_STORAGE [MAIN_PROGRAM] = TRUE;  ! Indicate that this is the activation
;  459          0688  3     END                                 !  of a main program
;  460          0689  2 ELSE
;  461          0690  3     BEGIN
;  462          0691  3
;  463          0692  3     BIND CONTEXT = IAC$GL_IMAGCTX : VECTOR [2,WORD];
;  464          0693  3
;  465          0694  3     CONTEXT [1] = 0;                    ! Only clear flags passed to $IMGFIX
;  466          0695  2     END;
;  467          0696  2
;  468          0697  2 ! Before we open the image file, we need to check whether the image is already
;  469          0698  2 ! mapped. If it is, we simply return successfully, passing back as much data
;  470          0699  2 ! as is available about the image.
;  471          0700  2
;  472          0701  2 STATUS = IMG$ALLOCATE_ICB (ICB_ADR);     ! Allocate an ICB for the primary image
;  473          0702  2
;  474          0703  2 IF NOT .STATUS THEN RETURN .STATUS;
;  475          0704  2
;  476          0705  2 ! The following check is only made for a merge activation
;  477          0706  2
;  478          0707  2 IF .FLAGS [IAC$V_MERGE]
;  479          0708  2 THEN
;  480          0709  3     BEGIN
;  481          0710  3
;  482          0711  3     BIND
;  483          0712  3         INPUT_NAME = OWN_STORAGE [IMAGE_NAME_DESC] : $BBLOCK,
```

```
;  484          0713  3          ICB_NAME = ICB_ADR [ICB$T_IMAGE_NAME] : VECTOR [,BYTE],
;  485          0714  3          RETADR = .OWN_STORAGE [RETURN_ARRAY_ADDRESS] : VECTOR [2],
;  486          0715  3          BUFFER = .OWN_STORAGE [BUFFER_ADDRESS] : VECTOR [128];
;  487          0716  3
;  488          0717  3      LOCAL
;  489          0718  3          MAPPED_ICB : REF $BBLOCK;
;  490          0719  3
;  491          0720  3      ICB_NAME [0] = .INPUT_NAME [DSC$W_LENGTH];
;  492          0721  3      MOVC5 (
;  493          0722  3          INPUT_NAME [DSC$W_LENGTH],
;  494          0723  3          .INPUT_NAME [DSC$A_POINTER],
;  495          0724  3          %REF(0),
;  496          0725  3          %REF(ICB$S_IMAGE_NAME-1),
;  497          0726  3          ICB_NAME [1]);
;  498          0727  3      STATUS = IMG$IS_IT_MAPPED (ICB_NAME; MAPPED_ICB);
;  499          0728  3
;  500          0729  3      ! If the shareable image has already been mapped, we return successfully
;  501          0730  3      ! after passing back to the caller whatever information is available.
;  502          0731  3
;  503          0732  3      IF .STATUS EQL SS$_NORMAL
;  504          0733  3      THEN
;  505          0734  4          BEGIN
;  506          0735  4
;  507          0736  4          IF RETADR NEQU 0
;  508          0737  4          THEN
;  509          0738  5              BEGIN
;  510          0739  5              RETADR [0] = .MAPPED_ICB [ICB$L_STARTING_ADDRESS];
;  511          0740  5              RETADR [1] = .MAPPED_ICB [ICB$L_END_ADDRESS];
;  512          0741  4              END;
;  513          0742  4
;  514          0743  4          IF BUFFER NEQU 0
;  515          0744  4          THEN
;  516          0745  5              BEGIN
;  517          0746  5
;  518          0747  5              BIND IFD = BUFFER [3] : $BBLOCK;
;  519          0748  5
;  520          0749  5              BUFFER [0] = 0;
;  521          0750  5              BUFFER [1] = IFD;
;  522          0751  5              BUFFER [2] = 0;
;  523          0752  5              IFD [IFD$W_CHAN] = .MAPPED_ICB [ICB$W_CHAN];
;  524          0753  5              IFD [IFD$W_FLAGS] = .IAC$GL_IMAGCTX;
;  525          0754  5
;  526          0755  4              END;
;  527          0756  4
;  528          0757  4          IMG$DEALLOCATE_ICB (.ICB_ADR);
;  529          0758  4
;  530          0759  4          RETURN SS$_NORMAL;
;  531          0760  4
;  532          0761  3          END;
;  533          0762  3
;  534          0763  2      END;
;  535          0764  2
;  536          0765  2  IAC$GL_FIRST_ICB = .ICB_ADR;              ! Remember address of this ICB
;  537          0766  2  IF .OWN_STORAGE [MAIN_PROGRAM]
;  538          0767  2  THEN IAC$GL_MAIN_ICB = .ICB_ADR;          ! Remember it here, too, if main program
;  539          0768  2  ICB_ADR [ICB$L_CONTEXT] = IRD_CTX;       ! Store address before opening image
;  540          0769  2
```

```
;  541     0770  2 STATUS = GET_LOCK();                          ! Lock the KFE data base for read access
;  542     0771  2 IF NOT .STATUS THEN RETURN .STATUS;
;  543     0772  2
;  544     0773  2 STATUS = IMG$OPEN_IMAGE (                     ! Open the image file
;  545     0774  2     OWN_STORAGE [IMAGE_NAME_DESC],
;  546     0775  2     OWN_STORAGE [DFLT_NAME_DESC],
;  547     0776  2     PRIMARY_FAB,
;  548     0777  2     PRIMARY_NAM,
;  549     0778  2     RESULT_NAME,
;  550     0779  2     .ICB_ADR);
;  551     0780  2 IF NOT .STATUS
;  552     0781  2 THEN
;  553     0782  3     BEGIN
;  554     0783  3     IMG$DEALLOCATE_ICB (.ICB_ADR);
;  555     0784  3     RELEASE_LOCK ();
;  556     0785  3     RETURN .STATUS
;  557     0786  2     END;
;  558     0787  2
;  559     0788  2 ! The IHD_CTX context block stores the image header data that does not need
;  560     0789  2 ! to exist once the image activator is done. Because this ICB represents the
;  561     0790  2 ! image whose name was passed directly to the image activator, the primary
;  562     0791  2 ! input buffer and IHD buffer are used.
;  563     0792  2
;  564     0793  2 IHD_CTX [CTX_L_BUFFER] = INPUT_BUFFER;
;  565     0794  2 IHD_CTX [CTX_L_IHDBUF] = PRIMARY_IHD;
;  566     0795  2
;  567     0796  2 STATUS = IMG$GET_HEADER (.ICB_ADR);          ! Decode and store away the IHD contents
;  568     0797  2 IF NOT .STATUS
;  569     0798  2 THEN
;  570     0799  3     BEGIN
;  571     0800  3     $DASSGN (CHAN = .ICB_ADR [ICB$W_CHAN]);
;  572     0801  3     ERROR_CLEAN_UP ();
;  573     0802  3     IMG$DEALLOCATE_ICB (.ICB_ADR);
;  574     0803  3     RELEASE_LOCK ();
;  575     0804  3     RETURN .STATUS
;  576     0805  2     END;
;  577     0806  2
;  578     0807  2 ! There are several alias images that cause a secondary image to be activated,
;  579     0808  2 ! leaving the primary image opened and its name stored in P1 space for later
;  580     0809  2 ! possible use by the secondary image, the one actually activated.
;  581     0810  2
;  582     0811  2 IF .IHD_CTX [CTX_W_ALIAS] NEQ IHD$C_NATIVE
;  583     0812  2 THEN
;  584     0813  3     BEGIN
;  585     0814  3     STATUS = GET_OTHER_IMAGE (.ICB_ADR);
;  586     0815  3     IF NOT .STATUS
;  587     0816  3     THEN
;  588     0817  4         BEGIN
;  589     0818  4         IMG$DEALLOCATE_ICB (.ICB_ADR);
;  590     0819  4         RELEASE_LOCK ();
;  591     0820  4         RETURN .STATUS
;  592     0821  3         END;
;  593     0822  2     END;
;  594     0823  2
;  595     0824  2 ! Several other fields in the ICB must be loaded with information obtained
;  596     0825  2 ! from the input parameter list.
;  597     0826  2
```

```
;  598          0827  2 ! Note that EXPREG is only valid when mapping into P0 space. Note further that
;  599          0828  2 ! the input map range is specified explicitly, rather than using the EXPREG
;  600          0829  2 ! flag in the ICB, which is only meaningful in ICBs for shareable images
;  601          0830  2 ! implicitly referenced during the activation of the primary image.
;  602          0831  2
;  603          0832  2 IF .FLAGS [IAC$V_EXPREG]
;  604          0833  2 THEN
;  605          0834  3     BEGIN
;  606          0835  3
;  607          0836  3     BIND
;  608          0837  3         PHD = .CTL$GL_PHD       : $BBLOCK;
;  609          0838  3
;  610          0839  3     ICB_ADR [ICB$L_STARTING_ADDRESS] = .PHD [PHD$L_FREPOVA];
;  611          0840  3     ICB_ADR [ICB$L_END_ADDRESS] = END_OF_P0_SPACE;
;  612          0841  3     END
;  613          0842  2 ELSE
;  614          0843  3     BEGIN
;  615          0844  3     ICB_ADR [ICB$L_STARTING_ADDRESS] = .OWN_STORAGE [INPUT_START_ADDRESS];
;  616          0845  3     ICB_ADR [ICB$L_END_ADDRESS] = .OWN_STORAGE [INPUT_END_ADDRESS];
;  617          0846  2     END;
;  618          0847  2
;  619          0848  2 ICB_ADR [ICB$B_ACCESS_MODE] = .OWN_STORAGE [ACCESS_MODE];
;  620          0849  2 ICB_ADR [ICB$B_ACT_CODE] =
;  621          0850  3     (IF .FLAGS [IAC$V_MERGE]
;  622          0851  3      THEN ICB$K_MERGED_IMAGE
;  623          0852  3      ELSE ICB$K_MAIN_PROGRAM);
;  624          0853  2 ICB_ADR [ICB$L_MATCH_CONTROL] = .OWN_STORAGE [MATCH_CONTROL];
;  625          0854  2 ICB_ADR [ICB$L_VERSION] = .OWN_STORAGE [VERSION];
;  626          0855  2
;  627          0856  2 ! The image control block is inserted into the work list where it can be
;  628          0857  2 ! retrieved by the routine that converts ISDs into mapping requests.
;  629          0858  2
;  630          0859  2 INSQUE (.ICB_ADR , .IAC$GL_WORK_LIST [1]);      ! Insert at tail of work list
;  631          0860  2 STATUS = IMG$DO_WORK_LIST();
;  632          0861  2 IF NOT .STATUS THEN
;  633          0862  3     BEGIN
;  634          0863  3     ERROR_CLEAN_UP ();
;  635          0864  3     RELEASE_LOCK ();
;  636          0865  3     RETURN .STATUS
;  637          0866  2     END;
;  638          0867  2
;  639          0868  2 STATUS = END_PROCESSING (.ICB_ADR);      ! Set final state for image
;  640          0869  2 IF NOT .STATUS THEN
;  641          0870  3     BEGIN
;  642          0871  3     ERROR_CLEAN_UP ();
;  643          0872  3     RELEASE_LOCK ();
;  644          0873  3     RETURN .STATUS
;  645          0874  2     END;
;  646          0875  2
;  647          0876  2 RELEASE_LOCK();                          ! Allow exclusive access again
;  648          0877  2 RETURN .OWN_STORAGE [FINAL_STATUS];      ! ... and return
;  649          0878  2
;  650          0879  1 END;                                     ! End of routine EXE$IMGACT main routine


                                                        .TITLE  SYS$IMGACT SYSIMGACT - Image Activator System S
;                                                               ervice
```

```
                                                                N 11

                                                        .IDENT  \V04-001\

                                                        .EXTRN  EXE$ALOP1PROC, EXE$DEAP1
                                                        .EXTRN  EXE$MAXACMODE, EXE$PROBER_DSC
                                                        .EXTRN  EXE$PROBEW_DSC, IMG$IS_IT_MAPPED
                                                        .EXTRN  IOC$VERIFYCHAN, FIL$INIWCB
                                                        .EXTRN  RM$RESET, RM$SET
                                                        .EXTRN  FIL$OPENFILE, IMG$DECODE_IHD
                                                        .EXTRN  IMG$DO_WORK_LIST
                                                        .EXTRN  MMG$CRETVA, CTL$AG_CMEDATA
                                                        .EXTRN  CTL$AL_STACK, CTL$A_DISPVEC
                                                        .EXTRN  CTL$GL_CTLBASVA
                                                        .EXTRN  CTL$GL_FIXUPLNK
                                                        .EXTRN  CTL$GL_IMGHDRBF
                                                        .EXTRN  CTL$GL_PHD, CTL$GL_RMSBASE
                                                        .EXTRN  CTL$GL_VOLUMES, CTL$GQ_PROCPRIV
                                                        .EXTRN  IAC$AW_VECSET, IAC$GL_ICBFL
                                                        .EXTRN  IAC$AL_IMGACTBUF
                                                        .EXTRN  IAC$GL_FIRST_ICB
                                                        .EXTRN  IAC$GL_IMAGCTX, IAC$GL_MAIN_ICB
                                                        .EXTRN  IAC$GL_WORK_LIST
                                                        .EXTRN  IAC$GL_IMAGE_LIST
                                                        .EXTRN  IAC$GL_STACK_SIZE
                                                        .EXTRN  PIO$GW_IIOIMPA, CTL$GL_ICPUTIM
                                                        .EXTRN  CTL$GL_IFAULTS, CTL$GL_IFAULTIO
                                                        .EXTRN  CTL$GL_IWSPEAK, CTL$GL_IPAGEFL
                                                        .EXTRN  CTL$GL_IDIOCNT, CTL$GL_IBIOCNT
                                                        .EXTRN  CTL$GL_IVOLUMES
                                                        .EXTRN  CTL$GQ_ISTART, EXE$GL_ACMFLAGS
                                                        .EXTRN  EXE$GL_FLAGS, EXE$GL_RNOWN_FILES
                                                        .EXTRN  EXE$GL_SYSID_LOCK
                                                        .EXTRN  EXE$GQ_KFE_LCKNAM
                                                        .EXTRN  EXE$GQ_SYSTIME, SGN$GW_IMGIOCNT
                                                        .EXTRN  EXE$C_SYSEFN, EXE$V_INIT
                                                        .EXTRN  SYS$K_VERSION, SYS$DASSGN

                                                        .PSECT  YF$$SYSIMGACT,2

                                        0FFC 00000      .ENTRY  EXE$$IMGACT, Save R2,R3,R4,R5,R6,R7,R8,R9,-      ; 0601
                                                                R10,R11
                                5B 00000000G    00  9E 00002   MOVAB   IAC$GL_IMAGCTX, R11                      ;
                                5A 00000000G    00  9E 00009   MOVAB   OWN_STORAGE+68, R10                     ;
                                5E      B8      AE  9E 00010   MOVAB   -72(SP), SP                             ;
0068    8F      00              6E              00  2C 00014   MOVC5   #0, (SP), #0, #104, OWN_STORAGE         ; 0641
                                        BC      AA      0001B                                                   ;
                        CC      AA              01  D0 0001D   MOVL    #1, OWN_STORAGE+16                      ; 0642
                        D8      AA              02  D0 00021   MOVL    #2, OWN_STORAGE+28                      ; 0643
                                6A      10      AC  D0 00025   MOVL    CONTROL_FLAGS, OWN_STORAGE+68           ; 0650
                06      02      AA              05  E1 00029   BBC     #5, FLAGS+2, 1$                         ; 0652
                        0000V   CF              00  FB 0002E   CALLS   #0, SET_VECTORS                         ; 0653
                                                04  00033   RET                                                 ;
                                        E0      AA  DC 00034 1$:MOVPSL  OWN_STORAGE+36                         ; 0657
                                                50  D4 00037   CLRL    I                                       ; 0661
                51              50              08  78 00039 2$:ASHL    #8, I, R1                               ; 0665
                                    00000000G0041   9F 0003D   PUSHAB  CTL$A_DISPVEC[R1]                       ; 0667
                                9E 00000000G0040    3C 00044   MOVZWL  IAC$AW_VECSET[I], @(SP)+               ;
                                                50  D6 0004C   INCL    I                                       ; 0661
```

```
                       03             50  D1 0004E         CMPL    I, #3                                    :
                                      E6  1B 00051         BLEQU   2$                                       :
            0000V      CF             6C  FA 00053         CALLG   (AP), CHECK_PARAMS                       : 0674
                       59             50  D0 00058         MOVL    R0, STATUS                               :
                       25             59  E9 0005B         BLBC    STATUS, 5$                               : 0675
                         00000000G    00  D4 0005E         CLRL    CTL$GL_FIXUPLNK                          : 0681
      0E               6A             04  E0 00064         BBS     #4, FLAGS, 3$                            : 0682
                         00000000G    00  16 00068         JSB     RM$RESET                                 : 0685
                                      6B  D4 0006E         CLRL    IAC$GL_IMAGCTX                           : 0686
              BC       AA             01  88 00070         BISB2   #1, OWN_STORAGE                          : 0687
                                      03  11 00074         BRB     4$                                       : 0682
                               02     AB  B4 00076 3$:     CLRW    CONTEXT+2                                : 0694
                                      5E  DD 00079 4$:     PUSHL   SP                                       : 0701
            0000V      CF             01  FB 0007B         CALLS   #1, IMG$ALLOCATE_ICB                     :
                       59             50  D0 00080         MOVL    R0, STATUS                               :
                       03             59  E8 00083 5$:     BLBS    STATUS, 6$                               : 0703
                                    0156  31 00086         BRW     21$                                      :
      5A               6A             04  E1 00089 6$:     BBC     #4, FLAGS, 9$                            : 0707
      58               6E             14  C1 0008D         ADDL3   #20, ICB_ADR, R8                         : 0713
                       57      0C     AA  D0 00091         MOVL    OWN_STORAGE+80, R7                       : 0714
                       56      FC     AA  D0 00095         MOVL    OWN_STORAGE+64, R6                       : 0715
                       68      EC     AA  90 00099         MOVB    INPUT_NAME, (R8)                         : 0720
                       50      F0     AA  D0 0009D         MOVL    INPUT_NAME+4, R0                         : 0723
  27  00               60      EC     AA  2C 000A1         MOVC5   INPUT_NAME, (R0), #0, #39, 1(R8)         : 0726
                               01     A8     000A7                                                      :
                       50             58  D0 000A9         MOVL    R8, R0                                   : 0727
                         00000000G    00  16 000AC         JSB     IMG$IS_IT_MAPPED                         :
                       59             50  D0 000B2         MOVL    R0, STATUS                               :
                       01             59  D1 000B5         CMPL    STATUS, #1                               : 0732
                                      2D  12 000B8         BNEQ    9$                                       :
                                      57  D5 000BA         TSTL    R7                                       : 0736
                                      04  13 000BC         BEQL    7$                                       :
                       67      48     A1  7D 000BE         MOVQ    72(MAPPED_ICB), (R7)                     : 0739
                                      56  D5 000C2 7$:     TSTL    R6                                       : 0743
                                      16  13 000C4         BEQL    8$                                       :
                       50      0C     A6  9E 000C6         MOVAB   12(R6), R0                               : 0747
                                      66  D4 000CA         CLRL    (R6)                                     : 0749
              04       A6             50  D0 000CC         MOVL    R0, 4(R6)                                : 0750
                               08     A6  D4 000D0         CLRL    8(R6)                                    : 0751
              08       A0      0E     A1  B0 000D3         MOVW    14(MAPPED_ICB), 8(R0)                    : 0752
              10       A0             6B  B0 000D8         MOVW    IAC$GL_IMAGCTX, 16(R0)                   : 0753
                                      6E  DD 000DC 8$:     PUSHL   ICB_ADR                                  : 0757
            0000V      CF             01  FB 000DE         CALLS   #1, IMG$DEALLOCATE_ICB                   :
                       50             01  D0 000E3         MOVL    #1, R0                                   : 0759
                                          04 000E6         RET                                              :
                       52             6E  D0 000E7 9$:     MOVL    ICB_ADR, R2                              : 0765
        00000000G      00             52  D0 000EA         MOVL    R2, IAC$GL_FIRST_ICB                     :
                       07      BC     AA  E9 000F1         BLBC    OWN_STORAGE, 10$                         : 0766
        00000000G      00             52  D0 000F5         MOVL    R2, IAC$GL_MAIN_ICB                      : 0767
              58       A2      04     AE  9E 000FC 10$:    MOVAB   IHD_CTX, 88(R2)                          : 0768
            0000V      CF             00  FB 00101         CALLS   #0, GET_LOCK                             : 0770
                       59             50  D0 00106         MOVL    R0, STATUS                               :
                       03             59  E8 00109         BLBS    STATUS, 11$                              : 0771
                                    00D0  31 0010C         BRW     21$                                      :
                                      52  DD 0010F 11$:    PUSHL   R2                                       : 0779
                             FEBD     CA  9F 00111         PUSHAB  RESULT_NAME                              : 0775
                             FDAD     CA  9F 00115         PUSHAB  PRIMARY_NAM                              :
```

```
                                FD5D   CA  9F 00119          PUSHAB  PRIMARY_FAB                                     ;
                                  F4   AA  9F 0011D          PUSHAB  OWN_STORAGE+56                                  ;
                                  EC   AA  9F 00120          PUSHAB  OWN_STORAGE+48                                  ; 0774
                    0000V  CF         06  FB 00123          CALLS   #6, IMG$OPEN_IMAGE                              ; 0775
                           59         50  D0 00128          MOVL    R0, STATUS                                      ;
                           40         59  E9 0012B          BLBC    STATUS, 13$                                     ; 0780
                      04   AE   F55D   CA  9E 0012E          MOVAB   INPUT_BUFFER, IHD_CTX                           ; 0793
                      08   AE   F75D   CA  9E 00134          MOVAB   PRIMARY_IHD, IHD_CTX+4                          ; 0794
                                      52  DD 0013A          PUSHL   R2                                              ; 0796
                    0000V  CF         01  FB 0013C          CALLS   #1, IMG$GET_HEADER                              ;
                           59         50  D0 00141          MOVL    R0, STATUS                                      ;
                           12         59  E8 00144          BLBS    STATUS, 12$                                     ; 0797
                           7E     0E  A2  3C 00147          MOVZWL  14(R2), -(SP)                                   ; 0800
               00000000G   00         01  FB 0014B          CALLS   #1, SYS$DASSGN                                  ;
                    0000V  CF         00  FB 00152          CALLS   #0, ERROR_CLEAN_UP                              ; 0801
                           09         59  E8 00157          BLBS    STATUS, 13$                                     ; 0802
                    FFFF   8F     12  AE  B1 00159 12$:      CMPW    IHD_CTX+14, #-1                                 ; 0811
                           04         16  13 0015F          BEQL    14$                                             ;
                                      52  DD 00161          PUSHL   R2                                              ; 0814
                    0000V  CF         01  FB 00163          CALLS   #1, GET_OTHER_IMAGE                             ;
                           59         50  D0 00168          MOVL    R0, STATUS                                      ;
                           09         59  E8 0016B          BLBS    STATUS, 14$                                     ; 0815
                                      52  DD 0016E 13$:      PUSHL   R2                                              ; 0818
                    0000V  CF         01  FB 00170          CALLS   #1, IMG$DEALLOCATE_ICB                          ;
                           63         11  00175          BRB     20$                                             ; 0819
                16         6A         05  E1 00177 14$:      BBC     #5, FLAGS, 15$                                  ; 0832
                           50 00000000G  00  D0 0017B          MOVL    CTL$GL_PHD, R0                                  ; 0837
                      48   A2     28    A0  D0 00182          MOVL    40(R0), 72(R2)                                  ; 0839
                      4C   A2 3FFFFFFF   8F  D0 00187          MOVL    #1073741823, 76(R2)                             ; 0840
                           05         11  0018F          BRB     16$                                             ; 0832
                      48   A2     04    AA  7D 00191 15$:      MOVQ    OWN_STORAGE+72, 72(R2)                          ; 0844
                      0C   A2     20    AA  90 00196 16$:      MOVB    OWN_STORAGE+100, 12(R2)                         ; 0848
                05         6A         04  E1 0019B          BBC     #4, FLAGS, 17$                                  ; 0850
                           50         02  D0 0019F          MOVL    #2, R0                                          ;
                           03         11  001A2          BRB     18$                                             ;
                           50         01  D0 001A4 17$:      MOVL    #1, R0                                          ;
                      0D   A2         50  90 001A7 18$:      MOVB    R0, 13(R2)                                      ;
                      40   A2     18    AA  7D 001AB          MOVQ    OWN_STORAGE+92, 64(R2)                          ; 0853
                           50 00000000G  00  9E 001B0          MOVAB   IAC$GL_WORK_LIST+4, R0                          ; 0859
                      00   B0         62  0E 001B7          INSQUE  (R2), @0(R0)                                    ;
               00000000G   00         00  FB 001BB          CALLS   #0, IMG$DO_WORK_LIST                            ; 0860
                           59         50  D0 001C2          MOVL    R0, STATUS                                      ;
                           0D         59  E9 001C5          BLBC    STATUS, 19$                                     ; 0861
                                      6E  DD 001C8          PUSHL   ICB_ADR                                         ; 0868
                    0000V  CF         01  FB 001CA          CALLS   #1, END_PROCESSING                              ;
                           59         50  D0 001CF          MOVL    R0, STATUS                                      ;
                           0E         59  E8 001D2          BLBS    STATUS, 22$                                     ; 0869
                    0000V  CF         00  FB 001D5 19$:      CALLS   #0, ERROR_CLEAN_UP                              ; 0871
                    0000V  CF         00  FB 001DA 20$:      CALLS   #0, RELEASE_LOCK                                ; 0872
                           50         59  D0 001DF 21$:      MOVL    STATUS, R0                                      ; 0873
                                          04  001E2          RET                                                     ;
                    0000V  CF         00  FB 001E3 22$:      CALLS   #0, RELEASE_LOCK                                ; 0876
                           50     CC  AA  D0 001E8          MOVL    OWN_STORAGE+16, R0                              ; 0877
                                          04  001EC          RET                                                     ; 0879

