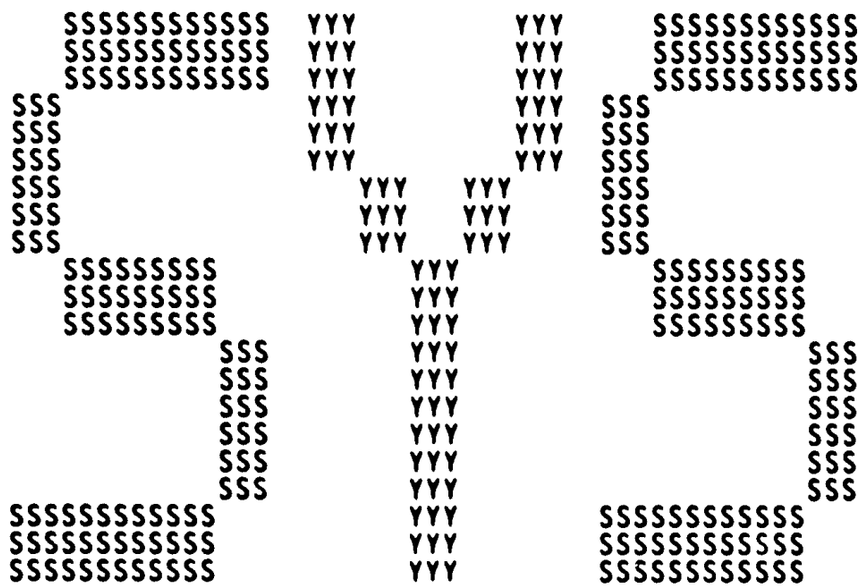

FILEID**PROCSTRT

```
PPPPPPPP    RRRRRRRR      000000      CCCCCCCC    SSSSSSSS  TTTTTTTTTT  RRRRRRRR    TTTTTTTTTT
PPPPPPPP    RRRRRRRR      000000      CCCCCCCC    SSSSSSSS  TTTTTTTTTT  RRRRRRRR    TTTTTTTTTT
PP      PP  RR      RR  OO      OO  CC          SS            TT      RR      RR      TT
PP      PP  RR      RR  OO      OO  CC          SS            TT      RR      RR      TT
PP      PP  RR      RR  OO      OO  CC          SS            TT      RR      RR      TT
PP      PP  RR      RR  OO      OO  CC          SS            TT      RR      RR      TT
PPPPPPPP    RRRRRRRR    OO      OO  CC            SSSSSS      TT      RRRRRRRR        TT
PPPPPPPP    RRRRRRRR    OO      OO  CC            SSSSSS      TT      RRRRRRRR        TT
PP          RR  RR      OO      OO  CC                  SS    TT      RR  RR          TT
PP          RR  RR      OO      OO  CC                  SS    TT      RR  RR          TT
PP          RR    RR    OO      OO  CC                  SS    TT      RR    RR        TT
PP          RR    RR    OO      OO  CC                  SS    TT      RR    RR        TT
PP          RR      RR    000000      CCCCCCCC  SSSSSSSS      TT      RR      RR      TT
PP          RR      RR    000000      CCCCCCCC  SSSSSSSS      TT      RR      RR      TT

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

Table of contents

```
0000     1          .TITLE  PROCSTRT - PROCESS STARTUP AND INITIALIZATION
0000     2          .IDENT  'V04-002'
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27
0000    28 ;++
0000    29 ; FACILITY: EXECUTIVE, PROCESS CREATION SYSTEM SERVICE
0000    30 ;
0000    31 ; ABSTRACT:
0000    32 ;       PROCSTRT CONTAINS THE CODE NECESSARY TO CONCLUDE THE CREATION
0000    33 ;       OF A PROCESS WHICH MUST BE EXECUTED IN THE CONTEXT OF THAT PROCESS.
