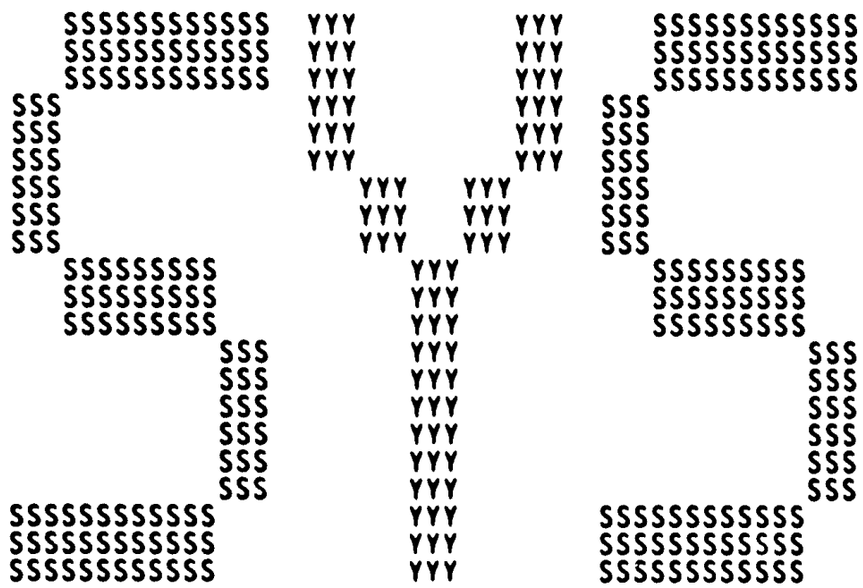

```
  CCCCCCCC  MM      MM    OOOOOO    DDDDDDDD     SSSSSSSS   SSSSSSSS  DDDDDDDD     SSSSSSSS  PPPPPPPP
  CCCCCCCC  MM      MM    OOOOOO    DDDDDDDD     SSSSSSSS   SSSSSSSS  DDDDDDDD     SSSSSSSS  PPPPPPPP
CC          MMMM  MMMM  OO      OO  DD      DD  SS         SS         DD      DD  SS         PP      PP
CC          MMMM  MMMM  OO      OO  DD      DD  SS         SS         DD      DD  SS         PP      PP
CC          MM  MM  MM  OO      OO  DD      DD  SS         SS         DD      DD  SS         PP      PP
CC          MM  MM  MM  OO      OO  DD      DD  SS         SS         DD      DD  SS         PP      PP
CC          MM      MM  OO      OO  DD      DD    SSSSSS     SSSSSS   DD      DD    SSSSSS   PPPPPPPP
CC          MM      MM  OO      OO  DD      DD    SSSSSS     SSSSSS   DD      DD    SSSSSS   PPPPPPPP
CC          MM      MM  OO      OO  DD      DD          SS         SS  DD      DD          SS  PP
CC          MM      MM  OO      OO  DD      DD          SS         SS  DD      DD          SS  PP
CC          MM      MM  OO      OO  DD      DD          SS         SS  DD      DD          SS  PP
CC          MM      MM  OO      OO  DD      DD          SS         SS  DD      DD          SS  PP
  CCCCCCCC  MM      MM    OOOOOO    DDDDDDDD   SSSSSSSS   SSSSSSSS  DDDDDDDD   SSSSSSSS  PP
  CCCCCCCC  MM      MM    OOOOOO    DDDDDDDD   SSSSSSSS   SSSSSSSS  DDDDDDDD   SSSSSSSS  PP

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

....
....
....

Table of contents

```
0000     1          .NLIST  CND
0000     5          .TITLE  CMODSSDSP - CHANGE MODE SYSTEM SERVICE DISPATCHER
0000    19          .IDENT  'V04-000'
0000    20
0000    21 ;
0000    22 ;*****************************************************************************
0000    23 ;*                                                                           *
0000    24 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    25 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000    26 ;*  ALL RIGHTS RESERVED.                                                     *
0000    27 ;*                                                                           *
0000    28 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    29 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    30 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    31 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    32 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    33 ;*  TRANSFERRED.                                                             *
0000    34 ;*                                                                           *
0000    35 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    36 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    37 ;*  CORPORATION.                                                             *
0000    38 ;*                                                                           *
0000    39 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    40 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    41 ;*                                                                           *
0000    42 ;*                                                                           *
0000    43 ;*****************************************************************************
0000    44 ;
0000    45 ; D. N. CUTLER 22-JUN-76
0000    46 ;
0000    47 ;  MODIFIED BY:
0000    48 ;
0000    49 ;       V03-041 LJK0287         Lawrence J. Kenah       27-Jun-1984
0000    50 ;               Add R5 to entry mask for $CANEXH system service.
0000    51 ;
0000    52 ;       V03-040 LMP0239         L. Mark Pilant,         23-Apr-1984  9:21
0000    53 ;               Change $CHKPRO from an exec mode service to a kernel mode
0000    54 ;               service.  This was made necessary by the $CHKPRO (internal
0000    55 ;               entry point) interface change.
0000    56 ;
0000    57 ;       V03-039 MMD0250         Meg Dumont,     27-Feb-1984  17:49
0000    58 ;               Add support for $MTACCESS installation specific accessibility
0000    59 ;               routine
0000    60 ;
0000    61 ;       V03-038 DAS0001         David Solomon           20-Feb-1984
0000    62 ;               Implement new design for RMS echo SYS$INPUT to SYS$OUTPUT
0000    63 ;               (vs V03-019). Echo is now performed by a caller's mode AST
0000    64 ;               routine declared in RMS\RMSEXRMS. Change INCB/DECB of FAB/RAB
0000    65 ;               busy bit to BISB/BICB, now that we have room.
0000    66 ;
0000    67 ;       V03-037 SSA0004         Stan Amway              28-Dec-1983
0000    68 ;               For $SETPFM, changed number of parameters from 1 to 4
0000    69 ;               and changed entry mask to save R2-R11.
0000    70 ;
0000    71 ;       V03-036 TMK0002         Todd M. Katz            19-Nov-1983
0000    72 ;               The entry point for $ASCTOID can no longer be reached as a
0000    73 ;               branch destination from the executive mode dispatcher.
```

```
0000     74 ;              A temporary entry point (EXE$ASCTOID) has been placed within
0000     75 ;              this module, and a JMP is made from it to the real system
0000     76 ;              service entry point (EXE$$ASCTOID).
0000     77 ;
0000     78 ;              Also, change the entry mask for SYS$TRNLOG, so that R8 is
0000     79 ;              now saved.
0000     80 ;
0000     81 ;       V03-035 TMK0001         Todd M. Katz            22-Oct-1983
0000     82 ;              The entry points for $FINISH_RDB and $IDTOASC can no
0000     83 ;              longer be reached as branch destinations from the executive
0000     84 ;              mode dispatcher. Temporary entry points (EXE$FINISH_RDB and
0000     85 ;              EXE$IDTOASC) have been placed within this module, and from
0000     86 ;              each a JMP is made to the real system service entry points
0000     87 ;              (EXE$$FINISH_RDB and EXE$$IDTOASC).
0000     88 ;
0000     89 ;       V03-034 PRB0254         Paul Beck       15-Sep-1983  14:49
0000     90 ;              (1) Correct the way synchronous CJF services are defined.
0000     91 ;              (2) Define loadable RUF services.
0000     92 ;
0000     93 ;       V03-033 WMC0029         Wayne Cardoza           31-Aug-1983
0000     94 ;              Loadable services should not be unconditionally inhibited.
0000     95 ;              Add an alternate CHMx argument to LDBSRV.
0000     96 ;
0000     97 ;       V03-032 DWT0125         David W. Thiel          22-Aug-1983
0000     98 ;              Remove CHECKARGLIST and calls to same.
0000     99 ;
0000    100 ;       V03-031 MKL0167         Mary Kay Lyons          19-Aug-1983
0000    101 ;              Generate loadable service vector for CJF$GETCJI.
0000    102 ;
0000    103 ;       V03-030 KBT0578         Keith B. Thompson       8-Aug-1983
0000    104 ;              Add parameter to $FILESCAN
0000    105 ;
0000    106 ;       V03-029 RAS0178         Ron Schaefer            29-Jul-1983
0000    107 ;              Add code to detect the AST/non-AST RMS FAB/RAB race
0000    108 ;              condition where an RMS operation is initiated while
0000    109 ;              the user FAB/RAB is still waiting for completion of
0000    110 ;              previous operation.
0000    111 ;
0000    112 ;       V03-028 WMC0028         Wayne Cardoza           29-Jun-1983
0000    113 ;              Add CJF services.
0000    114 ;
0000    115 ;       V03-027 WMC0027         Wayne Cardoza           23-Jun-1983
0000    116 ;              Make old logical name services "all mode".
0000    117 ;              Changes to image activator vectors.
0000    118 ;
0000    119 ;       V03-026 JWH0222         Jeffrey W. Horn          2-May-1983
0000    120 ;              Add LDBSRV macro for vector definitions of loadable
0000    121 ;              services.
0000    122 ;
0000    123 ;       V03-025 DMW4035         DMWalp                  26-May-1983
0000    124 ;              Intergate new logical name structures.
0000    125 ;
0000    126 ;       V03-024 LMP0109         L. Mark Pilant,         28-Apr-1983  15:53
0000    127 ;              Make $CHKPRO an EXEC mode system service to allow examination
0000    128 ;              of various system data structures.
0000    129 ;
0000    130 ;       V03-024 RAS0147         Ron Schaefer            28-APR-1983
```

```
0000   131 ;               Add $FILESCAN. Add R8 and R9 to $SETPRN register mask.
0000   132 ;
0000   133 ;       V03-023 JLV0244         Jake VanNoy             27-APR-1983
0000   134 ;               Add $BRKTHRUW. Change $BRDCST to all mode service.
0000   135 ;               $BRDCST now uses $BRKTHRU to do real work.
0000   136 ;
0000   137 ;       V03-022 LMP0099         L. Mark Pilant,         13-Apr-1983  19:15
0000   138 ;               Add the $CHKPRO system service.
0000   139 ;
0000   140 ;       V03-021 ACG0319         Andrew C. Goldstein,    21-Mar-1983  13:51
0000   141 ;               Add $GRANTID and $REVOKID services
0000   142 ;
0000   143 ;       V03-020 JLV0234         Jake VanNoy              1-MAR-1983
0000   144 ;               Add $BRKTHRU service.
0000   145 ;
0000   146 ;       V03-019 RAS0120         Ron Schaefer            25-Feb-1983
0000   147 ;               Add suppor   o echo SYS$INPUT to SYS$OUTPUT.
0000   148 ;               This involves examining the return code from RMS for $GET;
0000   149 ;               if the special status RMS$_ECHO (not returned to users)
0000   150 ;               is found, then create a RAB on the caller's stack and
0000   151 ;               execute a $PUT operation to echo the line.
0000   152 ;               A certain amount of RMS synchronization code was
0000   153 ;               shuffled around in order to make room for this.
0000   154 ;
0000   155 ;       V03-018 ACG0317         Andrew C. Goldstein,    22-Feb-1983  15:16
0000   156 ;               Fix off-by-one in kernel arg vector
0000   157 ;
0000   158 ;       V03-017 RSH0004         R. Scott Hanna          10-Feb-1983
0000   159 ;               Added $ASCTOID, $FINISH_RDB, and $IDTOASC to system service list
0000   160 ;
0000   161 ;       V03-016 RNG0016         Rod N. Gamache          1-Feb-1983
0000   162 ;               Added $GETLKI to system service list
0000   163 ;
0000   164 ;       V03-015 WMC0015         Wayne Cardoza           12-Jan-1983
0000   165 ;               Put back accidentally deleted space holder for RMS synchronization.
0000   166 ;
0000   167 ;       V03-014 DMW4023         DMWalp                  7-Jan-1983
0000   168 ;               Added $CRELNT, $CRELNM, $DELLNM and $TRNLNM
0000   169 ;
0000   170 ;       V03-013 KDM0033         Kathleen D. Morse       13-Dec-1982
0000   171 ;               Correct usage of an interlocked instruction to flush
0000   172 ;               the hardware cache queue.
0000   173 ;
0000   174 ;       V03-012 ROW0146         Ralph O. Weber          6-DEC-1982
0000   175 ;               Insert routine header comments for INHEXCP, CHECKARGLIST,
0000   176 ;               and EXE$CMODKRNLX (MPS$CMODKRNLX).  Move things around so
0000   177 ;               that EXE$CMODKRNL (MPS$CMODKRNL) header comments are near
0000   178 ;               EXE$CMODRKNL (MPS$CMODKRNL) and ASTEXIT comments are near
0000   179 ;               ASTEXIT.  Make basic kernal-mode .PSECT definition for Y$CMODK
0000   180 ;               or MP$CMOD1 immediately after executive mode code so that new
0000   181 ;               code can be inserted in a way that preserves routine headers,
0000   182 ;               conditional assembly, and .PSECT definitions.  Backout ROW145,
0000   183 ;               and in its place, correct conditional assembly of BGEQU 10$
0000   184 ;               after ACCVIO_RET so that it is assembled only for MPCMOD and
0000   185 ;               so that it is located before ACCVIO_RET.  Change PCB address
0000   186 ;               lookup at KERDSP in MPCMOD to use CTL$GL_PCB so that it works
0000   187 ;               correctly regardless of which processor executes it.
```

```
0000   188 ;
0000   189 ;       V03-011 ROW0145         Ralph O. Weber          29-NOV-1982
0000   190 ;               Move EXE$EXCPTN (and MPS$EXCPTN) to before ASTEXIT (or
0000   191 ;               MPS$ASTEXIT) in an attempt to make branch destinations in
0000   192 ;               EXE$CMODKRNL reach.
0000   193 ;
0000   194 ;       V03-010 KDM0030         Kathleen D. Morse       18-Nov-1982
0000   195 ;               Add logic to MPCMOD that allows the primary to execute
0000   196 ;               secondary-specific code, without turning into a secondary.
0000   197 ;
0000   198 ;       V03-009 MLJ0099         Martin L. Jack, 20-Oct-1982  19:42
0000   199 ;               Complete V03-002 by correcting mode and argument count of
0000   200 ;               $SNDJBC and removing temporary stubs.
0000   201 ;
0000   202 ;       V03-008 RIH0001         Richard I. Hustvedt     1-Jun-1982
0000   203 ;               Correct handling of AST queue by secondary processor to
0000   204 ;               avoid losing some AST notifications by incorrectly computing
0000   205 ;               PHD$B_ASTLVL.
0000   206 ;
0000   207 ;       V03-007 KDM0018         Kathleen D. Morse       30-Sep-1982
0000   208 ;               Add MPSWITCH logic to create a kernel system service
0000   209 ;               dispatcher for the secondary processor of an 11/782.
0000   210 ;
0000   211 ;       V03-006 STJ3028         Steven T. Jeffreys      26-Sep-1982
0000   212 ;               Added $ERAPAT system service vector.
0000   213 ;
0000   214 ;       V03-005 DWT0058         David Thiel             11-Aug-1982
0000   215 ;               Eliminate use of R2 while waiting for service
0000   216 ;               completion.
0000   217 ;
0000   218 ;       V03-004 JWH0001         Jeffrey W. Horn         26-Jul-1982
0000   219 ;               Add new RMS service, RMSRUHNDLR, an un-documented service
0000   220 ;               which acts as the Recovery Unit handler for RMS.
0000   221 ;
0000   222 ;       V03-003 PHL0102         Peter H. Lipman         16-Jul-1982
0000   223 ;               Fix new SYNCH logic to always return SS$_NORMAL,
0000   224 ;               not access IOSB if error from service, and return
0000   225 ;               error status from $SETEF if event flag cluster went away
0000   226 ;
0000   227 ;       V03-002 PHL0101         Peter H. Lipman         17-Jun-1982
0000   228 ;               Add $SYNCH system service and fix $QIOW and $ENQW to use the
0000   229 ;               new code for waiting for the combination of EFN and IOSB
0000   230 ;
0000   231 ;               Improve readability of conditionals.
0000   232 ;
0000   233 ;               Add $GETDVIW, $GETJPIW, $GETSYIW, $SNDJBC, $SNDJBCW, and
0000   234 ;               $UPDSECW.  All the waiting versions use common code.
0000   235 ;
0000   236 ;
0000   237 ;
0000   238 ; CHANGE MODE SYSTEM SERVICE DISPATCHER
0000   239 ;
0000   240 ; MACRO LIBRARY CALLS
0000   241 ;
0000   242
0000   243         $ACBDEF                         ;DEFINE AST CONTROL BLOCK OFFSETS
0000   244         $CHFDEF                         ;DEFINE CONDITION HANDLING OFFSETS
```

```
                     0000   245          $ENQDEF                         ;DEFINE ENQ SYSTEM SERVICE ARGS
                     0000   246          $GETDVIDEF                      ;DEFINE GETDVI SYSTEM SERVICE ARGS
                     0000   247          $GETJPIDEF                      ;DEFINE GETJPI SYSTEM SERVICE ARGS
                     0000   248          $GETLKIDEF                      ;DEFINE GETLKI SYSTEM SERVICE ARGS
                     0000   249          $GETSYIDEF                      ;DEFINE GETSYI SYSTEM SERVICE ARGS
                     0000   250          $IPLDEF                         ;DEFINE INTERRUPT PRIORITY LEVELS
                     0000   254          $PCBDEF                         ;DEFINE PCB OFFSETS
                     0000   255          $PHDDEF                         ;DEFINE PHD OFFSETS
                     0000   256          $PRDEF                          ;DEFINE PROCESSOR REGISTERS
                     0000   257          $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
                     0000   258          $RABDEF                         ;DEFINE RMS RAB FIELDS
                     0000   259          $RPBDEF                         ;DEFINE REBOOT PARAME*ER BLOCK
                     0000   260          $QIODEF                         ;DEFINE QIO SYSTEM SERVICE ARGS
                     0000   261          $SGNDEF                         ;DEFINE SYSGEN PARAMETERS
                     0000   262          $SNDJBCDEF                      ;DEFINE SNDJBC SYSTEM SERVICE ARGS
                     0000   263          $SSDEF                          ;DEFINE SYSTEM STATUS VALUES
                     0000   264          $SYNCHDEF                       ;DEFINE SYNCH SYSTEM SERVICE ARGS
                     0000   265          $UPDSECDEF                      ;DEFINE UPDATE SECTION SYS SRV ARGS
                     0000   266 ;
                     0000   267 ; LOCAL EQUATES
                     0000   268 ;
00000001             0000   269          CAT0 =          1@0
00000080             0000   270          CAT7 =          1@7
00000081             0000   271          DEF_MASK =      CAT0!CAT7       ;INHIBIT FOR 'ALL' AND 'NOT EXIT'
00000080             0000   272          EXC_MASK =      CAT7            ;INHIBIT ONLY FOR 'ALL' CASE
                     0000   273 ;
                     0000   274 ; LOCAL MACROS
                     0000   275 ;
                     0000   276 ;          GSYSSRV - GENERATE SYSTEM SERVICE ENTRY VECTOR
                     0000   277 ;
                     0000   278 ;          GSYSSRV SRVNAME,MODE,NARG,REGISTERS,MASK,NOSYNC
                     0000   279 ;
                     0000   280 ;          WHERE:
                     0000   281 ;                  SRVNAME - SERVICE NAME LESS ANY PREFIX (SYS$,EXE$,RMS$$)
                     0000   282 ;                  MODE - MODE DESIGNATOR FOR SERVICE (K,E,ALL,R)
                     0000   283 ;                  NARG - REQUIRED NUMBER OF ARGUMENTS
                     0000   284 ;                  REGISTERS - REGISTER SAVE LIST
                     0000   285 ;                  MASK - SERVICE INHIBIT MASK(BIT SET IN CAT INHIBITS)
                     0000   286 ;                  NOSYNC - NON-ZERO IF RMS SYNCHRONIZATION CODE NOT TO BE INCLUDED
                     0000   287 ;
                     0000   288
                     0000   289          .MACRO  GSYSSRV,SRVNAME,MODE,NARG,REGS,MASK=DEF_MASK,NOSYNC
                     0000   290          .IF     NDF,RMSSWITCH
                     0000   291          .IF     DF,LIBSWITCH
                     0000   292          .PSECT  $$$0000,QUAD
                     0000   293          .IFF
                     0000   294          .PSECT  $$$000,QUAD
                     0000   295          .ENDC
                     0000   296          .ALIGN  QUAD
                     0000   297          .IF DF  LIBSWITCH
                     0000   298 SYS$'SRVNAME::
                     0000   299          .IFF
                     0000   300          .IF     NDF,MPSWITCH
                     0000   301          .WORD   ^M<REGS>
                     0000   302          SRVNAME'_MASK = ^M<REGS>
                     0000   303          .IFTF   ;MPSWITCH
                     0000   304          .IF B   NOSYNC
```