; Routine Size:  493 bytes,    Routine Base:  YF$$SYSIMGACT + 0000
```

```
;  652          0880  1 %SBTTL 'CHECK_PARAMS - Check Accessibility and Store Parameters'
;  653          0881  1
;  654          0882  1 ROUTINE CHECK_PARAMS (NAME,DEFAULT,BUFFER,FLAGS,INADR,RETADR,IDENT,ACMODE) =
;  655          0883  1
;  656          0884  1 !+
;  657          0885  1 ! Functional Description:
;  658          0886  1 !
;  659          0887  1 !     This routine probes each input parameter for read access and stores
;  660          0888  1 !     the value of the parameter in a safe place. Output parameters are probed
;  661          0889  1 !     for write access and their addresses are stored for later use.
;  662          0890  1 !
;  663          0891  1 ! Calling Sequence:
;  664          0892  1 !
;  665          0893  1 !     CHECK_PARAMS (NAME,DEFAULT,BUFFER,FLAGS,INADR,RETADR,IDENT,ACMODE)
;  666          0894  1 !
;  667          0895  1 !-
;  668          0896  1
;  669          0897  2 BEGIN
;  670          0898  2
;  671          0899  2 LOCAL
;  672          0900  2     SAFE_PLACE,
;  673          0901  2     STATUS;
;  674          0902  2
;  675          0903  2 ! Define some synonyms for the local storage called SAFE_PLACE
;  676          0904  2
;  677          0905  2 BIND
;  678          0906  2     LOCAL_NAME   = SAFE_PLACE : REF $BBLOCK,
;  679          0907  2     LOCAL_DFLT   = SAFE_PLACE : REF $BBLOCK,
;  680          0908  2     LOCAL_BUFFER = SAFE_PLACE,
;  681          0909  2     LOCAL_INADR  = INADR      : REF VECTOR,
;  682          0910  2     LOCAL_RETADR = RETADR     : REF VECTOR,
;  683          0911  2     LOCAL_IDENT  = SAFE_PLACE : REF VECTOR;
;  684          0912  2
;  685          0913  2 ! Define some more synonyms for the two file name descriptors
;  686          0914  2
;  687          0915  2 BIND
;  688          0916  2     NAM_DSC  = OWN_STORAGE [IMAGE_NAME_DESC] : $BBLOCK,
;  689          0917  2     DFLT_DSC = OWN_STORAGE [DFLT_NAME_DESC]  : $BBLOCK;
;  690          0918  2
;  691          0919  2 ! The image name is the only required parameter for the image activator. If an
;  692          0920  2 ! image name is not present, this routine returns with an IMG$_NONAME error.
;  693          0921  2
;  694          0922  2 LOCAL_NAME = .NAME;
;  695          0923  2 IF .LOCAL_NAME EQL 0
;  696          0924  2 THEN
;  697          0925  2     RETURN IMG$_NONAME
;  698          0926  2 ELSE
;  699          0927  3     BEGIN
;  700          0928  4     IF NOT (STATUS = EXE$PROBER_DSC (
;  701          0929  4         .LOCAL_NAME;
;  702          0930  4         NAM_DSC [DSC$W_LENGTH],
;  703          0931  4         NAM_DSC [DSC$A_POINTER]))
;  704          0932  3     THEN
;  705          0933  3         RETURN .STATUS
;  706          0934  2     END;
;  707          0935  2
;  708          0936  2 ! The default name descriptor is probed and stored in a similar fashion. No
```

```
;  709          0937  2 ! error occurs if the default name is missing.
;  710          0938  2
;  711          0939  2 LOCAL_DFLT = .DEFAULT;
;  712          0940  2 IF .LOCAL_DFLT EQL 0
;  713          0941  2 THEN
;  714          0942  3     BEGIN
;  715          0943  3     DFLT_DSC [DSC$W_LENGTH]  = 0;
;  716          0944  3     DFLT_DSC [DSC$A_POINTER] = 0;
;  717          0945  3     END
;  718          0946  2 ELSE
;  719          0947  3     BEGIN
;  720          0948  4     IF NOT (STATUS = EXE$PROBER_DSC (
;  721          0949  4         .LOCAL_DFLT;
;  722          0950  4         DFLT_DSC [DSC$W_LENGTH],
;  723          0951  4         DFLT_DSC [DSC$A_POINTER]))
;  724          0952  3     THEN
;  725          0953  3         RETURN .STATUS
;  726          0954  2     END;
;  727          0955  2
;  728          0956  2 ! The address of a 512-byte buffer is tucked away after the buffer is
;  729          0957  2 ! checked for write access.
;  730          0958  2 !
;  731          0959  2 ! NOTE WELL
;  732          0960  2 !
;  733          0961  2 ! Because the image activator issues calls to other memory management system
;  734          0962  2 ! services, the protection on this buffer may change. This buffer must be probed
;  735          0963  2 ! again by the completion code before anything is written to the buffer.
;  736          0964  2
;  737          0965  2 LOCAL_BUFFER = .BUFFER;
;  738          0966  3 IF (.LOCAL_BUFFER NEQ 0)
;  739          0967  2     AND
;  740          0968  3     (NOT PROBEW (%REF(0), %REF(RETURN_BUFFER_SIZE), .LOCAL_BUFFER))
;  741          0969  2 THEN RETURN SS$_ACCVIO
;  742          0970  2 ELSE OWN_STORAGE [BUFFER_ADDRESS] = .LOCAL_BUFFER;
;  743          0971  2
;  744          0972  2 !!! NEED TO PROPOGATE SOME OF THE FLAGS INFORMATION INTO OTHER OWN_STORAGE
;  745          0973  2 !!! CELLS. ALSO NEED TO DO CONSISTENCY CHECKS BETWEEN FLAGS AND OTHER
;  746          0974  2 !!! INPUT PARAMETERS.
;  747          0975  2
;  748          0976  2 ! The input address range array specifies the address range into which the
;  749          0977  2 ! image is to be mapped. It must be readable by the caller.
;  750          0978  2
;  751          0979  2 LOCAL_INADR = .INADR;
;  752          0980  3 IF (.LOCAL_INADR NEQ 0) AND (NOT PROBER (%REF(0), %REF(8), .LOCAL_INADR))
;  753          0981  2 THEN RETURN SS$_ACCVIO;
;  754          0982  2 IF .LOCAL_INADR NEQ 0
;  755          0983  2 THEN
;  756          0984  3     BEGIN
;  757          0985  3
;  758          0986  3     BIND
;  759          0987  3         FLAG_BITS = OWN_STORAGE [INPUT_FLAGS] : $BBLOCK,
;  760          0988  3         ADDRESS_RANGE = OWN_STORAGE [INPUT_START_ADDRESS] : $BBLOCK;
;  761          0989  3
;  762          0990  3     OWN_STORAGE [INPUT_START_ADDRESS] = .LOCAL_INADR [0];
;  763          0991  3     OWN_STORAGE [INPUT_END_ADDRESS] = .LOCAL_INADR [1];
;  764          0992  3
;  765          0993  3     ! We need to remember that this is the P0 part of a P1 merge operation
```

```
;  766          0994  3     ! because certain steps taken in a "real" activation must not take place.
;  767          0995  3
;  768          0996  4     IF (
;  769          0997  5         (.FLAG_BITS [IAC$V_P1MERGE])
;  770          0998  4         AND
;  771          0999  5         (NOT (.ADDRESS_RANGE [VA$V_P1]))
;  772          1000  4        )
;  773          1001  3     THEN
;  774          1002  3         OWN_STORAGE [P1_MERGE_PO] = TRUE;
;  775          1003  3
;  776          1004  2     END;
;  777          1005  2
;  778          1006  2 ! The caller can specify the address of an array that will receive the
;  779          1007  2 ! address range into which a collection of images was mapped. This array
;  780          1008  2 ! is probed for write access and set to contain FFFFFFFF in both elements,
;  781          1009  2 ! indicating that no mapping has yet occurred. Note that this array, like
;  782          1010  2 ! the output buffer must be probed again by the completion code before the
;  783          1011  2 ! actual return address arra is written to make sure that the accessibility
;  784          1012  2 ! of the array has not changed as a side effect of a system service call
;  785          1013  2 ! issued by the image activator.
;  786          1014  2
;  787          1015  2 LOCAL_RETADR = .RETADR;
;  788          1016  3 IF (.LOCAL_RETADR NEQ 0) AND (NOT PROBEW (%REF(0), %REF(8), .LOCAL_RETADR))
;  789          1017  2 THEN RETURN SS$_ACCVIO;
;  790          1018  2 OWN_STORAGE [RETURN_ARRAY_ADDRESS] = .LOCAL_RETADR;
;  791          1019  2 OWN_STORAGE [RETURN_START_ADDRESS] = -1;
;  792          1020  2 OWN_STORAGE [RETURN_END_ADDRESS] = -1;
;  793          1021  2 IF .LOCAL_RETADR NEQ 0
;  794          1022  2 THEN
;  795          1023  3     BEGIN
;  796          1024  3     LOCAL_RETADR [0] = -1;
;  797          1025  3     LOCAL_RETADR [1] = -1;
;  798          1026  2     END;
;  799          1027  2
;  800          1028  2 ! The caller can specify explicit match control information that will be
;  801          1029  2 ! compared to the version number contained in the image that is actually
;  802          1030  2 ! activated.
;  803          1031  2
;  804          1032  2 LOCAL_IDENT = .IDENT;
;  805          1033  3 IF (.LOCAL_IDENT NEQ 0) AND (NOT PROBER (%REF(0), %REF(8), .LOCAL_IDENT))
;  806          1034  2 THEN RETURN SS$_ACCVIO;
;  807          1035  2 IF .LOCAL_IDENT NEQ 0
;  808          1036  2 THEN
;  809          1037  3     BEGIN
;  810          1038  3     OWN_STORAGE [MATCH_CONTROL] = .LOCAL_IDENT [0];
;  811          1039  3     OWN_STORAGE [VERSION] = .LOCAL_IDENT [1];
;  812          1040  2     END;
;  813          1041  2
;  814          1042  2 ! The final input parameter is the access mode that will be used for two
;  815          1043  2 ! purposes. The channel on which the image file is opened will have this
;  816          1044  2 ! access mode associated with it. The pages that are mapped will be owned
;  817          1045  2 ! by the mode specified by this parameter.
;  818          1046  2 !
;  819          1047  2 ! Because an access mode of kernel is meaningless, a missing ACCESS_MODE
;  820          1048  2 ! parameter is interpreted as USER mode. If a nonzero value is present in the
;  821          1049  2 ! argument list, it is first maximized with caller's mode and then stored in
;  822          1050  2 ! a safe place.
```

```
;  823          1051  2
;  824          1052  2 IF .ACMODE EQL 0
;  825          1053  2 THEN
;  826          1054  2     OWN_STORAGE [ACCESS_MODE] = PSL$C_USER
;  827          1055  2 ELSE
;  828          1056  2     EXE$MAXACMODE (.ACMODE; OWN_STORAGE [ACCESS_MODE]);
;  829          1057  2
;  830          1058  2 RETURN SS$_NORMAL
;  831          1059  2
;  832          1060  1 END;                                    ! End of routine CHECK_PARAMS


                                        007C 00000 CHECK_PARAMS:
                                                    .WORD   Save R2,R3,R4,R5,R6                     ; 0882
                    56 00000000G   00    9E 00002           MOVAB   EXE$PROBER_DSC, R6              ;
                    55 00000000G   00    9E 00009           MOVAB   DFLT_DSC, R5                    ;
                    54         04  AC    D0 00010           MOVL    NAME, LOCAL_NAME                ; 0922
                                   08    12 00014           BNEQ    1$                              ; 0923
                    50 084D8CCC    8F    D0 00016           MOVL    #139300044, R0                  ; 0925
                                         04 0001D           RET                                     ;
                    51                   54 D0 0001E 1$:    MOVL    LOCAL_NAME, R1                  ; 0930
                                         66 16 00021        JSB     EXE$PROBER_DSC                  ;
                F8  A5                   51 B0 00023        MOVW    R1, NAM_DSC                     ;
                FC  A5                   52 D0 00027        MOVL    R2, NAM_DSC+4                   ; 0931
                    19                   50 E9 0002B        BLBC    STATUS, 3$                      ; 0930
                    54         08    AC  D0 0002E           MOVL    DEFAULT, LOCAL_DFLT             ; 0939
                                   07    12 00032           BNEQ    2$                              ; 0940
                                         65 B4 00034        CLRW    DFLT_DSC                        ; 0943
                                   04    A5 D4 00036        CLRL    DFLT_DSC+4                      ; 0944
                                         10 11 00039        BRB     4$                              ; 0940
                    51                   54 D0 0003B 2$:    MOVL    LOCAL_DFLT, R1                  ; 0950
                                         66 16 0003E        JSB     EXE$PROBER_DSC                  ;
                    65                   51 B0 00040        MOVW    R1, DFLT_DSC                    ;
                 04 A5                   52 D0 00043        MOVL    R2, DFLT_DSC+4                  ; 0951
                    01                   50 E8 00047 3$:    BLBS    STATUS, 4$                      ; 0950
                                         04 0004A           RET                                     ;
                    54         0C    AC  D0 0004B 4$:       MOVL    BUFFER, LOCAL_BUFFER            ; 0965
                                   08    13 0004F           BEQL    5$                              ; 0966
            64  0200               8F    00 0D 00051        PROBEW  #0, #512, (LOCAL_BUFFER)         ; 0968
                                         65 13 00057        BEQL    10$                             ;
                 08 A5                   54 D0 00059 5$:    MOVL    LOCAL_BUFFER, OWN_STORAGE+64    ; 0970
                    50         14    AC  D0 0005D           MOVL    LOCAL_INADR, R0                 ; 0980
                                         51 D4 00061        CLRL    R1                              ;
                                         50 D5 00063        TSTL    R0                              ;
                                         08 13 00065        BEQL    6$                              ;
                                         51 D6 00067        INCL    R1                              ;
            60                   08      00 0C 00069        PROBER  #0, #8, (R0)                    ;
                                         4F 13 0006D        BEQL    10$                             ;
                    12                   51 E9 0006F 6$:    BLBC    R1, 7$                          ; 0982
                 10 A5                   60 7D 00072        MOVQ    (R0), OWN_STORAGE+72            ; 0990
            09   0C A5                   06 E1 00076        BBC     #6, FLAG_BITS, 7$               ; 0997
            04   13 A5                   06 E0 0007B        BBS     #6, ADDRESS_RANGE+3, 7$         ; 0999
                 C8 A5                   20 88 00080        BISB2   #32, OWN_STORAGE                ; 1002
                    50         18    AC  D0 00084 7$:       MOVL    LOCAL_RETADR, R0                ; 1016
```

```
                                    51  D4 00088          CLRL    R1                                   ;
                                    50  D5 0008A          TSTL    R0                                   ;
                                    08  13 0008C          BEQL    8$                                   ;
                                    51  D6 0008E          INCL    R1                                   ;
             60          08         00  0D 00090          PROBEW  #0, #8, (R0)                         ;
                                    28  13 00094          BEQL    10$                                  ;
                    18   A5         50  D0 00096 8$:      MOVL    R0, OWN_STORAGE+80                   ; 1018
                    1C   A5         01  CE 0009A          MNEGL   #1, OWN_STORAGE+84                   ; 1019
                    20   A5         01  CE 0009E          MNEGL   #1, OWN_STORAGE+88                   ; 1020
                         07         51  E9 000A2          BLBC    R1, 9$                               ; 1021
                         60         01  CE 000A5          MNEGL   #1, (R0)                             ; 1024
                    04   A0         01  CE 000A8          MNEGL   #1, 4(R0)                            ; 1025
                         54    1C   AC  D0 000AC 9$:      MOVL    IDENT, LOCAL_IDENT                   ; 1032
                                    50  D4 000B0          CLRL    R0                                   ; 1033
                                    54  D5 000B2          TSTL    LOCAL_IDENT                          ;
                                    0C  13 000B4          BEQL    11$                                  ;
                                    50  D6 000B6          INCL    R0                                   ;
             64          08         00  0C 000B8          PROBER  #0, #8, (LOCAL_IDENT)                ;
                                    04  12 000BC          BNEQ    11$                                  ;
                         50         0C  D0 000BE 10$:     MOVL    #12, R0                              ; 1034
                                        04 000C1          RET                                          ;
                         04         50  E9 000C2 11$:     BLBC    R0, 12$                              ; 1035
                    24   A5         64  7D 000C5          MOVQ    (LOCAL_IDENT), OWN_STORAGE+92        ; 1038
                               20   AC  D5 000C9 12$:     TSTL    ACMODE                               ; 1052
                                    06  12 000CC          BNEQ    13$                                  ;
                    2C   A5         03  D0 000CE          MOVL    #3, OWN_STORAGE+100                  ; 1054
                                    0E  11 000D2          BRB     14$                                  ;
                         50    20   AC  D0 000D4 13$:     MOVL    ACMODE, R0                           ; 1056
                          00000000G 00  16 000D8          JSB     EXE$MAXACMODE                        ;
                    2C   A5         50  D0 000DE          MOVL    R0, OWN_STORAGE+100                  ;
                         50         01  D0 000E2 14$:     MOVL    #1, R0                               ; 1058
                                        04 000E5          RET                                          ; 1060

; Routine Size:  230 bytes,    Routine Base:  YF$$SYSIMGACT + 01ED
```

```
;  834     1061  1 %SBTTL 'IMG$OPEN_IMAGE - Open Next Image File'
;  835     1062  1
;  836     1063  1 GLOBAL ROUTINE IMG$OPEN_IMAGE (NAME_DESC , DFLT_DESC ,
;  837     1064  1                               FAB_ADDRESS , NAM_BLOCK_ADDRESS ,
;  838     1065  1                               NAME_STRING , ICB_POINTER) =
;  839     1066  1
;  840     1067  1 !+
;  841     1068  1 ! Functional Description:
;  842     1069  1 !
;  843     1070  1 !    This routine opens an image file as a process permanent file for
;  844     1071  1 !    subsequent use by one of the memory management system services.
;  845     1072  1 !
;  846     1073  1 ! Calling Sequence:
;  847     1074  1 !
;  848     1075  1 !    IMG$OPEN_IMAGE ()
;  849     1076  1 !
;  850     1077  1 ! Input Parameters:
;  851     1078  1 !
;  852     1079  1 !    NAME_DESC - Address of string descriptor for image file name
;  853     1080  1 !
;  854     1081  1 !    DFLT_DESC - Address of default name descriptor
;  855     1082  1 !-
;  856     1083  1
;  857     1084  2 BEGIN
;  858     1085  2
;  859     1086  2 MAP
;  860     1087  2     NAME_STRING : REF VECTOR;
;  861     1088  2
;  862     1089  2 BIND
;  863     1090  2     FAB         = .FAB_ADDRESS           : $BBLOCK,
;  864     1091  2     NAME_BLOCK  = .NAM_BLOCK_ADDRESS     : $BBLOCK,
;  865     1092  2     ICB         = .ICB_POINTER           : $BBLOCK,
;  866     1093  2     DEV_CHAR    = FAB [FAB$L_DEV]        : $BBLOCK,
;  867     1094  2     ICB_NAME    = ICB [ICB$T_IMAGE_NAME] : VECTOR [, BYTE];
;  868     1095  2
;  869     1096  2 ! Create synonyms for the CTX and STV fields in the FAB
;  870     1097  2
;  871     1098  2 BIND
;  872     1099  2     CHAN        = FAB [FAB$L_STV],
;  873     1100  2     KFE         = FAB [FAB$L_CTX]        : REF $BBLOCK;
;  874     1101  2
;  875     1102  2 MAP
;  876     1103  2     NAME_DESC                    : REF $BBLOCK,
;  877     1104  2     DFLT_DESC                    : REF $BBLOCK;
;  878     1105  2
;  879     1106  2 LOCAL
;  880     1107  2     STATUS;
;  881     1108  2
;  882     1109  2 ! Once the system is fully initialized, RMS and the file system can be used
;  883     1110  2 ! to open the various image files. Until that time, the bootstrap file
;  884     1111  2 ! routines must be used. At least two images, SYSINIT.EXE and F11BXQP.EXE,
;  885     1112  2 ! are activated along this alternate code path.
;  886     1113  2
;  887     1114  2 IF .EXE$GL_FLAGS [EXE_V_INIT]
;  888     1115  2 THEN
;  889     1116  3     BEGIN
;  890     1117  3
```

```
;  891        P 1118  3     $FAB_INIT (
;  892        P 1119  3         FAB = FAB,
;  893        P 1120  3         FNS = .NAME_DESC [DSC$W_LENGTH],
;  894        P 1121  3         FNA = .NAME_DESC [DSC$A_POINTER],
;  895        P 1122  3         DNS = .DFLT_DESC [DSC$W_LENGTH],
;  896        P 1123  3         DNA = .DFLT_DESC [DSC$A_POINTER],
;  897        P 1124  3         NAM = NAME_BLOCK,
;  898        P 1125  3         FOP = (
;  899        P 1126  3             KFO ,                   ! Use the known file data base
;  900        P 1127  3             PPF ,                   ! Process permanent file
;  901        P 1128  3             SQO ,                   ! Network optimization
;  902        P 1129  3             UFO ),                  ! User file open
;  903        P 1130  3         CTX = 0,                    ! KFE address will be returned here
;  904        P 1131  3         RTV = -1                    ! Insure that WCB completely maps file
;  905           1132  3         );
;  906           1133  3
;  907           1134  3     FAB [FAB$V_CHAN_MODE] = .OWN_STORAGE [ACCESS_MODE];
;  908           1135  3
;  909           1136  3     ! The image files associated with writable global sections are opened
;  910           1137  3     ! for write access. All other image files are opened for execute access.
;  911           1138  3
;  912           1139  3     IF .ICB [ICB$V_OPEN_FOR_WRITE]
;  913           1140  3     THEN
;  914           1141  4         BEGIN
;  915           1142  4         FAB [FAB$B_FAC] = FAB$M_PUT;
;  916           1143  4         FAB [FAB$B_SHR] = FAB$M_SHRGET OR FAB$M_SHRPUT OR FAB$M_UPI;
;  917           1144  4         END
;  918           1145  3     ELSE
;  919           1146  4         BEGIN
;  920           1147  4         FAB [FAB$B_FAC] = FAB$M_EXE;
;  921           1148  4         FAB [FAB$B_SHR] = 0;
;  922           1149  3         END;
;  923           1150  3
;  924           1151  3     ! If we are currently running an executable image installed with privilege,
;  925           1152  3     ! we must direct RMS to only use the logical name tables that cannot be
;  926           1153  3     ! redefined by user mode code.                                                ; Re
;  927           1154  3
;  928           1155  3     FAB [FAB$V_LNM_MODE] =
;  929           1156  4         (IF .IAC$GL_IMAGCTX [IMAGCTX$V_PRIV]
;  930           1157  4         THEN PSL$C_EXEC
;  931           1158  3         ELSE PSL$C_USER);
;  932           1159  3
;  933           1160  3     ! Load NAM block with descriptor for resultant (or merely expanded) name
;  934           1161  3     ! string. (If a known file entry is found, then RMS does not return an
;  935           1162  3     ! resultant string but merely the expanded string.)
;  936           1163  3
;  937        P 1164  3     $NAM_INIT
;  938        P 1165  3         (NAM = NAME_BLOCK,
;  939        P 1166  3         RSS = NAM$C_MAXRSS,
;  940        P 1167  3         RSA = .NAME_STRING,
;  941        P 1168  3         ESS = NAM$C_MAXRSS,
;  942        P 1169  3         ESA = .NAME_BLOCK [NAM$L_RSA],
;  943        P 1170  3         NOP = <NOCONCEAL>
;  944           1171  3         );
;  945           1172  3
;  946           1173  3     STATUS = $OPEN (FAB = FAB);
;  947           1174  3
```