0000    34 ;
0000    35 ; ENVIRONMENT:
0000    36 ;       MODE=KERNEL, EXECUTING IN CONTEXT OF NEW PROCESS
0000    37 ;
0000    38 ; AUTHOR: R. I. HUSTVEDT        , CREATION DATE: 27-DEC-76
0000    39 ;
0000    40 ; MODIFIED BY:
0000    41 ;
0000    42 ;       V03-002 RAS0332         Ron Schaefer            14-Sep-1984
0000    43 ;               Check for RMS$_BUSY status in the RMS exit handler
0000    44 ;               so as to prevent an infinite loop if the handler
0000    45 ;               has interrupted RMS rundown badly.   In that case,
0000    46 ;               give up on trying to do rundown cleanly.
0000    47 ;               Also, change the rundown type to do a full PPF rundown.
0000    48 ;
0000    49 ;       V04-001 JWT0195         Jim Teague              11-Sep-1984
0000    50 ;               Replace RMS exec mode exit handler for proper rundown
0000    51 ;               of single-image processes.
0000    52 ;
0000    53 ;       V03-043 LJK0288         Lawrence J. Kenah       9-Aug-1984
0000    54 ;               The AUTHPRI cell now exists in both the PCB and the PHD.
0000    55 ;
0000    56 ;       V03-042 ACG0440         Andrew C. Goldstein,    24-Jul-1984  10:45
0000    57 ;               Add ref count field to ORB
```

```
0000     58 ;
0000     59 ;       V03-041 HH0040          Hai Huang                19-Jul-1984
0000     60 ;               Add routine EXE$CRE_GTABLE.
0000     61 ;
0000     62 ;       V03-040 LMP0275         L. Mark Pilant,          12-Jul-1984  20:14
0000     63 ;               Initialize the ACL info in the ORB to be a null descriptor
0000     64 ;               list rather than an empty queue.  This avoids the overhead
0000     65 ;               of locking and unlocking the ACL mutex, only to find out
0000     66 ;               that the ACL was empty.
0000     67 ;
0000     68 ;       V03-039 RAS0319         Ron Schaefer             29-Jun-1984
0000     69 ;               Initialize the logical name table name translation
0000     70 ;               cache queue to empty.
0000     71 ;
0000     72 ;       V03-038 LJK0272         Lawrence J. Kenah        10-Apr-1984
0000     73 ;               Initialize VECSET array at the same time that the VECRESET
0000     74 ;               array is set up.
0000     75 ;
0000     76 ;       V03-037 MHB0121         Mark Bramhall             9-Apr-1984
0000     77 ;               Move new spawn CLI information to P1 space.
0000     78 ;
0000     79 ;       V03-036 TMK0010         Todd M. Katz             27-Mar-1984
0000     80 ;               Modify the logical name system services to make use of the
0000     81 ;               updated internal protection checking mechanisms. What this
0000     82 ;               involves is replacing the CHIP protection template in the
0000     83 ;               templates for the group and job-wide logical name table with
0000     84 ;               a template for a quad-word aligned Object Rights Block, and
0000     85 ;               making sure that the appropriate fields within the Object Rights
0000     86 ;               blocks are initialized when group and job-wide logical name
0000     87 ;               tables are created.
0000     88 ;
0000     89 ;       V03-035 WMC0007         Wayne Cardoza            21-Mar-1984
0000     90 ;               Create the default image I/O segment.
0000     91 ;
0000     92 ;       V03-034 TMK0009         Todd M. Katz             07-Mar-1984
0000     93 ;               Add a hash code field, LNMX$W_HASH, to every translation block
0000     94 ;               of every logical name and logical name table template defined.
0000     95 ;               This hash code field will be used in an optimization of logical
0000     96 ;               name table name processing.