```
0000    305          SRV'MODE         SRVNAME,NARG,MASK
0000    306          .IFF
0000    307          SRV'MODE         SRVNAME,NARG,MASK,NOSYNC
0000    308          .ENDC
0000    309          .ENDC    ;MPSWITCH
0000    310          .IFT
0000    311          .BLKL    2
0000    312          .ENDC
0000    313          .IFF
0000    314          SRV'MODE         SRVNAME,NARG,MASK
0000    315          .ENDC
0000    316          .ENDM    GSYSSRV
0000    317
0000    318 ;
0000    319 ;        GCOMPSRVB - GENERATE COMPOSITE SYSTEM SERVICE ENTRY VECTOR BEGIN
0000    320 ;
0000    321 ;        GCOMPSRVB  SRVNAME,REGISTER_MASK[,PREFIX]
0000    322 ;
0000    323 ;        WHERE:
0000    324 ;                SRVNAME - SERVICE NAME LESS ANY PREFIX (SYS$, EXE$)
0000    325 ;                REGISTER_MASK - SYMBOLIC REGISTER MASK, E.G QIO_MASK
0000    326 ;                PREFIX - IF SUPPLIED, THE PREFIX FOR THE SERVICE NAME.
0000    327 ;                         IF OMITTED, ''SYS$'' IS ASSUMED.
0000    328 ;
0000    329
0000    330          .MACRO   GCOMPSRVB,SRVNAME,REGMSK,PREFIX=SYS$
0000    331          .IF      NDF,MPSWITCH
0000    332          .IF      NDF,RMSSWITCH
0000    333          .IF      DF,LIBSWITCH
0000    334          .PSECT   $$$0000,QUAD
0000    335          .IFF
0000    336          .PSECT   $$$000,QUAD
0000    337          .ENDC
0000    338          .ALIGN   QUAD
0000    339          .IF DF   LIBSWITCH
0000    340          .IIF     NOT_BLANK, <SRVNAME>,-
0000    341 'PREFIX'SRVNAME::
0000    342          .IFF
0000    343          .ENABL   LSB
0000    344 COMPSTRT=.
0000    345          .IIF     NOT_BLANK, <REGMSK>,-
0000    346          .WORD    <REGMSK>
0000    347          .ENDC
0000    348          .ENDC
0000    349          .ENDC    ;MPSWITCH
0000    350          .ENDM    GCOMPSRVB
0000    351
0000    352 ;
0000    353 ;        GCOMPSRVE - GENERATE COMPOSITE SYSTEM SERVICE ENTRY VECTOR END
0000    354 ;
0000    355 ;        GCOMPSRVE        QUADWORDS
0000    356 ;
0000    357 ;        WHERE:
0000    358 ;                QUADWORDS - NUMBER OF QUADWORDS TO RESERVE FOR VECTOR
0000    359 ;
0000    360
0000    361          .MACRO   GCOMPSRVE,QUADS
```

```
0000    362          .IF     NDF,MPSWITCH
0000    363          .IF     NDF,RMSSWITCH
0000    364          .IF     DF,LIBSWITCH
0000    365          .BLKQ   QUADS
0000    366          .IFF
0000    367 COMPSIZE=.-COMPSTRT
0000    368          .IF     GE,QUADS*8-COMPSIZE
0000    369          .BLKB   QUADS*8-COMPSIZE
0000    370          .IFF
0000    371          .ERROR          ; VECTOR EXCEEDS ALLOCATED SIZE ;
0000    372          .ENDC
0000    373          .DSABL  LSB
0000    374          .ENDC
0000    375          .ENDC
0000    376          .ENDC   ;MPSWITCH
0000    377          .ENDM   GCOMPSRVE
0000    378
0000    379
0000    380 ;
0000    381 ;        SRVK - GENERATE ENTRY FOR KERNEL MODE SERVICE
0000    382 ;
0000    383 ;        SRVK    SRVNAME,NARG,MASK
0000    384 ;
0000    385
0000    386          .MACRO  SRVK,SRVNAME,NARG,MASK
0000    387          .IF     NDF,RMSSWITCH
0000    388          .IF     DF,MPSWITCH
0000    389 CMK$C_'SRVNAME==KCASCTR
0000    390          .IFF    ;MPSWITCH DEFINED
0000    391 CMK$C_'SRVNAME=KCASCTR
0000    392          CHMK    #SRVNAME
0000    393          RET
0000    394          .PSECT  Y$CMODKN,BYTE
0000    395          .=KCASCTR
0000    396          ASSUME NARG LE 127
0000    397          .BYTE   NARG
0000    398          .PSECT  Y$CMODKX,BYTE
0000    399          .=KCASCTR
0000    400          .BYTE   MASK
0000    401          .PSECT  Y$CMODK,BYTE
0000    402          .SIGNED_WORD    EXE$'SRVNAME-KCASE+2
0000    403          .IFTF   ;MPSWITCH
0000    404 SRVNAME=KCASCTR
0000    405 KCASCTR=KCASCTR+1
0000    406          .ENDC   ;MPSWITCH
0000    407          .ENDC
0000    408          .ENDM   SRVK
0000    409
0000    410 ;
0000    411 ;        SRVE - GENERATE ENTRY FOR EXECUTIVE MODE SERVICE
0000    412 ;
0000    413
0000    414          .MACRO  SRVE,SRVNAME,NARG,MASK
0000    415          .IF     NDF,MPSWITCH
0000    416          .IF     NDF,RMSSWITCH
0000    417 CME$C_'SRVNAME=ECASCTR
0000    418          CHME    #SRVNAME
```

```
0000    419          RET
0000    420          .PSECT  Y$CMODEN,BYTE
0000    421          .=ECASCTR
0000    422          ASSUME NARG LE 127
0000    423          .BYTE   NARG
0000    424          .PSECT  Y$CMODEX,BYTE
0000    425          .=ECASCTR
0000    426          .BYTE   MASK
0000    427          .PSECT  Y$CMODE,BYTE
0000    428          .SIGNED_WORD    EXE$'SRVNAME-ECASE+2
0000    429          .ENDC
0000    430 SRVNAME=ECASCTR
0000    431 ECASCTR=ECASCTR+1
0000    432          .ENDC   ;MPSWITCH
0000    433          .ENDM   $RVE
0000    434 ;
0000    435 ;
0000    436 ;    MACROS FOR GENERATING RMS SYSTEM VECTORS
0000    437 ;
0000    438          .MACRO  RMSSRV  SRVNAME NARG=1,REGS=<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>,-
0000    439                          MASK,NOSYNC=0
0000    440          GSYSSRV SRVNAME,R,NARG,<REGS>,MASK,NOSYNC
0000    441          .ENDM   RMSSRV
0000    442 ;
0000    443 ;  SRVR - GENERATE ENTRY FOR RMS SERVICE (EXEC MODE)
0000    444 ;
0000    445          .MACRO  SRVR    SRVNAME,NARG,MASK,NOSYNC
0000    446          .IF     NDF,MPSWITCH
0000    447          .IF     NDF,RMSSWITCH
0000    448 CME$C_'SRVNAME=RCASCTR
0000    449          CHME    #SRVNAME
0000    450          .IF EQ NOSYNC
0000    451          .IIF GT <.+2-RMSSYNC>-127,-
0000    452 RMSSYNC=RMSWBR                              ;RESET BRANCH DESTINATION
0000    453 RMSWBR=.
0000    454          BRB     RMSSYNC
0000    455          .IFF
0000    456          RET
0000    457          .ENDC
0000    458          .PSECT  Y$CMODEN,BYTE
0000    459          .=RCASCTR
0000    460          ASSUME NARG LE 127
0000    461          .BYTE   NARG
0000    462          .PSECT  Y$CMODEX,BYTE
0000    463          .=RCASCTR
0000    464          .BYTE   MASK
0000    465          .IFF
0000    466          .PSECT  $$$RMSVEC,BYTE,NOWRT
0000    467          .SIGNED_WORD    RMS$'SRVNAME-RCASE+2
0000    468          .ENDC
0000    469 SRVNAME=RCASCTR
0000    470 RCASCTR=RCASCTR+1
0000    471          .ENDC   ;MPSWITCH
0000    472          .ENDM   $RVR
0000    473
0000    474 ;
0000    475 ;        SRVALL - GENERATE ENTRY FOR ALL MODE SERVICE
```

```
0000    476 ;
0000    477
0000    478          .MACRO  SRVALL,SRVNAME,NARG,MASK
0000    479          .IF     NDF,MPSWITCH
0000    480          .IF NDF,RMSSWITCH
0000    481          JMP     @#EXE$'SRVNAME+2
0000    482          .ENDC
0000    483          .ENDC   ;MPSWITCH
0000    484          .ENDM   SRVALL
0000    485
```

```
0000    487          .SBTTL  Macros for Loadable Services
0000    488
0000    489 ;
0000    490 ;       LDBSRV - Generate Loadable Service Vector
0000    491 ;
0000    492 ;       LDBSRV  PREFIX,SRVNAME,MODE,REGS,SYN_EFN,SYN_IOSB,ALT_CHMX
0000    493 ;
0000    494 ;       Where:
0000    495 ;               PREFIX          - Prefix for system service vector entry point name
0000    496 ;               SRVNAME         - Service name less any prefix (SYS$,CJF$, etc.)
0000    497 ;               MODE            - Mode designator for service (K,E,ALL)
0000    498 ;               REGS            - Register save list
0000    499 ;               SYN_EFN         - Event flag argument number for $SYNCH
0000    500 ;               SYN_IOSB        - IOSB argument number for $SYNCH
0000    501 ;               ALT_CHMX        - Use same CHMx number as this service
0000    502 ;
0000    503
0000    504          .MACRO  LDBSRV,PREFIX,SRVNAME,MODE,REGS,SYN_EFN,SYN_IOSB,ALT_CHMX
0000    505          .IF NDF,RMSSWITCH
0000    506          .IF NDF,MPSWITCH
0000    507              .IF DF,LIBSWITCH
0000    508                  .PSECT  $$$0000,QUAD
0000    509                  .ALIGN  QUAD
0000    510 PREFIX''SRVNAME::
0000    511                  .IF BLANK SYN_EFN
0000    512                      .BLKL    2
0000    513                  .IFF
0000    514                      .BLKL    4
0000    515                  .ENDC
0000    516              .IFF
0000    517                  .PSECT  $$$000,QUAD
0000    518                  .ALIGN  QUAD
0000    519                  .WORD   ^M<REGS>
0000    520                  SRVNAME'_MASK = ^M<REGS>
0000    521                  LVEC_'MODE PREFIX,SRVNAME,SYN_EFN,SYN_IOSB,ALT_CHMX
0000    522              .ENDC
0000    523          .ENDC   ; MPSWITCH
0000    524          .ENDC   ; RMSSWITCH
0000    525          .ENDM   LDBSRV
0000    526
0000    527 ;
0000    528 ;       LVEC_K  - Kernel Mode Loadable System Service Vector
0000    529 ;
0000    530 ;       LVEC_K  PREFIX,SERVICE,EFN,IOSB
0000    531 ;
0000    532
0000    533          .MACRO  LVEC_K,PREFIX,SERVICE,EFN,IOSB,ALT_CHMK
0000    534          .IF BLANK ALT_CHMK
0000    535              CMK$C_'SERVICE = PREFIX'KCASCTR
0000    536          .IFF
0000    537              CMK$C_'SERVICE = ALT_CHMK
0000    538          .ENDC
0000    539          CHMK #SERVICE
0000    540          .IF NOT BLANK EFN
0000    541              PUSHL       #EFN
0000    542              PUSHL       #IOSB
0000    543              JMP         @#EXE$LDB_SYNCH
```

```
0000    544          .IFF
0000    545              RET
0000    546          .ENDC
0000    547          .IF BLANK ALT_CHMK
0000    548              SERVICE = PREFIX'KCASCTR
0000    549              PREFIX'KCASCTR = PREFIX'KCASCTR + 1
0000    550          .IFF
0000    551              SERVICE = ALT_CHMK
0000    552          .ENDC
0000    553          .ENDM   LVEC_K
0000    554
0000    555 ;
0000    556 ;        LVEC_E  - Exec Mode Loadable System Service Vector
0000    557 ;
0000    558 ;        LVEC_E  PREFIX,SERVICE,EFN,IOSB
0000    559 ;
0000    560
0000    561          .MACRO  LVEC_E,PREFIX,SERVICE,EFN,IOSB,ALT_CHME
0000    562          .IF BLANK ALT_CHME
0000    563              CME$C_'SERVICE = PREFIX'ECASCTR
0000    564          .IFF
0000    565              CME$C_'SERVICE = ALT_CHME
0000    566          .ENDC
0000    567          CHME    #SERVICE
0000    568          .IF NOT_BLANK EFN
0000    569              PUSHL       #EFN
0000    570              PUSHL       #IOSB
0000    571              JMP         @#EXE$LDB_SYNCH
0000    572          .IFF
0000    573              RET
0000    574          .ENDC
0000    575          RET
0000    576          .IF BLANK ALT_CHME
0000    577              SERVICE = PREFIX'ECASCTR
0000    578              PREFIX'ECASCTR = PREFIX'ECASCTR + 1
0000    579          .IFF
0000    580              SERVICE = ALT_CHME
0000    581          .ENDC
0000    582          .ENDM   LVEC_E
0000    583
0000    584 ;
0000    585 ;        LVEC_ALL - Mode of caller Loadable System Service Vector
0000    586 ;
0000    587 ;        LVEC_ALL   PREFIX,SERVICE,EFN,IOSB
0000    588 ;
0000    589          .MACRO  LVEC_ALL,PREFIX,SERVICE,EFN,IOSB,ALT_CHMK
0000    590          JMP     @#EXE$'SERVICE
0000    591          .IF NOT_BLANK EFN
0000    592              .ERROR      ; SYNCH NOT ALLOWED FOR ALL-MODE SERVICES
0000    593          .ENDC
0000    594          .ENDM   LVEC_ALL
0000    595
0000    596
0000    602
0000    603 ;
0000    604 ; GLOBAL SYMBOLS
0000    605 ;
```

```
                     0000   606
00000014             0000   607 EXE$C_CMSTKSZ==4+5                  ;NUMBER OF LONGWORDS IN DISPATCH CALL FRAME
```

```
                     0000   609          .SBTTL  CHANGE MODE TO EXECUTIVE DISPATCHER
                     0000   610 ;+
                     0000   611 ; EXE$CMODEXEC - CHANGE MODE TO EXECUTIVE DISPATCHER
                     0000   612 ;
                     0000   613 ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A CHANGE MODE TO EXECUTIVE
                     0000   614 ; INSTRUCTION IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                     0000   615 ;
                     0000   616 ; INPUTS:
                     0000   617 ;
                     0000   618 ;       00(SP) = CHANGE MODE PARAMETER CODE.
                     0000   619 ;       04(SP) = SAVED PC OF EXCEPTION.
                     0000   620 ;       08(SP) = SAVED PSL OF EXCEPTION.
                     0000   621 ;
                     0000   622 ;       00(AP) = NUMBER OF SYSTEM SERVICE CALL ARGUMENTS.
                     0000   623 ;       04(AP) = FIRST ARGUMENT.
                     0000   624 ;              .
                     0000   625 ;              .
                     0000   626 ;              .
                     0000   627 ;       4*N(AP) = N'TH ARGUMENT.
                     0000   628 ;
                     0000   629 ; OUTPUTS:
                     0000   630 ;
                     0000   631 ;       ***TBS***
                     0000   632 ;
                     0000   633 ; NOTE:
                     0000   634 ;
                     0000   635 ;       DISPATCH TO RMS ROUTINES ASSUMES THAT R3, R4, & R8 ARE NOT DESTROYED
                     0000   636 ;       BY THE THE SERVICE EXIT CODE FOR SUCCESSFUL RETURNS.
                     0000   637 ;-
                     0000   638
                00000000    639          .PSECT  Y$CMODEX,BYTE           ;START OF THE MASK TABLE
                     0000   640 B_EMASK:
                00000000    641          .PSECT  Y$CMODE,QUAD
                     0000   642 EXACCVIO:                                 ;CHANGE MODE TO EXEC ACCESS VIOLATION
          5D    5E   D0  0000  643          MOVL    SP,FP                  ;SET FP TO POINT TO CALL FRAME
    0035'8F     50   B1  0003  644          CMPW    R0,#RCASCTR            ;IS THIS A BUILTIN OR RMS FUNCTION?
           7A        1E  0008  645          BGEQU   EXEDSP                 ;NO, NOT NECESSARILY ACCVIO
        0038'        31  000A  646          BRW     ACCVIO_RET             ;
                     000D  647 EXE$EXCPTNE::                             ;EXECMODE SYSTEM SERVICE EXCEPTION
              0000   000D  648          .WORD   0                      ;NULL ENTRY MASK
                     000F  649          BUG_CHECK SSRVEXCEPT           ;NON-FATAL EXCEPTION IF IN EXEC MODE
      51    04 AC    D0  0013  650          MOVL    CHF$L_SIGARGLST(AP),R1 ;GET ADDRESS OF SIGNAL ARGUMENTS
                     0017  651          $EXIT_S CHF$L_SIG_NAME(R1)     ;AND EXIT WITH SIGNAL AS STATUS
                     0021  652 EXINSARG:                                 ;CHANGE MODE TO EXEC INSUFFICIENT ARGS
    0035'8F     50   B1  0021  653          CMPW    R0,#RCASCTR            ;IS THIS A BUILTIN OR RMS FUNCTION?
           5C        1E  0026  654          BGEQU   EXEDSP                 ;NO, NOT NECESSARILY INSARG
        0025'        31  0028  655          BRW     INSARG                 ;
                     002B  656          .ALIGN  QUAD
                     0030  657 EXE$CMODEXECX::
50  03000000 8F    08 AE   CB  0030  658          BICL3   8(SP),#PSL$M_CURMOD,R0 ;CHECK THE PREVIOUS MODE
           1D        12  0039  659          BNEQ    EXE$CMODEXEC           ;NO CHECK NEEDED FOR NON-USER MODE
          50   6E    9A  003B  660          MOVZBL  (SP),R0                ;PICK UP THE CHME CODE (MOD 256)
 00000000'9F  0000'CF40   93  003E  661          BITB    W^B_EMASK[R0],@#CTL$GB_SSFILTER ;'AND' WITH THE INHIBIT MASK
           0E        13  0048  662          BEQL    EXE$CMODEXEC           ;THIS CODE IS ALLOWED
    51  04D4 8F    3C  004A  663          MOVZWL  #SS$_INHCHME,R1        ;SET THE EXECPTION CODE
        FFB7'        31  004F  664          BRW     INHEXCP                ;AND REFLECT IT
                     0052  665          .ALIGN  QUAD
```

```
                                0058   666 EXE$CMODEXEC::                          ;CHANGE MODE TO EXECUTIVE DISPATCH
                                0058   667                                         ;NOTE: MEMORY WRITING INSTRUCTIONS ARE
                                0058   668                                         ;CAREFULLY INTERLACED WITH REGISTER TO
                                0058   669                                         ;REGISTER OPERATIONS FOR SPEED.
                          50 8EDO   0058   670         POPL    R0                   ;REMOVE CHANGE MODE PARAMETER FROM STACK
                0056'CF   9F    005B   671         PUSHAB  W^SRVEXIT            ;RETURN ADDRESS FOR CALL FRAME
                51    50    9A    005F   672         MOVZBL  R0,R1                ;BOUND RANGE OF CHME CODE VALUES
                      5D    DD    0062   673         PUSHL   FP                   ;SAVE FP
          51    0000'CF41   9A    0064   674         MOVZBL  W^B_EXECNARG[R1] R1  ;GET REQUIRED NUMBER OF ARGUMENTS
                      5C    DD    006A   675         PUSHL   AP                   ;SAVE AP
    5D    00000004 9F41   DE    006C   676         MOVAL   @#4[R1],FP           ;CALCULATE LENGTH OF ARGUMENT LIST
                      7E    7C    0074   677         CLRQ    -(SP)                ;PSW, REGISTER SAVE MASK FOR CALL FRAME
                            0076   678         IFNORD  FP,(AP),EXACCVIO     ;BR IF ARGLIST INACCESSIBLE
                5D    5E    D0    007C   679         MOVL    SP,FP                ;SET FP TO POINT TO CALL FRAME
                51    6C    91    007F   680         CMPB    (AP),R1              ;CHECK FOR REQUIRED NUMBER OF ARGUMENTS
                      9D    1F    0082   681         BLSSU   EXINSARG             ;INSUFFICIENT NUMBER OF ARGUMENTS
                            0084   682                                         ;   (R0 HAS CHME CODE)
          0B'   00    50    AF    0084   683 EXEDSP: CASEW   R0,#0,S^#ECASMAX      ;DISPATCH TO PROPER SERVICE ROUTINE
                    00000000    0088   684 ECASCTR=0                              ;START WITH 0 FOR CHME CODE
                            0088   685 ECASE:                                    ;BASE OF CHME CASE TABLE
                          00000000 686         .PSECT  Y$CMODEN,BYTE        ;REQUIRED NUMBER OF ARG TABLE
                            0000   687 B_EXECNARG:                               ;DEFINE TABLE BASE
                            0000   688
                            0000   689 ;
                            0000   690 ;       NOTE THAT THE OUT OF RANGE FALL THROUGH FROM THE CASEW FOLLOWS
                            0000   691 ;       MANY PAGES LATER IN THIS LISTING (SEE "ILLEGAL CHME" SUBTITLE).
                            0000   692 ;
                            0000   694
                            0000   695
                            0000   696
                            0000   697
                            0000   698 ;
                            0000   699 ; Establish .PSECT for kernel-mode servicing code which follows
                            0000   700 ;
                          00000000 702         .PSECT  Y$CMODK,QUAD
```