```
;  948          1175  3      !!! THE FOLLOWING HACK ALLOWS INFORMATION IN THE FAB TO BE PASSED   !!!!!!
;  949          1176  3      !!! BACK TO THE CALLER IN THE EVENT THAT THE ACTIVATION FAILS        !!!!!!
;  950          1177  3      !!!                                                                  !!!!!!
;  951          1178  3      if .own_storage [buffer_address] nequ 0                              !!!!!!
;  952          1179  3      then                                                                 !!!!!!
;  953          1180  4          begin                                                            !!!!!!
;  954          1181  4                                                                           !!!!!!
;  955          1182  4          bind                                                             !!!!!!
;  956          1183  4              bufhdr = .own_storage [buffer_address]      : vector [3,long]; !!!!!!
;  957          1184  4                                                                           !!!!!!
;  958          1185  4          if probew (%ref(0), %ref(4), bufhdr [2])                         !!!!!!
;  959          1186  4          then                                                             !!!!!!
;  960          1187  4              bufhdr [2] = fab;                                            !!!!!!
;  961          1188  4                                                                           !!!!!!
;  962          1189  4          end;                                                             !!!!!!
;  963          1190  3      !!!                                                                  !!!!!!
;  964          1191  3      !!! THIS IS THE END OF THE ERROR REPORTING HACK                       !!!!!!
;  965          1192  3
;  966          1193  3      IF NOT .STATUS THEN RETURN .STATUS; ! ??? WHAT ELSE ???
;  967          1194  3      END
;  968          1195  2  ELSE
;  969          1196  3      BEGIN
;  970          1197  3
;  971          1198  3      ! This is the code path that is used to open image files before the file
;  972          1199  3      ! system and RMS exist.
;  973          1200  3
;  974          1201  3      LOCAL
;  975          1202  3          RTVRBUF   : VECTOR [512, BYTE],  ! Retrieval pointer buffer
;  976          1203  3          IXFHDR    : VECTOR [512, BYTE],  ! Index file header buffer
;  977          1204  3          FILHDR    : VECTOR [512, BYTE],  ! File header buffer
;  978          1205  3          STATBLK   : VECTOR [2],          ! Statistics block
;  979          1206  3          RTVRLEN,                         ! Returned length of retrieval buffer
;  980          1207  3          BUF_DESC  : VECTOR [2]           ! Retrieval pointer buffer descriptor
;  981          1208  3                      INITIAL (512, RTVRBUF);
;  982          1209  3
;  983          1210  3      LOCAL
;  984          1211  3          ARG_LIST : VECTOR [3]
;  985          1212  3                     INITIAL (2, 0, BUF_DESC);
;  986          1213  3
;  987          1214  3      STATUS = FIL$OPENFILE (
;  988          1215  3          FAB [FAB$L_STV],             ! Store channel here
;  989          1216  3          .NAME_DESC,                  ! Address of name descriptor
;  990          1217  3          IXFHDR,                      ! Address of index file header buffer
;  991          1218  3          FILHDR,                      ! Address of file header buffer
;  992          1219  3          STATBLK,                     ! Address of statistics array
;  993          1220  3          RTVRLEN,                     ! Returned buffer size
;  994          1221  3          BUF_DESC);                   ! Retrieval buffer descriptor
;  995          1222  3      IF NOT .STATUS
;  996          1223  3      THEN RETURN .STATUS;             ! Exit if error detected
;  997          1224  3
;  998          1225  3      IF .RTVRLEN GTR .BUF_DESC [0]    ! If the mapping pointers can't fit on
;  999          1226  3      THEN RETURN SS$_BADIMGHDR;       !  a single page, then return an error
; 1000          1227  3
; 1001          1228  3      IF .RTVRLEN EQL 0                ! If the file is empty,
; 1002          1229  3      THEN RETURN SS$_BADIMGHDR        !  then return an error
; 1003          1230  3      ELSE BUF_DESC [0] = .RTVRLEN;    ! Otherwise, use correct buffer size
; 1004          1231  3
```

```
;  1005          1232  3      ARG_LIST [1] = .CHAN;                   ! Store channel number in argument list
;  1006        P 1233  3      STATUS = $CMKRNL (                      ! Allocate and initialize a window
;  1007        P 1234  3          ROUTIN = INIT_WINDOW,               !  control block in kernel mode
;  1008          1235  3          ARGLST = ARG_LIST);
;  1009          1236  3      IF NOT .STATUS
;  1010          1237  3      THEN RETURN .STATUS;                    ! Exit if error detected
;  1011          1238  3
;  1012          1239  3      ! The image file name string passed as input will be stored internally and
;  1013          1240  3      ! passed back to the caller if so requested. The entire file specification
;  1014          1241  3      ! (and not merely the file name portion) will also be stored in the ICB.
;  1015          1242  3
;  1016          1243  3      NAME_BLOCK [NAM$B_RSL] = .NAME_DESC [DSC$W_LENGTH];
;  1017          1244  3      CH$MOVE (
;  1018          1245  3          .NAME_BLOCK [NAM$B_RSL],            ! Store the input file specification
;  1019          1246  3          .NAME_DESC [DSC$A_POINTER],         !  in the name string buffer
;  1020          1247  3          .NAME_STRING);
;  1021          1248  3
;  1022          1249  3      ! Rather than storing the name in the ICB, this code segment will merely
;  1023          1250  3      ! set up the NAM block fields so that the code below that loads the image
;  1024          1251  3      ! name into the ICB will work correctly. Note that ICB$S_IMAGE_NAME includes
;  1025          1252  3      ! one byte of count and four bytes of _00n suffix. These must be subtracted
;  1026          1253  3      ! before the minimization takes place.
;  1027          1254  3
;  1028          1255  3      NAME_BLOCK [NAM$B_NAME] = MINU (        ! Only use the beginning of the string
;  1029          1256  3          .NAME_DESC [DSC$W_LENGTH],          !  if the input file specification is
;  1030          1257  3          ICB$S_IMAGE_NAME - (1+4));          !  longer than 39 characters
;  1031          1258  3      NAME_BLOCK [NAM$L_NAME] = .NAME_STRING;
;  1032          1259  3
;  1033          1260  2      END;
;  1034          1261  2
;  1035          1262  2  ICB [ICB$W_CHAN] = .CHAN;                   ! Store channel number in ICB
;  1036          1263  2  ICB [ICB$L_KFE] = .FAB [FAB$L_CTX];        ! Pick up address of KFE
;  1037          1264  2
;  1038          1265  2  ! If ICB does not yet contain an image name, load the name contained in the
;  1039          1266  2  ! NAM block. The name stored in the ICB will be the filename portion of the
;  1040          1267  2  ! full file spec, the same name that the linker uses when constructing the
;  1041          1268  2  ! names of global ISDs.
;  1042          1269  2
;  1043          1270  2  IF .ICB_NAME [0] EQL 0
;  1044          1271  2  THEN
;  1045          1272  3      BEGIN
;  1046          1273  3      ICB_NAME [0] = .NAME_BLOCK [NAM$B_NAME];
;  1047          1274  3      CH$MOVE (
;  1048          1275  3          .ICB_NAME [0],
;  1049          1276  3          .NAME_BLOCK [NAM$L_NAME],
;  1050          1277  3          ICB_NAME [1]);
;  1051          1278  2      END;
;  1052          1279  2
;  1053          1280  2  ! If activating an image across the network or from magtape, the file is
;  1054          1281  2  ! not mapped. Rather, the address space is created and the contents of the
;  1055          1282  2  ! image file are read directly into the newly created pages.
;  1056          1283  2
;  1057          1284  3  IF (.DEV_CHAR [DEV$V_NET] OR
;  1058          1285  3      (.DEV_CHAR [DEV$V_SQD] AND (.FAB [FAB$W_BLS] EQL 512)))
;  1059          1286  2  THEN ICB [ICB$V_LOAD_IMAGE] = TRUE;
;  1060          1287  2
;  1061          1288  2  IF .ICB [ICB$L_KFE] EQL 0
```

```
;  1062          1289  2 THEN
;  1063          1290  3     BEGIN
;  1064          1291  3     IF (.IAC$GL_IMAGCTX [IMAGCTX$V_PRIV]) THEN RETURN SS$_PRIVINSTALL
;  1065          1292  3     END
;  1066          1293  2 ELSE
;  1067          1294  3     BEGIN
;  1068          1295  3
;  1069          1296  3     ! Make sure that the count byte immediately precedes the name string
;  1070          1297  3
;  1071        P 1298  3     $ASSUME (($BYTEOFFSET (KFE$B_FILNAMLEN))
;  1072        P 1299  3              EQLU
;  1073          1300  3              ($BYTEOFFSET (KFE$T_FILNAM) - 1));
;  1074          1301  3
;  1075          1302  3     BIND
;  1076          1303  3         KFE_NAME = KFE [KFE$T_FILNAM] - 1 : VECTOR [,BYTE],     ! Include count byte
;  1077          1304  3         CTX = .ICB [ICB$L_CONTEXT]        : $BBLOCK,
;  1078          1305  3         GSD_NAME = CTX [CTX_T_GSD_NAME]   : VECTOR [,BYTE];
;  1079          1306  3
;  1080          1307  3     ! Several bits of housekeeping are in order if this was a successful
;  1081          1308  3     ! $OPEN of a known file.
;  1082          1309  3
;  1083          1310  3     IF .KFE [KFE$V_PROCPRIV] AND .OWN_STORAGE [MAIN_PROGRAM]
;  1084          1311  3         THEN IAC$GL_IMAGCTX [IMAGCTX$V_PRIV] = TRUE;
;  1085          1312  3
;  1086          1313  3     ! Images installed with the /ACCOUNT qualifier can cause image accounting
;  1087          1314  3     ! records to be written each time they are run as main programs. The
;  1088          1315  3     ! ACCOUNT flag is ignored for shareable images and images that are merged
;  1089          1316  3     ! into an existing image's address space.
;  1090          1317  3
;  1091          1318  3     IF .KFE [KFE$V_ACCOUNT] AND .OWN_STORAGE [MAIN_PROGRAM]
;  1092          1319  3         THEN OWN_STORAGE [IMAGE_ACCOUNT] = TRUE;
;  1093          1320  3
;  1094          1321  3     ! RMS returned an expanded string. We will store the size of the image
;  1095          1322  3     ! file in the RSL field in the NAM block. We will also strip off the
;  1096          1323  3     ! trailing semicolon (or dot) that would inhibit further known file
;  1097          1324  3     ! lookups if the file name returned by the image activator were to be
;  1098          1325  3     ! passed back into the image activator at a later time.
;  1099          1326  3
;  1100          1327  3     IF .NAME_BLOCK [NAM$B_VER] EQL 1
;  1101          1328  3     THEN NAME_BLOCK [NAM$B_RSL] = .NAME_BLOCK [NAM$B_ESL] - 1;
;  1102          1329  3
;  1103          1330  3     IF .KFE [KFE$V_SHARED]
;  1104          1331  3     THEN
;  1105          1332  4         BEGIN
;  1106          1333  4         CH$MOVE (
;  1107          1334  4             (.KFE [KFE$B_FILNAMLEN] + 1),
;  1108          1335  4             KFE_NAME [0],
;  1109          1336  4             GSD_NAME [0]);
;  1110          1337  4         CH$MOVE (
;  1111          1338  4             4,
;  1112          1339  4             CH$PTR (UPLIT (' 000')),
;  1113          1340  4             GSD_NAME [.KFE [KFE$B_FILNAMLEN] + 1]);
;  1114          1341  4         ICB [ICB$V_SHAREABLE] = TRUE;
;  1115          1342  3         END;
;  1116          1343  3
;  1117          1344  3     !!! SOME MORE KFE FLAGS SHOULD BE COPIED INTO THE ICB
;  1118          1345  3
```

```
; 1119          1346  3      !!! THE CURRENT CODE WORRIES ABOUT THIS "SHMIDENT" STUFF
; 1120          1347  3
; 1121          1348  2      END;
; 1122          1349  2
; 1123          1350  2 RETURN SS$_NORMAL;
; 1124          1351  2
; 1125          1352  1 END;                                         ! End of routine IMG$OPEN_IMAGE


                                                         002D3          .BLKB    1
                              00000000  00000002        002D4 P.AAA:    .LONG    2, 0                                        ;
                                        00000000        002DC           .LONG    0                                           ;
                                   30   30   30   5F    002E0 P.AAB:    .ASCII   \_000\                                      ;

                                                                        .EXTRN   SYS$OPEN, SYS$CMKRNL

                                               0FFC     00000           .ENTRY   IMG$OPEN_IMAGE, Save R2,R3,R4,R5,R6,R7,R8,- ; 1063
                                                                                 R9,R10,R11                                  ;
                           5B 0000000G       00    9E   00002           MOVAB    OWN_STORAGE, R11                            ;
                           5E     F9E0       CE    9E   00009           MOVAB    -1568(SP), SP                               ;
                           56       0C       AC    7D   0000E           MOVQ     FAB_ADDRESS, R6                             ; 1090
                           59       18       AC    D0   00012           MOVL     ICB_POINTER, R9                             ; 1092
                           5A       14       A9    9E   00016           MOVAB    20(R9), R10                                 ; 1094
                           58       04       AC    D0   0001A           MOVL     NAME_DESC, R8                               ; 1132
                03 0000000G        0C       00G    E0   0001E           BBS      S^EXE$V_INIT, EXE$GL_FLAGS, 1$              ; 1114
                                          00AF     31   00026           BRW      8$                                          ;
    0050    8F          00         6E        00    2C   00029 1$:       MOVC5    #0, (SP), #0, #80, (R6)                     ; 1132
                                             66         00030                                                                ;
                                   66     5003       8F    B0   00031   MOVW     #20483, (R6)                                ;
                          04       A6 40060040       8F    D0   00036   MOVL     #1074135104, 4(R6)                          ;
                          16       A6                02    90   0003E   MOVB     #2, 22(R6)                                  ;
                          1C       A6                01    8E   00042   MNEGB    #1, 28(R6)                                  ;
                          1F       A6                02    90   00046   MOVB     #2, 31(R6)                                  ;
                          28       A6                57    D0   0004A   MOVL     R7, 40(R6)                                  ;
                          2C       A6       04       A8    D0   0004E   MOVL     4(R8), 44(R6)                               ;
                          50                08       AC    D0   00053   MOVL     DFLT_DESC, R0                               ;
                          30       A6       04       A0    D0   00057   MOVL     4(R0), 48(R6)                               ;
                          34       A6       04       BC    90   0005C   MOVB     @NAME_DESC, 52(R6)                          ;
                          35       A6                60    90   00061   MOVB     (R0), 53(R6)                                ;
      4A    A6          02         02       64       AB    F0   00065   INSV     OWN_STORAGE+100, #2, #2, 74(R6)             ; 1134
                        08         10       A9       02    E1   0006C   BBC      #2, 16(R9), 2$                              ; 1139
                                   16       A6     4301    8F    B0   00071 MOVW  #17153, 22(R6)                             ; 1142
                                                     05    11   00077   BRB      3$                                          ; 1139
                          16       A6       80       8F    9B   00079 2$: MOVZBW #128, 22(R6)                                ; 1147
                05 0000000G        00                01    E1   0007E 3$: BBC    #1, IAC$GL_IMAGCTX, 4$                      ; 1156
                                   50                01    D0   00086   MOVL     #1, R0                                      ;
                                                     03    11   00089   BRB      5$                                          ;
                                   50                03    D0   0008B 4$: MOVL   #3, R0                                      ;
      4A    A6          02         00                50    F0   0008E 5$: INSV   R0, #0, #2, 74(R6)                          ;
    0060    8F          00         6E                00    2C   00094   MOVC5    #0, (SP), #0, #96, (R7)                     ; 1171
                                                     67         0009B                                                        ;
                                   67     6002       8F    B0   0009C   MOVW     #24578, (R7)                                ;
                          02       A7                01    8E   000A1   MNEGB    #1, 2(R7)                                   ;
                          04       A7       14       AC    D0   000A5   MOVL     NAME_STRING, 4(R7)                          ;
                          08       A7                10    90   000AA   MOVB     #16, 8(R7)                                  ;
                          0A       A7                01    8E   000AE   MNEGB    #1, 10(R7)                                  ;
```

```
                 0C  A7      04  A7  D0 000B2        MOVL    4(R7), 12(R7)                   :
                                 56  DD 000B7        PUSHL   R6                              :
          00000000G  00          01  FB 000B9        CALLS   #1, SYS$OPEN                    : 1173
                     51      40  AB  D0 000C0        MOVL    OWN_STORAGE+64, R1              : 1178
                                 0B  13 000C4        BEQL    6$                              :
     08  A1          04          00  0D 000C6        PROBEW  #0, #4, 8(R1)                   : 1185
                                 04  13 000CB        BEQL    6$                              :
                 08  A1          56  D0 000CD        MOVL    R6, 8(R1)                       : 1187
                     03          50  E9 000D1 6$:    BLBC    STATUS, 7$                      : 1193
                               0087  31 000D4        BRW     14$                             :
                                     04 000D7 7$:    RET                                     :
                 10  AE    0200  8F  3C 000D8 8$:    MOVZWL  #512, BUF_DESC                  : 1196
                 14  AE    FE00  CD  9E 000DE        MOVAB   RTVRBUF, BUF_DESC+4             :
     04  AE    FF07  CF          0C  28 000E4        MOVC3   #12, P.AAA, ARG_LIST            : 1212
                 0C  AE      10  AE  9E 000EB        MOVAB   BUF_DESC, ARG_LIST+8            : 1196
                             10  AE  9F 000F0        PUSHAB  BUF_DESC                        : 1215
                             04  AE  9F 000F3        PUSHAB  RTVRLEN                         :
                             20  AE  9F 000F6        PUSHAB  STATBLK                         :
                             2C  AE  9F 000F9        PUSHAB  FILHDR                          :
                           0230  CE  9F 000FC        PUSHAB  IXFHDR                          :
                             04  AC  DD 00100        PUSHL   NAME_DESC                       : 1216
                             0C  A6  9F 00103        PUSHAB  12(R6)                          : 1215
          00000000G  00          07  FB 00106        CALLS   #7, FIL$OPENFILE                :
                     26          50  E9 0010D        BLBC    STATUS, 11$                     : 1222
                 10  AE          6E  D1 00110        CMPL    RTVRLEN, BUF_DESC               : 1225
                                 04  14 00114        BGTR    9$                              :
                                 6E  D5 00116        TSTL    RTVRLEN                         : 1228
                                 05  12 00118        BNEQ    10$                             :
                     50      44  8F  9A 0011A 9$:    MOVZBL  #68, R0                         : 1229
                                     04 0011E        RET                                     :
                 10  AE          6E  D0 0011F 10$:   MOVL    RTVRLEN, BUF_DESC               : 1230
                 08  AE      0C  A6  D0 00123        MOVL    12(R6), ARG_LIST+4              : 1232
                             04  AE  9F 00128        PUSHAB  ARG_LIST                        : 1235
                          0000V  CF  9F 0012B        PUSHAB  INIT_WINDOW                     :
          00000000G  00          02  FB 0012F        CALLS   #2, SYS$CMKRNL                  :
                     01          50  E8 00136 11$:   BLBS    STATUS, 12$                     : 1236
                                     04 00139        RET                                     :
                 03  A7      04  BC  90 0013A 12$:   MOVB    @NAME_DESC, 3(R7)               : 1243
                     50      03  A7  9A 0013F        MOVZBL  3(R7), R0                       : 1245
     14  BC          04  B8      50  28 00143        MOVC3   R0, @4(R8), @NAME_STRING        : 1247
                     50      04  BC  3C 00149        MOVZWL  @NAME_DESC, R0                  : 1257
                     23          50  B1 0014D        CMPW    R0, #35                         :
                                 03  1B 00150        BLEQU   13$                             :
                     50          23  D0 00152        MOVL    #35, R0                         :
                 3B  A7          50  90 00155 13$:   MOVB    R0, 59(R7)                      : 1255
                 4C  A7      14  AC  D0 00159        MOVL    NAME_STRING, 76(R7)             : 1258
                 0E  A9      0C  A6  B0 0015E 14$:   MOVW    12(R6), 14(R9)                  : 1262
                 54  A9      18  A6  D0 00163        MOVL    24(R6), 84(R9)                  : 1263
                                 6A  95 00168        TSTB    (R10)                           : 1270
                                 0D  12 0016A        BNEQ    15$                             :
                     6A      3B  A7  90 0016C        MOVB    59(R7), (R10)                   : 1273
                     50          6A  9A 00170        MOVZBL  (R10), R0                       : 1275
     01  AA          4C  B7      50  28 00173        MOVC3   R0, @76(R7), 1(R10)             : 1277
         0D          41  A6      05  E0 00179 15$:   BBS     #5, 65(R6), 16$                 : 1284
         0C          40  A6      05  E1 0017E        BBC     #5, 64(R6), 17$                 : 1285
               0200  8F      3C  A6  B1 00183        CMPW    60(R6), #512                    :
                                 04  12 00189        BNEQ    17$                             :
```

```
                    10   A9           10  88 0018B 16$:   BISB2   #16, 16(R9)                   ; 1286
                              54      A9  D5 0018F 17$:   TSTL    84(R9)                        ; 1288
                                      0E  12 00192        BNEQ    18$
         51 00000000G    00           01  E1 00194        BBC     #1, IAC$GL IMAGCTX, 22$       ; 1291
                         50   2054    8F  3C 0019C        MOVZWL  #8276, R0
                                          04 001A1        RET
                         50     18    A6  D0 001A2 18$:   MOVL    24(R6), R0                    ; 1303
         56         58   A9           14  C1 001A6        ADDL3   #20, 88(R9), R6               ; 1305
         0A         10   A0           02  E1 001AB        BBC     #2, 16(R0), 19$               ; 1310
                         07           6B  E9 001B0        BLBC    OWN_STORAGE, 19$
            00000000G    00           02  88 001B3        BISB2   #2, IAC$GL_IMAGCTX            ; 1311
         06         11   A0           01  E1 001BA 19$:   BBC     #1, 17(R0), 20$               ; 1318
                         03           6B  E9 001BF        BLBC    OWN_STORAGE, 20$
                         6B           04  88 001C2        BISB2   #4, OWN_STORAGE               ; 1319
                         01     3D    A7  91 001C5 20$:   CMPB    61(R7), #1                    ; 1327
                                      06  12 001C9        BNEQ    21$
      03 A7         0B   A7           01  83 001CB        SUBB3   #1, 11(R7), 3(R7)             ; 1328
         17         10   A0           05  E1 001D1 21$:   BBC     #5, 16(R0), 22$               ; 1330
                         57     36    A0  9A 001D6        MOVZBL  54(R0), R7                    ; 1334
                                      57  D6 001DA        INCL    R7
         66         36   A0           57  28 001DC        MOVC3   R7, 54(R0), (R6)              ; 1336
                                    6746  9F 001E1        PUSHAB  (R7)[R6]                      ; 1340
                         9E   FE14    CF  D0 001E4        MOVL    P.AAB, @(SP)+
                    10   A9           02  88 001E9        BISB2   #2, 16(R9)                    ; 1341
                         50           01  D0 001ED 22$:   MOVL    #1, R0                        ; 1350
                                          04 001F0        RET                                   ; 1352

; Routine Size:  497 bytes,    Routine Base:  YF$$SYSIMGACT + 02E4

; 1126         1353  1
```

```
1 %SBTTL 'INIT_WINDOW - Interface Routine to FIL$INIWCB'
1
1 ROUTINE INIT_WINDOW (CHANNEL, BUFFER_DESC) : SYS_CMKRNL =
1
1 !+
1 ! Functional Description:
1 !
1 !     This routine acts as a jacket to the routine called FIL$INIWCB, which
1 !     allocates a window control block in the absence of a complete file
1 !     system.
1 !
1 ! Calling Sequence:
1 !
1 !     $CMKRNL (INIT_WINDOW, ARGUMENT_LIST)
1 !
1 ! Formal Parameters:
1 !
1 !     CHANNEL - Address of channel on which image file is opened
1 !
1 !     BUFFER_DESC - Address of descriptor of retrieval pointer buffer
1 !-
1
2 BEGIN
2
2 EXTERNAL REGISTER                          ! We enter this procedure with the PCB
2     PCB = 4 : REF $BBLOCK;                 !  address contained in R4
2
2 BIND
2     JIB     = .PCB [PCB$L_JIB] : $BBLOCK,
2     BUF_DSC = .BUFFER_DESC     : VECTOR [2];
2
2 LOCAL
2     STATUS,
2     CCB  : REF $BBLOCK,
2     WCB;
2
2 STATUS = IOC$VERIFYCHAN (.CHANNEL; CCB);
2 IF NOT .STATUS THEN RETURN .STATUS;
2 STATUS = FIL$INIWCB (.BUF_DSC [0], .BUF_DSC [1], .CCB [CCB$L_UCB]; WCB);
2 IF NOT .STATUS THEN RETURN .STATUS;
2 CCB [CCB$L_WIND] = .WCB;
2 CCB [CCB$B_AMOD] = PSL$C_USER + 1;
2 JIB [JIB$W_FILCNT] = .JIB [JIB$W_FILCNT] -1;
2 RETURN SS$_NORMAL
2
1 END;