0000     97 ;
0000     98 ;       V03-033 TMK0008         Todd M. Katz             17-Feb-1984
0000     99 ;               Fix alignment problems with LNM$GROUP and LNM$JOB introduced
0000    100 ;               by one of the two logical name table alignment bug fixes
0000    101 ;               below. This is done by defining the symbols GROUP_XEND_SIZE
0000    102 ;               and JOB_XEND_SIZE. These two symbols represent the actual
0000    103 ;               amount of storage utilized by LNM$GROUP and LNM$JOB
0000    104 ;               respectively while the two symbols GROUP_SIZE and JOB_SIZE
0000    105 ;               represent the amount of storage actually allocated for these
0000    106 ;               logical names. These two new symbols are needed by the PROCSTRT
0000    107 ;               code that constructs the equivalence strings for these logical
0000    108 ;               names. This code depends upon knowledge of the actual amount of
0000    109 ;               storage, allocated to the logical names, which is utilized by
0000    110 ;               the logical names.
0000    111 ;
0000    112 ;       V03-032 LY00b8          Larry Yetto              17-FEB-1984 14:36
0000    113 ;               Fix alignment of logical name tables
0000    114 ;
```

```
0000   115 ;       V03-031 LY00b6          Larry Yetto             16-FEB-1984 14:21
0000   116 ;               Fix alignment of logical name tables
0000   117 ;
0000   118 ;       V03-030 WMC0006         Wayne Cardoza           12-Jan-1983
0000   119 ;               Create DZRO space for XQP, CRF no longer an option.
0000   120 ;
0000   121 ;       V03-029 TMK0007         Todd M. Katz            26-Jan-1984
0000   122 ;               Fix a bug introduced in TMK0006. In EXE$CRE_JGTABLE, if an
0000   123 ;               existing group table is found when an attempt is made to
0000   124 ;               create-if a new group table, then the paged pool for what would
0000   125 ;               have become a new group logical name table must be deleted. This
0000   126 ;               was not being done. This resulted in multiple group tables with
0000   127 ;               identical names, and had the further undesirable affect of
0000   128 ;               causing paged pool to disappear over time as more and more of
0000   129 ;               these duplicate group logical name tables were created as a
0000   130 ;               by-product of process creation.
0000   131 ;
0000   132 ;       V03-028 LJK0258         Lawrence J. Kenah       18-Jan-1984
0000   133 ;               Fix bug introduced in LJK0257. Save R4 and R5 before they
0000   134 ;               are destroyed by a MOVC3 instruction.
0000   135 ;
0000   136 ;       V03-027 LJK0257         Lawrence J. Kenah       28-Dec-1983
0000   137 ;               Add support for longer text strings in the PQB. Fix error
0000   138 ;               paths. Add initialization code for P1 lookaside list.
0000   139 ;
0000   140 ;       V03-026 SHZ0001         Stephen H. Zalewski     27-Dec-1983
0000   141 ;               Remove RMS executive mode exit handler.
0000   142 ;
0000   143 ;       V03-025 TMK0006         Todd M. Katz            10-Nov-1983
0000   144 ;               Optimize the logical name and logical name table creations that
0000   145 ;               are required to be done at process creation time. This is done
0000   146 ;               by replacing all $CRELNT and $CRELNM system service calls with
0000   147 ;               the corresponding code that hand constructs the logical name
0000   148 ;               blocks and oversees their insertion into the overall logical
0000   149 ;               name structure. In addition, group logical name tables will no
0000   150 ;               longer be created for sub-processes. As in the case of the
0000   151 ;               job-wide logical name table, it will be assumed that the group
0000   152 ;               logical name table for a sub-process already exists.
0000   153 ;
0000   154 ;       V03-024 TMK0005         Todd M. Katz            12-Oct-1983
0000   155 ;               If the process being created is not a sub-process, create the
0000   156 ;               job-wide logical name table.
0000   157 ;
0000   158 ;       V03-023 TMK0004         Todd M. Katz            26-Sep-1983
0000   159 ;               Change the protection on the group logical name table to
0000   160 ;               SYSTEM:RWED OWNER: GROUP:R WORLD: so that processes with system
0000   161 ;               access rights can access and modify any group table.