```
                        0000    707          .SBTTL  INHEXCP - Inhibited CHMK or CHME code handling
                        0000    708
                        0000    709 ;+
                        0000    710 ;
                        0000    711 ; INHEXCP - Inhibited CHMK or CHME code handling
                        0000    712 ;
                        0000    713 ; FUNCTIONAL DESCRIPTION:
                        0000    714 ;
                        0000    715 ; When the ability to use specified system services is inhibited
                        0000    716 ; via the $SETSSF system service, this routine receives control
                        0000    717 ; when an attempt to execute an inhibited system service occurs.
                        0000    718 ;
                        0000    720 ; INHEXCP  is called when no stack frame cleanup is required.
                        0000    721 ; INHEXCP1 is called when a call frame must be cleared from the stack.
                        0000    722 ;
                        0000    723 ; The result of this code is a signaled exception whose signal arguments are:
                        0000    724 ;       1) SS$_INHCHMK or SS$_INHCHME
                        0000    725 ;       2) the inhibited change mode code whose use was attempted
                        0000    726 ;       3) the offending PC and PSL
                        0000    727 ;
                        0000    728 ; INPUTS:
                        0000    729 ;
                        0000    730 ;    INHEXCP
                        0000    731 ;       R1      = SS error code (SS$_INHCHMK or SS$_INHCHME)
                        0000    732 ;       00(SP) = Change mode parameter code
                        0000    733 ;       04(SP) = Saved PC of exception
                        0000    734 ;       08(SP) = Saved PSL of exception
                        0000    735 ;
                        0000    736 ;    INHEXCP1
                        0000    737 ;       A change mode dispatcher call frame to be cleaned up
                        0000    738 ;       R0      = Change mode parameter code
                        0000    739 ;       R1      = SS error code (SS$_INHCHMK or SS$_INHCHME)
                        0000    740 ;       04(SP) = Saved PC of exception
                        0000    741 ;       08(SP) = Saved PSL of exception
                        0000    742 ;
                        0000    743 ;-
                        0000    762 INHEXCP1:
 5D   0C AE    D0       0000    763          MOVL    12(SP),FP               ;PICK UP THE OLD FP FROM FRAME
    5E    14    C0       0004    764          ADDL    #5*4,SP                 ;CLEAN OFF THE FRAME
          50    DD       0007    765          PUSHL   R0                      ;RESTORE THE CHMX CODE
                         0009    767 INHEXCP:
          51    DD       0009    768          PUSHL   R1                      ;PUSH THE EXECPTION CODE
          04    DD       000B    769          PUSHL   #4                      ;PUSH THE NUMBER OF ARGUMENTS
00000000'GF     17       000D    771          JMP     G^EXE$REFLECT           ;REFLECT THE EXCEPTION
```

```
                     0013   781          .SBTTL  ASTEXIT SYSTEM SERVICE
                     0013   782 ;+
                     0013   783 ; ASTEXIT - SERVICE TO EXIT AN ACTIVE AST AND ALLOW PENDING ASTS TO
                     0013   784 ;           BE DELIVERED.
                     0013   785 ;
                     0013   786 ; THIS SYSTEM SERVICE IS INVOKED WITH A CHMK #ASTEXIT NOT CONTAINED IN
                     0013   787 ; A STANDARD SYSTEM SERVICE VECTOR TO AVOID CLUTTERING THE STACK WITH AN
                     0013   788 ; ADDITIONAL CALL FRAME DURING AST EXIT PROCESSING.
                     0013   789 ;
                     0013   790 ; INPUTS:
                     0013   791 ;       NONE
                     0013   792 ;
                     0013   793 ; OUTPUTS:
                     0013   794 ;       PCB$B_ASTACT IS CLEARED FOR THE ISSUING MODE
                     0013   795 ;       PHD$B_ASTLVL IS SET TO THE ACCESS MODE OF THE NEXT PENDING AST, IF ANY.
                     0013   796 ;
                     0013   797 ;-
                     0013   798
                     0013   799
                     0013   800          .ALIGN  QUAD                        ;** THIS IS ADDED TO FIX
                     0018   801                                              ;** A BROKEN BRANCH INST. -
                     0018   802                                              ;** BEQL ASTEXIT IN EXE$CMODKRNL
                     0018   803
                     0018   804 ASTEXIT:                                     ;EXIT ACTIVE AST
 50   04 AE   02   18   EF  0018   805          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,4(SP),R0 ;GET PREVIOUS MODE
                        52   DD  001E   806          PUSHL   R2                  ;SAVE R2 (PUSHR IS SLOWER!)
                        54   DD  0020   807          PUSHL   R4                  ;SAVE R4
      54    00000000'EF   D0  0022   808          MOVL    SCH$GL_CURPCB,R4    ;GET PCB CURRENT PCB ADDRESS
                              0029   809          SETIPL  #IPL$_ASTDEL        ;DISABLE KERNEL AST DELIVERY
         00 0C A4   50   E7  002C   810          BBCCI   R0,PCB$B_ASTACT(R4),10$ ;CLEAR AST ACTIVE BIT FOR MODE
                 FFCC'   30  0031   811 10$:     BSBW    SCH$NEWLVL          ;COMPUTE NEW AST LEVEL SETTING
                   54 8ED0   0034   812          POPL    R4                  ;RESTORE R4
                   52 8ED0   0037   813          POPL    R2                  ;RESTORE R2
                        02   003A   814          REI                         ;AND EXIT
```

```
                     003B   872          .SBTTL  CHANGE MODE DETECTED ERROR HANDLING
                     003B   873 ;+
                     003B   874 ; ACCVIO - ACCESS VIOLATION DETECTED IN ARGUMENT LIST
                     003B   875 ; INSARG - INSUFFICIENT ARGUMENTS SUPPLIED FOR SERVICE
                     003B   876 ; SSFAIL - ABNORMAL STATUS RETURNED BY SERVICE ROUTINE
                     003B   877 ;
                     003B   878 ; THESE ROUTINES TAKE THE APPROPRIATE ACTION TO RETURN THE ERROR INDICATION
                     003B   879 ; TO THE ORIGINAL CALLER.
                     003B   880 ;
                     003B   881 ;-
                     003B   882          .ENABL  LSB
                     003B   883 ACCVIO:                                      ;
     5D   5E   D0    003B   884          MOVL    SP,FP                      ;SET FRAME POINTER BEFORE RET
0057'8F   50   B1    003E   885          CMPW    R0,#KCASCTR                ;IS THIS AN UNRECOGNIZED CODE?
          78   1E    0043   889          BGEQU   KERDSP                     ;YES, NOT NECESSARILY ACCVIO
                     0045   890 ACCVIO_RET:                                 ;
     50   0C   3C    0045   892          MOVZWL  #SS$_ACCVIO,R0             ;SET ACCESS VIOLATION
                     04     0048   893          RET                        ;
                     0049   894
0057'8F   50   B1    0049   895 KINSARG:CMPW    R0,#KCASCTR                ;IS THIS AN UNRECOGNIZED CODE?
          6D   1E    004E   896 10$:    BGEQU   KERDSP                     ;YES, NOT NECESSARILY INSARG
50   0114 8F   3C    0050   898 INSARG: MOVZWL  #SS$_INSFARG,R0            ;SET INSUFFICIENT NUMBER OF ARGUMENTS
                     04     0055   902          RET                        ;
                     0056   903 SRVEXIT:                                   ;SERVICE EXIT
       07 50   E9    0056   904          BLBC    R0,SSFAIL                  ;BR IF ABNORMAL COMPLETION
                     02     0059   905 SRVREI: REI                         ;
                     005A   907 EXE$EXCPTN::                               ;SYSTEM SERVICE EXCEPTION
               0000  005A   911          .WORD   0                          ;ENTRY MASK
                     005C   913          BUG_CHECK SSRVEXCEPT,FATAL         ;UNEXPECTED SYSTEM SERVICE EXCEPTION
     50   07   D3    0060   917 SSFAIL: BITL    #7,R0                      ;TEST SEVERITY FIELD
          F4   13    0063   918          BEQL    SRVREI                     ;IF EQL WARNING
        0148   31    0065   919          BRW     SSFAILMAIN                 ;GOTO MAIN SSFAIL LOGIC
                     0068   920          .DSABL  LSB
```

```
                     0068   922          .SBTTL  Filtered Change Mode to Kernel Dispatcher
                     0068   923 ;+
                     0068   924 ;
                     0068   926 ; EXE$CMODKRNLX - Filtered Change Mode to Kernel Dispatcher
                     0068   930 ;
                     0068   931 ; When inhibiting of user mode system service calls has been enabled via the
                     0068   933 ; SSINHIBIT SYSGEN parameter, this routine -- not EXE$CMODKRNLX -- is called
                     0068   937 ; whenever a CHMK instruction is executed.  The state of the stack on entry
                     0068   938 ; is:
                     0068   939 ;
                     0068   940 ; INPUTS:
                     0068   941 ;
                     0068   942 ;       00(SP) = CHANGE MODE PARAMETER CODE.
                     0068   943 ;       04(SP) = SAVED PC OF EXCEPTION.
                     0068   944 ;       08(SP) = SAVED PSL OF EXCEPTION.
                     0068   945 ;
                     0068   946 ;       00(AP) = NUMBER OF SYSTEM SERVICE CALL ARGUMENTS.
                     0068   947 ;       04(AP) = FIRST ARGUMENT.
                     0068   948 ;              .
                     0068   949 ;              .
                     0068   950 ;              .
                     0068   951 ;       4*N(AP) = N'TH ARGUMENT.
                     0068   952 ;
                     0068   953 ; OUTPUTS:
                     0068   954 ;
                     0068   955 ;       THE APPROPRIATE KERNEL MODE SYSTEM SERVICE IS INVOKED.
                     0068   956 ;-
                     0068   957
                00000000   959          .PSECT  Y$CMODKX,BYTE           ;START OF THE MASK TABLE
                     0000   960 SYS$GB_KMASK::
                00  0000   961          .BYTE   0                       ;ALLOW FOR ASTEXIT (CHMK #0)!!!
                00000068   962          .PSECT  Y$CMODK,QUAD
                     0068   966
                     0068   967          .ALIGN  QUAD
                     0068   969 EXE$CMODKRNLX::
50   03000000 8F    08 AE   CB  0068   973          BICL3   8(SP),#PSL$M_CURMOD,R0  ;CHECK THE PREVIOUS MODE
                       1D   12  0071   975          BNEQ    EXE$CMODKRNL            ;NO CHECK NEEDED FOR NON-USER MODE
                    50 6E   9A  0073   979          MOVZBL  (SP),R0                 ;PICK UP THE CHMK CODE
 00000000'GF  0000'CF40     93  0076   981          BITB    W^SYS$GB_KMASK[R0],G^CTL$GB_SSFILTER ;'AND' WITH INHIBIT MASK
                       0E   13  0080   982          BEQL    EXE$CMODKRNL            ;THIS CODE IS ALLOWED
            51   04CC 8F    3C  0082   987          MOVZWL  #SS$_INHCHMK,R1         ;SET THE EXECPTION CODE
                 FF7F       31  0087   988          BRW     INHEXCP                 ;AND REFLECT IT
                            008A   989
```

```
                     008A   991          .SBTTL  CHANGE MODE TO KERNEL DISPATCHER
                     008A   992 ;+
                     008A   994 ; EXE$CMODKRNL - CHANGE MODE TO KERNEL DISPATCHER
                     008A   998 ;
                     008A   999 ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A CHANGE MODE TO KERNEL
                     008A  1000 ; INSTRUCTION IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                     008A  1001 ;
                     008A  1002 ; INPUTS:
                     008A  1003 ;
                     008A  1004 ;       00(SP) = CHANGE MODE PARAMETER CODE.
                     008A  1005 ;       04(SP) = SAVED PC OF EXCEPTION.
                     008A  1006 ;       08(SP) = SAVED PSL OF EXCEPTION.
                     008A  1007 ;
                     008A  1008 ;       00(AP) = NUMBER OF SYSTEM SERVICE CALL ARGUMENTS.
                     008A  1009 ;       04(AP) = FIRST ARGUMENT.
                     008A  1010 ;              .
                     008A  1011 ;              .
                     008A  1012 ;              .
                     008A  1013 ;       4*N(AP) = N'TH ARGUMENT.
                     008A  1014 ;
                     008A  1015 ; OUTPUTS:
                     008A  1016 ;
                     008A  1017 ;       THE APPROPRIATE KERNEL MODE SYSTEM SERVICE IS INVOKED.
                     008A  1018 ;-
                     008A  1019
                     008A  1020          .ALIGN  QUAD
                     0090  1022 EXE$CMODKRNL::                           ;CHANGE MODE TO KERNEL DISPATCH
                     0090  1026                                          ;NOTE: MEMORY WRITING INSTRUCTIONS ARE
                     0090  1027                                          ;CAREFULLY INTERLACED WITH REGISTER
                     0090  1028                                          ;INSTRUCTIONS FOR SPEED.
                     0090  1029
                  50 8ED0  0090  1035          POPL    R0                 ;REMOVE CHANGE MODE PARAMETER FROM STACK
                     83    13  0093  1037          BEQL    ASTEXIT            ;IF ZERO, AST EXIT SYSTEM SERVICE
                BE AF    9F  0095  1041          PUSHAB  B^SRVEXIT          ;RETURN ADDRESS
              51   50    9A  0098  1042          MOVZBL  R0,R1              ;BOUND RANGE OF CHMK CODES TO 0,255
                           009B  1043                                     ;AND 256 BYTES ACCESSIBLE FROM B_KRNLNARG
                     5D    DD  009B  1044          PUSHL   FP                 ;SAVE FP
    51    0000'CF41    9A  009D  1046          MOVZBL  W^SYS$GB_KRNLNARG[R1],R1 ;GET NUMBER OF REQUIRED ARGUMENTS
                     5C    DD  00A3  1050          PUSHL   AP                 ;SAVE AP
5D    00000004 9F41    DE  00A5  1051          MOVAL   @#4[R1],FP         ;CALCULATE LENGTH OF ARGUMENT LIST
                     7E    7C  00AD  1052          CLRQ    -(SP)              ;PSW AND REGISTER SAVE MASK
                           00AF  1054          IFNORD  FP,(AP),ACCVIO     ;DECLARE ACCESS VIOLATION
              5D   5E    D0  00B5  1058          MOVL    SP,FP              ;SET FRAME POINTER FOR CALL FRAME
              51   6C    91  00B8  1059          CMPB    (AP),R1            ;CHECK FOR REQUIRED NUMBER OF ARGS
                   8C    1F  00BB  1061          BLSSU   KINSARG            ;IF LSSU, INSUFFICIENT ARGUMENTS
 54   00000000'GF    D0  00BD  1062 KERDSP: MOVL    G^SCH$GL_CURPCB,R4 ;GET CURRENT PROCESS PCB ADDRESS
0055'8F    01    50  AF  00C4  1063          CASEW   R0,#1,#KCASMAX     ;DISPATCH TO PROPER SERVICE ROUTINE
                           00CA  1101 KCASE:                              ;BASE OF CHMK CASE TABLE
                  00000001 00CA  1102 KCASCTR=1                           ;CHMK CODES START AT 1
                           00000000  1104          .PSECT  Y$CMODKN,BYTE      ;REQUIRED NUMBER OF ARG TABLE
                  00000000 0000  1105 SYS$GB_KRNLNARG==.
                        00 0000  1106          .BYTE   0                  ;ENTRY FOR CODE ZERO
```

```
                     0001  1112          .SBTTL  SYSTEM SERVICE VECTOR DEFINITION
                     0001  1113 ;
                     0001  1114 ;
                     0001  1115 ;       DEFINE ALL SYSTEM SERVICE VECTOR POSITIONS
                     0001  1116 ;
                     0001  1117 ;
                     0001  1118
                00000000   1120          .PSECT  $$$000,QUAD
                     0000  1132 VECBASE:                                ;VECTOR AREA BASE
                     0000  1133
                     0000  1134 ;
                     0000  1135 ;       QIO AND WAIT COMPOSITE SERVICE
                     0000  1136 ;
                     0000  1137 ;       THE QIO AND WAITFR COMPOSITE SERVICE OCCUPIES THE FIRST TWO
                     0000  1138 ;       SYSTEM SERVICE VECTOR POSITIONS.  IT IS CONSTRUCTED BY
                     0000  1139 ;       FROM TWO DISCRETE CHMK INSTRUCTIONS, ONE PERFORMING THE QIO
                     0000  1140 ;       AND THE OTHER PERFORMING THE WAITFR, WHICH RELY UPON THE
                     0000  1141 ;       COMPATIBLE ARGUMENT LISTS OF THESE TWO SERVICES. WAITFR HAS
                     0000  1142 ;       A SINGLE ARGUMENT, THE EVENT FLAG, WHICH IS THE FIRST ARGUMENT
                     0000  1143 ;       IN THE QIO ARGUMENT LIST.
                     0000  1144 ;
                     0000  1145
                     0000  1146          GCOMPSRVB QIOW,-                  ;QIO AND WAIT
                     0000  1147                    <QIO_MASK ! WAITFR_MASK ! CLREF_MASK ! SETEF_MASK>
0028'8F    BC        0002  1149          CHMK    #QIO                      ;ISSUE QI/O
  0C 50    E9        0006  1150          BLBC    R0,QIOW_RET               ;DON'T WAIT IF ERROR QUEUEING REQUEST
  10 AC    DD        0009  1151          PUSHL   QIO$_IOSB(AP)             ;FETCH IOSB ADDRESS IF SPECIFIED
   0636    31        000C  1152          BRW     QIO_ENQ_SYNCH             ;USE COMMON QIOW, ENQW SYNCH CODE
                     000F  1154          GCOMPSRVE       2                 ;RESERVE 2 QUADWORDS FOR VECTOR
                     0010  1158
                     0010  1159 ;
                     0010  1160 ;  CONDITION HANDLER DISPATCH VECTOR
                     0010  1161 ;
                     0010  1162 ;  THE FOLLOWING VECTOR IS INCLUDED IN THE SYSTEM VECTOR SPACE SO THAT BOTH
                     0010  1163 ;  HARDWARE-DETECTED (EXCEPTIONS) AND SOFTWARE-DETECTED (SIGNALS) CONDITIONS
                     0010  1164 ;  CAN BE DISPATCHED FROM THE SAME CALL INSTRUCTION. THIS IS NECESSARY SO
                     0010  1165 ;  THAT THE STACK SEARCH ALGORITHM AND THE UNWIND SYSTEM SERVICE CAN DETECT
                     0010  1166 ;  AND PROPERLY PROCESS MULTIPLE ACTIVE SIGNALS AND/OR EXCEPTIONS.
                     0010  1167 ;
                     0010  1168
                     0010  1169          .ALIGN  QUAD
  61   04 AE    FA     0010  1177          CALLG   4(SP),(R1)                ;CALL CONDITION HANDLER
                05     0014  1178          RSB                               ;
                     0015  1179 ;
                     0015  1180 ;  RET INSTRUCTION FOR QIOW ABOVE
                     0015  1181 ;
                     0015  1182 QIOW_RET:
                04   0015  1183          RET
                     0016  1190
                     0016  1191 ;
                     0016  1192 ;  COMMAND INTERPRETER DISPATCH VECTOR
                     0016  1193 ;
                     0016  1194 ;  THE FOLLOWING VECTOR IS INCLUDED IN THE SYSTEM VECTOR SPACE SO THAT DIRECT
                     0016  1195 ;  CALLS CAN BE MADE TO THE CURRENT COMMAND INTERPRETER WITHOUT HAVING TO KNOW
                     0016  1196 ;  THE ADDRESS OF ITS SERVICE ROUTINE.
                     0016  1197 ;
                     0016  1198
```

```
                     0016  1199          .ALIGN  QUAD
                0FFC 0018  1203          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;SAVE R2-R11
0000088F'EF   17  001A  1204          JMP     CLIJMP                      ;INDIRECT DISPATCH TO CURRENT COMMAND INTERP
```

```
                     ;
                     ;       DEFINE REMAINING SERVICES
                     ;
                     
                     GSYSSRV ADJSTK,K,3,-            ;ADJUST OUTER MODE STACK POINTER
                             <R2,R3,R4,R5,R6>,-      ;REGISTERS R2-R6
                             EXC_MASK                ;EXCEPTION MASK
                     GSYSSRV ADJWSL,K,2,-            ;ADJUST WORKING SET LIMIT
                             <R2,R3,R4,R5>           ;REGISTERS R2-R5
                     GSYSSRV ALCDNP,K,4,-            ;ALLOCATE DIAGNOSTIC PAGE
                             <R2,R3,R4,R5,R6,R7>     ;REGISTERS R2-R7
                     GSYSSRV ALLOC,K,4,-             ;ALLOCATE DEVICE
                             <R2,R3,R4,R5,R6>        ;REGISTERS R2-R6
                     GSYSSRV ASCEFC,K,4,-            ;ASSOCIATE COMMON EVENT FLAG CLUSTER
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     GSYSSRV ASCTIM,ALL,3,-          ;CONVERT TO ASCII TIME
                             <R2,R3,R4,R5,R6>        ;REGISTERS R2-R6
                     GSYSSRV ASSIGN,K,4,-            ;ASSIGN I/O CHANNEL
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     GSYSSRV BINTIM,ALL,2,-          ;CONVERT TO BINARY TIME
                             <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
                     GSYSSRV CANCEL,K,1,-            ;CANCEL I/O ON CHANNEL
                             <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
                     GSYSSRV CANTIM,K,2,-            ;CANCEL TIMER REQUEST
                             <R2,R3,R4,R5>           ;REGISTERS R2-R5
                     GSYSSRV CANWAK,K,2,-            ;CANCEL WAKE UP REQUESTS
                             <R2,R3,R4,R5>           ;REGISTERS R2-R5
                     GSYSSRV CRMPSC,K,12,-           ;CREATE AND MAP SECTION
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     GSYSSRV CLRPAR,K,2,-            ;CLEAR HARD PARITY ERROR
                             <R2,R3,R4,R5>           ;REGISTERS R2-R5
                     GSYSSRV CMEXEC,E,2,-            ;CHANGE MODE TO EXECUTIVE
                             <R4>                    ;REGISTER R4
                     GSYSSRV CMKRNL,K,2,-            ;CHANGE MODE TO KERNEL
                             <R4>                    ;REGISTER R4
                     GSYSSRV CLREF,K,1,-             ;CLEAR EVENT FLAG
                             <R2,R3,R4,R5>           ;REGISTERS R2-R5. SEE WAITFR COMMENTS.
                     GSYSSRV CNTREG,K,4,-            ;CONTRACT REGION
                             <R2,R3,R4,R5,R6,R7>     ;REGISTERS R2-R7
                     GSYSSRV GETPTI,K,5,-            ;GET PAGE TABLE INFORMATION
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10> ;REGISTERS R2-R10
                     GSYSSRV CRELOG,ALL,4,-          ;CREATE LOGICAL NAME
                             <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
                     GSYSSRV CREMBX,K,7,-            ;CREATE MAILBOX
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     GSYSSRV CREPRC,K,12,-           ;CREATE PROCESS
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     GSYSSRV CRETVA,K,3,-            ;CREATE VIRTUAL ADDRESS
                             <R2,R3,R4,R5,R6,R7,R8>,-;REGISTERS R2-R8
                             EXC_MASK                ;EXCEPTION MASK
                     GSYSSRV DACEFC,K,1,-            ;DISASSOCIATE EVENT FLAG CLUSTER
                             <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     GSYSSRV DALLOC,K,2,-            ;DEALLOCATE DEVICE
                             <R2,R3,R4,R5,R8>        ;REGISTERS R2-R5,R8
                     GSYSSRV DASSGN,K,1,-            ;DEASSIGN I/O CHANNEL
                             <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
                     GSYSSRV DCLAST,K,3,-            ;DECLARE AST SYSTEM SERVICE
```