                                  00EC 00000 INIT_WINDOW:
                                                     .WORD   Save R2,R3,R5,R6,R7          ; 1356
           56      0080   C4  D0 00002               MOVL    128(PCB), R6                 ; 1382
           57        08   AC  D0 00007               MOVL    BUFFER_DESC, R7              ; 1383
           50        04   AC  D0 0000B               MOVL    CHANNEL, R0                  ; 1390
              00000000G   00  16 0000F               JSB     IOC$VERIFYCHAN               ;
```

```
                                55              51  D0 00015          MOVL    R1, R5                                  ;
                                1D              50  E9 00018          BLBC    STATUS, 1$                              ; 1391
                                53              65  D0 0001B          MOVL    (CCB), R3                               ; 1392
                                51              67  7D 0001E          MOVQ    (R7), R1                                ;
                                   00000000G    00  16 00021          JSB     FIL$INIWCB                              ;
                                0E              50  E9 00027          BLBC    STATUS, 1$                              ; 1393
                          04    A5              52  D0 0002A          MOVL    WCB, 4(CCB)                             ; 1394
                          09    A5              04  90 0002E          MOVB    #4, 9(CCB)                              ; 1395
                                      30        A6  B7 00032          DECW    48(R6)                                  ; 1396
                                50              01  D0 00035          MOVL    #1, R0                                  ; 1397
                                                    04 00038 1$:      RET                                             ; 1399

; Routine Size:  57 bytes,    Routine Base:  YF$$SYSIMGACT + 04D5
```

```
;  1175          1400  1 %SBTTL 'IMG$GET_HEADER - Get Parameters from Image Header'
;  1176          1401  1
;  1177          1402  1 GLOBAL ROUTINE IMG$GET_HEADER (ICB_ADDRESS) =
;  1178          1403  1
;  1179          1404  1 !+
;  1180          1405  1 ! Functional Description:
;  1181          1406  1 !
;  1182          1407  1 !     This routine is a jacket routine that handler two kinds of image header.
;  1183          1408  1 !     If an image was installed header resident, then there is no need to decode
;  1184          1409  1 !     the header or verify its contents. Other images must have their headers
;  1185          1410  1 !     read into memory before the decode and verification can occur.
;  1186          1411  1 !
;  1187          1412  1 ! Calling Sequence:
;  1188          1413  1 !
;  1189          1414  1 !     IMG$GET_HEADER (ICB_ADDRESS)
;  1190          1415  1 !
;  1191          1416  1 ! Formal Parameter:
;  1192          1417  1 !
;  1193          1418  1 !     ICB_ADDRESS - Address of image control block that describes the image
;  1194          1419  1 !         that is currently being activated.
;  1195          1420  1 !-
;  1196          1421  1
;  1197          1422  2 BEGIN
;  1198          1423  2
;  1199          1424  2 BIND
;  1200          1425  2     ICB_ADR = .ICB_ADDRESS              : $BBLOCK,
;  1201          1426  2     KFE     = .ICB_ADR [ICB$L_KFE]      : $BBLOCK,
;  1202          1427  2     IHD_CTX = .ICB_ADR [ICB$L_CONTEXT]  : $BBLOCK;
;  1203          1428  2
;  1204          1429  2 LOCAL
;  1205          1430  2     STATUS;
;  1206          1431  2
;  1207          1432  2 IF
;  1208          1433  3     BEGIN
;  1209          1434  3     IF KFE EQL 0
;  1210          1435  3     THEN TRUE
;  1211          1436  3     ELSE NOT .KFE [KFE$V_HDRRES]
;  1212          1437  3     END
;  1213          1438  2 THEN
;  1214          1439  3     BEGIN
;  1215          1440  3
;  1216          1441  3     STATUS = IMG$DECODE_IHD (
;  1217          1442  3         .ICB_ADR [ICB$W_CHAN],
;  1218          1443  3         .IHD_CTX [CTX_L_BUFFER],
;  1219          1444  3         .IHD_CTX [CTX_L_IHDBUF],
;  1220          1445  3         IHD_CTX [CTX_L_VBN],
;  1221          1446  3         IHD_CTX [CTX_W_ISD_OFFSET],
;  1222          1447  3         IHD_CTX [CTX_W_GENERATION],
;  1223          1448  3         IHD_CTX [CTX_W_ALIAS]);
;  1224          1449  3
;  1225          1450  3     IF NOT .STATUS THEN RETURN .STATUS;
;  1226          1451  3
;  1227          1452  3     ! Images that contain an alias of either IHD$C_RSX or IHD$C_BPA are not
;  1228          1453  3     ! native images produced by the linker and, as a result, do not require
;  1229          1454  3     ! a check against the system version field.
;  1230          1455  3
;  1231          1456  3     IF
```

```
;  1232          1457  4          (.IHD_CTX [CTX_W_ALIAS] NEQ IHD$C_RSX)
;  1233          1458  3          AND
;  1234          1459  4          (.IHD_CTX [CTX_W_ALIAS] NEQ IHD$C_BPA)
;  1235          1460  3          AND
;  1236          1461  4          (.IHD_CTX [CTX_W_ALIAS] NEQ IHD$C_ALIAS)
;  1237          1462  3      THEN
;  1238          1463  4          BEGIN
;  1239          1464  4
;  1240          1465  4          BIND
;  1241          1466  4              IHD = .IHD_CTX [CTX_L_IHDBUF]       : $BBLOCK,
;  1242          1467  4              MINORID_DIGIT = IHD [IHD$W_MINORID] : VECTOR [2,BYTE],
;  1243          1468  4              IHA  = IHD + .IHD [IHD$W_ACTIVOFF]  : VECTOR [5];
;  1244          1469  4
;  1245          1470  4          ! MINORID_DIGIT [0] maps the tens digit.
;  1246          1471  4          ! MINORID_DIGIT [1] maps the units digit.
;  1247          1472  4
;  1248          1473  4          LITERAL
;  1249          1474  4              MINOR_ID_TENS = IHD$K_MINORID AND %X'FF',
;  1250          1475  4              MINOR_ID_ONES = IHD$K_MINORID ^ -8;
;  1251          1476  4
;  1252          1477  4          ! The major ID in the image header must be identically equal to
;  1253          1478  4          ! the constant IHD$K_MAJORID. The minor ID in the image header
;  1254          1479  4          ! must be LEQU the constant IHD$K_MINORID. Both IDs are stored
;  1255          1480  4          ! as ASCII strings.
;  1256          1481  4
;  1257          1482  5          IF (.IHD [IHD$W_MAJORID] NEQU IHD$K_MAJORID)
;  1258          1483  4          THEN RETURN SS$_BADIMGHDR;
;  1259          1484  4
;  1260          1485  5          IF (
;  1261          1486  6              (.MINORID_DIGIT [0] GTRU MINOR_ID_TENS)
;  1262          1487  5              OR
;  1263          1488  6              (
;  1264          1489  7                  (.MINORID_DIGIT [0] EQLU MINOR_ID_TENS)
;  1265          1490  6                  AND
;  1266          1491  7                  (.MINORID_DIGIT [1] GTRU MINOR_ID_ONES)
;  1267          1492  6              )
;  1268          1493  5          )
;  1269          1494  4          THEN RETURN SS$_BADIMGHDR;
;  1270          1495  4
;  1271          1496  4          ! Check match control data for shareable images that are being
;  1272          1497  4          ! activated because of global ISD references.
;  1273          1498  4
;  1274          1499  4          IF .ICB_ADR [ICB$B_ACT_CODE] EQLU ICB$K_GLOBAL_IMAGE_SECTION
;  1275          1500  4          THEN
;  1276          1501  5              BEGIN
;  1277          1502  5              STATUS = CHECK_MATCH_CONTROL (ICB_ADR, IHD);
;  1278          1503  5              IF NOT .STATUS THEN RETURN .STATUS;
;  1279          1504  4              END;
;  1280          1505  4
;  1281          1506  4          ! We must record whether we are activating a shareable image with an
;  1282          1507  4          ! initialization section.
;  1283          1508  4
;  1284          1509  4          IF .IHD [IHD$V_INISHR]
;  1285          1510  4          THEN
;  1286          1511  5              BEGIN
;  1287          1512  5              ICB_ADR [ICB$L_INITIALIZE] = .IHA [4];
;  1288          1513  5              ICB_ADR [ICB$V_INITIALIZE] = TRUE;
```

```
;  1289          1514  5              IAC$GL_IMAGCTX [IMAGCTX$V_INITIALIZE] = TRUE;
;  1290          1515  4              END;
;  1291          1516  4
;  1292          1517  4          ! If the image was linked against a SYS.STB for other than the current
;  1293          1518  4          ! system, then CMEXEC and CMKRNL privilege will be removed.
;  1294          1519  4
;  1295          1520  5          IF  (.IHD [IHD$L_SYSVER] NEQU 0)
;  1296          1521  4          THEN
;  1297          1522  5              BEGIN
;  1298          1523  5              ICB_ADR [ICB$V_SYS_STB] = TRUE;
;  1299          1524  6              IF  (.IHD [IHD$L_SYSVER] NEQU SYS$K_VERSION)
;  1300          1525  5              THEN OWN_STORAGE [REMOVE_PRIVILEGE] = TRUE;
;  1301          1526  4              END;
;  1302          1527  4
;  1303          1528  3          END;
;  1304          1529  3      END
;  1305          1530  2  ELSE              ! Header is already resident
;  1306          1531  3      BEGIN
;  1307          1532  3
;  1308          1533  3      BIND
;  1309          1534  3          IHD  = .KFE [KFE$L_IMGHDR]          : $BBLOCK,
;  1310          1535  3          KFRH = IHD - KFRH$K_LENGTH          : $BBLOCK,
;  1311          1536  3          IHA  = IHD + .IHD [IHD$W_ACTIVOFF] : VECTOR [5];
;  1312          1537  3
;  1313          1538  3      ICB_ADR [ICB$V_RES_HEADER] = TRUE;  ! Indicate that header is already in memory
;  1314          1539  3      ICB_ADR [ICB$L_IHD] = IHD;          ! Remember the address of the resident IHD
;  1315          1540  3      IHD_CTX [CTX_L_IHDBUF] = 0;         ! Clear this so it won't get used by something else later
;  1316          1541  3      IHD_CTX [CTX_W_ISD_OFFSET] = .IHD [IHD$W_SIZE];
;  1317          1542  3      IHD_CTX [CTX_W_GENERATION] = .KFRH [KFRH$B_HDRVER];
;  1318          1543  3      IHD_CTX [CTX_W_ALIAS] = .KFRH [KFRH$W_ALIAS];
;  1319          1544  3
;  1320          1545  3      IF .ICB_ADR [ICB$B_ACT_CODE] EQLU ICB$K_GLOBAL_IMAGE_SECTION
;  1321          1546  3      THEN
;  1322          1547  4          BEGIN
;  1323          1548  4          STATUS = CHECK_MATCH_CONTROL (ICB_ADR, IHD);
;  1324          1549  4          IF NOT .STATUS THEN RETURN .STATUS;
;  1325          1550  3          END;
;  1326          1551  3
;  1327          1552  3      ! We must record whether we are activating a shareable image with an
;  1328          1553  3      ! initialization section.
;  1329          1554  3
;  1330          1555  3      IF .IHD [IHD$V_INISHR]
;  1331          1556  3      THEN
;  1332          1557  4          BEGIN
;  1333          1558  4          ICB_ADR [ICB$L_INITIALIZE] = .IHA [4];
;  1334          1559  4          ICB_ADR [ICB$V_INITIALIZE] = TRUE;
;  1335          1560  3          END;
;  1336          1561  3
;  1337          1562  3      ! If the image was linked against a SYS.STB for other than the current
;  1338          1563  3      ! system, then CMEXEC and CMKRNL privilege will be removed.
;  1339          1564  3
;  1340          1565  4      IF  (.IHD [IHD$L_SYSVER] NEQU 0)
;  1341          1566  3      THEN
;  1342          1567  4          BEGIN
;  1343          1568  4          ICB_ADR [ICB$V_SYS_STB] = TRUE;
;  1344          1569  5          IF  (.IHD [IHD$L_SYSVER] NEQU SYS$K_VERSION)
;  1345          1570  4          THEN OWN_STORAGE [REMOVE_PRIVILEGE] = TRUE;
```

```
;  1346          1571  3          END;
;  1347          1572  3
;  1348          1573  3      STATUS = SS$_NORMAL;
;  1349          1574  2      END;
;  1350          1575  2 RETURN .STATUS;
;  1351          1576  2
;  1352          1577  1 END;


                                                00FC 00000          .ENTRY  IMG$GET_HEADER, Save R2,R3,R4,R5,R6,R7      ; 1402
                                    57 00000000G  00   9E 00002     MOVAB   OWN_STORAGE, R7                             ;
                                    56 00000000G  8F   D0 00009     MOVL    #SYS$K_VERSION, R6                          ;
                                    53        04  AC   D0 00010     MOVL    ICB_ADDRESS, R3                             ; 1425
                                    50        54  A3   D0 00014     MOVL    84(R3), R0                                  ; 1426
                                    52        58  A3   D0 00018     MOVL    88(R3), R2                                  ; 1427
                                                  50   D5 0001C     TSTL    R0                                          ; 1434
                                                  08   13 0001E     BEQL    1$                                          ;
                          03        10  A0            04   E1 00020     BBC     #4, 16(R0), 1$                          ; 1436
                                                  0090 31 00025     BRW     8$                                          ;
                                              0E  A2   9F 00028 1$: PUSHAB  14(R2)                                      ; 1448
                                              10  A2   9F 0002B     PUSHAB  16(R2)                                      ; 1447
                                              0C  A2   9F 0002E     PUSHAB  12(R2)                                      ; 1446
                                              08  A2   9F 00031     PUSHAB  8(R2)                                       ; 1445
                                    7E            62   7D 00034     MOVQ    (R2), -(SP)                                 ; 1448
                                    7E        0E  A3   3C 00037     MOVZWL  14(R3), -(SP)                               ;
                              00000000G       00        07   FB 0003B     CALLS   #7, IMG$DECODE_IHD                      ;
                                    47              50   E9 00042     BLBC    STATUS, 4$                                  ; 1450
                                    51        0E  A2   32 00045     CVTWL   14(R2), R1                                  ; 1457
                                                  67   13 00049     BEQL    7$                                          ;
                                    01            51   B1 0004B     CMPW    R1, #1                                      ; 1459
                                                  62   13 0004E     BEQL    7$                                          ;
                                    02            51   B1 00050     CMPW    R1, #2                                      ; 1461
                                                  5D   13 00053     BEQL    7$                                          ;
                                    54        04  A2   D0 00055     MOVL    4(R2), R4                                   ; 1466
                                    51        0E  A4   9E 00059     MOVAB   14(R4), R1                                  ; 1467
                                    55        02  A4   3C 0005D     MOVZWL  2(R4), R5                                   ; 1468
                          52        54            55   C1 00061     ADDL3   R5, R4, R2                                  ;
                                3230  8F        0C  A4   B1 00065     CMPW    12(R4), #12848                          ; 1482
                                                  0D   12 0006B     BNEQ    2$                                          ;
                                    30            61   91 0006D     CMPB    (R1), #48                                   ; 1486
                                                  08   1A 00070     BGTRU   2$                                          ;
                                                  0B   12 00072     BNEQ    3$                                          ; 1489
                                    35        01  A1   91 00074     CMPB    1(R1), #53                                  ; 1491
                                                  05   1B 00078     BLEQU   3$                                          ;
                                    50        44  8F   9A 0007A 2$: MOVZBL  #68, R0                                    ; 1494
                                                  04   0007E     RET                                                     ;
                                    03        0D  A3   91 0007F 3$: CMPB    13(R3), #3                                  ; 1499
                                                  0A   12 00083     BNEQ    5$                                          ;
                                                  18   BB 00085     PUSHR   #^M<R3,R4>                                  ; 1502
                                0000V       CF        02   FB 00087     CALLS   #2, CHECK_MATCH_CONTROL                  ;
                                    5F              50   E9 0008C 4$: BLBC    STATUS, 9$                                  ; 1503
                          10        20  A4            06   E1 0008F 5$: BBC     #6, 32(R4), 6$                          ; 1509
                                    60  A3        10  A2   D0 00094     MOVL    16(R2), 96(R3)                          ; 1512
                                    10  A3            20   88 00099     BISB2   #32, 16(R3)                             ; 1513
```

```
                00000000G  00              02  88  0009D                BISB2   #2, IAC$GL_IMAGCTX+2                    ; 1514
                                       28   A4  D5  000A4  6$:          TSTL    40(R4)                                  ; 1520
                                            6C  13  000A7                BEQL    13$                                     ;
                         10    A3      80   8F  88  000A9                BISB2   #128, 16(R3)                           ; 1523
                               56      28   A4  D1  000AE                CMPL    40(R4), R6                              ; 1524
                                            61  13  000B2  7$:           BEQL    13$                                     ;
                               67            08  88  000B4                BISB2   #8, OWN_STORAGE                         ; 1525
                                             04  000B7                    RET                                             ; 1432
                               54      1C    A0  D0  000B8  8$:          MOVL    28(R0), R4                              ; 1534
                               51      F4    A4  9E  000BC               MOVAB   -12(R4), R1                             ; 1535
                               50      02    A4  3C  000C0               MOVZWL  2(R4), R0                               ; 1536
                    55         54            50  C1  000C4               ADDL3   R0, R4, R5                              ;
                         10    A3            08  88  000C8               BISB2   #8, 16(R3)                              ; 1538
                         50    A3            54  D0  000CC               MOVL    R4, 80(R3)                              ; 1539
                                       04   A2  D4  000D0               CLRL    4(R2)                                   ; 1540
                         0C    A2            64  B0  000D3               MOVW    (R4), 12(R2)                            ; 1541
                         10    A2      0B    A1  9B  000D7               MOVZBW  11(R1), 16(R2)                          ; 1542
                         0E    A2      04    A1  B0  000DC               MOVW    4(R1), 14(R2)                           ; 1543
                               03      0D    A3  91  000E1               CMPB    13(R3), #3                              ; 1545
                                             0A  12  000E5               BNEQ    10$                                     ;
                                             18  BB  000E7               PUSHR   #^M<R3,R4>                              ; 1548
                      0000V    CF            02  FB  000E9               CALLS   #2, CHECK_MATCH_CONTROL                  ;
                               24            50  E9  000EE  9$:          BLBC    STATUS, 13$                             ; 1549
                    09   20    A4            06  E1  000F1  10$:         BBC     #6, 32(R4), 11$                         ; 1555
                         60    A3      10    A5  D0  000F6               MOVL    16(R5), 96(R3)                          ; 1558
                         10    A3            20  88  000FB               BISB2   #32, 16(R3)                             ; 1559
                                       28   A4  D5  000FF  11$:          TSTL    40(R4)                                  ; 1565
                                            0E  13  00102                BEQL    12$                                     ;
                         10    A3      80   8F  88  00104               BISB2   #128, 16(R3)                            ; 1568
                               56      28   A4  D1  00109               CMPL    40(R4), R6                              ; 1569
                                            03  13  0010D                BEQL    12$                                     ;
                               67            08  88  0010F               BISB2   #8, OWN_STORAGE                         ; 1570
                               50            01  D0  00112  12$:         MOVL    #1, STATUS                              ; 1573
                                             04  00115  13$:             RET                                             ; 1577

; Routine Size:  278 bytes,    Routine Base:  YF$$SYSIMGACT + 050E
```

```
;  1354     1578  1 %SBTTL 'CHECK_MATCH_CONTROL - Check Match Control Identification'
;  1355     1579  1
;  1356     1580  1 ROUTINE CHECK_MATCH_CONTROL (ICB, IHD ) =
;  1357     1581  1
;  1358     1582  1 !+
;  1359     1583  1 ! Functional Description:
;  1360     1584  1 !
;  1361     1585  1 !    This routine checks that the match control information in the image header
;  1362     1586  1 !    is consistent with the match control information that is actually being
;  1363     1587  1 !    requested. The check is made in three steps.
;  1364     1588  1 !
;  1365     1589  1 !    1.  The match control flag that is being requested must be at least as
;  1366     1590  1 !        restrictive as the match control flag in the image header.
;  1367     1591  1 !
;  1368     1592  1 !    2.  If the resultant match control is MATCH ALWAYS, no further checks are
;  1369     1593  1 !        made. If the resultant check is MATCH NEVER, ?????.
;  1370     1594  1 !
;  1371     1595  1 !    3.  If the match control is either MATCH EQUAL or MATCH LEQUAL, two
;  1372     1596  1 !        further checks are made.
;  1373     1597  1 !
;  1374     1598  1 !        a.  The two major IDs (one in ICB and one in IHD) must be equal.
;  1375     1599  1 !
;  1376     1600  1 !        b.  The two minor IDs must be related according to the match control.
;  1377     1601  1 !            In the case of MATCH EQUAL, they must be equal. In the case of
;  1378     1602  1 !            MATCH LEQUAL, the requested minor ID (located in the ICB) must be
;  1379     1603  1 !            LEQU the minor ID in the image header of the image being
;  1380     1604  1 !            activated.
;  1381     1605  1 !
;  1382     1606  1 ! Calling Sequence:
;  1383     1607  1 !
;  1384     1608  1 !    CHECK_MATCH_CONTROL (ICB, IHD )
;  1385     1609  1 !
;  1386     1610  1 ! Formal Parameter:
;  1387     1611  1 !
;  1388     1612  1 !    ICB - Address of image control block that describes the image
;  1389     1613  1 !        that is currently being activated.
;  1390     1614  1 !
;  1391     1615  1 !    IHD - Address of image header of image being activated.
;  1392     1616  1 !-
;  1393     1617  1
;  1394     1618  2 BEGIN
;  1395     1619  2
;  1396     1620  2 MACRO
;  1397     1621  2     IHD_V_MINOR_ID = $BYTEOFFSET (IHD$L_IDENT), 0, 24, 0 %,
;  1398     1622  2     IHD_V_MAJOR_ID = $BYTEOFFSET (IHD$L_IDENT), 24, 8, 0 %;
;  1399     1623  2
;  1400     1624  2 MAP
;  1401     1625  2     ICB : REF $BBLOCK,
;  1402     1626  2     IHD : REF $BBLOCK;
;  1403     1627  2
;  1404     1628  2 CASE .IHD [IHD$V_MATCHCTL]
;  1405     1629  2 FROM ISD$K_MATALL TO ISD$K_MATNEV OF
;  1406     1630  2 SET
;  1407     1631  2
;  1408     1632  2 [ISD$K_MATALL]:
;  1409     1633  2
;  1410     1634  2     SS$_NORMAL;                        ! Do nothing
```