0000   162 ;
0000   163 ;       V03-022 RAS0181         Ron Schaefer            5-Sep-1983
0000   164 ;               Convert creation of SYS$INPUT, SYS$OUTPUT, SYS$ERROR,
0000   165 ;               SYS$DISK and TT logical names to use $CRELNM.
0000   166 ;
0000   167 ;       V03-021 TMK0003         Todd M. Katz            22-Aug-1983
0000   168 ;               Create the Group Logical Name Table with the protection mask
0000   169 ;               G:R as part of the changes being made to the logical name table
0000   170 ;               protection mechanism to provide for upwards compatibility
0000   171 ;               between V3 and V4. In addition, specify the GROUP and NO_ALIAS
```

```
0000   172 ;               attributes while creating the Group Logical Name Table so that
0000   173 ;               the table can not be aliased, and so it will get marked
0000   174 ;               specially as a Group Logical Name Table.
0000   175 ;
0000   176 ;               There is no need to perform any protection checking of
0000   177 ;               process-private logical name tables; therefore, process-private
0000   178 ;               logical name tables are no longer created with CHIP protection
0000   179 ;               structures. Remove the CHIP protection structure from the
0000   180 ;               process space logical name directory template as well as
0000   181 ;               any fixing up of this CHIP which was being done as part of
0000   182 ;               process creation.
0000   183 ;
0000   184 ;       V03-020 WMC0005         Wayne Cardoza           02-Jul-1983
0000   185 ;               assorted performance improvements.
0000   186 ;
0000   187 ;       V03-019 RAS0176         Ron Schaefer            28-Jul-1983
0000   188 ;               Fix group logical name table creation to be in octal;
0000   189 ;               and clean up the code somewhat.
0000   190 ;
0000   191 ;       V03-018 LJK0221         Lawrence J. Kenah       5-Jul-1983
0000   192 ;               Initialize listheads for image descriptor blocks.
0000   193 ;
0000   194 ;       V03-017 DMW4061         DMWalp                  23-Jun-1983
0000   195 ;               Change $xxLNM value parameters to be by reference
0000   196 ;
0000   197 ;       V03-016 DMW4048         DMWalp                  13-Jun-1983
0000   198 ;               Fix proctection problems with new logical name structures.
0000   199 ;               Add execmode entry point to IMGDMP.
0000   200 ;
0000   201 ;       V03-015 ADE9005         Alan D. Eldridge        31-May-1983
0000   202 ;               Make BSBW to MMG$IMGRESET a JSB.
0000   203 ;
0000   204 ;       V03-014 RAS0158         Ron Schaefer            23-May-1983
0000   205 ;               Add CHIP protection to logical name structures.
0000   206 ;               Currently only SOGW protection is supported.
0000   207 ;               Fix quota of LNM$PROCESS_DIRECTORY.
0000   208 ;
0000   209 ;       V03-103 WMC0003         Wayne Cardoza           10-May-1983
0000   210 ;               Change XQP merge to use global sections rather than IMGACT.
0000   211 ;
0000   212 ;       V03-012 TMK0002         Todd M. Katz            26-Apr-1983
0000   213 ;               Create the following logical name structures at process creation
0000   214 ;               time:
0000   215 ;
0000   216 ;               1. LNM$PROCESS_TABLE.
0000   217 ;               2. LNM$GROUP_xxxxxxxx (The Group Logical Name Table).
0000   218 ;               3. LNM$GROUP.
0000   219 ;               4. LNM$PROCESS.
0000   220 ;
0000   221 ;               Change the name of LNT$PROCESS_DIRECTORY to
0000   222 ;               LNM$PROCESS_DIRECTORY.
0000   223 ;
0000   224 ;       V03-011 CDS0003         Christian D. Saether    20-Apr-1983
0000   225 ;               Fix to V03-009.  Don't merge xqp if EXE$V_INIT is clear.