```
                00F4  1270                      <R2,R3,R4,R5>                   ;REGISTERS R2-R5
                00F6  1271              GSYSSRV DCLEXH,K,1,-                    ;DECLARE EXIT HANDLER
                00F6  1272                      <R2,R3,R4>                      ;REGISTERS R2-R4
                00F8  1273              GSYSSRV DELLOG,ALL,3,-                  ;DELETE LOGICAL NAME
                00F8  1274                      <R2,R3,R4,R5,R6,R7,R8>          ;REGISTERS R2-R8
                0100  1275              GSYSSRV DELMBX,K,1,-                    ;DELETE MAILBOX
                0100  1276                      <R2,R3,R4,R5>                   ;REGISTERS R2-R5
                00FA  1277              GSYSSRV DELPRC,K,2,-                    ;DELETE PROCESS
                00FA  1278                      <R2,R3,R4,R5,R6,R7>             ;REGISTERS R2-R5
                00FC  1279              GSYSSRV DELTVA,K,3,-                    ;DELETE VIRTUAL ADDRESS
                00FC  1280                      <R2,R3,R4,R5,R6,R7>,-           ;REGISTERS R2-R7
                00FC  1281                      EXC_MASK                        ;EXCEPTION MASK
                00FE  1282              GSYSSRV DGBLSC,K,3,-                    ;DELETE GLOBAL SECTION
                00FE  1283                      <R2,R3,R4,R5,R6,R7,R8,R9,R10> ;REGISTERS R2-R10
                0100  1284              GSYSSRV DLCDNP,K,2,-                    ;DEALLOCATE DIAGNOSTIC PAGE
                0100  1285                      <R2,R3,R4,R5,R6,R7>             ;REGISTERS R2-R7
                0102  1286              GSYSSRV DLCEFC,K,1,-                    ;DELETE COMMON EVENT CLUSTER
                0102  1287                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                0104  1288              GSYSSRV UPDSEC,K,8,-                    ;UPDATE SECTION FILE
                0104  1289                      <R2,R3,R4,R5,R6,R7,R8>          ;R2-R8
                0106  1290              GSYSSRV SNDERR,K,1,-                    ;SEND MSG TO ERROR LOGGER
                0106  1291                      <R2,R3,R4,R5>                   ;REGISTERS R2-R5
                0108  1292              GSYSSRV EXIT,K,1,-                      ;IMAGE EXIT
                0108  1293                      <R4>,0                          ;REGISTER R4, ALWAYS ALLOWED!
                010A  1294              GSYSSRV EXPREG,K,4,-                    ;EXPAND PROGRAM REGION
                010A  1295                      <R2,R3,R4,R5,R6,R7,R8>          ;REGISTERS R2-R8
                010C  1296              GSYSSRV FAO,ALL,0,-                     ;FORMAT ASCII OUTPUT
                010C  1297                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; REGISTERS R2-R11
                0158  1298              GSYSSRV FAOL,ALL,0,-                    ;FORMAT ASCII OUTPUT WITH VALUE LIST
                0158  1299                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                0160  1300              GSYSSRV FORCEX,K,3,-                    ;FORCE EXIT
                0160  1301                      <R2,R3,R4,R5>                   ;REGISTERS R2-R5
                010E  1302              GSYSSRV IMGSTA,ALL,6,-                  ;IMAGE STARTUP
                010E  1303                      <>                              ;REGISTERS NONE
                0170  1304              GSYSSRV SNDJBC,E,7,-                    ;SEND TO JOB CONTROLLER
                0170  1305                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                008C  1306              GSYSSRV GETTIM,E,1,-                    ;GET TIME
                008C  1307                      <>                              ;NO REGISTERS
                008E  1308              GCOMPSRVB UPDSECW,-                     ;UPDATE SECTION AND WAIT
                008E  1309                      <UPDSEC_MASK ! GETJPI_SYNCH_MASK>
000001EC'9F     17  0182  1313          JMP     @#EXE$UPDSECW
                0188  1317              GCOMPSRVE       1
                0188  1318              GSYSSRV HIBER,K,0,-                     ;HIBERNATE
                0188  1319                      <R2,R3,R4,R5>                   ;REGISTERS R2-R5
                0110  1320              GSYSSRV IMGACT,E,8,-                    ;IMAGE ACTIVATION
                0110  1321                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                0090  1322              GSYSSRV LCKPAG,K,3,-                    ;LOCK PAGE IN MEMORY
                0090  1323                      <R2,R3,R4,R5,R6,R7,R8>          ;REGISTERS R2-R8
                0112  1324              GSYSSRV LKWSET,K,3,-                    ;LOCK PAGES IN WORKING SET
                0112  1325                      <R2,R3,R4,R5,R6,R7,R8>          ;REGISTERS R2-R8
                0114  1326              GSYSSRV MGBLSC,K,7,-                    ;MAP GLOBAL SECTION
                0114  1327                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                0116  1328              GSYSSRV PURGWS,K,1,-                    ;PURGE WORKING SET
                0116  1329                      <R2,R3,R4,R5,R6,R7,R8>          ;R2-R8
                0118  1330              GSYSSRV NUMTIM,E,2,-                    ;CONVERT TIME TO NUMERIC
                0118  1331                      <R2,R3,R4,R5,R6,R7>             ;REGISTERS R2-R7
                0092  1332              GSYSSRV SNDOPR,E,2,-                    ;SEND MSG TO OPERATOR
```

```
                                                N 8
0092  1333                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>       ;REGISTERS R2-R11
0094  1334          GSYSSRV QIO,K,12,-                  ;QUEUE I/O REQUEST
0094  1335                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
011A  1336          GSYSSRV READEF,K,2,-                ;READ EVENT FLAG
011A  1337                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
011C  1338          GSYSSRV RESUME,K,2,-                ;RESUME PROCESS
011C  1339                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
011E  1340          GSYSSRV RUNDWN,K,1,-                ;RUNDOWN
011E  1341                      <R2,R3,R4,R5,R6,R7>     ;REGISTERS R2-R7
0120  1342          GSYSSRV SNDSMB,E,2,-                ;SEND MSG TO SYMBIONT MANAGER
0120  1343                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>       ;REGISTERS R2-R11
0096  1344          GSYSSRV SCHDWK,K,4,-                ;SCHEDULE WAKEUP
0096  1345                      <R2,R3,R4,R5,R6,R7,R8,R9> ;REGISTERS R2-R9
0122  1346          GSYSSRV SETAST,K,1,-                ;SET AST ENABLE SERVICE
0122  1347                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
0124  1348          GSYSSRV SETEF,K,1,-                 ;SET EVENT FLAG
0124  1349                      <R2,R3,R4,R5>           ;REGISTERS R2-R5. SEE WAITFR COMMENTS.
0126  1350          GSYSSRV SETEXV,K,4,-                ;SET EXCEPTION VECTOR
0126  1351                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
0128  1352          GSYSSRV SETPRN,K,1,-                ;SET PROCESS  NAME
0128  1353                      <R2,R3,R4,R5,R6,R7,R8,R9> ;REGISTERS R2-R9
012A  1354          GSYSSRV SETPRA,K,2,-                ;SET POWER RECOVERY AST
012A  1355                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
012C  1356          GSYSSRV SETIMR,K,4,-                ;SET TIMER
012C  1357                      <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
012E  1358          GSYSSRV SETPRI,K,4,-                ;SET PROCESS PRIORITY
012E  1359                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
0130  1360          GSYSSRV SETPRT,K,5,-                ;SET PAGE PROTECTION
0130  1361                      <R2,R3,R4,R5,R6,R7,R8,R9> ;REGISTERS R2-R9
0132  1362          GSYSSRV SETRWM,K,1,-                ;SET RESOURCE WAIT MODE
0132  1363                      <R4>                    ;REGISTER R4
0134  1364          GSYSSRV SETSFM,K,1,-                ;SET SYSTEM SERVICE FAILURE MODE
0134  1365                      <R4>,EXC_MASK           ;REGISTER R4, AND EXECPTION MASK
0136  1366          GSYSSRV SETSWM,K,1,-                ;SET PROCESS SWAP MODE
0136  1367                      <R4>                    ;REGISTER R4
0138  1368          GSYSSRV SUSPND,K,2,-                ;SUSPEND PROCESS
0138  1369                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
013A  1370          GSYSSRV TRNLOG,ALL,6,-              ;TRANSLATE LOGICAL NAME
013A  1371                      <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
0260  1372          GSYSSRV ULKPAG,K,3,-                ;UNLOCK PAGE FROM MEMORY
0260  1373                      <R2,R3,R4,R5,R6,R7,R8>  ,REGISTERS R2-R8
013C  1374          GSYSSRV ULWSET,K,3,-                ;UNLOCK PAGES FROM WORKING SET
013C  1375                      <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
013E  1376          GSYSSRV UNWIND,ALL,2,-              ;UNWIND PROCEDURE CALL STACK
013E  1377                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
0278  1378          GSYSSRV WAITFR,K,1,-                ;WAIT FOR EVENT FLAG
0278  1379                      <R2,R3,R4,R5,R6>        ;REGISTERS R2-R6.  IF R8 IS EVER USED
0140  1380                                              ;THE RMS SYCHRONIZATION CODE MUST BE
0140  1381                                              ;MODIFIED TO SAVE IT ALSO.
0140  1382          GSYSSRV WAKE,K,2,-                  ;WAKE PROCESS
0140  1383                      <R2,R3,R4,R5>           ;REGISTERS R2-R5
0142  1384          GSYSSRV WFLAND,K,2,-                ;WAIT FOR LOGICAL AND OF EVENT FLAGS
0142  1385                      <R2,R3,R4,R5,R6>        ;REGISTERS R2-R6
0144  1386          GSYSSRV WFLOR,K,2,-                 ;WAIT FOR LOGICAL OR OF EVENT FLAGS
0144  1387                      <R2,R3,R4,R5,R6>        ;REGISTERS R2-R5
0146  1388          GSYSSRV BRDCST,ALL,2,-              ;BROADCAST TO TERMINALS
0146  1389                      <R2,R3,R4,R5,R6>        ;REGISTERS R2-R6
```

```
02A0  1390          GSYSSRV DCLCMH,K,3,-            ;DECLARE CHANGE MODE HANDLER
02A0  1391                  <R4>                    ;SAVE R4
0148  1392          GSYSSRV SETPFM,K,4,-            ;SET PAGE FAULT MONITORING
0148  1393                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
014A  1394          GSYSSRV GETMSG,ALL,5,-          ;GET MESSAGE
014A  1395                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
02B8  1396          GSYSSRV DERLMB,K,1,-            ;DECLARE ERROR LOG MAILBOX
02B8  1397                  <R2,R3,R4,R5>           ;REGISTERS R2-R5
014C  1398          GSYSSRV CANEXH,K,1,-            ;CANCEL EXIT HANDLER
014C  1399                  <R2,R3,R4,R5>           ;REGISTERS R2-R5
014E  1400          GSYSSRV GETCHN,K,5,-            ;GET CHANNEL INFORMATION
014E  1401                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
0150  1402          GSYSSRV GETDEV,K,5,-            ;GET DEVICE INFORMATION
0150  1403                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
0152  1404          GSYSSRV GETJPI,K,7,-            ;GET JOB PROCESS INFORMATION
0152  1405                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
0154  1406          GSYSSRV PUTMSG,ALL,3,-          ;PUT FORMATTED ERROR MESSAGE
0154  1407                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
02E8  1408          GSYSSRV EXCMSG,ALL,2,-          ;OUTPUT EXCEPTION SUMMARY MESSAGE
02E8  1409                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
02F0  1410          GSYSSRV SNDACC,E,2,-            ;SEND MSG TO ACOUNTING MANAGER
02F0  1411                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
0098  1412          GSYSSRV SETIME,K,1,-            ;SET SYSTEM TIME
0098  1413                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
0156  1414          GSYSSRV SETPRV,K,4,-            ;SET PRIVILEGES
0156  1415                  <R2,R3,R4,R5,R6,R7,R8>  ;REGISTERS R2-R8
```

```
                     0158  1417 ;
                     0158  1418 ;       SPECIAL VECTORS FOR AST DELIVERY AND CLEARING
                     0158  1419 ;
                     0158  1420 ;       SYS$CLRAST CLEARS THE CURRENTLY ACTIVE AST STATUS
                     0158  1421 ;
                     0158  1422 ;       SYS$GL_ASTRET CONTAINS THE VALUE OF THE RETURN ADDRESS FROM
                     0158  1423 ;       THE CALL INSTRUCTION USED TO DISPATCH AN AST. THIS VALUE CAN
                     0158  1424 ;       BE USED WHEN SEARCHING UP THE STACK FOR THE AST CALL FRAME.
                     0158  1425 ;
                00000307   1431         .PSECT  $$$000,QUAD
                     0307  1433         .ALIGN  QUAD
                0000 0308  1438         .WORD   ^M<>                    ;SAVE NO REGISTERS
          0000'8F    BC   030A  1439    CHMK    #CLRAST                 ;DO SPECIAL CHMK
                     04   030E  1440    RET                             ;AND RETURN
            00000000  030F  1441 CLRAST=0                               ;
                     030F  1443         .ALIGN  QUAD                    ;
            00000000' 0310  1450         .LONG   EXE$ASTRET              ;RETURN ADDRESS FROM AST DISPATCHING
                     0314  1451                                         ;CALL
            00000000' 0314  1452         .LONG   CTL$GQ_COMMON           ;ADDRESS OF "CORE COMMON" DESCRIPTOR
                     0318  1454
                     0318  1455 ;
                     0318  1456 ; ENTRY VECTOR FOR CONDITION HANDLER SEARCH. LIB$SIGNAL USES THIS VECTOR
                     0318  1457 ; TO SHARE EXCEPTION'S CODE TO SEARCH FOR AND CALL CONDITION HANDLERS.
                     0318  1458 ; THIS ENTRY IS NOT CALLED; RATHER, IT IS JUMPED TO. NO RETURN IS MADE.
                     0318  1459 ;
                     0318  1460
                     0318  1461         .ALIGN  QUAD
00000000'9F    17    0318  1465         JMP     @#EXE$SRCHANDLER        ;JUMP TO COMMON CODE
                     031E  1469
                     031E  1471 ;
                     031E  1472 ;  NOTE THAT THE CODE IN PSECT $$$000 AT THIS POINT CANNOT EXCEED 320 (HEX)
                     031E  1473 ;  WITHOUT MODIFYING THE RMS SYNCHRONIZATION CODE WHICH PRECEDES THE RMS
                     031E  1474 ;  VECTORS WHICH CANNOT BE MOVED.
                     031E  1475 ;
                     031E  1476
```

```
                     031E  1478 ;
                     031E  1479 ; Set up the base for the RMS service codes. We leave a hole so that
                     031E  1480 ; other exec mode system services can be defined later in this module.
                     031E  1481 ; The hole is defined by the offset between ECASCTR and RCASCTR; it
                     031E  1482 ; is checked with an ASSUME at the end of all service definitions.
                     031E  1483 ;
00000012             031E  1485 RCASCTR=ECASCTR+10
                     031E  1487
00000012             031E  1503 RCASMIN=RCASCTR
```

```
                     031E  1507 ;++
                     031E  1508 ;
                     031E  1509 ;  RMS SERVICES
                     031E  1510 ;
                     031E  1511 ;
                     031E  1512 ;  RMS SYNCHRONIZATION ROUTINE
                     031E  1513 ;
                     031E  1514 ;  THE FOLLOWING ROUTINE IS USED BY THE VARIOUS RMS SERVICES IN ORDER
                     031E  1515 ;  TO AWAIT I/O COMPLETION.  THE ROUTINE IS IN THE VECTOR AREA IN ORDER
                     031E  1516 ;  TO WAIT AT THE CALLER'S MODE,  THUS ALLOWING AST ACTIVITY FOR EITHER
                     031E  1517 ;  USER OR SUPERVISOR MODE, OR BOTH.
                     031E  1518 ;
                     031E  1519 ;  THE FAB/RAB IS CHECKED FOR A LEGAL BLOCK ID, I.E., A 1 OR 3, AND
                     031E  1520 ;  AN ERROR RETURNED IF INVALID.  THE STRUCTURE IS NOT REPROBED.
                     031E  1521 ;
                     031E  1522 ;  NOTE THAT EACH RMS SERVICE VECTOR TERMINATES WITH A BRANCH TO THIS
                     031E  1523 ;  ROUTINE.
                     031E  1524 ;
                     031E  1525 ;  THIS ROUTINE ASSUMES THAT THE FOLLOWING REGISTERS HAVE BEEN SET BY THE
                     031E  1526 ;  EXITING RMS EXEC-LEVEL CODE WHENEVER A STALL IS REQUIRED:
                     031E  1527 ;
                     031E  1528 ;       R3      EFN TO WAIT ON
                     031E  1529 ;       R8      RAB/FAB ADDRESS TO WAIT ON
                     031E  1530 ;       R4      (RMSWAIT_BR ENTRY POINT ONLY, $WAIT SERVICE) FLAG FOR WAIT TYPE
                     031E  1531 ;               (0 = SAME RAB, 1 = DIFFERENT RABS)
                     031E  1532 ;
                     031E  1533 ;--
           00000031E 031E  1535         .PSECT  $$$000,QUAD
  00000320 00000320  031E  1539         .BLKB   ^X320-<.-VECBASE>
                     0320  1541 RMSWAIT_IO_DONE:
                     0320  1542 ;
                     0320  1543 ;  SET A FLAG IN THE USER'S CONTROL BLOCK THAT TELLS RMS THAT THE PROCESS
                     0320  1544 ;  IS WAITING ON THIS FAB/RAB.  WHEN RMS IS INITIALIZING FOR A NEW OPERATION
                     0320  1545 ;  IT CHECKS THIS FLAG AND REJECTS THE NEW OPERATION IF THE CONTROL BLOCK
                     0320  1546 ;  IS WAITING ON A PREVIOUS OPERATION.  THIS PREVENTS A HANG CONDITION
                     0320  1547 ;  CAUSED BY USING THE SAME STS/STV FIELD FOR 2 OPERATIONS AT ONCE.
                     0320  1548 ;  FAB$B_BLN = RAB$B_BLN
                     0320  1549 ;
01 A8    01    88    0320  1550         BISB    #1,RAB$B_BLN(R8)        ;LOW BIT OF BLN FIELD IS THE FLAG
                     0324  1551
                     0324  1552 ;
                     0324  1553 ;  THE ARGUMENTS ARE PUSHED ON THE STACK AND THE AP SET UP AS IF A 'CALLS'
                     0324  1554 ;  INSTRUCTION WERE BEING EXECUTED.  THE CHANGE MODE TO KERNEL SERVICE IS
                     0324  1555 ;  EXECUTED DIRECTLY.  THIS SAVES THE OVERHEAD OF A 'CALLS' INSTRUCTION.
                     0324  1556 ;  R8 MUST NOT BE DESTROYED BY ANY OF THE SERVICES USED HERE.
                     0324  1557 ;
      53    DD       0324  1558         PUSHL   R3                      ;EVENT FLAG TO WAIT FOR
5C    FC AE    9E    0326  1559         MOVAB   -4(SP),AP               ;SET UP AP AS IF USING CALLS INSTR.
        01    DD     032A  1560         PUSHL   #1                      ;NUMBER OF ARGUMENTS
                     032C  1561 USERWAIT:
   003B'8F    BC     032C  1562         CHMK    I^#WAITFR               ;DO 'NAKED' WAITFR TO SAVE CALLS TIME
                     0330  1563 ;
                     0330  1564 ;  CHECK TO SEE IF THE USER STRUCTURE POINTED TO BY R8 IS STILL VALID BY
                     0330  1565 ;  CHECKING THE BLOCK ID TO BE SURE THAT IT IS EITHER A RAB (BID=1) OR
                     0330  1566 ;  A FAB (BID=3).  THIS WON'T CATCH THE CASE WHERE WHAT SHOULD HAVE BEEN
                     0330  1567 ;  A FAB NOW LOOKS LIKE A RAB OR VICE VERSA  BUT WILL CATCH EVERYTHING
                     0330  1568 ;  ELSE.  IF THE STRUCTURE IS NOT READABLE OR WRITEABLE THEN THE USER
```