```
;  1411          1635  2
;  1412          1636  2 [ISD$K_MATEQU]:
;  1413          1637  2
;  1414          1638  2     IF
;  1415          1639  3         (.ICB [ICB$V_MATCH_CONTROL] EQLU ISD$K_MATALL)
;  1416          1640  2         OR
;  1417          1641  3         (.ICB [ICB$V_MATCH_CONTROL] EQLU ISD$K_MATLEQ)
;  1418          1642  2     THEN RETURN SS$_SHRIDMISMAT;
;  1419          1643  2
;  1420          1644  2 [ISD$K_MATLEQ]:
;  1421          1645  2
;  1422          1646  2     IF .ICB [ICB$V_MATCH_CONTROL] EQLU ISD$K_MATALL
;  1423          1647  2     THEN RETURN SS$_SHRIDMISMAT;
;  1424          1648  2
;  1425          1649  2 [ISD$K_MATNEV]:
;  1426          1650  2
;  1427          1651  2     RETURN SS$_SHRIDMISMAT;
;  1428          1652  2
;  1429          1653  2 TES;
;  1430          1654  2
;  1431          1655  2 CASE .ICB [ICB$V_MATCH_CONTROL]
;  1432          1656  2 FROM ISD$K_MATAL[ TO ISD$K_MATNEV OF
;  1433          1657  2 SET
;  1434          1658  2
;  1435          1659  2 [ISD$K_MATALL]:
;  1436          1660  2
;  1437          1661  2     RETURN SS$_NORMAL;
;  1438          1662  2
;  1439          1663  2 [ISD$K_MATEQU, ISD$K_MATNEV]:
;  1440          1664  2
;  1441          1665  3     BEGIN
;  1442          1666  3
;  1443          1667  3     IF .ICB [ICB$V_MAJOR_ID] NEQU .IHD [IHD_V_MAJOR_ID]
;  1444          1668  3     THEN RETURN SS$_SHRIDMISMAT;
;  1445          1669  3
;  1446          1670  3     IF .ICB [ICB$V_MINOR_ID] EQLU .IHD [IHD_V_MINOR_ID]
;  1447          1671  3     THEN RETURN SS$_NORMAL
;  1448          1672  3     ELSE RETURN SS$_SHRIDMISMAT;
;  1449          1673  3
;  1450          1674  2     END;
;  1451          1675  2
;  1452          1676  2 [ISD$K_MATLEQ]:
;  1453          1677  2
;  1454          1678  3     BEGIN
;  1455          1679  3
;  1456          1680  3     IF .ICB [ICB$V_MAJOR_ID] NEQU .IHD [IHD_V_MAJOR_ID]
;  1457          1681  3     THEN RETURN SS$_SHRIDMISMAT;
;  1458          1682  3
;  1459          1683  3     IF .ICB [ICB$V_MINOR_ID] LEQU .IHD [IHD_V_MINOR_ID]
;  1460          1684  3     THEN RETURN SS$_NORMAL
;  1461          1685  3     ELSE RETURN SS$_SHRIDMISMAT;
;  1462          1686  3
;  1463          1687  2     END;
;  1464          1688  2
;  1465          1689  2 TES;
;  1466          1690  2
;  1467          1691  1 END;
```

```
                                             000C 00000 CHECK_MATCH_CONTROL:
                                                               .WORD   Save R2,R3                         ; 1580
                          51       08    AC   D0 00002         MOVL    IHD, R1                            ; 1628
     52       23    A1    03             00   EF 00006         EXTZV   #0, #3, 35(R1), R2
                    03    00             52   CF 0000C         CASEL   R2, #0, #3
   006E           001C  000A           0026      00010 1$:     .WORD   5$-1$,-
                                                                       2$-1$,-
                                                                       3$-1$,-
                                                                       10$-1$
                                         1C   11 00018         BRB     5$
                          50       04    AC   D0 0001A 2$:     MOVL    ICB, R0                            ; 1639
                          07       40    A0   93 0001E         BITB    64(R0), #7
                                         5A   13 00022         BEQL    10$
     02       40    A0    03             00   ED 00024         CMPZV   #0, #3, 64(R0), #2                 ; 1641
                                         08   11 0002A         BRB     4$
                          50       04    AC   D0 0002C 3$:     MOVL    ICB, R0                            ; 1646
                          07       40    A0   93 00030         BITB    64(R0), #7
                                         48   13 00034 4$:     BEQL    10$
                          50       04    AC   D0 00036 5$:     MOVL    ICB, R0                            ; 1655
     52       40    A0    03             00   EF 0003A         EXTZV   #0, #3, 64(R0), R2                 ; 1678
                    03    00             52   CF 00040         CASEL   R2, #0, #3
   000A           0021  000A           0036      00044 6$:     .WORD   9$-6$,-
                                                                       7$-6$,-
                                                                       8$-6$,-
                                                                       7$-6$
                                         2C   11 0004C         BRB     9$
                    27    A1       47    A0   91 0004E 7$:     CMPB    71(R0), 39(R1)                     ; 1667
                                         29   12 00053         BNEQ    10$
     53       24    A1    18             00   EF 00055         EXTZV   #0, #24, 36(R1), R3                ; 1670
     53       44    A0    18             00   ED 0005B         CMPZV   #0, #24, 68(R0), R3
                                         1B   12 00061         BNEQ    10$
                                         15   11 00063         BRB     9$                                 ; 1672
                    27    A1       47    A0   91 00065 8$:     CMPB    71(R0), 39(R1)                     ; 1680
                                         12   12 0006A         BNEQ    10$
     53       24    A1    18             00   EF 0006C         EXTZV   #0, #24, 36(R1), R3                ; 1683
     53       44    A0    18             00   ED 00072         CMPZV   #0, #24, 68(R0), R3
                                         04   1A 00078         BGTRU   10$
                          50             01   D0 0007A 9$:     MOVL    #1, R0                             ; 1685
                                              04 0007D         RET
                          50     20BC    8F   3C 0007E 10$:    MOVZWL  #8380, R0
                                              04 00083         RET                                        ; 1691

; Routine Size:  132 bytes,    Routine Base:  YF$$SYSIMGACT + 0624
```

```
;  1469          1692  1 %SBTTL 'END_PROCESSING - Final Steps of Image Activation'
;  1470          1693  1
;  1471          1694  1 ROUTINE END_PROCESSING (ICB_ADDRESS) =
;  1472          1695  1
;  1473          1696  1 !+
;  1474          1697  1 ! Functional Description:
;  1475          1698  1 !
;  1476          1699  1 !    This routine performs the steps involved in image activation after all
;  1477          1700  1 !    of the ISDs have been processed. (Some of the details are actually
;  1478          1701  1 !    performed by the kernel mode routine SET_CONTROL_REGION.) The operations
;  1479          1702  1 !    that need to be completed include
;  1480          1703  1 !
;  1481          1704  1 !    o mapping the image I/O segment
;  1482          1705  1 !
;  1483          1706  1 !    o mapping the user stack and setting the initial value of USP
;  1484          1707  1 !
;  1485          1708  1 !    o setting up the process privilege mask
;  1486          1709  1 !
;  1487          1710  1 !    o optionally setting up image accounting information
;  1488          1711  1 !
;  1489          1712  1 !    o bumping the use count in any KFEs referenced by this activation
;  1490          1713  1 !
;  1491          1714  1 !    o bumping the reference count in any shared WCBs referenced by this
;  1492          1715  1 !      activation
;  1493          1716  1 !
;  1494          1717  1 !    o returning image parameters to the caller
;  1495          1718  1 !
;  1496          1719  1 ! Calling Sequence:
;  1497          1720  1 !
;  1498          1721  1 !    END_PROCESSING (ICB_ADDRESS)
;  1499          1722  1 !-
;  1500          1723  1
;  1501          1724  2 BEGIN
;  1502          1725  2
;  1503          1726  2 LOCAL
;  1504          1727  2     IMAGE_IO_PAGE_COUNT,
;  1505          1728  2     STACK_BASE,
;  1506          1729  2     CRETVA_RANGE        : VECTOR [2],
;  1507          1730  2     ARG_LIST            : VECTOR [4] INITIAL (
;  1508          1731  2         3,                                  ! Argument count
;  1509          1732  2         CRETVA_RANGE,                       ! INADR
;  1510          1733  2         0,                                  ! Null retadr
;  1511          1734  2         EXEC_PROT),                         ! Access mode and protection
;  1512          1735  2     IMGIO_SEG_DESC      : VECTOR [2] INITIAL (0,0),
;  1513          1736  2     STATUS;
;  1514          1737  2
;  1515          1738  2 BIND
;  1516          1739  2     ICB = .ICB_ADDRESS          : $BBLOCK,
;  1517          1740  2     KFE = .ICB [ICB$L_KFE]      : $BBLOCK,
;  1518          1741  2     PHD = .CTL$GL_PHD           : $BBLOCK,
;  1519          1742  2     STACK_ARRAY = CRETVA_RANGE  : VECTOR;
;  1520          1743  2
;  1521          1744  2 ! The location of the image header depends on whether the image was installed
;  1522          1745  2 ! with its header resident.
;  1523          1746  2
;  1524          1747  2 BIND
;  1525          1748  2     IHD =
```

```
;  1526     1749  3          (IF .ICB [ICB$L_IHD] EQL 0
;  1527     1750  3          THEN
;  1528     1751  4              BEGIN
;  1529     1752  4
;  1530     1753  4              BIND
;  1531     1754  4                  CTX = .ICB [ICB$L_CONTEXT]      : $BBLOCK;
;  1532     1755  4
;  1533     1756  4              .CTX [CTX_L_IHDBUF]
;  1534     1757  4
;  1535     1758  4              END
;  1536     1759  3          ELSE
;  1537     1760  2              .ICB [ICB$L_IHD])                   : $BBLOCK;
;  1538     1761  2
;  1539     1762  2 ! If this is the activation of a main program, then the image I/O segment and
;  1540     1763  2 ! user stack must be mapped.
;  1541     1764  2
;  1542     1765  2 IF .OWN_STORAGE [MAIN_PROGRAM]
;  1543     1766  2 THEN
;  1544     1767  3     BEGIN
;  1545     1768  4     IMAGE_IO_PAGE_COUNT = (
;  1546     1769  4         IF .IHD [IHD$W_IMGIOCNT] NEQ 0
;  1547     1770  4         THEN
;  1548     1771  4             .IHD [IHD$W_IMGIOCNT]
;  1549     1772  4         ELSE
;  1550     1773  3             .SGN$GW_IMGIOCNT);
;  1551     1774  3
;  1552     1775  3     IF .IMAGE_IO_PAGE_COUNT GTR .SGN$GW_IMGIOCNT
;  1553     1776  3     THEN
;  1554     1777  4     BEGIN
;  1555     1778  4         IF .IHD [IHD$V_POIMAGE]
;  1556     1779  4         THEN
;  1557     1780  5             BEGIN
;  1558     1781  5             CRETVA_RANGE [0] = .PHD [PHD$L_FREPOVA];
;  1559     1782  5             CRETVA_RANGE [1] =
;  1560     1783  5                 .CRETVA_RANGE [0]
;  1561     1784  6                 + (.IMAGE_IO_PAGE_COUNT * BYTES_PER_PAGE)
;  1562     1785  5                 - 1;
;  1563     1786  5             END
;  1564     1787  4         ELSE
;  1565     1788  5             BEGIN
;  1566     1789  5             CRETVA_RANGE [1] = .CTL$GL_CTLBASVA - 1;
;  1567     1790  5             CRETVA_RANGE [0] =
;  1568     1791  5                 .CRETVA_RANGE [1]
;  1569     1792  5                 - (.IMAGE_IO_PAGE_COUNT * BYTES_PER_PAGE) + 1;
;  1570     1793  4             END;
;  1571     1794  4
;  1572     1795  4         ! Create the image I/O segment using the internal routine that allows
;  1573     1796  4         ! a mixed-mode protection mask to be specified.
;  1574     1797  4
;  1575   P 1798  4         STATUS = $CMKRNL (
;  1576   P 1799  4             ROUTIN = MMG$CRETVA,
;  1577     1800  4             ARGLST = ARG_LIST);
;  1578     1801  4         IF NOT .STATUS
;  1579     1802  4         THEN RETURN .STATUS;
;  1580     1803  4
;  1581     1804  4         IMGIO_SEG_DESC [0] = .IMAGE_IO_PAGE_COUNT * BYTES_PER_PAGE;
;  1582     1805  4         IMGIO_SEG_DESC [1] = .CRETVA_RANGE [0];
```

```
;  1583          1806  4
;  1584          1807  3      END;
;  1585          1808  3
;  1586          1809  3      RM$SET (.IHD [IHD$L_LNKFLAGS], IMGIO_SEG_DESC);
;  1587          1810  3
;  1588          1811  3      IF .IHD [IHD$V_POIMAGE]
;  1589          1812  3      THEN
;  1590          1813  4          BEGIN
;  1591        P 1814  4          STATUS = $EXPREG (
;  1592        P 1815  4              PAGCNT = .OWN_STORAGE [USER_STACK_SIZE],
;  1593        P 1816  4              RETADR = STACK_ARRAY,
;  1594        P 1817  4              ACMODE = PSL$C_USER,
;  1595          1818  4              REGION = 0);
;  1596          1819  4          STACK_BASE = .STACK_ARRAY [1] + 1;
;  1597          1820  4          END
;  1598          1821  3      ELSE
;  1599          1822  4          BEGIN
;  1600        P 1823  4          STATUS = $EXPREG (
;  1601        P 1824  4              PAGCNT = .OWN_STORAGE [USER_STACK_SIZE],
;  1602        P 1825  4              RETADR = STACK_ARRAY,
;  1603        P 1826  4              ACMODE = PSL$C_USER,
;  1604          1827  4              REGION = 1);
;  1605          1828  4          STACK_BASE = .STACK_ARRAY [0] + 1;
;  1606          1829  3          END;
;  1607          1830  3
;  1608          1831  3      IF NOT .STATUS
;  1609          1832  3      THEN RETURN .STATUS;
;  1610          1833  2      END;                                    ! End of test for main program
;  1611          1834  2
;  1612          1835  2 ! If the caller so requested, information about the image just activated is
;  1613          1836  2 ! returned in a 512-byte buffer whose address was passed as an input parameter.
;  1614          1837  2 !
;  1615          1838  2 ! The two main pieces of information are
;  1616          1839  2 !
;  1617          1840  2 !    the image header except for all of the ISDs
;  1618          1841  2 !
;  1619          1842  2 !    an image file descriptor whose primary piece is a string and descriptor
;  1620          1843  2 !    for the image file just activated
;  1621          1844  2 !
;  1622          1845  2 ! Note that the buffer must be probed again because one of the mapping requests
;  1623          1846  2 ! issued by the image activator may have changed its protection.
;  1624          1847  2
;  1625          1848  2 IF .OWN_STORAGE [BUFFER_ADDRESS] NEQA 0
;  1626          1849  2 THEN
;  1627          1850  3     BEGIN
;  1628          1851  3
;  1629          1852  3     ! Because this buffer contains a variety of information, we need to look
;  1630          1853  3     ! at it in several different ways.
;  1631          1854  3
;  1632          1855  3     BIND
;  1633          1856  3         BUFFER = .OWN_STORAGE [BUFFER_ADDRESS]  : VECTOR [RETURN_BUFFER_SIZE,BYTE],
;  1634          1857  3         BUFHDR = .OWN_STORAGE [BUFFER_ADDRESS]  : VECTOR [3,LONG],
;  1635          1858  3         NAM    = PRIMARY_NAM                    : $BBLOCK;
;  1636          1859  3
;  1637          1860  3     LOCAL
;  1638          1861  3         IFD             : REF $BBLOCK,
;  1639          1862  3         FILE_NAME       : REF VECTOR [,BYTE];
```

```
;  1640          1863  3
;  1641          1864  3      ! Insure that buffer is still writable by caller
;  1642          1865  3
;  1643          1866  3      IF NOT PROBEW (%REF(0), %REF(RETURN_BUFFER_SIZE), .OWN_STORAGE [BUFFER_ADDRESS])
;  1644          1867  3      THEN RETURN SS$_ACCVIO;
;  1645          1868  3
;  1646          1869  3      ! Insure that image header information will fit into a single page
;  1647          1870  3
;  1648          1871  4      IF (
;  1649          1872  4          12 +                                ! Three longword header
;  1650          1873  4          .IHD [IHD$W_SIZE]                   ! Image header less ISDs
;  1651          1874  4          )
;  1652          1875  3          LEQU RETURN_BUFFER_SIZE
;  1653          1876  3      THEN                                                                                                    ; R
;  1654          1877  4          BEGIN
;  1655          1878  4
;  1656          1879  4          ! Fill in the three pointers
;  1657          1880  4
;  1658          1881  4          BUFHDR [0] = BUFFER [12];               ! Image header starts right after three longword pointers
;  1659          1882  4          BUFHDR [1] = 0;                         ! 0 until we see if IFD fits
;  1660          1883  4          BUFHDR [2] = 0;                         ! Third longword is unused on success path
;  1661          1884  4
;  1662          1885  4          ! Copy the image header
;  1663          1886  4
;  1664          1887  4          CH$MOVE (.IHD [IHD$W_SIZE], IHD, BUFFER [12]);
;  1665          1888  4
;  1666          1889  4          ! Any transfer address array elements that are not located in the
;  1667          1890  4          ! permanent portion of P1 space or in system space must be relocated
;  1668          1891  4          ! by an amount equal to the base address of the image just mapped.
;  1669          1892  4
;  1670          1893  4          IF
;  1671          1894  5              (.OWN_STORAGE [TRANSFER_ARRAY_BIAS] NEQ 0)
;  1672          1895  4              AND
;  1673          1896  5              (NOT .IHD [IHD$V_LNKNOTFR])
;  1674          1897  4          THEN
;  1675          1898  5              BEGIN
;  1676          1899  5
;  1677          1900  5              BIND
;  1678          1901  5                  IHA = .BUFHDR [0] + .IHD [IHD$W_ACTIVOFF]       : VECTOR [3];
;  1679          1902  5
;  1680          1903  5              LOCAL
;  1681          1904  5                  I;
;  1682          1905  5
;  1683          1906  5              INCR I FROM 0 TO 2 DO
;  1684          1907  5                  IF .IHA [.I] LSSU .CTL$GL_CTLBASVA
;  1685          1908  5                  THEN
;  1686          1909  5                      IHA [.I] = .IHA [.I] + .OWN_STORAGE [TRANSFER_ARRAY_BIAS];
;  1687          1910  4              END;
;  1688          1911  4
;  1689          1912  4          ! See if we can also fit in the IFD
;  1690          1913  4
;  1691          1914  5          IF (
;  1692          1915  5              12 +                            ! Three longword header
;  1693          1916  5              .IHD [IHD$W_SIZE] +             ! Image header less ISDs
;  1694          1917  5              IFD$K_LENGTH +                  ! Fixed portion of IFD
;  1695          1918  5              .NAM [NAM$B_RSL] +              ! Length of image file name
;  1696          1919  5              1 )                             ! Count byte in ASCIC string
```

```
;  1697          1920  4              LEQU RETURN_BUFFER_SIZE
;  1698          1921  4          THEN
;  1699          1922  5              BEGIN
;  1700          1923  5              IFD = BUFFER [12 + .IHD [IHD$W_SIZE]];      ! IFD follows the image header
;  1701          1924  5              BUFHDR [1] = .IFD;                          ! Store its address
;  1702          1925  5
;  1703          1926  5              ! Fill in the fixed portion of the IFD
;  1704          1927  5
;  1705          1928  5              IFD [IFD$W_SIZE] = IFD$K_LENGTH + .NAM [NAM$B_RSL];
;  1706          1929  5              IFD [IFD$W_FILNAMOFF] = IFD$K_LENGTH;
;  1707          1930  5              IFD [IFD$W_CHAN] = .ICB [ICB$W_CHAN];
;  1708          1931  5
;  1709          1932  5              IFD [IFD$W_CMCHAN] = .OWN_STORAGE [OTHER_CHANNEL];
;  1710          1933  5              IFD [IFD$L_CMKFIADR] = .OWN_STORAGE [OTHER_KFE_ADDRESS];
;  1711          1934  5
;  1712          1935  5              IFD [IFD$W_FLAGS] = .IAC$GL_IMAGCTX;
;  1713          1936  5
;  1714          1937  5              FILE_NAME = .IFD + IFD$K_LENGTH;
;  1715          1938  5
;  1716          1939  6              BEGIN
;  1717          1940  6
;  1718          1941  6              BIND
;  1719          1942  6                  FILE_NAME_DESC = IFD [IFD$Q_CURPROG]    : $BBLOCK;
;  1720          1943  6
;  1721          1944  6              FILE_NAME_DESC [DSC$W_LENGTH] = .NAM [NAM$B_RSL];
;  1722          1945  6              FILE_NAME_DESC [DSC$A_POINTER] = .FILE_NAME + 1;
;  1723          1946  6
;  1724          1947  5              END;
;  1725          1948  5
;  1726          1949  5              FILE_NAME [0] = .NAM [NAM$B_RSL];
;  1727          1950  5              CH$MOVE (
;  1728          1951  5                  .NAM [NAM$B_RSL],
;  1729          1952  5                  RESULT_NAME,
;  1730          1953  5                  FILE_NAME [1]);
;  1731          1954  5
;  1732          1955  5              END
;  1733          1956  5
;  1734          1957  4          ELSE
;  1735          1958  4                                                          ; R
;  1736          1959  4              ! Fill the IFD portion with zeros
;  1737          1960  4
;  1738          1961  4              CH$FILL (0, RETURN_BUFFER_SIZE - 12 - .IHD [IHD$W_SIZE],
;  1739          1962  4                       BUFFER [12 + .IHD [IHD$W_SIZE]] );
;  1740          1963  4          END
;  1741          1964  4
;  1742          1965  3      ELSE
;  1743          1966  3
;  1744          1967  3          !!! FOR LACK OF ANYTHING BETTER TO DO IN THIS CASE, I WILL FILL
;  1745          1968  3          !!! THE ENTIRE 512-BYTE BUFFER WITH ZEROS
;  1746          1969  3
;  1747          1970  3          CH$FILL (0, RETURN_BUFFER_SIZE, .OWN_STORAGE [BUFFER_ADDRESS]);
;  1748          1971  3
;  1749          1972  2      END;
;  1750          1973  2
;  1751          1974  2 ! If the caller so requested, the address range into which the image and all
;  1752          1975  2 ! of its associated shareable images were mapped is returned. Like the previous
;  1753          1976  2 ! buffer, this address range must be probed in case the protection on the
```

```
; 1754          1977  2 ! pages changed since the argument was first validated.
; 1755          1978  2
; 1756          1979  2 IF .OWN_STORAGE [RETURN_ARRAY_ADDRESS] NEQA 0
; 1757          1980  2 THEN
; 1758          1981  3     BEGIN
; 1759          1982  3
; 1760          1983  3     BIND
; 1761          1984  3         RETADR = .OWN_STORAGE [RETURN_ARRAY_ADDRESS]    : VECTOR;
; 1762          1985  3
; 1763          1986  3     ! Insure that two longwords are still writable
; 1764          1987  3
; 1765          1988  3     IF NOT PROBEW (%REF(0), %REF(8), .OWN_STORAGE [RETURN_ARRAY_ADDRESS])
; 1766          1989  3     THEN RETURN SS$_ACCVIO;
; 1767          1990  3
; 1768          1991  3     ! Return the start and end addresses
; 1769          1992  3
; 1770          1993  3     RETADR [0] = .OWN_STORAGE [RETURN_START_ADDRESS];
; 1771          1994  3     RETADR [1] = .OWN_STORAGE [RETURN_END_ADDRESS];
; 1772          1995  3
; 1773          1996  2     END;
; 1774          1997  2
; 1775          1998  2 ! The following kernel mode routine executes unconditionally to perform those
; 1776          1999  2 ! completion chores that must be executed in kernel mode.
; 1777          2000  2
; 1778          2001  2 ARG_LIST [1] = .STACK_BASE;
; 1779          2002  2 ARG_LIST [2] = IHD;
; 1780          2003  2 ARG_LIST [3] = KFE;
; 1781          2004  2
; 1782        P 2005  2 STATUS = $CMKRNL (
; 1783        P 2006  2     ROUTIN = SET_CONTROL_REGION,
; 1784          2007  2     ARGLST = ARG_LIST);
; 1785          2008  2 IF NOT .STATUS
; 1786          2009  2 THEN RETURN .STATUS;
; 1787          2010  2
; 1788          2011  2 RETURN SS$_NORMAL
; 1789          2012  2
; 1790          2013  1 END;


                                      00000003  006A8 P.AAC:  .LONG   3                                   ;
               00000D01  00000000  00000000  006AC           .LONG   0, 0, 3329                          ;

                                                             .EXTRN  SYS$EXPREG

                                          0FFC  00000 END_PROCESSING:
                                                             .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 ; 1694
                        5E            24  C2 00002           SUBL2   #36, SP                              ;
      0C     AE      E7 AF            10  28 00005           MOVC3   #16, P.AAC, ARG_LIST                 ; 1734
                     10 AE        1C  AE  9E 0000B           MOVAB   CRETVA_RANGE, ARG_LIST+4             ; 1724
                                  04  AE  7C 00010           CLRQ    IMGIO_SEG_DESC                       ; 1734
                        59        04  AC  D0 00013           MOVL    ICB_ADDRESS, R9                      ; 1739
                        51 00000000G  00  D0 00017           MOVL    CTL$GL_PHD, R1                       ; 1741
                                  50  A9  D5 0001E           TSTL    80(R9)                               ; 1749
                                      0A  12 00021           BNEQ    1$                                   ;
                        50        58  A9  D0 00023           MOVL    88(R9), R0                           ; 1754
                        58        04  A0  D0 00027           MOVL    4(R0), R8                            ; 1756
```

```
                                         04  11 0002B          BRB     2$                                         ;
                     58          50      A9  D0 0002D  1$:     MOVL    80(R9), R8                                 ; 1760
                     03  00000000G       00  E8 00031  2$:     BLBS    OWN_STORAGE, 3$                            ; 1765
                                       00BA  31 00038          BRW     11$                                        ;
                                 1E      A8  B5 0003B  3$:     TSTW    30(R8)                                     ; 1769
                                         06  13 0003E          BEQL    4$                                         ;
                     50          1E      A8  3C 00040          MOVZWL  30(R8), IMAGE_IO_PAGE_COUNT                ; 1771
                                         07  11 00044          BRB     5$                                         ;
                     50  00000000G       00  3C 00046  4$:     MOVZWL  SGN$GW_IMGIOCNT, IMAGE_IO_PAGE_COUNT       ; 1773
 50 00000000G    00  10                  00  ED 0004D  5$:     CMPZV   #0, #16, SGN$GW_IMGIOCNT, -                ; 1775
                                                                       IMAGE_IO_PAGE_COUNT                        ;
                                         4C  18 00056          BGEQ    8$                                         ;
             52          50              09  78 00058          ASHL    #9, IMAGE_IO_PAGE_COUNT, R2                ; 1784
             11     20   A8              04  E1 0005C          BBC     #4, 32(R8), 6$                             ; 1778
                    1C   AE      28      A1  D0 00061          MOVL    40(R1), CRETVA_RANGE                       ; 1781
             50          52      1C      AE  C1 00066          ADDL3   CRETVA_RANGE, R2, R0                       ; 1784
                    20   AE      FF      A0  9E 0006B          MOVAB   -1(R0), CRETVA_RANGE+4                     ; 1785
                                         13  11 00070          BRB     7$                                         ; 1778
        20   AE 00000000G  00            01  C3 00072  6$:     SUBL3   #1, CTL$GL_CTLBASVA, CRETVA_RANGE+4        ; 1789
             50     20   AE              52  C3 0007B          SUBL3   R2, CRETVA_RANGE+4, R0                     ; 1792
                    1C   AE      01      A0  9E 00080          MOVAB   1(R0), CRETVA_RANGE                        ;
                                 0C      AE  9F 00085  7$:     PUSHAB  ARG_LIST                                   ; 1800
                         00000000G       00  9F 00088          PUSHAB  MMG$CRETVA                                 ;
                00000000G  00            02  FB 0008E          CALLS   #2, SYS$CMKRNL                             ;
                           5B            50  D0 00095          MOVL    R0, STATUS                                 ;
                           54            5B  E9 00098          BLBC    STATUS, 10$                                ; 1801
                    04   AE              52  D0 0009B          MOVL    R2, IMGIO_SEG_DESC                         ; 1804
                    08   AE      1C      AE  D0 0009F          MOVL    CRETVA_RANGE, IMGIO_SEG_DESC+4             ; 1805
                         51      04      AE  9E 000A4  8$:     MOVAB   IMGIO_SEG_DESC, R1                         ; 1809
                         50      20      A8  D0 000A8          MOVL    32(R8), R0                                 ;
                         00000000G       00  16 000AC          JSB     RM$SET                                     ;
                     50  00000000G       00  D0 000B2          MOVL    OWN_STORAGE+28, R0                         ; 1818
             19     20   A8              04  E1 000B9          BBC     #4, 32(R8), 9$                             ; 1811
                         7E              03  7D 000BE          MOVQ    #3, -(SP)                                  ; 1818
                                 24      AE  9F 000C1          PUSHAB  STACK_ARRAY                                ;
                                         50  DD 000C4          PUSHL   R0                                         ;
                00000000G  00            04  FB 000C6          CALLS   #4, SYS$EXPREG                             ;
                           5B            50  D0 000CD          MOVL    R0, STATUS                                 ;
             6E     20   AE              01  C1 000D0          ADDL3   #1, STACK_ARRAY+4, STACK_BASE              ; 1819
                                         18  11 000D5          BRB     10$                                        ; 1811
                                         01  DD 000D7  9$:     PUSHL   #1                                         ; 1827
                                         03  DD 000D9          PUSHL   #3                                         ;
                                 24      AE  9F 000DB          PUSHAB  STACK_ARRAY                                ;
                                         50  DD 000DE          PUSHL   R0                                         ;
                00000000G  00            04  FB 000E0          CALLS   #4, SYS$EXPREG                             ;
                           5B            50  D0 000E7          MOVL    R0, STATUS                                 ;
             6E     1C   AE              01  C1 000EA          ADDL3   #1, STACK_ARRAY, STACK_BASE                ; 1828
                           03            5B  E8 000EF  10$:    BLBS    STATUS, 11$                                ; 1831
                                       0112  31 000F2          BRW     24$                                        ;
                     56  00000000G       00  D0 000F5  11$:    MOVL    OWN_STORAGE+64, R6                         ; 1848
                                         03  12 000FC          BNEQ    12$                                        ;
                                       00CB  31 000FE          BRW     20$                                        ;
             66   0200   8F              00  0D 00101  12$:    PROBEW  #0, #512, (R6)                             ; 1866
                                         03  12 00107          BNEQ    13$                                        ;
                                       00CF  31 00109          BRW     21$                                        ;
                         57              68  3C 0010C  13$:    MOVZWL  (R8), R7                                   ; 1873
                         5A      0C      A7  9E 0010F          MOVAB   12(R7), R10                                ; 1871
```

```
                  00000200  8F          5A  D1 00113        CMPL    R10, #512                               ; 1875
                                        03  1B 0011A        BLEQU   14$                                     ;
                                      00A5  31 0011C        BRW     19$                                     ;
                            66      0C  A6  9E 0011F  14$:  MOVAB   12(R6), (R6)                            ; 1881
                                    04  A6  7C 00123        CLRQ    4(R6)                                   ; 1882
          0C    A6          68          57  28 00126        MOVC3   R7, (R8), 12(R6)                        ; 1887
                            52 00000000G 00 D0 0012B        MOVL    OWN_STORAGE+24, R2                      ; 1894
                                        20  13 00132        BEQL    17$                                     ;
          1B         20     A8          01  E0 00134        BBS     #1, 32(R8), 17$                         ; 1896
                            51      02  A8  3C 00139        MOVZWL  2(R8), R1                               ; 1901
                            51          66  C0 0013D        ADDL2   (R6), R1                                ;
                                        50  D4 00140        CLRL    I                                       ; 1907
                  00000000G 00        6140  D1 00142  15$:  CMPL    (R1)[I], CTL$GL_CTLBASVA                ;
                                        04  1E 0014A        BGEQU   16$                                     ;
                         6140           52  C0 0014C        ADDL2   R2, (R1)[I]                             ; 1909
                EE          50          02  F3 00150  16$:  AOBLEQ  #2, I, 15$                              ; 1907
                            52 00000000G 00 9A 00154  17$:  MOVZBL  NAM+3, R2                               ; 1918
                            50      29 A247 9E 0015B        MOVAB   41(R2)[R7], R0                          ; 1914
                  00000200  8F          50  D1 00160        CMPL    R0, #512                                ; 1920
                                        4A  1A 00167        BGTRU   18$                                     ;
                50          56          5A  C1 00169        ADDL3   R10, R6, IFD                            ; 1923
                       04   A6          50  D0 0016D        MOVL    IFD, 4(R6)                              ; 1924
                60          52          1C  A1 00171        ADDW3   #28, R2, (IFD)                          ; 1928
                       02   A0          1C  B0 00175        MOVW    #28, 2(IFD)                             ; 1929
                       08   A0      0E  A9  B0 00179        MOVW    14(R9), 8(IFD)                          ; 1930
                       0A   A0 00000000G 00 B0 0017E        MOVW    OWN_STORAGE+14, 10(IFD)                 ; 1932
                       0C   A0 00000000G 00 D0 00186        MOVL    OWN_STORAGE+20, 12(IFD)                 ; 1933
                       10   A0 00000000G 00 B0 0018E        MOVW    IAC$GL_IMAGCTX, 16(IFD)                 ; 1935
                            51      1C  A0  9E 00196        MOVAB   28(R0), FILE_NAME                       ; 1937
                            50          14  C0 0019A        ADDL2   #20, R0                                 ; 1942
                            60          52  B0 0019D        MOVW    R2, (R0)                                ; 1944
                       04   A0      01  A1  9E 001A0        MOVAB   1(FILE_NAME), 4(R0)                     ; 1945
                            61          52  90 001A5        MOVB    R2, (FILE_NAME)                         ; 1949
          01    A1 00000000G 00         52  28 001A8        MOVC3   R2, RESULT_NAME, 1(FILE_NAME)           ; 1953
                                        19  11 001B1        BRB     20$                                     ; 1914
                57 000001F4 8F          57  C3 001B3  18$:  SUBL3   R7, #500, R7                            ; 1961
      57        00          6E          00  2C 001BB        MOVC5   #0, (SP), #0, R7, (R10)[R6]             ; 1962
                                      6A46     001C0                                                    ;
                                        08  11 001C2        BRB     20$                                     ; 1871
0200  8F        00          6E          00  2C 001C4  19$:  MOVC5   #0, (SP), #0, #512, (R6)                ; 1970
                                        66     001CB                                                    ;
                            50 00000000G 00 D0 001CC  20$:  MOVL    OWN_STORAGE+80, R0                      ; 1979
                                        11  13 001D3        BEQL    23$                                     ;
                60          08          00  0D 001D5        PROBEW  #0, #8, (R0)                            ; 1988
                                        04  12 001D9        BNEQ    22$                                     ;
                            50          0C  D0 001DB  21$:  MOVL    #12, R0                                 ; 1989
                                            04 001DE        RET                                             ;
                            60 00000000G 00 7D 001DF  22$:  MOVQ    OWN_STORAGE+84, (R0)                    ; 1993
                       10   AE          6E  D0 001E6  23$:  MOVL    STACK_BASE, ARG_LIST+4                  ; 2001
                       14   AE          58  D0 001EA        MOVL    R8, ARG_LIST+8                          ; 2002
                       18   AE      54  A9  D0 001EE        MOVL    84(R9), ARG_LIST+12                     ; 2003
                                    0C  AE  9F 001F3        PUSHAB  ARG_LIST                                ; 2007
                                 0000V  CF  9F 001F6        PUSHAB  SET_CONTROL_REGION                      ;
                  00000000G 00          02  FB 001FA        CALLS   #2, SYS$CMKRNL                          ;
                            5B          50  D0 00201        MOVL    R0, STATUS                              ;
                            04          5B  E8 00204        BLBS    STATUS, 25$                             ; 2008
                            50          5B  D0 00207  24$:  MOVL    STATUS, R0                              ; 2009
```

```
                                    04 0020A                RET
                          50        01 D0 0020B 25$:        MOVL    #1, R0                          ;
                                       04 0020E             RET                                     ; 2011
                                                                                                    ; 2013

; Routine Size:  527 bytes,    Routine Base:  YF$$SYSIMGACT + 06B8
```

```
;  1792          2014  1 %SBTTL 'SET_CONTROL_REGION - Kernel Mode Completion Routine'
;  1793          2015  1
;  1794          2016  1 ROUTINE SET_CONTROL_REGION (
;  1795          2017  1     USER_STACK_ADDRESS,
;  1796          2018  1     IHD_POINTER,
;  1797          2019  1     KFE_POINTER) : SYS_CMKRNL =
;  1798          2020  1
;  1799          2021  1 !+
;  1800          2022  1 ! Functional Description:
;  1801          2023  1 !
;  1802          2024  1 !     This routine performs the end processing that must be done in kernel
;  1803          2025  1 !     mode. This includes loading the user stack pointer processor register,
;  1804          2026  1 !     and setting up the process privilege mask. Note that these steps are
;  1805          2027  1 !     only taken during the activation of a main program.
;  1806          2028  1 !
;  1807          2029  1 ! Calling Sequence:
;  1808          2030  1 !
;  1809          2031  1 !     $CMKRNL (SET_CONTROL_REGION, ARGUMENT_LIST)
;  1810          2032  1 !
;  1811          2033  1 ! Formal Parameters:
;  1812          2034  1 !
;  1813          2035  1 !     USER_STACK_ADDRESS - Address of base (high address end) of user stack
;  1814          2036  1 !
;  1815          2037  1 !     IHD_POINTER - Pointer to image header for this image
;  1816          2038  1 !
;  1817          2039  1 !     KFE_POINTER - Pointer to known file entry for this image, if one exists
;  1818          2040  1 !-
;  1819          2041  1
;  1820          2042  2 BEGIN
;  1821          2043  2
;  1822          2044  2 MACRO
;  1823          2045  2
;  1824        M 2046  2     MOVE_PRIV_MASK (SRC, DST) =
;  1825        M 2047  2
;  1826        M 2048  2         BEGIN
;  1827        M 2049  2         VECTOR [DST,0] = .VECTOR [SRC,0];
;  1828        M 2050  2         VECTOR [DST,1] = .VECTOR [SRC,1];
;  1829          2051  2         END% ,
;  1830          2052  2
;  1831        M 2053  2     CLEAR_PRIV_MASK (DST) =
;  1832        M 2054  2
;  1833        M 2055  2         BEGIN
;  1834        M 2056  2         VECTOR [DST,0] = 0;
;  1835        M 2057  2         VECTOR [DST,1] = 0;
;  1836          2058  2         END% ,
;  1837          2059  2
;  1838        M 2060  2     EXCLUDE_PRIV_MASK (MASK, DST) =
;  1839        M 2061  2
;  1840        M 2062  2         BEGIN
;  1841        M 2063  2         VECTOR [DST,0] = .VECTOR [MASK,0] AND .VECTOR [DST,0];
;  1842        M 2064  2         VECTOR [DST,1] = .VECTOR [MASK,1] AND .VECTOR [DST,1];
;  1843          2065  2         END% ,
;  1844          2066  2
;  1845        M 2067  2     INCLUDE_PRIV_MASK (MASK, DST) =
;  1846        M 2068  2
;  1847        M 2069  2         BEGIN
;  1848        M 2070  2         VECTOR [DST,0] = .VECTOR [MASK,0] OR .VECTOR [DST,0];
```

```
;  1849        M 2071  2          VECTOR [DST,1] = .VECTOR [MASK,1] OR .VECTOR [DST,1];
;  1850          2072  2          END% ;
;  1851          2073  2
;  1852          2074  2  EXTERNAL REGISTER                          ! We enter this procedure with the PCB
;  1853          2075  2      PCB = 4 : REF $BBLOCK;                 !  address contained in R4
;  1854          2076  2
;  1855          2077  2  BIND
;  1856          2078  2      IHD = .IHD_POINTER  : $BBLOCK,
;  1857          2079  2      KFE = .KFE_POINTER  : $BBLOCK,
;  1858          2080  2      PHD = .CTL$GL_PHD   : $BBLOCK;
;  1859          2081  2
;  1860          2082  2  LITERAL
;  1861          2083  3      CMKRNL_OR_CMEXEC = (1 ^ $BITPOSITION (PRV$V_CMKRNL))
;  1862          2084  2                         OR
;  1863          2085  2                         (1 ^ $BITPOSITION (PRV$V_CMEXEC));
;  1864          2086  2
;  1865          2087  2  LOCAL
;  1866          2088  2      PRIVILEGES  : VECTOR [2],
;  1867          2089  2      ICB         : REF $BBLOCK;
;  1868          2090  2
;  1869          2091  2  ! Most of the operations in this routine are only performed when activating
;  1870          2092  2  ! a main program. The only step that must be performed during a merged
;  1871          2093  2  ! activation is the USECNT adjustment for the KFEs.
;  1872          2094  2
;  1873          2095  2  IF .OWN_STORAGE [MAIN_PROGRAM]
;  1874          2096  2  THEN
;  1875          2097  3      BEGIN
;  1876          2098  3
;  1877          2099  3      ! The high address end of the user stack is loaded into the stack limit
;  1878          2100  3      ! array. The user stack pointer is initialized with a value that is smaller
;  1879          2101  3      ! than the input value by a value given by the EXTRA_USER_STACK compile
;  1880          2102  3      ! time constant. The size of the user stack is stored ina  cell that will
;  1881          2103  3      ! be used by the automatic stack expansion logic in EXCEPTION. Note that
;  1882          2104  3      ! this number can never be smaller than 2.
;  1883          2105  3
;  1884          2106  3      CTL$AL_STACK [PSL$C_USER] = .USER_STACK_ADDRESS;
;  1885          2107  3      MTPR (%REF (.USER_STACK_ADDRESS - (EXTRA_USER_STACK*BYTES_PER_PAGE))
;  1886          2108  3          , PR$_USP);
;  1887          2109  3      IAC$GL_STACK_SIZE = .OWN_STORAGE [USER_STACK_SIZE];
;  1888          2110  3
;  1889          2111  3      ! The privilege mask that will be used while this image is executing must                    ; R
;  1890          2112  3      ! be fabricated.
;  1891          2113  3
;  1892          2114  3      MOVE_PRIV_MASK (CTL$GQ_PROCPRIV, PRIVILEGES);       ! Start with process  privileges
;  1893          2115  3
;  1894          2116  3      ! Eliminate those not present in the image header
;  1895          2117  3
;  1896          2118  3      EXCLUDE_PRIV_MASK (IHD [IHD$Q_PRIVREQS], PRIVILEGES);
;  1897          2119  3
;  1898          2120  3      ! If the image was installed with privilege and we were called from other than
;  1899          2121  3      ! user mode, then add the privileges from the KFE
;  1900          2122  3
;  1901          2123  3      IF .OWN_STORAGE [CALL_MODE] NEQ PSL$C_USER
;  1902          2124  3          AND
;  1903          2125  4          BEGIN
;  1904          2126  4          IF .KFE_POINTER EQL 0
;  1905          2127  4          THEN FALSE
```

```
;  1906          2128  4          ELSE .KFE [KFE$V_PROCPRIV]
;  1907          2129  4          END
;  1908          2130  3      THEN
;  1909          2131  4          BEGIN
;  1910          2132  4          INCLUDE_PRIV_MASK (KFE [KFE$Q_PROCPRIV], PRIVILEGES);
;  1911          2133  4          MOVE_PRIV_MASK (KFE [KFE$Q_PROCPRIV], PHD [PHD$Q_IMAGPRIV]);
;  1912          2134  4          END
;  1913          2135  3      ELSE
;  1914          2136  3          CLEAR_PRIV_MASK (PHD [PHD$Q_IMAGPRIV]);
;  1915          2137  3
;  1916          2138  3      ! Before the final privileges get stored, we need to check whether there
;  1917          2139  3      ! is a system version mismatch in any of the images that was mapped. (This
;  1918          2140  3      ! mismatch was detected by the image header decode routine and remembered
;  1919          2141  3      ! in the REMOVE_PRIVILEGE flag.) If a mismatch was detected, and either
;  1920          2142  3      ! CMEXEC or CMKRNL is set, then CMKRNL and CMEXEC privileges are turned
;  1921          2143  3      ! off and an alternate status (SS$_SYSVERDIF) is returned.
;  1922          2144  3
;  1923          2145  4      IF  (.OWN_STORAGE [REMOVE_PRIVILEGE])
;  1924          2146  3          AND
;  1925          2147  4          ((.PRIVILEGES [0] AND CMKRNL_OR_CMEXEC) NEQU 0)
;  1926          2148  3      THEN
;  1927          2149  4          BEGIN
;  1928          2150  4          PRIVILEGES [0] = .PRIVILEGES [0] AND (NOT CMKRNL_OR_CMEXEC);
;  1929          2151  4          OWN_STORAGE [FINAL_STATUS] = SS$_SYSVERDIF;
;  1930          2152  3          END;
;  1931          2153  3
;  1932          2154  3      ! Store the privileges in the process header and in the PCB
;  1933          2155  3
;  1934          2156  3      MOVE_PRIV_MASK (PRIVILEGES, PCB [PCB$Q_PRIV]);
;  1935          2157  3      MOVE_PRIV_MASK (PRIVILEGES, PHD [PHD$Q_PRIVMSK]);
;  1936          2158  3
;  1937          2159  3      ! The address of the image header buffer must be stored in the pointer page
;  1938          2160  3
;  1939          2161  3      CTL$GL_IMGHDRBF = .OWN_STORAGE [BUFFER_ADDRESS];
;  1940          2162  3
;  1941          2163  3      ! Finally, if image accounting was requested for this image, then the various
;  1942          2164  3      ! image accounting cells must be initialized.
;  1943          2165  3
;  1944          2166  3      IF .OWN_STORAGE [IMAGE_ACCOUNT] OR .EXE$GL_ACMFLAGS [ACM$V_IMAGE]
;  1945          2167  3      THEN
;  1946          2168  4          BEGIN
;  1947          2169  4          CTL$GL_ICPUTIM = .PHD [PHD$L_CPUTIM];
;  1948          2170  4          CTL$GL_IFAULTS = .PHD [PHD$L_PAGEFLTS];
;  1949          2171  4          CTL$GL_IFAULTIO = .PHD [PHD$L_PGFLTIO];
;  1950          2172  4          CTL$GL_IWSPEAK = 0;
;  1951          2173  4          CTL$GL_IPAGEFL = 0;
;  1952          2174  4          CTL$GL_IDIOCNT = .PHD [PHD$L_DIOCNT];
;  1953          2175  4          CTL$GL_IBIOCNT = .PHD [PHD$L_BIOCNT];
;  1954          2176  4          CTL$GL_IVOLUMES = .CTL$GL_VOLUMES;
;  1955          2177  4          CTL$GQ_ISTART [0] = .EXE$GQ_SYSTIME [0];
;  1956          2178  4          CTL$GQ_ISTART [1] = .EXE$GQ_SYSTIME [1];
;  1957          2179  3          END;
;  1958          2180  3
;  1959          2181  2      END;                                    ! End of test for main program
;  1960          2182  2
;  1961          2183  2 ! The USECNT cell in each KFE must be incremented. In addition, any shared WCB
;  1962          2184  2 ! must have its REFCNT incremented. If the adjusted REFCNT is larger than the
```

```
2 ! current high water REFCNT in the KFE, the KFE cell is adjusted. Finally, the
2 ! DONE bit in the ICB is turned ON, indicating that the activation for each of
2 ! these ICBs is complete.
2
2      ICB = .IAC$GL_IMAGE_LIST;
2      DO
3          BEGIN
3
3          LOCAL
3              KFE : REF $BBLOCK,
3              WCB : REF $BBLOCK;
3
3          IF NOT .ICB [ICB$V_DONE]
3          THEN
4              BEGIN
4              ICB [ICB$V_DONE] = TRUE;
4              KFE = .ICB [ICB$L_KFE];
4              IF .KFE NEQU 0
4              THEN
5                  BEGIN
5                  KFE [KFE$L_USECNT] = .KFE [KFE$L_USECNT] + 1;
5                  IF .KFE [KFE$V_OPEN]
5                  THEN
6                      BEGIN
6                      WCB = .KFE [KFE$L_WCB];
6                      IF .WCB [WCB$W_REFCNT] GTRU .KFE [KFE$W_SHRCNT]
6                      THEN KFE [KFE$Q_SHRCNT] = .WCB [WCB$W_REFCNT];
5                      END;
4                  END;
3              END;
3          ICB = .ICB [ICB$L_FLINK];
3          END
2      UNTIL .ICB EQLA IAC$GL_IMAGE_LIST;
2
2 IF .OWN_STORAGE [RMS_BASE] NEQU 0
2 THEN CTL$GL_RMSBASE = .OWN_STORAGE [RMS_BASE];
2
2 RETURN SS$_NORMAL
2
1 END;
```

```
                                        006C 00000 SET_CONTROL_REGION:
                                                        .WORD   Save R2,R3,R5,R6                 ; 2016
                 56 00000000G   00   9E 00002           MOVAB   IAC$GL_IMAGE_LIST, R6            ;
                 55 00000000G   00   9E 00009           MOVAB   OWN_STORAGE, R5                  ;
                 5E             08   C2 00010           SUBL2   #8, SP                           ;
                 53       08    AC   D0 00013           MOVL    IHD_POINTER, R3                  ; 2078
                 52       0C    AC   D0 00017           MOVL    KFE_POINTER, R2                  ; 2079
                 51 00000000G   00   D0 0001B           MOVL    CTL$GL_PHD, R1                   ; 2080
                 03             65   E8 00022           BLBS    OWN_STORAGE, 1$                  ; 2095
                              00DE   31 00025           BRW     6$                               ;
     00000000G   00       04    AC   D0 00028 1$:       MOVL    USER_STACK_ADDRESS, CTL$AL_STACK+12 ; 2106
                 50       04    AC   D0 00030           MOVL    USER_STACK_ADDRESS, R0           ; 2107
```