```
                     0330  1569 ;  WILL GET AN ACCESS VIOLATION.  THE BID FOR A FAB/RAB IS AT BYTE 0,
                     0330  1570 ;  THE STS FOR A FAB/RAB IS AT BYTE 8.
                     0330  1571 ;
      23 68    E9    0330  1572 10$:    BLBC    (R8),30$                ;NOT SET, THEN NOT A FAB OR RAB
68    FC 8F    93    0333  1573         BITB    #^B11111100,(R8)        ;IS IT A 1 OR 3?
         1D    12    0337  1574         BNEQ    30$                     ;NEQ NO SO BLOW THE WHISTLE
50    08 A8    D0    0339  1575         MOVL    8(R8),R0                ;GET RMS STATUS CODE
         08    13    033D  1576         BEQL    20$                     ;AND WAIT AGAIN IF NOT SET
01 A8    01    8A    033F  1577         BICB    #1,RAB$B_BLN(R8)        ;CLEAR WAITING FLAG
   10 50       E9    0343  1578         BLBC    R0,30$                  ;BRANCH IF FAILURE CODE
               04    0346  1579         RET                             ;RETURN TO CALLER
                     0347  1580 ;
                     0347  1581 ;  CLEAR THE RMS EVENT FLAG, CHECK STATUS AGAIN AND WAIT 1 MORE TIME IF
                     0347  1582 ;  OPERATION STILL NOT DONE.  THE APPROPRIATE ARGUMENTS FOR THE CLREF
                     0347  1583 ;  AND SETEF (IF EXECUTED) REMAIN ON THE STACK FROM THE WAITFR ABOVE.
                     0347  1584 ;  THE AP MUST BE PRESERVED.
                     0347  1585 ;
000D'8F              BC    0347  1586 20$:    CHMK    I^#CLREF                ;DO A 'NAKED' CLREF, THE ARGUMENTS
                     034B  1587                                             ;ARE ON STACK AND AP STILL SET UP
                     034B  1588                                             ;FROM THE WAITFR ABOVE
   08 A8             D5    034B  1589         TSTL    8(R8)                   ;AND RE-CHECK STATUS
      DC             13    034E  1590         BEQL    USERWAIT                ;BRANCH TO WAIT FOR FLAG AGAIN..
                     0350  1591                                             ;... IF STATUS STILL ZERO
002E'8F              BC    0350  1592         CHMK    I^#SETEF                ;I/O COMPLETE - LEAVE EFN SET
      DA             11    0354  1593         BRB     10$                     ;AND RESTORE R0 STATUS CODE
                     0356  1594 ;
                     0356  1595 ;  BRANCH TO CHECK STATUS CODE FOR ERROR OR SEVERE ERROR
                     0356  1596 ;  A SUCCESS STATUS IN R0 (FROM THE $WAITFR) INDICATES AN INVALID FAB/RAB.
                     0356  1597 ;
   0127              31    0356  1598 30$:    BRW     RMS_ERR_BR
                     0359  1599 ;
                     0359  1600 ;  ENTRY HERE FROM $WAIT SERVICE.  THIS SERVES AS AN EXTENDED BRANCH
                     0359  1601 ;  TO THE $WAIT SYNCHRONIZATION CODE IN THE Y$CMODE PSECT.
                     0359  1602 ;
                     0359  1603 RMSWAIT_BR:
000000D5'9F          16    0359  1604         JSB     @#RMS_WAIT_SYNC                 ;DO $WAIT SYNCHRONIZATION
                     035F  1605 ;
                     035F  1606 ;
                     035F  1607 ;  ENTRY HERE FROM EACH VECTOR
                     035F  1608 ;  CHECK FOR POSSIBLE STALL
                     035F  1609 ;
                     035F  1610 RMSCHK_STALL:
0000'8F    50        B1    035F  1611         CMPW    R0,#RMS$_STALL&^XFFFF   ;IS THE STATUS CODE I/O STALL?
      BA             13    0364  1612         BEQL    RMSWAIT_IO_DONE         ;BRANCH IF YES
               04    0366  1613         RET                             ;BACK TO CALLER
                     0367  1614         .ALIGN  QUAD
```

```
                     0368  1621 ;
                     0368  1622 ;
                     0368  1623 ;  DEFINE RMS SERVICES
                     0368  1624 ;
0000035F             0368  1626 RMSSYNC=RMSCHK_STALL
                     0368  1629 ;
                     0368  1630 ;  HIGH USE RECORD OPERATIONS
                     0368  1631 ;
                     0368  1632          RMSSRV  DELETE          ;DELETE A RECORD
                     0013  1633          .NLIST CND
                     0013  1634          RMSSRV  FIND            ;FIND RECORD
                     0014  1635          RMSSRV  FREE            ;RELEASE LOCK ON ALL RECORDS
                     0015  1636          RMSSRV  GET             ;GET A RECORD
                     0016  1637          RMSSRV  PUT             ;PUT A RECORD
                     0017  1638          RMSSRV  READ            ;READ A BLOCK
                     0018  1639          RMSSRV  RELEASE         ;RELEASE LOCK ON NAMED RECORD
                     0019  1640          RMSSRV  UPDATE          ;REWRITE EXISTING RECORD
00000359             001A  1643 RMSSYNC=RMSWAIT_BR               ;REDEFINE FOR $WAIT ONLY
                     001A  1646          RMSSRV  WAIT            ;STALL FOR RECORD OPERATION COMPLETE
0000035F             001B  1649 RMSSYNC=RMSCHK_STALL             ;RESTORE STANDARD SYNC ADDR
                     001B  1652          RMSSRV  WRITE           ;WRITE BLOCK
                     001C  1653 ;
                     001C  1654 ;  LOWER USAGE OPERATIONS
                     001C  1655 ;
                     001C  1656          RMSSRV  CLOSE           ;CLOSE FILE
                     001D  1657          RMSSRV  CONNECT         ;CONNECT RAB
                     001E  1658          RMSSRV  CREATE          ;CREATE FILE
                     001F  1659          RMSSRV  DISCONNECT      ;DISCONNECT RAB
                     0020  1660          RMSSRV  DISPLAY         ;DISPLAY FILE INFORMATION
                     0021  1661          RMSSRV  ERASE           ;ERASE (DELETE) FILE
                     0022  1662          RMSSRV  EXTEND          ;EXTEND FILE ALLOCATION
                     0023  1663          RMSSRV  FLUSH           ;FINISH I/O ACTIVITY FOR STREAM
                     0024  1664          RMSSRV  MODIFY          ;MODIFY FILE ATTRIBUTES
                     0025  1665          RMSSRV  NXTVOL          ;NEXT VOLUME
                     0026  1666          RMSSRV  OPEN            ;OPEN FILE
                     0027  1667          RMSSRV  REWIND          ;REWIND FILE
                     0028  1668          RMSSRV  SPACE           ;POSITION FOR TRANSFER
                     0029  1669          RMSSRV  TRUNCATE        ;TRUNCATE FILE
                     002A  1670          RMSSRV  ENTER           ;ENTER FILENAME INTO DIRECTORY
                     002B  1671          RMSSRV  PARSE           ;PARSE FILENAME SPECIFICATION
                     002C  1672          RMSSRV  REMOVE          ;REMOVE FILENAME FROM DIRECTORY
                     002D  1673          RMSSRV  RENAME,NARG=4   ;RENAME A FILE
                     002E  1674          RMSSRV  SEARCH          ;SEARCH A FILE DIRECTORY
                     002F  1675          RMSSRV  SETDDIR,NARG=3,NOSYNC=1
                     0030  1676                                  ;SET DEFAULT DIRECTORY STRING
                     0030  1677          RMSSRV  SETDFPROT,REGS=<R2,R3>,NARG=2,NOSYNC=1
                     0031  1678                                  ;SET DEFAULT FILE PROTECTION MASK
                     0031  1679          RMSSRV  SSVEXC,REGS=<>,NOSYNC=1
                     0032  1680                                  ;GENERATE SYS SERV EXCEPTION
                     0032  1681          RMSSRV  RMSRUNDWN,NARG=2,NOSYNC=1
                     0033  1682                                  ;PERFORM RUNDOWN ON RMS FILES
                     0033  1683          RMSSRV  RMSRUHNDLR,NARG=5,NOSYNC=1
                     0034  1684                                  ;RMS Recovery Unit Handler
                     0034  1685          RMSSRV  FILESCAN,NARG=3,NOSYNC=1
                     0035  1686                                  ;Perform syntax check for file specs
                     0035  1687 ;
                     0035  1688 ; ADD NEW RMS SERVICES IN FRONT OF THIS CODE!
```

```
                     0035  1689 ;
                     0035  1690 ; Now we add special non-vector code.  Because of the CASE instruction
                     0035  1691 ; used at the front of RMS, this code (and any future additional code)
                     0035  1692 ; must be the last element of the RMS area.
                     0035  1693 ;
                     0035  1694
                     0035  1695         GCOMPSRVB                   ;Helper branch to error processing
                     0480  1699 RMS_ERR_BR:
000000F2'9F    17    0480  1700         JMP     @#RMS_ERR
                     0486  1704         GCOMPSRVE       1
                     0488  1705
                     0488  1707
                     0488  1708 ; NOTE:  RMSVECEND MARKS THE END OF THE CURRENTLY DEFINED RMS VECTORS.
                     0488  1709 ;        SSVECREG2 MARKS THE START OF THE SECOND REGION OF SYSTEM
                     0488  1710 ;        SERVICE VECTORS.  THERE IS EMPTY SPACE BETWEEN THESE REGIONS
                     0488  1711 ;        FOR FUTURE RMS VECTORS.  IF NECESSARY, THIS SPACE CAN ALSO
                     0488  1712 ;        BE USED FOR SYSTEM SERVICE VECTORS BY BACKING UP SSVECREG2
                     0488  1713 ;        (TOWARDS THE RMS VECTORS) AND ADDING NEW SYSTEM SERVICE VECTORS
                     0488  1714 ;        BEFORE THE ALREADY DEFINED ONES.  IN OTHER WORDS, THESE TWO
                     0488  1715 ;        VECTOR REGIONS MAY GROW TOWARDS EACH OTHER.  IF THEY COLLIDE,
                     0488  1716 ;        AN ASSEMBLY ERROR IS GENERATED.
                     0488  1717
                00000488  1721         .PSECT  $$$000,QUAD                 ; CMODSSDSP
                     0488  1723
                     0488  1724 RMSVECEND:
                000005C0  1725 .=VECBASE+^X5C0
                     05C0  1726 SSVECREG2:                          ; START OF SYSTEM SERVICE VECTOR REGION 2
                     05C0  1732
```

```
                     05C0  1734          .SBTTL  REGION 2 OF SYS. SERV. VECTOR DEFINITIONS
                     05C0  1735
                     05C0  1736 ;
                     05C0  1737 ; Note: Service codes for exec mode services in this region are
                     05C0  1738 ; reserved by the offset defined above between RCASCTR and ECASCTR.
                     05C0  1739 ; If the ASSUME at the end of this section breaks, the offset must
                     05C0  1740 ; be increased.
                     05C0  1741 ;
                     05C0  1742
                     05C0  1743          GSYSSRV ENQ,K,11,-                      ; ENQUEUE
                     05C0  1744                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; REGISTERS R2-R11
                     015A  1745          GSYSSRV DEQ,K,4,-                       ; DEQUEUE
                     015A  1746                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; REGISTERS R2-R11
                     015C  1747          GCOMPSRVB ENQW,-                        ; ENQUEUE AND WAIT
                     015C  1748                  <ENQ_MASK ! WAITFR_MASK ! CLREF_MASK ! SETEF_MASK>
        0048 8F   BC 05D2  1752          CHMK    #ENQ                            ; EXECUTE ENQ SYSTEM SERVICE
0689 8F    50   B1 05D6  1753          CMPW    R0,#SS$_SYNCH                   ; IF COMPLETED SYNCHRONOUSLY
           01   12 05DB  1754          BNEQ    10$
           04   05DD  1755 5$:        RET                                     ; THEN RETURN WITHOUT ANY WAITING
      FC 50   E9 05DE  1756 10$:      BLBC    R0,5$                           ; DON'T WAIT IF ERROR
      0C AC   DD 05E1  1757           PUSHL   ENQ$_LKSB(AP)                   ; OTHERWISE GET IOSB ADDRESS IF SPECIFIED
         5F   11 05E4  1758           BRB     QIO_ENQ_SYNCH                   ; AND USE COMMON SYNCH CODE
                     05E6  1762          GCOMPSRVE       3                       ; RESERVE 3 QUADWORDS FOR VECTOR
                     05E8  1763          GSYSSRV SETSSF,K,1,-                    ; SET SYSTEM SERVICE FILTER MASK
                     05E8  1764                  <R4>                            ; REGISTER R4
                     015E  1765          GSYSSRV SETSTK,K,3,-                    ; SET STACK LIMITS
                     015E  1766                  <R2,R3,R4>                      ; REGISTERS R2,R3,R4
                     0160  1767          GSYSSRV GETSYI,K,7,-                    ; GET SYSTEM INFORMATION
                     0160  1768                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; REGISTERS R2-R11
                     0162  1769          GSYSSRV IMGFIX,ALL,0,-                  ; IMAGE ADDRESS RELOCATION FIXUP
                     0162  1770                  <R2,R3,R4,R5>                   ; REGISTERS R2-R5
                     0608  1771          GCOMPSRVB       IMGFIX_2,-              ; ********** TEMP **********
                     0608  1772                  <0>
                     060A  1773          GCOMPSRVE       1                       ; ********** TEMP **********
                     0610  1774          GSYSSRV GETDVI,K,8,-                    ; GET DEVICE AND VOLUME INFORMATION
                     0610  1775                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; REGISTERS R2-R11
                     0164  1776          GCOMPSRVB GETDVIW,-                     ; GET DEVICE INFORMATION AND WAIT
                     0164  1777                  <GETDVI_MASK ! GETJPI_SYNCH_MASK>
   004D'8F    BC 061A  1781          CHMK    I^#GETDVI
         06   11 061E  1782          BRB     GETJPI_COMMON
                     0620  1786          GCOMPSRVE       1
                     0620  1787          GCOMPSRVB GETJPIW,-                     ; GET JOB/PROCESS INFORMATION AND WAIT
                     0620  1788                  <GETJPI_MASK ! GETJPI_SYNCH_MASK>
   0045'8F    BC 0622  1792          CHMK    I^#GETJPI
                     0626  1793 GETJPI_COMMON:
000001FC'9F    17 0626  1794          JMP     @#GETJPI_SYNCH
                     062C  1798          GCOMPSRVE       2
                     0630  1799          GCOMPSRVB GETSYIW,-                     ; GET SYSTEM INFORMATION AND WAIT
                     0630  1800                  <GETSYI_MASK ! GETJPI_SYNCH_MASK>
   004C'8F    BC 0632  1804          CHMK    I^#GETSYI
         EE   11 0636  1805          BRB     GETJPI_COMMON
                     0638  1809          GCOMPSRVE       1
                     0638  1810          GCOMPSRVB SNDJBCW,-                     ; SEND TO JOB CONTROLLER AND WAIT
                     0638  1811                  <SNDJBC_MASK ! GETJPI_SYNCH_MASK>
   0001'8F    BD 063A  1815          CHME    I^#SNDJBC                       ; SEND TO JOB CONTROLLER
         E6   11 063E  1816          BRB     GETJPI_COMMON
                     0640  1820          GCOMPSRVE       1
```

```
                     0640  1821          GCOMPSRVB SYNCH,-                       ; SYNCHRONIZE EFN AND IOSB
                     0640  1822                    <WAITFR_MASK ! CLREF_MASK ! SETEF_MASK>
 08 AC    DD         0642  1826          PUSHL   SYNCH$_IOSB(AP)                 ; GET ADDRESS OF IOSB IF SPECIFIED
                     0645  1827 ;
                     0645  1828 ; CONDITION CODES SET FROM PUSH OF IOSB ADR ONTO STACK
                     0645  1829 ; THE EFN STATE AND IOSB STATUS MAY HAVE ONLY THE FOLLOWING COMBINATIONS
                     0645  1830 ;       EFN CLEAR, (IOSB) = 0
                     0645  1831 ;       EFN SET, (IOSB) NON ZERO
                     0645  1832 ;       EFN SET, (IOSB) CLEAR - the EFN was set by another I/O operation
                     0645  1833 ;
                     0645  1834 ; IF THE EFN COULD BE CLEAR AND (IOSB) WAS NON-ZERO, THIS SERVICE WOULD
                     0645  1835 ; EXIT WITH THE EVENT FLAG CLEAR WHICH IS NOT CORRECT.
                     0645  1836 ;
                     0645  1837
                     0645  1838 QIO_ENQ_SYNCH:
    23    13         0645  1839          BEQL    50$                             ; BRANCH IF NO IOSB SPECIFIED
 00 BE    B5         0647  1840          TSTW    @(SP)                           ; IS COMPLETION STATUS SET?
    1A    12         064A  1841          BNEQ    40$                             ; BRANCH IF SET
003B'8F    BC         064C  1842 10$:     CHMK    I^#WAITFR                       ; MUST WAIT FOR EFN TO BE SET
 00 BE    B5         0650  1843          TSTW    @(SP)                           ; COMPLETION STATUS SET YET?
    01    13         0653  1844          BEQL    30$                             ; BRANCH IF NOT
          04         0655  1845 20$:     RET                                     ; YES, RETURN STATUS
 FC 50    E9         0656  1846 30$:     BLBC    R0,20$                          ; IF ERROR, RETURN STATUS
000D'8F    BC         0659  1847          CHMK    I^#CLREF                        ; NO, CLEAR EVENT FLAG
 00 BE    B5         065D  1848          TSTW    @(SP)                           ; AND IF STILL NOT DONE
    EA    13         0660  1849          BEQL    10$                             ; WAIT SOME MORE
002E'8F    BC         0662  1850          CHMK    I^#SETEF                        ; OTHERWISE EXIT WITH IT SET
50  01    D0         0666  1851 40$:     MOVL    S^#SS$_NORMAL,R0                ; FORCE NORMAL SUCCESS
          04         0669  1852          RET                                     ; AND RETURN
                     066A  1853 ;
                     066A  1854 ; NO IOSB GIVEN, JUST WAIT FOR THE EVENT FLAG TO BE SET
                     066A  1855 ;
003B'8F    BC         066A  1856 50$:     CHMK    I^#WAITFR                       ; WAIT FOR SPECIFIED EVENT FLAG
          04         066E  1857          RET                                     ; AND RETURN
                     066F  1861          GCOMPSRVE       6                       ; RESERVE 6 QUADWORDS FOR VECTOR
                     0670  1862          GSYSSRV ERAPAT,K,3,-                    ; GENERATE A SECURITY  ERASE PATTERN
                     0670  1863                  <R4>                            ; SAVE R4
                     0166  1864          GSYSSRV CRELNT,K,8,-                    ; CREATE LOGICAL NAME TABLE
                     0166  1865                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     0168  1866          GSYSSRV CRELNM,K,5,-                    ; CREATE LOGICAL NAME
                     0168  1867                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     016A  1868          GSYSSRV DELLNM,K,3,-                    ; DELETE LOGICAL NAME
                     016A  1869                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     016C  1870          GSYSSRV TRNLNM,K,5,-                    ; TRANSLATE LOGICAL NAME
                     016C  1871                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     016E  1872          GSYSSRV GETLKI,K,7,-                    ; GET LOCK INFORMATION
                     016E  1873                  <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     0170  1874          GCOMPSRVB GETLKIW,-                     ; GET LOCK INFORMATION AND WAIT
                     0170  1875                  <GETLKI_MASK ! WAITFR_MASK ! CLREF_MASK ! SETEF_MASK>
0053'8F    BC         06A2  1879          CHMK    I^#GETLKI
 05 50    E9         06A6  1880          BLBC    R0,10$                          ; DON'T WAIT IF ERROR
 10 AC    DD         06A9  1881          PUSHL   GETLKI$_IOSB(AP)                ; OTHERWISE GET IOSB ADDRESS IF SPECIFIED
    97    11         06AC  1882          BRB     QIO_ENQ_SYNCH                   ; AND USE COMMON SYNCH CODE
          04         06AE  1883 10$:     RET                                     ; RETURN ON ERROR
                     06AF  1887          GCOMPSRVE       2                       ; RESERVE 2 QUADWORDS FOR VECTOR
                     06B0  1888
                     06B0  1889          GSYSSRV ASCTOID,E,3,-                   ;ASCII TO IDENTIFIER CONVERSION
```