```
                                50      FC00    C0  9E 00034          MOVAB   -1024(R0), R0                           ;
                                03              50  DA 00039          MTPR    R0, #3                                  ;
                    00000000G   00        1C    A5  D0 0003C          MOVL    OWN_STORAGE+28, IAC$GL_STACK_SIZE       ; 2109
                                6E 00000000G    00  7D 00044          MOVQ    CTL$GQ_PROCPRIV, PRIVILEGES             ; 2114
                                50        14    A3  9E 0004B          MOVAB   20(R3), R0                              ; 2118
                                53              60  D2 0004F          MCOML   (R0), R3                                ;
                                6E              53  CA 00052          BICL2   R3, PRIVILEGES                          ;
                                53        04    A0  D2 00055          MCOML   4(R0), R3                               ;
                          04    AE              53  CA 00059          BICL2   R3, PRIVILEGES+4                        ;
                                03        24    A5  91 0005D          CMPB    OWN_STORAGE+36, #3                      ; 2123
                                                20  13 00061          BEQL    2$                                      ;
                                          0C    AC  D5 00063          TSTL    KFE_POINTER                             ; 2126
                                                1B  13 00066          BEQL    2$                                      ;
                      16        10    A2        02  E1 00068          BBC     #2, 16(R2), 2$                          ; 2128
                                50        20    A2  9E 0006D          MOVAB   32(R2), R0                              ; 2132
                                6E              60  C8 00071          BISL2   (R0), PRIVILEGES                        ;
                          04    AE        04    A0  C8 00074          BISL2   4(R0), PRIVILEGES+4                     ;
                                52      00E8    C1  9E 00079          MOVAB   232(R1), R2                             ; 2133
                                62              60  7D 0007E          MOVQ    (R0), (R2)                              ;
                                                07  11 00081          BRB     3$                                      ; 2123
                                50      00E8    C1  9E 00083 2$:      MOVAB   232(R1), R0                             ; 2136
                                                60  7C 00088          CLRQ    (R0)                                    ;
                      0E                65        03  E1 0008A 3$:      BBC     #3, OWN_STORAGE, 4$                     ; 2145
                                03              6E  93 0008E          BITB    PRIVILEGES, #3                          ; 2147
                                                09  13 00091          BEQL    4$                                      ;
                                6E              03  8A 00093          BICB2   #3, PRIVILEGES                          ; 2150
                          10    A5      0671    8F  3C 00096          MOVZWL  #1649, OWN_STORAGE+16                   ; 2151
                                50      0084    C4  9E 0009C 4$:      MOVAB   132(PCB), R0                            ; 2156
                                60              6E  7D 000A1          MOVQ    PRIVILEGES, (R0)                        ;
                                61              6E  7D 000A4          MOVQ    PRIVILEGES, (R1)                        ; 2157
                    00000000G   00        40    A5  D0 000A7          MOVL    OWN_STORAGE+64, CTL$GL_IMGHDRBF         ; 2161
                08              65              02  E0 000AF          BBS     #2, OWN_STORAGE, 5$                     ; 2166
                4B  00000000G   00              01  E1 000B3          BBC     #1, EXE$GL_ACMFLAGS, 6$                 ;
                    00000000G   00        38    A1  D0 000BB 5$:      MOVL    56(R1), CTL$GL_ICPUTIM                  ; 2169
                    00000000G   00        4C    A1  D0 000C3          MOVL    76(R1), CTL$GL_IFAULTS                  ; 2170
                    00000000G   00      0108    C1  D0 000CB          MOVL    264(R1), CTL$GL_IFAULTIO                ; 2171
                                00000000G   00  D4 000D4              CLRL    CTL$GL_IWSPEAK                          ; 2172
                                00000000G   00  D4 000DA              CLRL    CTL$GL_IPAGEFL                          ; 2173
                    00000000G   00        54    A1  D0 000E0          MOVL    84(R1), CTL$GL_IDIOCNT                  ; 2174
                    00000000G   00        58    A1  D0 000E8          MOVL    88(R1), CTL$GL_IBIOCNT                  ; 2175
                    00000000G   00 00000000G    00  D0 000F0          MOVL    CTL$GL_VOLUMES, CTL$GL_IVOLUMES         ; 2176
                    00000000G   00 00000000G    00  7D 000FB          MOVQ    EXE$GQ_SYSTIME, CTL$GQ_ISTART           ; 2177
                                52              66  D0 00106 6$:      MOVL    IAC$GL_IMAGE_LIST, ICB                  ; 2189
                23        10    A2              06  E0 00109 7$:      BBS     #6, 16(ICB), 8$                         ; 2197
                          10    A2        40    8F  88 0010E          BISB2   #64, 16(ICB)                            ; 2200
                                50        54    A2  D0 00113          MOVL    84(ICB), KFE                            ; 2201
                                                18  13 00117          BEQL    8$                                      ; 2202
                                          14    A0  D6 00119          INCL    20(KFE)                                 ; 2205
                10        10    A0              03  E1 0011C          BBC     #3, 16(KFE), 8$                         ; 2206
                                51        18    A0  D0 00121          MOVL    24(KFE), WCB                            ; 2209
                          34    A0        0E    A1  B1 00125          CMPW    14(WCB), 52(KFE)                        ; 2210
                                                05  1B 0012A          BLEQU   8$                                      ;
                          34    A0        0E    A1  B0 0012C          MOVW    14(WCB), 52(KFE)                        ; 2211
                                52              62  D0 00131 8$:      MOVL    (ICB), ICB                              ; 2215
                                50              66  9E 00134          MOVAB   IAC$GL_IMAGE_LIST, R0                   ; 2217
                                50              52  D1 00137          CMPL    ICB, R0                                 ;
                                                CD  12 0013A          BNEQ    7$                                      ;
```

```
                                50       2C    A5  D0 0013C          MOVL    OWN_STORAGE+44, R0                      ; 2219
                                                 07  13 00140          BEQL    9$                                      ;
                     00000000G  00              50  D0 00142          MOVL    R0, CTL$GL_RMSBASE                      ; 2220
                                50              01  D0 00149 9$:      MOVL    #1, R0                                  ; 2222
                                                    04 0014C          RET                                             ; 2224

; Routine Size:  333 bytes,    Routine Base:  YF$$SYSIMGACT + 08C7
```

```
;  2004          2225  1 %SBTTL 'GET_LOCK - Lock Known File Data Base for Read Access'
;  2005          2226  1
;  2006          2227  1 ROUTINE GET_LOCK =
;  2007          2228  1
;  2008          2229  1 !+
;  2009          2230  1 ! Functional Description:
;  2010          2231  1 !
;  2011          2232  1 !     This routine locks the known file data base for read access. The image
;  2012          2233  1 !     activator maintains this lock for the entire time that it will be opening
;  2013          2234  1 !     files. Note that no lock is taken out during system initialization. until
;  2014          2235  1 !     INSTALL first executes and sets up the known file lists (and loads
;  2015          2236  1 !     EXE$GL_KNOWN_FILES with nonzero contents.
;  2016          2237  1 !
;  2017          2238  1 ! Calling Sequence:
;  2018          2239  1 !
;  2019          2240  1 !     GET_LOCK ()
;  2020          2241  1 !-
;  2021          2242  1
;  2022          2243  2 BEGIN
;  2023          2244  2
;  2024          2245  2 LOCAL STATUS;
;  2025          2246  2
;  2026          2247  2 IF .EXE$GL_KNOWN_FILES NEQ 0
;  2027          2248  2 THEN
;  2028          2249  3     BEGIN
;  2029          2250  3
;  2030        P 2251  3     STATUS = $ENQW (
;  2031        P 2252  3         EFN = EXE$C_SYSEFN,
;  2032        P 2253  3         LKMODE = LCK$K_PRMODE,
;  2033        P 2254  3         LKSB = OWN_STORAGE [LOCK_STATUS_BLOCK],
;  2034        P 2255  3         FLAGS = LCK$M_SYSTEM,
;  2035        P 2256  3         RESNAM = EXE$GQ_KFE_LCKNAM,
;  2036        P 2257  3         PARID = .EXE$GL_SYSID_LOCK,
;  2037          2258  3         ACMODE = PSL$C_EXEC);           ! End of routine GET_LOCK
;  2038          2259  3
;  2039          2260  3     IF NOT .STATUS
;  2040          2261  3     THEN RETURN .STATUS
;  2041          2262  3     ELSE RETURN .OWN_STORAGE [LOCK_STATUS];
;  2042          2263  3
;  2043          2264  3     END
;  2044          2265  2 ELSE     ! No need to take out a lock yet
;  2045          2266  2     RETURN SS$_NORMAL
;  2046          2267  2
;  2047          2268  1 END;


                                                .EXTRN  SYS$ENQW

                                0004 00000 GET_LOCK:
                                                .WORD   Save R2                                 ;  2227
                    52 00000000G   00   9E 00002        MOVAB   OWN_STORAGE+4, R2               ;
                       00000000G   00   D5 00009        TSTL    EXE$GL_KNOWN_FILES              ;  2247
                                   2A   13 0000F        BEQL    1$                              ;
                    7E             01   7D 00011        MOVQ    #1, -(SP)                       ;  2258
                                   7E   7C 00014        CLRQ    -(SP)                           ;
                                   7E   D4 00016        CLRL    -(SP)                           ;
```

```
                                    00000000G  00  DD 00018          PUSHL   EXE$GL_SYSID_LOCK                    ;
                                    00000000G  00  9F 0001E          PUSHAB  EXE$GQ_KFE_LCKNAM                   ;
                                               10  DD 00024          PUSHL   #16                                 ;
                                               52  DD 00026          PUSHL   R2                                  ;
                                               03  DD 00028          PUSHL   #3                                  ;
                                7E             00G 9A 0002A          MOVZBL  S^EXE$C_SYSEFN, -(SP)               ;
                     00000000G  00             0B  FB 0002D          CALLS   #11, SYS$ENQW                       ;
                                07             50  E9 00034          BLBC    STATUS, 2$                          ; 2260
                                50             62  3C 00037          MOVZWL  OWN_STORAGE+4, R0                   ; 2262
                                                   04 0003A          RET                                         ; 2266
                                50             01  D0 0003B 1$:      MOVL    #1, R0                              ;
                                                   04 0003E 2$:      RET                                         ; 2268

; Routine Size:  63 bytes,    Routine Base:  YF$$SYSIMGACT + 0A14
```

```
; 2049          2269   1 %SBTTL 'RELEASE_LOCK - Unlock Lock Known File Data Base'
; 2050          2270   1
; 2051          2271   1 ROUTINE RELEASE_LOCK =
; 2052          2272   1
; 2053          2273   1 !+
; 2054          2274   1 ! Functional Description:
; 2055          2275   1 !
; 2056          2276   1 !     This routine unlocks the known file data base. This happens when
; 2057          2277   1 !     the image activator reaches the point where there are no more files
; 2058          2278   1 !     to open, or when an error occurs with the lock granted.
; 2059          2279   1 !
; 2060          2280   1 ! Calling Sequence:
; 2061          2281   1 !
; 2062          2282   1 !     RELEASE_LOCK ()
; 2063          2283   1 !-
; 2064          2284   1
; 2065          2285   1 IF .OWN_STORAGE [LOCK_ID] NEQ 0
; 2066          2286   1 THEN
; 2067          2287   2     $DEQ (LKID = .OWN_STORAGE [LOCK_ID])
; 2068          2288   1 ELSE
; 2069          2289   1     SS$_NORMAL;                        ! End of routine RELEASE_LOCK


                                                        .EXTRN  SYS$DEQ

                                         0000 00000 RELEASE_LOCK:
                                                        .WORD   Save nothing                            ; 2271
                         50 00000000G  00   D0 00002    MOVL    OWN_STORAGE+8, R0                       ; 2285
                                       0E   13 00009    BEQL    1$                                      ;
                                       7E   7C 0000B    CLRQ    -(SP)                                   ; 2287
                                       7E   D4 0000D    CLRL    -(SP)                                   ;
                                       50   DD 0000F    PUSHL   R0                                      ;
                  00000000G  00        04   FB 00011    CALLS   #4, SYS$DEQ                             ;
                                            04 00018    RET                                             ;
                         50            01   D0 00019 1$: MOVL   #1, R0                                  ; 2285
                                            04 0001C    RET                                             ; 2289

; Routine Size:  29 bytes,    Routine Base:  YF$$SYSIMGACT + 0A53
```

```
; 2071          2290  1 %SBTTL 'IMG$ALLOCATE_ICB - Lock Known File Data Base for Read Access'
; 2072          2291  1
; 2073          2292  1 GLOBAL ROUTINE IMG$ALLOCATE_ICB (ICB_POINTER) =
; 2074          2293  1
; 2075          2294  1 !+
; 2076          2295  1 ! Functional Description:
; 2077          2296  1 !
; 2078          2297  1 !    This routine allocates an image control block for use by later stages
; 2079          2298  1 !    of the image activator. An allocation request is first made from a
; 2080          2299  1 !    pool of previously used ICBs. A trip into kernel mode is thus only
; 2081          2300  1 !    required if this lookaside list request fails. In either case, the
; 2082          2301  1 !    ICB is entirely filled with zeros.
; 2083          2302  1 !
; 2084          2303  1 !    Note that only the process allocation region is used to insure that
; 2085          2304  1 !    no ICBs are created in P0 space. This would be no problem on merged
; 2086          2305  1 !    activations but would mess up the simple execution of an image.
; 2087          2306  1 !
; 2088          2307  1 ! Calling Sequence:
; 2089          2308  1 !
; 2090          2309  1 !    IMG$ALLOCATE_ICB (ICB_POINTER)
; 2091          2310  1 !
; 2092          2311  1 ! Formal Parameter:
; 2093          2312  1 !
; 2094          2313  1 !    ICB_POINTER - Address of cell that will receive the address of a newly
; 2095          2314  1 !    allocated image control block.
; 2096          2315  1 !
; 2097          2316  1 ! Status Return.
; 2098          2317  1 !
; 2099          2318  1 !    SS$_NORMAL  - ICB successfully allocated
; 2100          2319  1 !
; 2101          2320  1 !    SS$_INSFMEM - Unable to allocate ICB
; 2102          2321  1 !-
; 2103          2322  1
; 2104          2323  2 BEGIN
; 2105          2324  2
; 2106          2325  2 LOCAL
; 2107          2326  2     SIZE,
; 2108          2327  2     ICB : REF $BBLOCK;
; 2109          2328  2
; 2110          2329  2 IF REMQUE (.IAC$GL_ICBFL, ICB)
; 2111          2330  2 THEN IF NOT EXE$ALOP1PROC (ICB$K_LENGTH; SIZE, ICB)
; 2112          2331  2      THEN RETURN SS$_INSFMEM;
; 2113          2332  2 CH$FILL (0, ICB$K_LENGTH, .ICB);
; 2114          2333  2 ICB [ICB$W_SIZE] = ICB$K_LENGTH;
; 2115          2334  2 ICB [ICB$B_TYPE] = ICB$K_ICB_TYPE_CODE;
; 2116          2335  2 ICB [ICB$L_STARTING_ADDRESS] = -1;
; 2117          2336  2 ICB [ICB$L_END_ADDRESS] = -1;
; 2118          2337  2 .ICB_POINTER = .ICB;
; 2119          2338  2 RETURN SS$_NORMAL;
; 2120          2339  2
; 2121          2340  1 END;


                                007C 00000          .ENTRY  IMG$ALLOCATE_ICB, Save R2,R3,R4,R5,R6       ; 2292
```

```
                                 50 00000000G   00   9E 00002          MOVAB   IAC$GL_ICBFL, R0                        ; 2329
                                 56       00    B0   0F 00009          REMQUE  @0(R0), ICB                             ;
                                                16   1C 0000D          BVC     1$                                      ;
                                 51       64    8F   9A 0000F          MOVZBL  #100, R1                                ; 2330
                                    00000000G   00   16 00013          JSB     EXE$ALOP1PROC                           ;
                                 56             52   D0 00019          MOVL    R2, R6                                  ;
                                 06             50   E8 0001C          BLBS    R0, 1$                                  ;
                                 50     0124    8F   3C 0001F          MOVZWL  #292, R0                                ; 2331
                                                04   00024          RET                                             ;
    0064    8F          00                    00   2C 00025 1$:      MOVC5   #0, (SP), #0, #100, (ICB)               ; 2332
                                                66        0002C                                                        ;
                           08   A6       64    8F   9B 0002D          MOVZBW  #100, 8(ICB)                            ; 2333
                           0A   A6       7F    8F   90 00032          MOVB    #127, 10(ICB)                           ; 2334
                           48   A6             01   CE 00037          MNEGL   #1, 72(ICB)                             ; 2335
                           4C   A6             01   CE 0003B          MNEGL   #1, 76(ICB)                             ; 2336
                           04   BC             56   D0 0003F          MOVL    ICB, @ICB_POINTER                       ; 2337
                                50             01   D0 00043          MOVL    #1, R0                                  ; 2338
                                                04   00046          RET                                             ; 2340

; Routine Size:  71 bytes,    Routine Base:  YF$$SYSIMGACT + 0A70
```

```
1 %SBTTL 'IMG$DEALLOCATE_ICB - Deallocate an Unused ICB'
1
1 GLOBAL ROUTINE IMG$DEALLOCATE_ICB (ICB) : NOVALUE =
1
1 !+
1 ! Functional Description:
1 !
1 !     This routine deallocates an ICB that was not used. This can be due to
1 !     one of three reasons:
1 !
1 !         The current image is being rundown, thus releasing all of its
1 !         image control blocks back to pool.
1 !
1 !         An ICB was allocated during image activation for an image that
1 !         has already been activated.
1 !
1 !         An error occurred, requiring that all images activated during the
1 !         current call be eliminated.
1 !
1 !     If the ICB was allocated from P1 space, it is deallocated to a linked
1 !     list of free ICBs that will be used during later activations. ICBs
1 !     allocated from P0 space use the normal deallocation routine.
1 !
1 ! Calling Sequence:
1 !
1 !     IMG$DEALLOCATE_ICB (ICB_ADDRESS)
1 !
1 ! Formal Parameter:
1 !
1 !     ICB_ADDRESS - Address of ICB that is being deallocated
1 !-
1
2 BEGIN
2
2 MAP
2     ICB : REF $BBLOCK;
2
2 BIND VA = ICB : $BBLOCK;
2
2 IF .VA [VA$V_P1]
2 THEN INSQUE (.ICB, .IAC$GL_ICBFL [1])    ! Insert at tail of lookaside list
2 ELSE EXE$DEAP1 (.ICB, .ICB [ICB$W_SIZE]);
2
1 END;
```

```
                                  000C 00000        .ENTRY  IMG$DEALLOCATE_ICB, Save R2,R3           ; 2343
               50          04  AC   D0 00002        MOVL    ICB, R0                                   ; 2381
   0C      07  AC              06   E1 00006        BBC     #6, VA+3, 1$                              ; 2380
               51   00000000G  00   9E 0000B        MOVAB   IAC$GL_ICBFL+4, R1                        ; 2381
           00  B1              60   0E 00012        INSQUE  (R0), @0(R1)                              ;
                                    04 00016        RET                                               ;
               51          08  A0   3C 00017 1$:    MOVZWL  8(R0), R1                                 ; 2382
                    00000000G  00   16 0001B        JSB     EXE$DEAP1                                 ;
```

```
                                                 04 00021        RET                                                 ; 2384

; Routine Size.  34 bytes,    Routine Base:  YF$$SYSIMGACT + 0AB7
```

```
;  2168          2385  1 %SBTTL 'SET_VECTORS - Prepare privileged vectors for execution'
;  2169          2386  1
;  2170          2387  1 ROUTINE SET_VECTORS =
;  2171          2388  1
;  2172          2389  1 !+
;  2173          2390  1 ! Functional Description:
;  2174          2391  1 !
;  2175          2392  1 !    This routine takes the privileged vector entries that contain RSB
;  2176          2393  1 !    instructions and replaces each with a JSB instruction.
;  2177          2394  1 !
;  2178          2395  1 ! Calling Sequence:
;  2179          2396  1 !
;  2180          2397  1 !    SET_VECTORS ()
;  2181          2398  1 !
;  2182          2399  1 ! Input Parameters:
;  2183          2400  1 !
;  2184          2401  1 !    none
;  2185          2402  1 !
;  2186          2403  1 ! Implicit Input:
;  2187          2404  1 !
;  2188          2405  1 !    IAC$AW_VECSET - Array that locates the dividing point in each vector
;  2189          2406  1 !       list between those vectors that already existed and those that were
;  2190          2407  1 !       added as part of the latest activation. This is the starting point for
;  2191          2408  1 !       the search.
;  2192          2409  1 !
;  2193          2410  1 !    CTL$A_DISPVEC - This address locates the start of the two-page area
;  2194          2411  1 !       containing the privileged vectors. The first longword of each area
;  2195          2412  1 !       contains the current end of the vector list. This is the end point for
;  2196          2413  1 !       the search.
;  2197          2414  1 !-
;  2198          2415  1
;  2199          2416  2 BEGIN
;  2200          2417  2
;  2201          2418  2 LITERAL
;  2202          2419  2     ABSOLUTE_MODE = %X'9F',
;  2203          2420  2     JSB_ABSOLUTE = (ABSOLUTE_MODE ^ 8) OR OP$_JSB : UNSIGNED (16),
;  2204          2421  2     RSB_ABSOLUTE = (ABSOLUTE_MODE ^ 8) OR OP$_RSB : UNSIGNED (16);
;  2205          2422  2
;  2206          2423  2 LOCAL
;  2207          2424  2     I, J;
;  2208          2425  2
;  2209          2426  2 INCRU I FROM 0 TO 3 DO
;  2210          2427  3     BEGIN
;  2211          2428  3
;  2212          2429  3     BIND
;  2213          2430  3         END_POINT = CTL$A_DISPVEC + (.I * 256) : LONG,
;  2214          2431  3         DISPVEC = CTL$A_DISPVEC + (.I * 256) : VECTOR [256,BYTE];
;  2215          2432  3
;  2216          2433  3     J = .IAC$AW_VECSET [.I];
;  2217          2434  3     WHILE .J LSSU .END_POINT DO
;  2218          2435  3
;  2219          2436  4         BEGIN
;  2220          2437  4
;  2221          2438  4         BIND OPCODE = DISPVEC [.J] : WORD;
;  2222          2439  4
;  2223          2440  4         IF .OPCODE EQLU RSB_ABSOLUTE
;  2224          2441  4         THEN OPCODE = JSB_ABSOLUTE;
```

```
2442 4
2443 4          J = .J + 6;
2444 4
2445 3          END;
2446 3
2447 3      IAC$AW_VECSET [.I] = .END_POINT;
2448 3
2449 2      END;
2450 2
2451 2  RETURN SS$_NORMAL;
2452 2
2453 1  END;
```

```
                                         000C 00000 SET_VECTORS:
                                                           .WORD   Save R2,R3                        ; 2387
                                          50  D4 00002     CLRL    I                                 ; 2426
       51            50                   08  78 00004 1$: ASHL    #8, I, R1                         ; 2430
                     51 00000000G0041         9E 00008     MOVAB   CTL$A_DISPVEC[R1], R1             ;
                     52 00000000G0040         3E 00010     MOVAW   IAC$AW_VECSET[I], R2              ; 2433
                     53                   62  3C 00018     MOVZWL  (R2), J                           ;
                     61                   53  D1 0001B 2$: CMPL    J, (R1)                           ; 2434
                                          17  1E 0001E     BGEQU   4$                                ;
                                        6341  9F 00020     PUSHAB  (J)[R1]                           ; 2440
              9F05   8F                   9E  B1 00023     CMPW    @(SP)+, #40709                    ;
                                          08  12 00028     BNEQ    3$                                ;
                                        6341  9F 0002A     PUSHAB  (J)[R1]                           ; 2441
                     9E        9F16       8F  B0 0002D     MOVW    #-24810, @(SP)+                   ;
                     53                   06  C0 00032 3$: ADDL2   #6, J                             ; 2443
                                          E4  11 00035     BRB     2$                                ; 2434
                     62                   61  B0 00037 4$: MOVW    (R1), (R2)                        ; 2447
                                          50  D6 0003A     INCL    I                                 ; 2426
                     03                   50  D1 0003C     CMPL    I, #3                             ;
                                          C3  1B 0003F     BLEQU   1$                                ;
                     50                   01  D0 00041     MOVL    #1, R0                            ; 2451
                                              04 00044     RET                                       ; 2453
```

; Routine Size: 69 bytes,   Routine Base: YF$$SYSIMGACT + 0AD9