```
                     06B0  1890                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     009A  1891        GSYSSRV FINISH_RDB,E,1,-          ;FINISH RDB CONTEXT STREAM
                     009A  1892                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     009C  1893        GSYSSRV IDTOASC,E,6,-             ;IDENTIFIER TO ASCII CONVERSION
                     009C  1894                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     009E  1895        GSYSSRV BRKTHRU,K,11,-            ;BREAK THROUGH WRITES
                     009E  1896                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     0172  1897        GSYSSRV GRANTID,ALL,5,-           ;GRANT IDENTIFIER TO PROCESS
                     0172  1898                <R2,R3>                   ;REGISTERS R2-R3
                     06D8  1899        GSYSSRV REVOKID,ALL,5,-           ;REVOKE IDENTIFIER FROM PROCESS
                     06D8  1900                <R2,R3>                   ;REGISTERS R2-R3
                     06E0  1901        GSYSSRV CHKPRO,K,1,-              ;GENERAL PROTECTION CHECK ROUTINE
                     06E0  1902                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     0174  1903        GCOMPSRVB BRKTHRUW,-              ; BREAK THOUGH WRITE AND WAIT
                     0174  1904                <BRKTHRU_MASK ! GETJPI_SYNCH_MASK>
0054'8F    BC        06EA  1908        CHMK    I^#BRKTHRU
  FF35     31        06EE  1909        BRW     GETJPI_COMMON
                     06F1  1913        GCOMPSRVE       2
                     06F8  1914        GSYSSRV GETQUI,E,7,-              ;GET QUEUE INFORMATION
                     06F8  1915                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     00A0  1916        GCOMPSRVB GETQUIW,-               ;GET QUEUE INFORMATION AND WAIT
                     00A0  1917                <GETQUI_MASK ! GETJPI_SYNCH_MASK>
000B'8F    BD        0702  1921        CHME    I^#GETQUI
  FF1D     31        0706  1922        BRW     GETJPI_COMMON
                     0709  1926        GCOMPSRVE       2
                     0710  1927
                     0710  1928 ;
00004028             0710  1929        CJF$KCASCTR = 16424
                     0710  1930 ;
                     0710  1931        LDBSRV  CJF$, ALLJDR,      K,   <R4>
                     0717  1932        LDBSRV  CJF$, ASSJNL,      K,   <R4>
                     071F  1933        LDBSRV  CJF$, CONUIC,      K,   <R4>
                     0727  1934        LDBSRV  CJF$, CREJNL,      K,   <R4>
                     072F  1935        LDBSRV  CJF$, DEALJDR,     K,   <R4>
                     0737  1936        LDBSRV  CJF$, DEASJNL,     ALL, <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                     0740  1937        LDBSRV  CJF$, DEASJNL_INT, K,   <R4>
                     0747  1938        LDBSRV  CJF$, DELJNL,      K,   <R4>
                     074F  1939        LDBSRV  CJF$, DMTJMD,      K,   <R4>
                     0757  1940        LDBSRV  CJF$, DSPJNL,      K,   <R4>
                     075F  1941        LDBSRV  CJF$, GETJNL,      K,   <R4>
                     0767  1942        LDBSRV  CJF$, GETRUI,      K,   <R4>
                     076F  1943        LDBSRV  CJF$, MODFLT,      K,   <R4>
                     0777  1944        LDBSRV  CJF$, POSJNL,      K,   <R4>
                     077F  1945        LDBSRV  CJF$, READJNL,     K,   <R4>
                     0787  1946        LDBSRV  CJF$, RECOVER,     K,   <R4>
                     078F  1947        LDBSRV  CJF$, MNTJMD,      K,   <R4>
                     0797  1948        LDBSRV  CJF$, CRENWV,      K,   <R4>
                     079F  1949        LDBSRV  CJF$, CONJNLF,     K,   <R4>
                     07A7  1950        LDBSRV  CJF$, DCNJNLF,     K,   <R4>
                     07AF  1951        LDBSRV  CJF$, FORCEJNL,    ALL, <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                     07B8  1952        LDBSRV  CJF$, FORCEJNLW,   ALL, <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                     07C0  1953        LDBSRV  CJF$, WRITEJNL,    ALL, <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                     07C8  1954        LDBSRV  CJF$, WRITEJNLW,   ALL, <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                     07D0  1955        LDBSRV  CJF$, GETCJI,      K,   <R4>
                     07D7  1956        LDBSRV  CJF$, DMTJMDW,     K,   <R4>, 4, 5, DMTJMD
                     07E8  1957        LDBSRV  CJF$, MODFLTW,     K,   <R4>, 4, 5, MODFLT
                     07F8  1958        LDBSRV  CJF$, POSJNLW,     K,   <R4>, 4, 5, POSJNL
```

```
                     0808  1959          LDBSRV  CJF$, READJNLW,    K,   <R4>, 4, 5, READJNL
                     0818  1960          LDBSRV  CJF$, RECOVERW,    K,   <R4>, 5, 6, RECOVER
                     0828  1961
                     0828  1962 ;
            00004010 0828  1963          RUF$KCASCTR = 16400
                     0828  1964 ;
                     0828  1965          LDBSRV  RUF$,  REENTERRU,    K,  <R2,R3,R4,R5,R6>
                     082F  1966          LDBSRV  RUF$,  STARTRU,      K,  <R2,R3,R4,R5,R6>
                     0837  1967          LDBSRV  RUF$,  PHASE1,       K,  <R2,R3,R4,R5,R6>
                     083F  1968          LDBSRV  RUF$,  PHASE2,       K,  <R2,R3,R4,R5,R6>
                     0847  1969          LDBSRV  RUF$,  CANCELRU,     K,  <R2,R3,R4,R5,R6>
                     084F  1970          LDBSRV  RUF$,  MARKPOINTRU,  K,  <R2,R3,R4,R5,R6>
                     0857  1971          LDBSRV  RUF$,  RESETRU,      K,  <R2,R3,R4,R5,R6>
                     085F  1972          LDBSRV  RUF$,  DCLRUH,       K,  <R2,R3,R4,R5,R6>
                     0867  1973          LDBSRV  RUF$,  CANRUH,       K,  <R2,R3,R4,R5,R6>
                     086F  1974          LDBSRV  RUF$,  RUSTATUS,     K,  <R2,R3,R4,R5,R6>
                     0877  1975 ;
                     0877  1976 ; End Recovery Unit consists of a two-phase commit, so we call each
                     0877  1977 ; phase separately.
                     0877  1978 ;
                     0877  1979          GCOMPSRVB ENDRU, <PHASE1_MASK ! PHASE2_MASK>, RUF$ ; End Recovery Unit
4012'8F    BC        087A  1983          CHMK    I^#PHASE1
  04 50    E9        087E  1984          BLBC    R0,10$
4013'8F    BC        0881  1985          CHMK    I^#PHASE2
           04        0885  1986 10$:     RET
                     0886  1990          GCOMPSRVE       2
                     0888  1991          GSYSSRV MTACCESS,K,6,-          ;Mag tape installation specific access routi
                     0888  1992                <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;REGISTERS R2-R11
                     0176  1993
                     0176  1994 ;
                     0176  1995 ; End of system service vector definitions. New system services are
                     0176  1996 ; to be added at this point.
                     0176  1997 ;
                     0176  2000          ASSUME  RCASMIN GE ECASCTR      ;Exec service codes must not collide with RM
                     0176  2003
```

```
                    0000088F  2008          .PSECT  $$$000,BYTE
                        088F  2009 CLIJMP:
00000000'9F    DD   088F  2010          PUSHL   @#CTL$AL_CLICALBK       ;PIC JUMP FOR CLI CALLBACK
         9E    17   0895  2011          JMP     @(SP)+
        00000A00   0897  2012          .BLKB   <SGN$C_SYSVECPGS@9>-<.-VECBASE> ;FILL REMAINDER OF RESERVED PAGES
                    0A00  2013
```

```
                                0A00  2015          .SBTTL  ILLEGAL CHME OR CHMK CODE VALUE HANDLING
                                0A00  2016 ;
                                0A00  2017 ;
                                0A00  2018 ;       END OF CHME DISPATCH TABLE
                                0A00  2019 ;
                                0A00  2020
                           000000A0   2021          .PSECT  Y$CMODE,QUAD
          00000000'FF    16    00A0  2022          JSB     @CTL$GL_RMSBASE         ;SEE IF RMS DOES THIS SERVICE
                                00A6  2023                                          ;  (R0 HAS CHME CODE)
          00000000'EF    16    00A6  2024          JSB     EXE$LOAD_EDISP          ; CALL LOADABLE CODE DISPATCHERS
                                00AC  2025
          00000000'9F    95    00AC  2026          TSTB    @#CTL$GB_SSFILTER       ; ANY INHIBIT BITS ON?
                       08    13    00B2  2027          BEQL    5$                      ; NO, ALL OKAY
                 51  04D4 8F    3C    00B4  2028          MOVZWL  #SS$_INHCHME,R1         ; YES, SET THE EXCEPTION CODE
                        FF44'    31    00B9  2029          BRW     INHEXCP1                ; DEAL WITH BAD CODE
                                00BC  2030
               51  00000000'9F    D0    00BC  2031 5$:      MOVL    @#CTL$GL_USRCHME,R1     ; GET PER-PROCESS USER CHME VECTOR
                       02    13    00C3  2032          BEQL    10$                     ; NOT PRESENT, TRY SYSTEM WIDE
                                00C5  2033
                                00C5  2034 ;
                                00C5  2035 ;       CALL PER-PROCESS "USER" SUPPLIED PLUG-ON HANDLER FOR CHME
                                00C5  2036 ;       WITH UNRECOGNIZED CODES.
                                00C5  2037 ;
                                00C5  2038 ;       R0 - CODE FROM CHME/CHMK (LONGWORD)
                                00C5  2039 ;       R1 - ADDRESS OF ROUTINE
                                00C5  2040 ;       (SP) - RETURN ADDRESS IN CASE CODE IS NOT LEGAL.
                                00C5  2041 ;              IF AN RSB IS ISSUED, THEN THE SYSTEM-WIDE HANDLER WILL BE
                                00C5  2042 ;              GIVEN AN OPPORTUNITY BEFORE DECIDING THAT THE CODE IS REALLY ILLEGAL.
                                00C5  2043 ;               (NORMAL RETURN IS A RET AFTER PERFORMING FUNCTION)
                                00C5  2044 ;
                       61    16    00C5  2045          JSB     (R1)                    ; CALL PER-PROCESS USR CHME HANDLER
                                00C7  2046                                          ; RETURNS ONLY IF ILLEGAL CODE
               51  00000000'EF    D0    00C7  2047 10$:     MOVL    L^EXE$GL_USRCHME,R1     ; ELSE TRY SYSTEM WIDE VECTOR
                       02    13    00CE  2048          BEQL    20$                     ; NOT PRESENT, ILLEGAL
                       61    16    00D0  2049          JSB     (R1)                    ; CALL SYSTEM WIDE USER CHME HANDLER
                                00D2  2050 ;
                                00D2  2051 ;       CALL SYSTEM-WIDE "USER" SUPPLIED PLUG-ON HANDLER FOR CHME
                                00D2  2052 ;       WITH UNRECOGNIZED CODES.
                                00D2  2053 ;
                                00D2  2054 ;       R0 - CODE FROM CHME/CHMK (LONGWORD)
                                00D2  2055 ;       R1 - ADDRESS OF ROUTINE
                                00D2  2056 ;       (SP) - RETURN ADDRESS TO GIVE SS$_ILLSER ERROR
                                00D2  2057 ;              (NORMAL RETURN IS A RET AFTER PERFORMING FUNCTION)
                                00D2  2058 ;
                                00D2  2059                                          ; RETURNS ONLY IF ILLEGAL CODE
                     00CF'    31    00D2  2060 20$:     BRW     ILLSER                  ;
                                00D5  2061
                     00000008   00D5  2062 ECASMAX=ECASCTR-1
                                00D5  2063 ;
                                00D5  2064 ;
                                00D5  2065 ;  RMS $WAIT SYNCHRONIZATION CODE.
                                00D5  2066 ;
                                00D5  2067 ;  LOOK AT FLAG IN R4 TO DETERMINE IF THIS IS A $WAIT FOR THE SAME OR DIFFERENT
                                00D5  2068 ;  PABS.  IF SAME, MERELY RSB;  IF DIFFERENT, WAIT ON EVENT FLAG AND THEN
                                00D5  2069 ;  RE-EXECUTE THE $WAIT SERVICE.
                                00D5  2070 ;
                                00D5  2071
```

```
                     00D5  2072 RMS_WAIT_SYNC:
          01 54   E8 00D5  2073         BLBS    R4,10$                  ;BRANCH IF DIFFERENT RABS
                  05 00D8  2074         RSB                             ;HANDLE WITH STANDARD STALL
             8E   D5 00D9  2075 10$:     TSTL    (SP)+                   ;POP RETURN PC FROM STACK
    0000'8F   50   B1 00DB  2076         CMPW    R0,#RMS$_STALL&^XFFFF   ;IS STALL REQUIRED?
              01   13 00E0  2077         BEQL    20$                     ;BRANCH IF YES
                   04 00E2  2078         RET                             ;NO - BACK TO USER
                     00E3  2079 20$:     $WAITFR_S       R3              ;WAIT ON SPECIFIED EVENT FLAG
 00000002'EF       17 00EC  2080         JMP     SYS$WAIT+2              ;RE-EXECUTE RMS $WAIT
                     00F2  2081 ;
                     00F2  2082 ;  THE FOLLOWING CODE IS AN ERROR PATH FROM THE RMS SYCHRONIZATION CODE
                     00F2  2083 ;  THAT PRECEDES THE RMS VECTORS.  IT WAS MOVED HERE BECAUSE CODE WAS
                     00F2  2084 ;  ADDED THERE AND BECAUSE THE RMS VECTORS CAN'T MOVE, THIS CODE DID.
                     00F2  2085 ;
                     00F2  2086 ;  CHECK STATUS CODE FOR ERROR OR SEVERE ERROR, IF SUCCESS THEN
                     00F2  2087 ;  BAD USER STRUCTURE DETECTED - RETURN ERROR IN R0, STATUS OF RECORD
                     00F2  2088 ;  OPERATION WILL BE LOST
                     00F2  2089 ;
                     00F2  2090 RMS_ERR:
    01 A8    01   8A 00F2  2091         BICB2   #1,RAB$B_BLN(R8)        ;CLEAR WAITING FLAG
          07 50   E9 00F6  2092         BLBC    R0,98$                  ;STALE SUCCESS => BAD STRUCTURE
50  00000000'8F   D0 00F9  2093         MOVL    #RMS$_STR,R0            ;CHANGE STATUS TO BAD STRUCTURE ERROR
       50   06   93 0100  2094 98$:     BITB    #6,R0                   ;ERROR OR SEVERE ERROR?
            07   13 0103  2095         BEQL    99$                     ;BRANCH IF NOT
                     0105  2096 ;
                     0105  2097 ;  MUST RETURN TO EXEC MODE TO GENERATE POSSIBLE SYSTEM SERVICE FAILURE EXCEPTION
                     0105  2098 ;
       52   50   D0 0105  2099         MOVL    R0,R2                   ;STATUS CODE TO R2
    0031'8F      BD 0108  2100         CHME    I^#SSVEXC               ;GENERATE EXCEPTION IF ENABLED
                   04 010C  2101 99$:    RET
```

```
                     010D  2103 ;
                     010D  2104 ;       END OF CHMK DISPATCH TABLE
                     010D  2105 ;
                     010D  2106
                00000176   2107         .PSECT  Y$CMODK,QUAD
                     0176  2108 ;
                     0176  2109 ; UNIMPLEMENTED SERVICES, DEFINED TO PROVIDE CLEAN LINK.
                     0176  2110 ; REMOVE NAME AND VERIFY GSYSSRV ENTRY WHEN SERVICE IS IMPLEMENTED.
                     0176  2111 ;
                     0176  2112 ;
                     0176  2113 ;       CALL PER-PROCESS "USER" SUPPLIED PLUG-ON HANDLER FOR CHMK
                     0176  2114 ;       WITH UNRECOGNIZED CODES.
                     0176  2115 ;
                     0176  2116 ;       R0 - CODE FROM CHME/CHMK (LONGWORD)
                     0176  2117 ;       R1 - ADDRESS OF ROUTINE
                     0176  2118 ;       (SP) - RETURN ADDRESS TO GIVE SS$_ILLSER ERROR
                     0176  2119 ;              (NORMAL RETURN IS A RET AFTER PERFORMING FUNCTION)
                     0176  2120 ;
                     0176  2121
 00000000'EF   16    0176  2122         JSB     EXE$LOAD_KDISP          ; CALL LOADABLE CODE DISPATCHERS
                     017C  2123
 00000000'9F   95    017C  2124         TSTB    @#CTL$GB_SSFILTER       ; ANY INHIBIT BITS ON?
           08   13   0182  2125         BEQL    5$                      ; NO, ALL OKAY
      51 04CC 8F   3C   0184  2126         MOVZWL  #SS$_INHCHMK,R1         ; YES, SET THE EXCEPTION CODE
            FE74   31   0189  2127         BRW     INHEXCP1                ; DEAL WITH BAD CODE
                     018C  2128
   51   00000000'9F   D0   018C  2129 5$:      MOVL    @#CTL$GL_USRCHMK,R1     ; GET PER-PROCESS VECTOR
           02   13   0193  2130         BEQL    10$                     ; NOT PRESENT, TRY FOR SYSTEM WIDE
           61   16   0195  2131         JSB     (R1)                    ; CALL PER-PROCESS HANDLER
                     0197  2132                                         ; RETURNS ONLY IF CODE IN R0 IS NOT
                     0197  2133 ;
                     0197  2134 ;       CALL SYSTEM-WIDE "USER" SUPPLIED PLUG-ON HANDLER FOR CHMK
                     0197  2135 ;       WITH UNRECOGNIZED CODES.
                     0197  2136 ;
                     0197  2137 ;       R0 - CODE FROM CHME/CHMK (LONGWORD)
                     0197  2138 ;       R1 - ADDRESS OF ROUTINE
                     0197  2139 ;       (SP) - RETURN ADDRESS TO GIVE SS$_ILLSER ERROR
                     0197  2140 ;              (NORMAL RETURN IS A RET AFTER PERFORMING FUNCTION)
                     0197  2141 ;
                     0197  2142                                         ; HANDLED BY PER PROCESS HANDLER
   51   00000000'EF   D0   0197  2143 10$:      MOVL    L^EXE$GL_USRCHMK,R1     ; ELSE GET SYSTEM WIDE VECTOR
           02   13   019E  2144         BEQL    20$                     ; NOT PRESENT, ILLEGAL SERVICE
           61   16   01A0  2145         JSB     (R1)                    ; CALL SYSTEM WIDE HANDLER
                     01A2  2146                                         ; RETURN ONLY IF ILLEGAL CODE
                     01A2  2147 20$:                                    ;
                     01A2  2148 EXE$ALCDNP:                             ;
                     01A2  2149 EXE$CLRPAR:                             ;
                     01A2  2150 EXE$DLCDNP:                             ;
                     01A2  2151
                     01A2  2152 EXE$FAILURE::                           ; THIS PROCEDURE ALWAYS FAILS
                     01A2  2153
           01   01A2  2154         NOP
           01   01A3  2155         NOP
                     01A4  2156
      50 0104 8F   3C   01A4  2157 ILLSER: MOVZWL  #SS$_ILLSER,R0          ;ILLEGAL SYSTEM SERVICE
           04   01A9  2158         RET                                    ;
                     01AA  2159
```

```
                                01AA  2160 EXE$SUCCESS::                            ; THIS PROCEDURE ALWAYS SUCCEEDS
                             01 01AA  2161          NOP                             ; THESE TWO INSTRUCTIONS CAN ALSO
                             01 01AB  2162          NOP                             ;  SERVE AS A HARMLESS ENTRY MASK
                      50     01 3C    01AC  2163          MOVZWL  #SS$_NORMAL,R0          ; RETURN SUCCESSFUL STATUS
                             04 01AF  2164          RET
                                01B0  2165
                                01B0  2169 SSFAILMAIN:                             ;SSFAIL MAIN LOGIC
               51    00000000'GF  D0  01B0  2170          MOVL    G^CTL$GL_PCB,R1         ;GET PCB ADDRESS
                        0E A1    B5  01B7  2171          TSTW    PCB$W_MTXCNT(R1)        ;MUTEX COUNT ZERO?
                           21    12  01BA  2172          BNEQ    20$                     ;IF NEQ NO
                        02 18    EF  01BC  2173          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,- ;EXTRACT PREVIOUS MODE FROM
                     7E    04 AE      01BF  2174                  4(SP),-(SP)             ;SAVED PSL
                        6E 06    C0  01C2  2175          ADDL    #PCB$V_SSFEXC,(SP)      ;ADD IN BASE BIT NUMBER
                  12 24 A1    8E    E1  01C5  2176          BBC     (SP)+,PCB$L_STS(R1),10$ ;IF CLEAR, FAILURE EXCEPTION DISABLED
                           7E    DC  01CA  2177          MOVPSL  -(SP)                   ;GET CURRENT PSL
               8E    6E    02 18    EF  01CC  2178          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,(SP),(SP)+ ;IF CURRENT MODE IS
                           03    12  01D1  2179          BNEQ    5$                      ;NOT KERNEL, THEN BRANCH
                                01D3  2180          SETIPL  #0                      ;FORCE IPL TO 0 FOR ERROR PATH
                     00000000'EF  17  01D6  2182 5$:     JMP     EXE$SSFAIL              ;GENERATE SYSTEM SERVICE FAILURE EXCEPTION
                                02  01DC  2183 10$:    REI                             ;AND RETURN FROM SERVICE WITH ERROR STATUS
                        05 10    EF  01DD  2184 20$:    EXTZV   #PSL$V_IPL,#PSL$S_IPL,- ;EXTRACT PREVIOUS IPL FROM
                     7E    04 AE      01E0  2185                  4(SP),-(SP)             ;SAVED PSL
                        02 8E    D1  01E3  2186          CMPL    (SP)+,#IPL$_ASTDEL      ;TEST IF AT ELEVATED IPL
                           F4    18  01E6  2187          BGEQ    10$                     ;IF SO DO NOT BUGCHECK
                                01E8  2188          BUG_CHECK MTXCNTNONZ,FATAL      ;MUTEX COUNT NONZERO AT SERVICE EXIT
                                01EC  2200 ;
                                01EC  2201 ; UPDSECW - UPDATE SECTION AND WAIT COMPOSITE SERVICE
                                01EC  2202 ;
                                01EC  2203          .ENABL  LSB
                                01EC  2204
                                01EC  2205 EXE$UPDSECW:
                  001E'8F       BC  01EC  2206          CHMK    I^#UPDSEC               ;UPDATE THE SECTION
                     22 50      E9  01F0  2207          BLBC    R0,40$                  ;BRANCH IF ERROR
                  52    50      D0  01F3  2208          MOVL    R0,R2                   ;SAVE STATUS FROM UPDSEC
                                01F6  2209
                                01F6  2210          ASSUME  UPDSEC$_EFN+4 EQ UPDSEC$_IOSB
               7E    14 AC      7D  01F6  2211          MOVQ    UPDSEC$_EFN(AP),-(SP)   ;PUSHL IOSB(AP), PUSHL EFN(AP)
                        0C      11  01FA  2212          BRB     20$                     ;SYNCHRONIZE EFN AND IOSB
                                01FC  2213 ;
                                01FC  2214 ; COMMON WAIT CODE FOR $GETDVIW, $GETJPIW, $GETSYIW, $SNDJBCW SYSTEM SERVICES
                                01FC  2215 ;
                                01FC  2216 ; INPUTS:
                                01FC  2217 ;
                                01FC  2218 ;       R0 = STATUS FROM THE NON-WAITING VERSION OF THE SERVICE
                                01FC  2219 ;       EFN(AP)  = EVENT FLAG
                                01FC  2220 ;       IOSB(AP) = I/O STATUS BLOCK ADDRESS
                                01FC  2221 ;
                           00000004 01FC  2222          GETJPI_SYNCH_MASK = ^M<R2>      ;REGISTERS USED BY THIS CODE
                                01FC  2223                                          ;OTHER THAN R0 AND R1
                                01FC  2224 GETJPI_SYNCH:
                     16 50      E9  01FC  2225          BLBC    R0,40$                  ;BRANCH IF ERROR FROM ORIGINAL SERVICE
                  52    50      D0  01FF  2226          MOVL    R0,R2                   ;SAVE STATUS FROM ORIGINAL SERVICE
                                0202  2227
                                0202  2228          ASSUME  GETJPI$_IOSB EQ GETDVI$_IOSB
                                0202  2229          ASSUME  GETJPI$_IOSB EQ GETSYI$_IOSB
                                0202  2230          ASSUME  GETJPI$_IOSB EQ SNDJBC$_IOSB
                     14 AC      DD  0202  2231          PUSHL   GETJPI$_IOSB(AP)        ;GET IOSB PARAMETER
```

```
                04 AC    DD  0205  2232          PUSHL   GETJPI$_EFN(AP)         ;GET EVENT FLAG PARAMETER
00000000'GF     02    FB  0208  2233 20$:       CALLS   #2,G^SYS$SYNCH          ;WAIT FOR EFN AND IOSB TO BE SET
             03 50    E9  020F  2234          BLBC    R0,40$                  ;IF ERROR, RETURN THAT STATUS
          50    52    D0  0212  2235          MOVL    R2,R0                   ;OTHERWISE RESTORE ORIGINAL STATUS
                      04  0215  2236 40$:      RET                             ;AND RETURN
                          0216  2237
                          0216  2238          .DSABL  LSB
                          0216  2239 ;
                          0216  2240 ;       JUMPS TO REAL SYSTEM SERVICE ENTRY POINT ARE DEFINED HERE IF THE CASE
                          0216  2241 ;       TABLE WON'T REACH
                          0216  2242 ;
                          0216  2243 ;       THESE ARE FOR USE WITHIN THIS MODULE ONLY - NOT GLOBAL ENTRY POINTS
                          0216  2244 ;       ENTRY MASKS ARE PLACEHOLDERS ONLY
                          0216  2245 ;
                          0216  2246 EXE$IMGACT:                              ; IMAGE ACTIVATION
                    0000  0216  2247          .WORD   0
          00000002'EF    17   0218  2248          JMP     EXE$$IMGACT + 2
                          021E  2249
                          021E  2250 EXE$ASCTOID:                             ; ASCII TO IDENTIFIER CONVERSION
                    0000  021E  2251          .WORD   0
          00000002'EF    17   0220  2252          JMP     EXE$$ASCTOID + 2
                          0226  2253
                          0226  2254 EXE$FINISH_RDB:                          ; FINISH RDB CONTEXT STREAM
                    0000  0226  2255          .WORD   0
          00000002'EF    17   0228  2256          JMP     EXE$$FINISH_RDB + 2
                          022E  2257
                          022E  2258 EXE$IDTOASC:                             ; IDENTIFIER TO ASCII CONVERSION
                    0000  022E  2259          .WORD   0
          00000002'EF    17   0230  2260          JMP     EXE$$IDTOASC + 2
                          0236  2261
                          0236  2262 ;
                          0236  2263 ;
                 00000055 0236  2265 KCASMAX=KCASCTR-2
                          0236  2266
                          0236  2269
                 00000022 0236  2273 RCASMAX=RCASCTR-<1+RCASMIN>
```

```
                     0236  2293          .SBTTL  EXE$LDB_SYNCH   -       Synchronize Loadable Services
                     0236  2294
                     0236  2295 ;
                     0236  2296 ; EXE$LDB_SYNCH - Synchronize Loadable Service
                     0236  2297 ;
                     0236  2298 ;       This routine performs a $SYNCH service in the mode of the
                     0236  2299 ;       caller of a loadable service
                     0236  2300 ;
                     0236  2301 ;       Inputs:
                     0236  2302 ;               R0      -       Main Service Status
                     0236  2303 ;               (SP)    -       IOSB argument number
                     0236  2304 ;               4(SP)   -       Event flag argument number
                     0236  2305 ;               (FP)    -       Service Call Frame
                     0236  2306 ;
                     0236  2307 ;       Outputs:
                     0236  2308 ;               R0      -       Status Code
                     0236  2309 ;
                     0236  2310 ;       Calling Sequence:
                     0236  2311 ;               JMP     a#EXE$LDB_SYNCH
                     0236  2312 ;
                     0236  2313 ;       Returns Via:
                     0236  2314 ;               RET     instruction
                     0236  2315 ;
                     0236  2316
                     0236  2317 EXE$LDB_SYNCH::
        2E 50    E9  0236  2318          BLBC    R0,50$                          ; get out if service had error
           50    DD  0239  2319          PUSHL   R0                              ; save service status
     04 AE    6C   B1  023B  2320          CMPW    (AP),4(SP)                      ; was an IOSB specified
           09    19  023F  2321          BLSS    10$                             ; branch if not
     50    04 AE   D0  0241  2322          MOVL    4(SP),R0                        ; get argument offset
          6C40   DD  0245  2323          PUSHL   (AP)[R0]                        ; push IOSB address
           02    11  0248  2324          BRB     20$
                     024A  2325
           7E    D4  024A  2326 10$:     CLRL    -(SP)                           ; no IOSB so pass 0 to synch
                     024C  2327
     0C AE    6C   B1  024C  2328 20$:     CMPW    (AP),12(SP)                     ; was an EFN specified?
           09    19  0250  2329          BLSS    30$                             ; branch if not
     50    0C AE   D0  0252  2330          MOVL    12(SP),R0                       ; get argument offset
          6C40   DD  0256  2331          PUSHL   (AP)[R0]                        ; push EFN number
           02    11  0259  2332          BRB     40$
                     025B  2333
           7E    D4  025B  2334 30$:     CLRL    -(SP)                           ; no EFN so pass 0
                     025D  2335
00000000'GF    02   FB  025D  2336 40$:     CALLS   #2,G^SYS$SYNCH                  ; call synch system service
           50    8E   D0  0264  2337          MOVL    (SP)+,R0                        ; restore main service status
                     0267  2338
                04   0267  2339 50$:     RET
                     0268  2340
                     0268  2341
                     0268  2345          .END
```

```
$$ARGS                  =  00000008
$$T1                    =  00000024
ACCVIO                     00000038 R      05
ACCVIO_RET                 00000045 R      05
ADJSTK                  =  00000001
ADJSTK_MASK             =  0000007C
ADJWSL                  =  00000002
ADJWSL_MASK             =  0000003C
ALCDNP                  =  00000003
ALCDNP_MASK             =  000000FC
ALLJDR                  =  00004028
ALLJDR_MASK             =  00000010
ALLOC                   =  00000004
ALLOC_MASK              =  0000007C
ASCEFC                  =  00000005
ASCEFC_MASK             =  00000FFC
ASCTIM_MASK             =  0000007C
ASCTOID                 =  00000008
ASCTOID_MASK            =  00000FFC
ASSIGN                  =  00000006
ASSIGN_MASK             =  00000FFC
ASSJNL                  =  00004029
ASSJNL_MASK             =  00000010
ASTEXIT                    00000018 R      05
BINTIM_MASK             =  000001FC
BRDCST_MASK             =  0000007C
BRKTHRU                 =  00000054
BRKTHRU_MASK            =  00000FFC
BUG$_MTXCNTNONZ            ********   X    05
BUG$_SSRVEXCEPT            ********   X    03
B_EMASK                    00000000 R      02
B_EXECNARG                 00000000 R      04
CANCEL                  =  00000007
CANCELRU                =  00004014
CANCELRU_MASK           =  0000007C
CANCEL_MASK             =  000001FC
CANEXH                  =  00000042
CANEXH_MASK             =  0000003C
CANRUH                  =  00004018
CANRUH_MASK             =  0000007C
CANTIM                  =  00000008
CANTIM_MASK             =  0000003C
CANWAK                  =  00000009
CANWAK_MASK             =  0000003C
CAT0                    =  00000001
CAT7                    =  00000080
CHF$L_SIGARGLST         =  00000004
CHF$L_SIG_NAME          =  00000004
CHKPRO                  =  00000055
CHKPRO_MASK             =  00000FFC
CJF$KCASCTR             =  0000403C
CLIJMP                     0000088F R      08
CLOSE                   =  0000001C
CLOSE_MASK              =  00000FFC
CLRAST                  =  00000000
CLREF                   =  0000000D
CLREF_MASK              =  0000003C
CLRPAR                  =  0000000B
CLRPAR_MASK             =  0000003C
CME$C_ASCTOID           =  00000008
CME$C_CLOSE             =  0000001C
CME$C_CMEXEC            =  00000000
CME$C_CONNECT           =  0000001D
CME$C_CREATE            =  0000001E
CME$C_DELETE            =  00000012
CME$C_DISCONNECT        =  0000001F
CME$C_DISPLAY           =  00000020
CME$C_ENTER             =  0000002A
CME$C_ERASE             =  00000021
CME$C_EXTEND            =  00000022
CME$C_FILESCAN          =  00000034
CME$C_FIND              =  00000013
CME$C_FINISH_RDB        =  00000009
CME$C_FLUSH             =  00000023
CME$C_FREE              =  00000014
CME$C_GET               =  00000015
CME$C_GETQUI            =  0000000B
CME$C_GETTIM            =  00000002
CME$C_IDTOASC           =  0000000A
CME$C_IMGACT            =  00000003
CME$C_MODIFY            =  00000024
CME$C_NUMTIM            =  00000004
CME$C_NXTVOL            =  00000025
CME$C_OPEN              =  00000026
CME$C_PARSE             =  0000002B
CME$C_PUT               =  00000016
CME$C_READ              =  00000017
CME$C_RELEASE           =  00000018
CME$C_REMOVE            =  0000002C
CME$C_RENAME            =  0000002D
CME$C_REWIND            =  00000027
CME$C_RMSRUHNDLR        =  00000033
CME$C_RMSRUNDWN         =  00000032
CME$C_SEARCH            =  0000002E
CME$C_SETDDIR           =  0000002F
CME$C_SETDFPROT         =  00000030
CME$C_SNDACC            =  00000007
CME$C_SNDJBC            =  00000001
CME$C_SNDOPR            =  00000005
CME$C_SNDSMB            =  00000006
CME$C_SPACE             =  00000028
CME$C_SSVEXC            =  00000031
CME$C_TRUNCATE          =  00000029
CME$C_UPDATE            =  00000019
CME$C_WAIT              =  0000001A
CME$C_WRITE             =  0000001B
CMEXEC                  =  00000000
CMEXEC_MASK             =  00000010
CMK$C_ADJSTK            =  00000001
CMK$C_ADJWSL            =  00000002
CMK$C_ALCDNP            =  00000003
CMK$C_ALLJDR            =  00004028
CMK$C_ALLOC             =  00000004
CMK$C_ASCEFC            =  00000005
```

```
CMK$C_ASSIGN            =  00000006
CMK$C_ASSJNL            =  00004029
CMK$C_BRKTHRU           =  00000054
CMK$C_CANCEL            =  00000007
CMK$C_CANCELRU          =  00004014
CMK$C_CANEXH            =  00000042
CMK$C_CANRUH            =  00004018
CMK$C_CANTIM            =  00000008
CMK$C_CANWAK            =  00000009
CMK$C_CHKPRO            =  00000055
CMK$C_CLREF             =  0000000D
CMK$C_CLRPAR            =  0000000B
CMK$C_CMKRNL            =  0000000C
CMK$C_CNTREG            =  0000000E
CMK$C_CONJNLF           =  00004039
CMK$C_CONUIC            =  0000402A
CMK$C_CREJNL            =  0000402B
CMK$C_CRELNM            =  00000050
CMK$C_CRELNT            =  0000004F
CMK$C_CREMBX            =  00000010
CMK$C_CRENWV            =  00004038
CMK$C_CREPRC            =  00000011
CMK$C_CRETVA            =  00000012
CMK$C_CRMPSC            =  0000000A
CMK$C_DACEFC            =  00000013
CMK$C_DALLOC            =  00000014
CMK$C_DASSGN            =  00000015
CMK$C_DCLAST            =  00000016
CMK$C_DCLCMH            =  0000003F
CMK$C_DCLEXH            =  00000017
CMK$C_DCLRUH            =  00004017
CMK$C_DCNJNLF           =  0000403A
CMK$C_DEALJDR           =  0000402C
CMK$C_DEASJNL_INT       =  0000402D
CMK$C_DELJNL            =  0000402E
CMK$C_DELLNM            =  00000051
CMK$C_DELMBX            =  00000018
CMK$C_DELPRC            =  00000019
CMK$C_DELTVA            =  0000001A
CMK$C_DEQ               =  00000049
CMK$C_DERLMB            =  00000041
CMK$C_DGBLSC            =  0000001B
CMK$C_DLCDNP            =  0000001C
CMK$C_DLCEFC            =  0000001D
CMK$C_DMTJMD            =  0000402F
CMK$C_DMTJMDW           =  0000402F
CMK$C_DSPJNL            =  00004030
CMK$C_ENQ               =  00000048
CMK$C_ERAPAT            =  0000004E
CMK$C_EXIT              =  00000020
CMK$C_EXPREG            =  00000021
CMK$C_FORCEX            =  00000022
CMK$C_GETCHN            =  00000043
CMK$C_GETCJI            =  0000403B
CMK$C_GETDEV            =  00000044
CMK$C_GETDVI            =  0000004D
CMK$C_GETJNL            =  00004031
CMK$C_GETJPI            =  00000045
CMK$C_GETLKI            =  00000053
CMK$C_GETPTI            =  0000000F
CMK$C_GETRUI            =  00004032
CMK$C_GETSYI            =  0000004C
CMK$C_HIBER             =  00000023
CMK$C_LCKPAG            =  00000024
CMK$C_LKWSET            =  00000025
CMK$C_MARKPOINTRU       =  00004015
CMK$C_MGBLSC            =  00000026
CMK$C_MNTJMD            =  00004037
CMK$C_MODFLT            =  00004033
CMK$C_MODFLTW           =  00004033
CMK$C_MTACCESS          =  00000056
CMK$C_PHASE1            =  00004012
CMK$C_PHASE2            =  00004013
CMK$C_POSJNL            =  00004034
CMK$C_POSJNLW           =  00004034
CMK$C_PURGWS            =  00000027
CMK$C_QIO               =  00000028
CMK$C_READEF            =  00000029
CMK$C_READJNL           =  00004035
CMK$C_READJNLW          =  00004035
CMK$C_RECOVER           =  00004036
CMK$C_RECOVERW          =  00004036
CMK$C_REENTERRU         =  00004010
CMK$C_RESETRU           =  00004016
CMK$C_RESUME            =  0000002A
CMK$C_RUNDWN            =  0000002B
CMK$C_RUSTATUS          =  00004019
CMK$C_SCHDWK            =  0000002C
CMK$C_SETAST            =  0000002D
CMK$C_SETEF             =  0000002E
CMK$C_SETEXV            =  0000002F
CMK$C_SETIME            =  00000046
CMK$C_SETIMR            =  00000032
CMK$C_SETPFM            =  00000040
CMK$C_SETPRA            =  00000031
CMK$C_SETPRI            =  00000033
CMK$C_SETPRN            =  00000030
CMK$C_SETPRT            =  00000034
CMK$C_SETPRV            =  00000047
CMK$C_SETRWM            =  00000035
CMK$C_SETSFM            =  00000036
CMK$C_SETSSF            =  0000004A
CMK$C_SETSTK            =  0000004B
CMK$C_SETSWM            =  00000037
CMK$C_SNDERR            =  0000001F
CMK$C_STARTRU           =  00004011
CMK$C_SUSPND            =  00000038
CMK$C_TRNLNM            =  00000052
CMK$C_ULKPAG            =  00000039
CMK$C_ULWSET            =  0000003A
CMK$C_UPDSEC            =  0000001E
CMK$C_WAITFR            =  0000003B
CMK$C_WAKE              =  0000003C
CMK$C_WFLAND            =  0000003D
```

```
CMK$C_WFLOR                     =  0000003E
CMKRNC                          =  0000000C
CMKRNL_MASK                     =  00000010
CNTREG                          =  0000000E
CNTREG_MASK                     =  000000FC
COMPSIZE                        =  0000000E
COMPSTRT                        =  00000878 R      08
CONJNLF                         =  00004039
CONJNLF_MASK                    =  00000010
CONNECT                         =  0000001D
CONNECT_MASK                    =  00000FFC
CONUIC                          =  0000402A
CONUIC_MASK                     =  00000010
CREATE                          =  0000001E
CREATE_MASK                     =  00000FFC
CREJNL                          =  0000402B
CREJNL_MASK                     =  00000010
CRELNM                          =  00000050
CRELNM_MASK                     =  00000FFC
CRELNT                          =  0000004F
CRELNT_MASK                     =  00000FFC
CRELOG_MASK                     =  000001FC
CREMBX                          =  00000010
CREMBX_MASK                     =  00000FFC
CRENWV                          =  00004038
CRENWV_MASK                     =  00000010
CREPRC                          =  00000011
CREPRC_MASK                     =  00000FFC
CRETVA                          =  00000012
CRETVA_MASK                     =  000001FC
CRMPSC                          =  0000000A
CRMPSC_MASK                     =  00000FFC
CTL$AL_CLICALBK                 ********    X    08
CTL$GB_SSFILTER                 ********    X    03
CTL$GL_PCB                      ********    X    05
CTL$GL_RMSBASE                  ********    X    03
CTL$GL_USRCHME                  ********    X    03
CTL$GL_USRCHMK                  ********    X    05
CTL$GQ_COMMON                   ********    X    08
DACEFC                          =  00000013
DACEFC_MASK                     =  00000FFC
DALLOC                          =  00000014
DALLOC_MASK                     =  0000013C
DASSGN                          =  00000015
DASSGN_MASK                     =  000001FC
DCLAST                          =  00000016
DCLAST_MASK                     =  0000003C
DCLCMH                          =  0000003F
DCLCMH_MASK                     =  00000010
DCLEXH                          =  00000017
DCLEXH_MASK                     =  0000001C
DCLRUH                          =  00004017
DCLRUH_MASK                     =  0000007C
DCNJNLF                         =  0000403A
DCNJNLF_MASK                    =  00000010
DEALJDR                         =  0000402C
DEALJDR_MASK                    =  00000010
DEASJNL_INT                     =  0000402D
DEASJNL_INT_MASK                =  00000010
DEASJNL_MASK                    =  00000FFC
DEF_MASK                        =  00000081
DELETE                          =  00000012
DELETE_MASK                     =  00000FFC
DELJNL                          =  0000402E
DELJNL_MASK                     =  00000010
DELLNM                          =  00000051
DELLNM_MASK                     =  00000FFC
DELLOG_MASK                     =  000001FC
DELMBX                          =  00000018
DELMBX_MASK                     =  0000003C
DELPRC                          =  00000019
DELPRC_MASK                     =  000000FC
DELTVA                          =  0000001A
DELTVA_MASK                     =  000000FC
DEQ                             =  00000049
DEQ_MASK                        =  00000FFC
DERCMB                          =  00000041
DERLMB_MASK                     =  0000003C
DGBLSC                          =  0000001B
DGBLSC_MASK                     =  000007FC
DISCONNECT                      =  0000001F
DISCONNECT_MASK                 =  00000FFC
DISPLAY                         =  00000020
DISPLAY_MASK                    =  00000FFC
DLCDNP                          =  0000001C
DLCDNP_MASK                     =  000000FC
DLCEFC                          =  0000001D
DLCEFC_MASK                     =  00000FFC
DMTJMD                          =  0000402F
DMTJMDW                         =  0000402F
DMTJMDW_MASK                    =  00000010
DMTJMD_MASK                     =  00000010
DSPJNL                          =  00004030
DSPJNL_MASK                     =  00000010
ECASCTR                         =  0000000C
ECASE                              00000088 R      03
ECASMAX                         =  0000000B
ENQ                             =  00000048
ENQ$_ACMODE                     =  00000028
ENQ$_ASTADR                     =  0000001C
ENQ$_ASTPRM                     =  00000020
ENQ$_BLKAST                     =  00000024
ENQ$_EFN                        =  00000004
ENQ$_FLAGS                      =  00000010
ENQ$_LKMODE                     =  00000008
ENQ$_LKSB                       =  0000000C
ENQ$_NARGS                      =  0000000B
ENQ$_PARID                      =  00000018
ENQ$_PROT                       =  00000020
ENQ$_RESNAM                     =  00000014
ENQ_MASK                        =  00000FFC
ENTER                           =  0000002A
ENTER_MASK                      =  00000FFC
ERAPAT                          =  0000004E
```