```
: 2238          2454  1 %SBTTL 'ERROR_CLEAN_UP - Clean Up after an Error is Detected'
: 2239          2455  1
: 2240          2456  1 ROUTINE ERROR_CLEAN_UP : NOVALUE =
: 2241          2457  1
: 2242          2458  1 !+
: 2243          2459  1 ! Functional Description:
: 2244          2460  1 !
: 2245          2461  1 !     This routine cleans up when an error is detected after some successful
: 2246          2462  1 !     work has been completed.
: 2247          2463  1 !
: 2248          2464  1 !         All ICBs on the WORK list are simply deallocated.
: 2249          2465  1 !
: 2250          2466  1 !         ICBs that exist on the IMAGE (so-called done) list with the DONE bit
: 2251          2467  1 !         clear indicate images that have been successfully activated as a part
: 2252          2468  1 !         of this activation before an error was detected.
: 2253          2469  1 !
: 2254          2470  1 !             The address space associated with these images is deleted.
: 2255          2471  1 !
: 2256          2472  1 !             The channel on which each image file was opened is closed.
: 2257          2473  1 !
: 2258          2474  1 !             Each ICB is then deallocated.
: 2259          2475  1 !
: 2260          2476  1 !         Note that an ICB with no addresses yet mapped will have an address
: 2261          2477  1 !         range of -1, -1. The error from $DELTVA in this case is ignored.
: 2262          2478  1 !
: 2263          2479  1 ! Calling Sequence:
: 2264          2480  1 !
: 2265          2481  1 !     ERROR_CLEAN_UP ()
: 2266          2482  1 !
: 2267          2483  1 ! Formal Parameters:
: 2268          2484  1 !
: 2269          2485  1 !     none
: 2270          2486  1 !
: 2271          2487  1 ! Implicit Input:
: 2272          2488  1 !
: 2273          2489  1 !     IAC$GL_IMAGE_LIST - List of ICBs representing images that have been
: 2274          2490  1 !         successfully activated
: 2275          2491  1 !
: 2276          2492  1 !     IAC$GL_WORK_LIST - List of ICBs representing work left to be done.
: 2277          2493  1 !-
: 2278          2494  1
: 2279          2495  2 BEGIN
: 2280          2496  2
: 2281          2497  2 LOCAL
: 2282          2498  2     ICB      : REF $BBLOCK,
: 2283          2499  2     NEXT_ICB : REF $BBLOCK;
: 2284          2500  2
: 2285          2501  2 ! Simply deallocate the ICBs in the work list
: 2286          2502  2
: 2287          2503  2 WHILE NOT (REMQUE (.IAC$GL_WORK_LIST, ICB)) DO
: 2288          2504  2     IMG$DEALLOCATE_ICB (.ICB);
: 2289          2505  2
: 2290          2506  2 ! Traverse the done list, looking for ICBs with the done bit not yet set.
: 2291          2507  2
: 2292          2508  2 NEXT_ICB = .IAC$GL_IMAGE_LIST;
: 2293          2509  2
: 2294          2510  2 WHILE .NEXT_ICB NEQA IAC$GL_IMAGE_LIST DO
```

```
;  2295          2511  2      IF .NEXT_ICB [ICB$V_DONE]
;  2296          2512  2      THEN
;  2297          2513  2          NEXT_ICB = .NEXT_ICB [ICB$L_FLINK]
;  2298          2514  2      ELSE
;  2299          2515  3          BEGIN
;  2300          2516  3          REMQUE (.NEXT_ICB, ICB);
;  2301          2517  3          $DELTVA (INADR = ICB [ICB$Q_ADDRESS_RANGE]);
;  2302          2518  3          $DASSGN (CHAN = .ICB [ICB$W_CHAN]);
;  2303          2519  3          NEXT_ICB = .ICB [ICB$L_FLINK];
;  2304          2520  3          IMG$DEALLOCATE_ICB (.ICB);
;  2305          2521  2          END;
;  2306          2522  2
;  2307          2523  2  IAC$GL_IMAGCTX [IMAGCTX$V_SETVECTOR] = FALSE;
;  2308          2524  2
;  2309          2525  1  END;


                                                        .EXTRN  SYS$DELTVA

                                        001C 00000 ERROR_CLEAN_UP:
                                                        .WORD   Save R2,R3,R4                           ;  2456
                          54 00000000G   00  9E 00002           MOVAB   IAC$GL_IMAGE_LIST, R4           ;
                          50 00000000G   00  9E 00009 1$:       MOVAB   IAC$GL_WORK_LIST, R0            ;  2503
                          53         00  B0  0F 00010           REMQUE  @0(R0), ICB                     ;
                                         09  1D 00014           BVS     2$                              ;
                                         53  DD 00016           PUSHL   ICB                             ;  2504
                              FF7C   CF  01  FB 00018           CALLS   #1, IMG$DEALLOCATE_ICB          ;
                                         EA  11 0001D           BRB     1$                              ;
                                 52      64  D0 0001F 2$:       MOVL    IAC$GL_IMAGE_LIST, NEXT_ICB     ;  2508
                                 50      64  9E 00022 3$:       MOVAB   IAC$GL_IMAGE_LIST, R0           ;  2510
                                 50      52  D1 00025           CMPL    NEXT_ICB, R0                    ;
                                         50  13 00028           BEQL    5$                              ;
                          05         10  A2  06  E1 0002A       BBC     #6, 16(NEXT_ICB), 4$            ;  2511
                                 52      62  D0 0002F           MOVL    (NEXT_ICB), NEXT_ICB            ;  2513
                                         EE  11 00032           BRB     3$                              ;
                                 53      62  0F 00034 4$:       REMQUE  (NEXT_ICB), ICB                 ;  2516
                                         7E  7C 00037           CLRQ    -(SP)                           ;  2517
                                     48  A3  9F 00039           PUSHAB  72(ICB)                         ;
                              00000000G  00  03  FB 0003C       CALLS   #3, SYS$DELTVA                  ;
                                         7E  0E  A3  3C 00043   MOVZWL  14(ICB), -(SP)                  ;  2518
                              00000000G  00  01  FB 00047       CALLS   #1, SYS$DASSGN                  ;
                                 52      63  D0 0004E           MOVL    (ICB), NEXT_ICB                 ;  2519
                                         53  DD 00051           PUSHL   ICB                             ;  2520
                                  FF41   CF  01  FB 00053       CALLS   #1, IMG$DEALLOCATE_ICB          ;
                                         C8  11 00058           BRB     3$                              ;  2511
                              00000000G  00  01  8A 0005A 5$:       BICB2   #1, IAC$GL_IMAGCTX+2        ;  2523
                                             04 00061           RET                                     ;  2525

; Routine Size:  98 bytes,    Routine Base:  YF$$SYSIMGACT + 0B1E
```

```
;  2311          2526  1 %SBTTL 'GET_OTHER_IMAGE - Open Primary Image File'
;  2312          2527  1
;  2313          2528  1 ROUTINE GET_OTHER_IMAGE (ICB) =
;  2314          2529  1
;  2315          2530  1 !+
;  2316          2531  1 ! Functional Description:
;  2317          2532  1 !
;  2318          2533  1 !    This routine is called when the image passed to the image activator is not
;  2319          2534  1 !    really the image that should be activated. This situation occurs in the
;  2320          2535  1 !    case of compatibility mode images and other specialized cases. The general
;  2321          2536  1 !    operation of this routine is as follows.
;  2322          2537  1 !
;  2323          2538  1 !        Information about the original image is stored in various places so
;  2324          2539  1 !        that it is available to the image that eventually gets activated.
;  2325          2540  1 !
;  2326          2541  1 !        A new image name is selected based on the alias code.
;  2327          2542  1 !
;  2328          2543  1 !        This new image file is opened and its header decoded.
;  2329          2544  1 !
;  2330          2545  1 !    Activation continues with the secondary image replacing the original image
;  2331          2546  1 !    as the target of activation.
;  2332          2547  1 !
;  2333          2548  1 ! Calling Sequence:
;  2334          2549  1 !
;  2335          2550  1 !    GET_OTHER_IMAGE (ICB address)
;  2336          2551  1 !
;  2337          2552  1 ! Input Parameter:
;  2338          2553  1 !
;  2339          2554  1 !    ICB - Address of image control block that describes the primary image
;  2340          2555  1 !
;  2341          2556  1 ! Output Parameters:
;  2342          2557  1 !
;  2343          2558  1 !    none
;  2344          2559  1 !
;  2345          2560  1 ! Implicit Output:
;  2346          2561  1 !
;  2347          2562  1 !    The image name of the primary image is stored in the compatibility mode
;  2348          2563  1 !    data page. The channel on which the primary image is opened (and its KFE
;  2349          2564  1 !    address if any) is stored for later return to the caller.
;  2350          2565  1 !
;  2351          2566  1 ! Assumption:
;  2352          2567  1 !
;  2353          2568  1 !    This routine can only be called from the main loop in the image activator.
;  2354          2569  1 !    This means that the primary buffers (FAB, NAM, IHD, etc.) describe the
;  2355          2570  1 !    original image.
;  2356          2571  1 !-
;  2357          2572  1
;  2358          2573  2 BEGIN
;  2359          2574  2
;  2360          2575  2 MAP
;  2361          2576  2     ICB                                         : REF $BBLOCK;
;  2362          2577  2
;  2363          2578  2 BIND
;  2364          2579  2     IHD_CTX     = .ICB [ICB$L_CONTEXT]          : $BBLOCK,
;  2365          2580  2     ICB_NAME    = ICB [ICB$T_IMAGE_NAME]        : VECTOR [, BYTE],
;  2366          2581  2     FAB         = PRIMARY_FAB                   : $BBLOCK,
;  2367          2582  2     NAM         = PRIMARY_NAM                   : $BBLOCK,
```

```
; 2368          2583  2     STORED_NAME = CTL$AG_CMEDATA                    : VECTOR [, BYTE];
; 2369          2584  2
; 2370          2585  2 ! The image name for a type 2 image is stored in the last 126 bytes of the
; 2371          2586  2 ! first block of the image header as a counted ASCII string.
; 2372          2587  2
; 2373          2588  2 MAP
; 2374          2589  2     INPUT_BUFFER                                    : VECTOR [512, BYTE];
; 2375          2590  2
; 2376          2591  2 BIND
; 2377          2592  2     TYPE_2_IMAGE_NAME =
; 2378          2593  2         INPUT_BUFFER [512 - 128]                    : VECTOR [128 - 2, BYTE];
; 2379          2594  2
; 2380          2595  2 ! Three of the four cases handled by this routine activate specific images
; 2381          2596  2 ! whose names are listed here. (The fourth case extracts the image name from
; 2382          2597  2 ! the end of the first block of the image header.)
; 2383          2598  2
; 2384          2599  2 BIND
; 2385          2600  2     RSX_NAME = $DESCRIPTOR ('RSX'),
; 2386          2601  2     BPA_NAME = $DESCRIPTOR ('BPA'),
; 2387          2602  2     LOGIN_NAME = $DESCRIPTOR ('LOGINOUT'),
; 2388          2603  2
; 2389          2604  2     ! All four cases use SYS$SYSTEM as the default directory string
; 2390          2605  2
; 2391          2606  2     SYSTEM_NAME = $DESCRIPTOR ('SYS$SYSTEM:.EXE');
; 2392          2607  2
; 2393          2608  2 LOCAL
; 2394          2609  2     NEW_IMAGE_NAME,
; 2395          2610  2     NEW_IMAGE_NAME_DESC                             : $BBLOCK [DSC$K_S_BLN],
; 2396          2611  2     STATUS;
; 2397          2612  2
; 2398          2613  2 OWN_STORAGE [OTHER_CHANNEL] = .ICB [ICB$W_CHAN];
; 2399          2614  2 OWN_STORAGE [OTHER_KFE_ADDRESS] = .FAB [FAB$L_CTX];
; 2400          2615  2
; 2401          2616  2 ! Now perform the steps that are specific to the type of other image that is
; 2402          2617  2 ! being selected. The name of the image to activate is the most important
; 2403          2618  2 ! part of this step.
; 2404          2619  2
; 2405          2620  2 CASE .IHD_CTX [CTX_W_ALIAS] FROM IHD$C_RSX TO IHD$C_CLI OF
; 2406          2621  2
; 2407          2622  2     SET
; 2408          2623  2
; 2409          2624  2     [IHD$C_RSX]:
; 2410          2625  2
; 2411          2626  2         ! This is an image produced by the RSX-11M task builder. Activate
; 2412          2627  2         ! SYS$SYSTEM:RSX.EXE in its stead.
; 2413          2628  2
; 2414          2629  2         NEW_IMAGE_NAME = RSX_NAME;
; 2415          2630  2
; 2416          2631  2     [IHD$C_BPA]:
; 2417          2632  2
; 2418          2633  2         ! There is no supported way that this type of image can be created. We
; 2419          2634  2         ! will activate SYS$SYSTEM:BPA.EXE anyway and let the chips fall ...
; 2420          2635  2
; 2421          2636  2         NEW_IMAGE_NAME = BPA_NAME;
; 2422          2637  2
; 2423          2638  2     [IHD$C_ALIAS]:
; 2424          2639  2
```

```
;  2425          2640  2          ! This is a special form of image that contains the name of a second
;  2426          2641  2          ! image in the last 128 bytes of the first block of the image header.
;  2427          2642  2          ! (The actual name is restricted to 125 bytes because the last word
;  2428          2643  2          ! of the first block is reserved to contain the code word and the
;  2429          2644  2          ! string contains a count byte.)
;  2430          2645  2
;  2431          2646  3          BEGIN
;  2432          2647  3
;  2433          2648  3          NEW_IMAGE_NAME_DESC [DSC$W_LENGTH] = .TYPE_2_IMAGE_NAME [0];
;  2434          2649  3          NEW_IMAGE_NAME_DESC [DSC$A_POINTER] = TYPE_2_IMAGE_NAME [1];
;  2435          2650  3
;  2436          2651  3          NEW_IMAGE_NAME = NEW_IMAGE_NAME_DESC;
;  2437          2652  3
;  2438          2653  2          END;
;  2439          2654  2
;  2440          2655  2      [IHD$C_CLI]:
;  2441          2656  2
;  2442          2657  2          ! The image is a command language interpreter whose name was passed to
;  2443          2658  2          ! the Create Process system service. If this is the activation of a main
;  2444          2659  2          ! program (and not a merged activation, the usual way to put a CLI into
;  2445          2660  2          ! P1 space), we will activate SYS$SYSTEM:LOGINOUT.EXE. In this case, we
;  2446          2661  2          ! will close the CLI image file first (by deassigning the channel) because
;  2447          2662  2          ! LOGINOUT uses a more restrictive form of $OPEN than occurred above.
;  2448          2663  2
;  2449          2664  2          IF .OWN_STORAGE [MAIN_PROGRAM]
;  2450          2665  2          THEN
;  2451          2666  3              BEGIN
;  2452          2667  3
;  2453          2668  3              $DASSGN (CHAN = .ICB [ICB$W_CHAN]);
;  2454          2669  3              NEW_IMAGE_NAME = LOGIN_NAME;
;  2455          2670  3
;  2456          2671  3              END
;  2457          2672  2          ELSE
;  2458          2673  2              RETURN SS$_NORMAL;
;  2459          2674  2
;  2460          2675  2      [OUTRANGE]:
;  2461          2676  2
;  2462          2677  2          RETURN SS$_BADIMGHDR;
;  2463          2678  2
;  2464          2679  2      TES;
;  2465          2680  2
;  2466          2681  2 ! Any context established by the original image must be cleared before
;  2467          2682  2 ! the activation continues.
;  2468          2683  2
;  2469          2684  2 ICB [ICB$L_FLAGS] = 0;                   ! Clear previous activation flags
;  2470          2685  2 ICB [ICB$L_IHD] = 0;                     ! Clear pointer to resident header
;  2471          2686  2 IHD_CTX [CTX_L_IHDBUF] = PRIMARY_IHD;    ! Reestablish IHD buffer
;  2472          2687  2
;  2473          2688  2 ! If the primary image was a CLI (type 3), only the file name is stored. In
;  2474          2689  2 ! all other cases, the entire resultant (or expanded) string is stored.
;  2475          2690  2
;  2476          2691  2 IF .IHD_CTX [CTX_W_ALIAS] EQL IHD$C_CLI
;  2477          2692  2 THEN
;  2478          2693  3     BEGIN
;  2479          2694  3     STORED_NAME [0] = .NAM [NAM$B_NAME];
;  2480          2695  3     CH$MOVE (.STORED_NAME [0], .NAM [NAM$L_NAME], STORED_NAME [1]);
;  2481          2696  3     END
```

```
;  2482          2697  2 ELSE
;  2483          2698  3     BEGIN
;  2484          2699  3     STORED_NAME [0] = .NAM [NAM$B_RSL];
;  2485          2700  3     CH$MOVE (.STORED_NAME [0], .NAM [NAM$L_RSA], STORED_NAME [1]);
;  2486          2701  2     END;
;  2487          2702  2
;  2488          2703  2 ! Open the secondary image file, decode the image header, and resume processing
;  2489          2704  2 ! in the main routine as if the secondary image were the one selected for
;  2490          2705  2 ! activation.
;  2491          2706  2
;  2492          2707  2 ICB_NAME [0] = 0;                        ! Force a new name to be stored in ICB
;  2493          2708  2
;  2494          2709  2 STATUS = IMG$OPEN_IMAGE (                ! Open the image file
;  2495          2710  2     .NEW_IMAGE_NAME,
;  2496          2711  2     SYSTEM_NAME,
;  2497          2712  2     PRIMARY_FAB,
;  2498          2713  2     PRIMARY_NAM,
;  2499          2714  2     RESULT_NAME,
;  2500          2715  2     .ICB);
;  2501          2716  2 IF NOT .STATUS
;  2502          2717  2 THEN RETURN .STATUS;
;  2503          2718  2
;  2504          2719  2 STATUS = IMG$GET_HEADER (.ICB);          ! Decode and store away the IHD contents
;  2505          2720  2 RETURN .STATUS;
;  2506          2721  2
;  2507          2722  1 END;


                                                58  53  52  00B80 P.AAE:   .ASCII  \RSX\                          ;
                                                        00B83           .BLKB   1                                   ;
                                              00000003  00B84 P.AAD:    .LONG   3                                   ;
                                              00000000' 00B88           .ADDRESS P.AAE                              ;
                                                41  50  42  00B8C P.AAG:  .ASCII  \BPA\                           ;
                                                        00B8F           .BLKB   1                                   ;
                                              00000003  00B90 P.AAF:    .LONG   3                                   ;
                                              00000000' 00B94           .ADDRESS P.AAG                              ;
                                  54  55  4F  4E  49  47  4F  4C  00B98 P.AAI:   .ASCII  \LOGINOUT\                  ;
                                              00000008  00BA0 P.AAH:    .LONG   8                                   ;
                                              00000000' 00BA4           .ADDRESS P.AAI                              ;
45  58  45  2E  3A  4D  45  54  53  59  53  24  53  59  53  00BA8 P.AAK:   .ASCII  \SYS$SYSTEM:.EXE\                 ;
                                                        00BB7           .BLKB   1                                   ;
                                              0000000F  00BB8 P.AAJ:    .LONG   15                                  ;
                                              00000000' 00BBC           .ADDRESS P.AAK                              ;

                                                                RSX_NAME=           P.AAD
                                                                BPA_NAME=           P.AAF
                                                                LOGIN_NAME=         P.AAH
                                                                SYSTEM_NAME=        P.AAJ


                                                      07FC 00000 GET_OTHER_IMAGE:
                                                                 .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10           ; 2528
                                          5A         BF   AF  9E 00002          MOVAB   RSX_NAME, R10               ;
                                          59 00000000G   00  9E 00006          MOVAB   STORED_NAME, R9             ;
                                          58 00000000G   00  9E 0000D          MOVAB   OWN_STORAGE+14, R8          ;
                                          5E             08  C2 00014          SUBL2   #8, SP                      ;
```

```
                                    56       04  AC  D0 00017          MOVL    ICB, R6                                    ; 2579
                                    52       58  A6  D0 0001B          MOVL    88(R6), R2                                 ;
                                    68       0E  A6  B0 0001F          MOVW    14(R6), OWN_STORAGE+14                     ; 2613
                              06    A8     FDAB  C8  D0 00023          MOVL    FAB+24, OWN_STORAGE+20                     ; 2614
                    03              00       0E  A2  AF 00029          CASEW   14(R2), #0, #3                             ; 2620
        0028        0018          0012         000D     0002E 1$:      .WORD   2$-1$,-                                    ;
                                                                               3$-1$,-                                    ;
                                                                               4$-1$,-                                    ;
                                                                               5$-1$                                      ;
                                    50       44  8F  9A 00036          MOVZBL  #68, R0                                    ; 2677
                                                     04 0003A          RET                                                ;
                                    57           6A  9E 0003B 2$:      MOVAB   RSX_NAME, NEW_IMAGE_NAME                   ; 2629
                                                 2F  11 0003E          BRB     7$                                         ;
                                    57       0C  AA  9E 00040 3$:      MOVAB   BPA_NAME, NEW_IMAGE_NAME                   ; 2636
                                                 29  11 00044          BRB     7$                                         ;
                                    6E     F713  C8  9B 00046 4$:      MOVZBW  TYPE_2_IMAGE_NAME, NEW_IMAGE_NAME_DESC     ; 2648
                              04    AE     F714  C8  9E 0004B          MOVAB   TYPE_2_IMAGE_NAME+1, NEW_IMAGE_NAME_DESC+4 ; 2649
                                    57           6E  9E 00051          MOVAB   NEW_IMAGE_NAME_DESC, NEW_IMAGE_NAME        ; 2651
                                                 19  11 00054          BRB     7$                                         ; 2620
                                    11       F2  A8  E9 00056 5$:      BLBC    OWN_STORAGE, 6$                            ; 2664
                                    7E       0E  A6  3C 0005A          MOVZWL  14(R6), -(SP)                              ; 2668
                       00000000G    00           01  FB 0005E          CALLS   #1, SYS$DASSGN                             ;
                                    57       1C  AA  9E 00065          MOVAB   LOGIN_NAME, NEW_IMAGE_NAME                 ; 2669
                                                 04  11 00069          BRB     7$                                         ; 2664
                                    50           01  D0 0006B 6$:      MOVL    #1, R0                                     ; 2673
                                                     04 0006E          RET                                                ;
                                             10  A6  D4 0006F 7$:      CLRL    16(R6)                                     ; 2684
                                             50  A6  D4 00072          CLRL    80(R6)                                     ; 2685
                              04    A2     F793  C8  9E 00075          MOVAB   PRIMARY_IHD, 4(R2)                         ; 2686
                                    03       0E  A2  B1 0007B          CMPW    14(R2), #3                                 ; 2691
                                                 0F  12 0007F          BNEQ    8$                                         ;
                                    69     FE1E  C8  90 00081          MOVB    NAM+59, STORED_NAME                        ; 2694
                                    51           69  9A 00086          MOVZBL  STORED_NAME, R1                            ; 2695
                                    50     FE2F  C8  D0 00089          MOVL    NAM+76, R0                                 ;
                                                 0D  11 0008E          BRB     9$                                         ;
                                    69     FDE6  C8  90 00090 8$:      MOVB    NAM+3, STORED_NAME                         ; 2699
                                    51           69  9A 00095          MOVZBL  STORED_NAME, R1                            ; 2700
                                    50     FDE7  C8  D0 00098          MOVL    NAM+4, R0                                  ;
                01    A9            60           51  28 0009D 9$:      MOVC3   R1, (R0), STORED_NAME+1                    ;
                                             14  A6  94 000A2          CLRB    20(R6)                                     ; 2707
                                                 56  DD 000A5          PUSHL   R6                                         ; 2715
                                           FEF3  C8  9F 000A7          PUSHAB  RESULT_NAME                                ; 2709
                                           FDE3  C8  9F 000AB          PUSHAB  PRIMARY_NAM                                ;
                                           FD93  C8  9F 000AF          PUSHAB  PRIMARY_FAB                                ;
                                             34  AA  9F 000B3          PUSHAB  SYSTEM_NAME                                ;
                                                 57  DD 000B6          PUSHL   NEW_IMAGE_NAME                             ; 2710
                            F667    CF           06  FB 000B8          CALLS   #6, IMG$OPEN_IMAGE                         ;
                                    07           50  E9 000BD          BLBC    STATUS, 10$                                ; 2716
                                                 56  DD 000C0          PUSHL   R6                                         ; 2719
                            F887    CF           01  FB 000C2          CALLS   #1, IMG$GET_HEADER                         ;
                                                     04 000C7 10$:     RET                                                ; 2722
```

; Routine Size: 200 bytes,   Routine Base: YF$$SYSIMGACT + 0BC0

```
; 2509          2723  1
; 2510          2724  1 END                                     ! End of module SYS$IMGACT
; 2511          2725  1
; 2512          2726  0 ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  YF$$SYSIMGACT                    3208  NOVEC,  WRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;                                 Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1           18619       174         0      1000          00:01.8
```

```
;                                       COMMAND QUALIFIERS

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

; Size:         3111 code + 97 data bytes
; Run Time:       01:07.5
; Elapsed Time:   01:34.6
; Lines/CPU Min:   2423
; Lexemes/CPU-Min: 24715
; Memory Used:  301 pages
; Compilation Complete
```

SYSGETSYI
LIS

SYSGETPTI
LIS

SYSTMGFIX
LIS

SYSGETTIM
LIS

SYSGETLKI
LIS

SYSGETMSG
LIS

SYSIMGACT
LIS