```
ERAPAT_MASK             = 00000010
ERASE                   = 00000021
ERASE_MASK              = 00000FFC
EXACCVIO                  00000000 R     03
EXCMSG_MASK             = 00000FFC
EXC_MASK                = 00000080
EXE$$ASCTOID              ********   X   05
EXE$$FINISH_RDB           ********   X   05
EXE$$IDTOASC              ********   X   05
EXE$$IMGACT               ********   X   05
EXE$ADJSTK                ********   X   05
EXE$ADJWSL                ********   X   05
EXE$ALCDNP                000001A2 R     05
EXE$ALLOC                 ********   X   05
EXE$ASCEFC                ********   X   05
EXE$ASCTIM                ********   X   08
EXE$ASCTOID               0000021E R     05
EXE$ASSIGN                ********   X   05
EXE$ASTRET                ********   X   08
EXE$BINTIM                ********   X   08
EXE$BRDCST                ********   X   08
EXE$BRKTHRU               ********   X   05
EXE$CANCEL                ********   X   05
EXE$CANEXH                ********   X   05
EXE$CANTIM                ********   X   05
EXE$CANWAK                ********   X   05
EXE$CHKPRO                ********   X   05
EXE$CLREF                 ********   X   05
EXE$CLRPAR                000001A2 R     05
EXE$CMEXEC                ********   X   03
EXE$CMKRNL                ********   X   05
EXE$CMODEXEC              00000058 RG    03
EXE$CMODEXECX             00000030 RG    03
EXE$CMODKRNL              00000090 RG    05
EXE$CMODKRNLX             00000068 RG    05
EXE$CNTREG                ********   X   05
EXE$CRELNM                ********   X   05
EXE$CRELNT                ********   X   05
EXE$CRELOG                ********   X   08
EXE$CREMBX                ********   X   05
EXE$CREPRC                ********   X   05
EXE$CRETVA                ********   X   05
EXE$CRMPSC                ********   X   05
EXE$C_CMSTKSZ           = 00000014   G
EXE$DACEFC                ********   X   05
EXE$DALLOC                ********   X   05
EXE$DASSGN                ********   X   05
EXE$DCLAST                ********   X   05
EXE$DCLCMH                ********   X   05
EXE$DCLEXH                ********   X   05
EXE$DEASJNL               ********   X   08
EXE$DELLNM                ********   X   05
EXE$DELLOG                ********   X   08
EXE$DELMBX                ********   X   05
EXE$DELPRC                ********   X   05
EXE$DELTVA                ********   X   05
EXE$DEQ                   ********   X   05
EXE$DERLMB                ********   X   05
EXE$DGBLSC                ********   X   05
EXE$DLCDNP                000001A2 R     05
EXE$DLCEFC                ********   X   05
EXE$ENQ                   ********   X   05
EXE$ERAPAT                ********   X   05
EXE$EXCMSG                ********   X   08
EXE$EXCPTN                0000005A RG    05
EXE$EXCPTNE               0000000D RG    03
EXE$EXIT                  ********   X   05
EXE$EXPREG                ********   X   05
EXE$FAILURE               000001A2 RG    05
EXE$FAO                   ********   X   08
EXE$FAOL                  ********   X   08
EXE$FINISH_RDB            00000226 R     05
EXE$FORCEJNL              ********   X   08
EXE$FORCEJNLW             ********   X   08
EXE$FORCEX                ********   X   05
EXE$GETCHN                ********   X   05
EXE$GETDEV                ********   X   05
EXE$GETDVI                ********   X   05
EXE$GETJPI                ********   X   05
EXE$GETLKI                ********   X   05
EXE$GETMSG                ********   X   08
EXE$GETPTI                ********   X   05
EXE$GETQUI                ********   X   03
EXE$GETSYI                ********   X   05
EXE$GETTIM                ********   X   03
EXE$GL_USRCHME            ********   X   03
EXE$GL_USRCHMK            ********   X   05
EXE$GRANTID               ********   X   08
EXE$HIBER                 ********   X   05
EXE$IDTOASC               0000022E R     05
EXE$IMGACT                00000216 R     05
EXE$IMGFIX                ********   X   08
EXE$IMGSTA                ********   X   08
EXE$LCKPAG                ********   X   05
EXE$LDB_SYNCH             00000236 RG    05
EXE$LKWSET                ********   X   05
EXE$LOAD_EDISP            ********   X   03
EXE$LOAD_KDISP            ********   X   05
EXE$MGBLSC                ********   X   05
EXE$MTACCESS              ********   X   05
EXE$NUMTIM                ********   X   03
EXE$PURGWS                ********   X   05
EXE$PUTMSG                ********   X   08
EXE$QIO                   ********   X   05
EXE$READEF                ********   X   05
EXE$REFLECT               ********   X   05
EXE$RESUME                ********   X   05
EXE$REVOKID               ********   X   08
EXE$RUNDWN                ********   X   05
EXE$SCHDWK                ********   X   05
EXE$SETAST                ********   X   05
EXE$SETEF                 ********   X   05
EXE$SETEXV                ********   X   05
EXE$SETIME                ********   X   05
```

EXE$SETIMR	********	X	05	FREE_MASK	=	00000FFC			
EXE$SETPFM	********	X	05	GET	=	00000015			
EXE$SETPRA	********	X	05	GETCHN	=	00000043			
EXE$SETPRI	********	X	05	GETCHN_MASK	=	00000FFC			
EXE$SETPRN	********	X	05	GETCJI	=	0000403B			
EXE$SETPRT	********	X	05	GETCJI_MASK	=	00000010			
EXE$SETPRV	********	X	05	GETDEV	=	00000044			
EXE$SETRWM	********	X	05	GETDEV_MASK	=	00000FFC			
EXE$SETSFM	********	X	05	GETDVI	=	0000004D			
EXE$SETSSF	********	X	05	GETDVI$_ASTADR	=	00000018			
EXE$SETSTK	********	X	05	GETDVI$_ASTPRM	=	0000001C			
EXE$SETSWM	********	X	05	GETDVI$_CHAN	=	00000008			
EXE$SNDACC	********	X	03	GETDVI$_DEVNAM	=	0000000C			
EXE$SNDERR	********	X	05	GETDVI$_EFN	=	00000004			
EXE$SNDJBC	********	X	03	GETDVI$_IOSB	=	00000014			
EXE$SNDOPR	********	X	03	GETDVI$_ITMLST	=	00000010			
EXE$SNDSMB	********	X	03	GETDVI$_NARGS	=	00000008			
EXE$SRCHANDLER	********	X	08	GETDVI$_NULLARG	=	00000020			
EXE$SSFAIL	********	X	05	GETDVI_MASK	=	00000FFC			
EXE$SUCCESS	000001AA	RG	05	GETJNL	=	00004031			
EXE$SUSPND	********	X	05	GETJNL_MASK	=	00000010			
EXE$TRNLNM	********	X	05	GETJPI	=	00000045			
EXE$TRNLOG	********	X	08	GETJPI$_ASTADR	=	00000018			
EXE$ULKPAG	********	X	05	GETJPI$_ASTPRM	=	0000001C			
EXE$ULWSET	********	X	05	GETJPI$_EFN	=	00000004			
EXE$UNWIND	********	X	08	GETJPI$_IOSB	=	00000014			
EXE$UPDSEC	********	X	05	GETJPI$_ITMLST	=	00000010			
EXE$UPDSECW	000001EC	R	05	GETJPI$_NARGS	=	00000007			
EXE$WAITFR	********	X	05	GETJPI$_PIDADR	=	00000008			
EXE$WAKE	********	X	05	GETJPI$_PRCNAM	=	0000000C			
EXE$WFLAND	********	X	05	GETJPI_COMMON		00000626	R	08	
EXE$WFLOR	********	X	05	GETJPI_MASK	=	00000FFC			
EXE$WRITEJNL	********	X	08	GETJPI_SYNCH		000001FC	R	05	
EXE$WRITEJNLW	********	X	08	GETJPI_SYNCH_MASK	=	00000004			
EXEDSP	00000084	R	03	GETLKI	=	00000053			
EXINSARG	00000021	R	03	GETLKI$_ASTADR	=	00000014			
EXIT	= 00000020			GETLKI$_ASTPRM	=	00000018			
EXIT_MASK	= 00000010			GETLKI$_EFN	=	00000004			
EXPREG	= 00000021			GETLKI$_IOSB	=	00000010			
EXPREG_MASK	= 000001FC			GETLKI$_ITMLST	=	0000000C			
EXTEND	= 00000022			GETLKI$_LKIDADR	=	00000008			
EXTEND_MASK	= 00000FFC			GETLKI$_NARGS	=	00000007			
FAOL_MASK	= 00000FFC			GETLKI$_RESERVED	=	0000001C			
FAO_MASK	= 00000FFC			GETLKI_MASK	=	00000FFC			
FILESCAN	= 00000034			GETMSG_MASK	=	00000FFC			
FILESCAN_MASK	= 00000FFC			GETPTI	=	0000000F			
FIND	= 00000013			GETPTI_MASK	=	000007FC			
FIND_MASK	= 00000FFC			GETQUI	=	0000000B			
FINISH_RDB	= 00000009			GETQUI_MASK	=	00000FFC			
FINISH_RDB_MASK	= 00000FFC			GETRUI	=	00004032			
FLUSH	= 00000023			GETRUI_MASK	=	00000010			
FLUSH_MASK	= 00000FFC			GETSYI	=	0000004C			
FORCEJNLW_MASK	= 00000FFC			GETSYI$_ASTADR	=	00000018			
FORCEJNL_MASK	= 00000FFC			GETSYI$_ASTPRM	=	0000001C			
FORCEX	= 00000022			GETSYI$_CSIDADR	=	00000008			
FORCEX_MASK	= 0000003C			GETSYI$_EFN	=	00000004			
FREE	= 00000014			GETSYI$_IOSB	=	00000014			

```
GETSYI$_ITMLST                  = 00000010
GETSYI$_NARGS                   = 00000007
GETSYI$_NODENAME                = 0000000C
GETSYI_MASK                     = 00000FFC
GETTIM                          = 00000002
GETTIM_MASK                     = 00000000
GET_MASK                        = 00000FFC
GRANTID_MASK                    = 0000000C
HIBER                           = 00000023
HIBER_MASK                      = 0000003C
IDTOASC                         = 0000000A
IDTOASC_MASK                    = 00000FFC
ILLSER                            000001A4 R      05
IMGACT                          = 00000003
IMGACT_MASK                     = 00000FFC
IMGFIX_MASK                     = 00000030
IMGSTA_MASK                     = 00000000
INHEXCP                           00000009 R      05
INHEXCP1                          00000000 R      05
INSARG                            00000050 R      05
IPL$_ASTDEL                     = 00000002
KCASCTR                         = 00000057
KCASE                             000000CA R      05
KCASMAX                         = 00000055
KERDSP                            000000BD R      05
KINSARG                           00000049 R      05
LCKPAG                          = 00000024
LCKPAG_MASK                     = 000001FC
LKWSET                          = 00000025
LKWSET_MASK                     = 000001FC
MARKPOINTRU                     = 00004015
MARKPOINTRU_MASK                = 0000007C
MGBLSC                          = 00000026
MGBLSC_MASK                     = 00000FFC
MNTJMD                          = 00004037
MNTJMD_MASK                     = 00000010
MODFLT                          = 00004033
MODFLTW                         = 00004033
MODFLTW_MASK                    = 00000010
MODFLT_MASK                     = 00000010
MODIFY                          = 00000024
MODIFY_MASK                     = 00000FFC
MTACCESS                        = 00000056
MTACCESS_MASK                   = 00000FFC
NUMTIM                          = 00000004
NUMTIM_MASK                     = 000000FC
NXTVOL                          = 00000025
NXTVOL_MASK                     = 00000FFC
OPEN                            = 00000026
OPEN_MASK                       = 00000FFC
PARSE                           = 0000002B
PARSE_MASK                      = 00000FFC
PCB$B_ASTACT                    = 0000000C
PCB$L_STS                       = 00000024
PCB$V_SSFEXC                    = 00000006
PCB$W_MTXCNT                    - 0000000E
PHASE1                          - 00004012
PHASE1_MASK                     = 0000007C
PHASE2                          = 00004013
PHASE2_MASK                     = 0000007C
POSJNL                          = 00004034
POSJNLW                         = 00004034
POSJNLW_MASK                    = 00000010
POSJNL_MASK                     = 00000010
PR$_IPL                         = 00000012
PSL$M_CURMOD                    = 03000000
PSL$S_CURMOD                    = 00000002
PSL$S_IPL                       = 00000005
PSL$V_CURMOD                    = 00000018
PSL$V_IPL                       = 00000010
PURGWS                          = 00000027
PURGWS_MASK                     = 000001FC
PUT                             = 00000016
PUTMSG_MASK                     = 00000FFC
PUT_MASK                        = 00000FFC
QIO                             = 00000028
QIO$_ASTADR                     = 00000014
QIO$_ASTPRM                     = 00000018
QIO$_CHAN                       = 00000008
QIO$_EFN                        = 00000004
QIO$_FUNC                       = 0000000C
QIO$_IOSB                       = 00000010
QIO$_NARGS                      = 0000000C
QIO$_P1                         = 0000001C
QIO$_P2                         = 00000020
QIO$_P3                         = 00000024
QIO$_P4                         = 00000028
QIO$_P5                         = 0000002C
QIO$_P6                         = 00000030
QIOW_RET                          00000015 R      08
QIO_ENQ_SYNCH                     00000645 R      08
QIO_MASK                        = 00000FFC
RAB$B_BLN                       = 00000001
RCASCTR                         = 00000035
RCASMAX                         = 00000022
RCASMIN                         = 00000012
READ                            = 00000017
READEF                          = 00000029
READEF_MASK                     = 0000003C
READJNL                         = 00004035
READJNLW                        = 00004035
READJNLW_MASK                   = 00000010
READJNL_MASK                    = 00000010
READ_MASK                       = 00000FFC
RECOVER                         = 00004036
RECOVERW                        = 00004036
RECOVERW_MASK                   = 00000010
RECOVER_MASK                    = 00000010
REENTERRU                       = 00004010
REENTERRU_MASK                  = 0000007C
RELEASE                         = 00000018
RELEASE_MASK                    = 00000FFC
REMOVE                          = 0000002C
REMOVE_MASK                     = 00000FFC
```

```
RENAME                  =  0000002D
RENAME_MASK             =  00000FFC
RESETRD                 =  00004016
RESETRU_MASK            =  0000007C
RESUME                  =  0000002A
RESUME_MASK             =  0000003C
REVOKID_MASK            =  0000000C
REWIND                  =  00000027
REWIND_MASK             =  00000FFC
RMS$_STALL                 ********    X    08
RMS$_STR                   ********    X    03
RMSCAK_STALL               0000035F R       08
RMSRUHNDLR              =  00000033
RMSRUHNDLR_MASK         =  00000FFC
RMSRUNDWN               =  00000032
RMSRUNDWN_MASK          =  00000FFC
RMSSYNC                 =  000003D6 R       08
RMSVECEND                  00000488 R       08
RMSWAIT_BR                 00000359 R       08
RMSWAIT_IO_DONE            00000320 R       08
RMSWBR                  =  0000044E R       08
RMS_ERR                    000000F2 R       03
RMS_ERR_BR                 00000480 R       08
RMS_WAIT_SYNC              000000D5 R       03
RUF$KCASCTR             =  0000401A
RUNDWN                  =  0000002B
RUNDWN_MASK             =  000000FC
RUSTATUS                =  00004019
RUSTATUS_MASK           =  0000007C
SCH$GL_CORPCB              ********    X    05
SCH$NEQLVL                 ********    X    05
SCHDWK                  =  0000002C
SCHDWK_MASK             =  000003FC
SEARCH                  =  0000002E
SEARCH_MASK             =  00000FFC
SETAST                  =  0000002D
SETAST_MASK             =  0000003C
SETDDIR                 =  0000002F
SETDDIR_MASK            =  00000FFC
SETDFPROT               =  00000030
SETDFPROT_MASK          =  0000000C
SETEF                   =  0000002E
SETEF_MASK              =  0000003C
SETEXD                  =  0000002F
SETEXV_MASK             =  0000003C
SETIME                  =  00000046
SETIME_MASK             =  00000FFC
SETIMR                  =  00000032
SETIMR_MASK             =  00000FFC
SETPFM                  =  00000040
SETPFM_MASK             =  00000FFC
SETPRA                  =  00000031
SETPRA_MASK             =  0000003C
SETPRI                  =  00000033
SETPRI_MASK             =  0000003C
SETPRN                  =  00000030
SETPRN_MASK             =  000C03FC
SETPRT                  =  00000034
SETPRT_MASK             =  000003FC
SETPRV                  =  00000047
SETPRV_MASK             =  000001FC
SETRWM                  =  00000035
SETRWM_MASK             =  00000010
SETSFM                  =  00000036
SETSFM_MASK             =  00000010
SETSSF                  =  0000004A
SETSSF_MASK             =  00000010
SETSTK                  =  0000004B
SETSTK_MASK             =  0000001C
SETSWM                  =  00000037
SETSWM_MASK             =  00000010
SGN$C_SYSVECPGS         =  00000005
SNDACC                  =  00000007
SNDACC_MASK             =  00000FFC
SNDERR                  =  0000001F
SNDERR_MASK             =  0000003C
SNDJBC                  =  00000001
SNDJBC$_ASTADR          =  00000018
SNDJBC$_ASTPRM          =  0000001C
SNDJBC$_EFN             =  00000004
SNDJBC$_FUNC            =  00000008
SNDJBC$_IOSB            =  00000014
SNDJBC$_ITMLST          =  00000010
SNDJBC$_NARGS           =  00000007
SNDJBC$_NULLARG         =  0000000C
SNDJBC_MASK             =  00000FFC
SNDOPR                  =  00000005
SNDOPR_MASK             =  00000FFC
SNDSMB                  =  00000006
SNDSMB_MASK             =  00000FFC
SPACE                   =  00000028
SPACE_MASK              =  00000FFC
SRVEXIT                    00000056 R       05
SRVREI                     00000059 R       05
SS$_ACCVIO              =  0000000C
SS$_ILLSER              =  00000104
SS$_INHCHME             =  000004D4
SS$_INHCHMK             =  000004CC
SS$_INSFARG             =  00000114
SS$_NORMAL              =  00000001
SS$_SYNCH               =  00000689
SSFAIL                     00000060 R       05
SSFAILMAIN                 000001B0 R       05
SSVECREG2                  000005C0 R       08
SSVEXC                  =  00000031
SSVEXC_MASK             =  00000FFC
STARTRD                 =  00004011
STARTRU_MASK            =  0000007C
SUSPND                  =  00000038
SUSPND_MASK             =  0000003C
SYNCH$_EFN              =  00000004
SYNCH$_IOSB             =  00000008
SYNCH$_NARGS            =  00000002
SYS$EXIT                   ********  GX    03
```

```
SYS$GB_KMASK                    00000000 RG    06
SYS$GB_KRNLNARG               = 00000000 RG    07
SYS$SYNCH                       ******** X     05
SYS$WAIT                        ******** X     03
SYS$WAITFR                      ******** GX    03
TRNLNM                        = 00000052
TRNLNM_MASK                   = 00000FFC
TRNLOG_MASK                   = 000001FC
TRUNCATE                      = 00000029
TRUNCATE_MASK                 = 00000FFC
ULKPAG                        = 00000039
ULKPAG_MASK                   = 000001FC
ULWSET                        = 0000003A
ULWSET_MASK                   = 000001FC
UNWIND_MASK                   = 0000003C
UPDATE                        = 00000019
UPDATE_MASK                   = 00000FFC
UPDSEC                        = 0000001E
UPDSEC$_ACMODE                = 0000000C
UPDSEC$_ASTADR                = 0000001C
UPDSEC$_ASTPRM                = 00000020
UPDSEC$_EFN                   = 00000014
UPDSEC$_INADR                 = 00000004
UPDSEC$_IOSB                  = 00000018
UPDSEC$_NARGS                 = 00000008
UPDSEC$_RETADR                = 00000008
UPDSEC$_UPDFLG                = 00000010
UPDSEC_MASK                   = 000001FC
USERWAIT                        0000032C R     08
VECBASE                         00000000 R     08
WAIT                          = 0000001A
WAITFR                        = 0000003B
WAITFR_MASK                   = 0000007C
WAIT_MASK                     = 00000FFC
WAKE                          = 0000003C
WAKE_MASK                     = 0000003C
WFLAND                        = 0000003D
WFLAND_MASK                   = 0000007C
WFLOR                         = 0000003E
WFLOR_MASK                    = 0000007C
WRITE                         = 0000001B
WRITEJNLW_MASK                = 00000FFC
WRITEJNL_MASK                 = 00000FFC
WRITE_MASK                    = 00000FFC
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
Y$CMODEX	00000035 (53.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
Y$CMODE	0000010D (269.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD
Y$CMODEN	00000035 (53.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
Y$CMODK	00000268 (616.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD
Y$CMODKX	00000057 (87.)	06 (6.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
Y$CMODKN	00000057 (87.)	07 (7.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$000	00000A00 (2560.)	08 (8.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:01.88
Command processing	111	00:00:00.51	00:00:05.35
Pass 1	793	00:00:33.81	00:01:50.20
Symbol table sort	0	00:00:02.80	00:00:09.34
Pass 2	353	00:00:08.25	00:00:25.65
Symbol table output	78	00:00:00.62	00:00:01.91
Psect synopsis output	0	00:00:00.04	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1366	00:00:46.10	00:02:34.37

The working set limit was 2700 pages.
264510 bytes (517 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1877 non-local and 34 local symbols.
2345 source lines were read in Pass 1, producing 53 object records in Pass 2.
49 pages of virtual memory were used to define 45 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	9
_255DUA28:[SYSLIB]STARLET.MLB;2	21
TOTALS (all libraries)	30

1236 GETS were required to define 30 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:CMODSSDSP/OBJ=OBJ$:CMODSSDSP MSRC$:CMODSSDSP/UPDATE=(ENH$:CMODSSDSP)+EXECML$/LIB
```

0373 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CMODSSDSP
LIS

BUGCHECK
LIS

COMDRVSUB
LIS

CLUSTRVEC
LIS

BUFFERCTL
LIS

DEADLOCK
LIS

CVTFILNAM
LIS

BOOPARAM
LIS

CJFSYSVEC
LIS

CVTATB
LIS

BUGCHKMSG
LIS

CONSOLIO
LIS