0000   226 ;
0000   227 ;       V03-010 TMK0001         Todd M. Katz            14-Apr-1983
0000   228 ;               Make the following changes to the setting up of the process
```

```
0000    229 ;               directory logical name table:
0000    230 ;
0000    231 ;               1. Make the table a kernel mode access table.
0000    232 ;               2. The address of the process directory table's table header
0000    233 ;                  is placed in LNMB$L_TABLE.
0000    234 ;               3. The bit LNMTH$V_DIRECTORY is set in LNMTH$B_FLAGS.
0000    235 ;               4. The field LNMTH$L_LOGNAM is eliminated.
0000    236 ;
0000    237 ;       V03-009 CDS0002         Christian D. Saether    12-Apr-1983
0000    238 ;               Always merge f11bxqp.  Check for xqpmerge errors.
0000    239 ;
0000    240 ;       V03-008 WMC0002         Wayne Cardoza           01-Apr-1983
0000    241 ;               Add second half of IMGDMP.
0000    242 ;
0000    243 ;       V03-007 WMC0001         Wayne Cardoza           14-Mar-1983
0000    244 ;               Add image dump interface.
0000    245 ;
0000    246 ;       V03-006 ACG0305         Andrew C. Goldstein,    16-Dec-1982  14:03
0000    247 ;               Get hibernate flag correctly in EXE$PROCIMGACT entry
0000    248 ;
0000    249 ;       V03-005 CDS0001         Christian D. Saether    16-Dec-1982
0000    250 ;               Add routine to merge F11BXQP into P1 space.
0000    251 ;
0000    252 ;       V03-004 DMW4017         DMWalp                  15-Dec-1982
0000    253 ;               Added creation of new logical name hash table and
0000    254 ;               logical name process directory
0000    255 ;
0000    256 ;       V03-003 JWH0117         Jeffrey W. Horn         01-Nov-1982
0000    257 ;               Make the sizes of the RMS Process IO Segment and the
0000    258 ;               Process Allocation Region controlable via SYSGEN
0000    259 ;               parameters.
0000    260 ;
0000    261 ;       V03-002 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    262 ;               Add $DYNDEF.
0000    263 ;
0000    264 ;--
```

```
0000    267          .SBTTL  DECLARATIONS
0000    268 ;
0000    269 ; INCLUDE FILES:
0000    270 ;
0000    271
0000    272          $CCBDEF                         ; CHANNEL CONTROL BLOCK DEFINITIONS
0000    273          $CHFDEF                         ; CONDITION HANDLER DEFINITIONS
0000    274          $CLIMSGDEF                      ; COMMAND INTERPRETER STATUS CODES
0000    275          $DYNDEF                         ; DYNAMIC STRUCTURE TYPE CODES
0000    276          $IACDEF                         ; IMAGE ACTIVATION FLAGS
0000    277          $IHDDEF                         ; IMAGE HEADER DESCRIPTOR DEFINITIONS
0000    278          $IMGACTDEF                      ; IMAGE ACTIVATOR ARGUMENTS
0000    279          $JIBDEF                         ; DEFINE JIB OFFSETS
0000    280          $JPIDEF                         ; JPI ITEM CODES
0000    281          $IMPDEF                         ; RMS IMPURE AREA DEFINITIONS
0000    282          $LNMDEF                         ; LOGICAL NAME DEFINITIONS
0000    283          $LNMSTRDEF                      ; LOGICAL NAME STRUCTURE DEFINITIONS
0000    284          $OPDEF                          ; SYMBOLIC NAMES FOR INSTRUCTION OPCODES
0000    285          $ORBDEF                         ; DEFINE OBJECT RIGHTS BLOCK OFFSETS
0000    286          $PCBDEF                         ; DEFINE PCB OFFSETS
0000    287          $PHDDEF                         ; DEFINE PROCESS HEADER
0000    288          $PQBDEF                         ; DEFINE PROCESS QUOTA BLOCK OFFSETS
0000    289          $PRDEF                          ; DEFINE PROCESSOR REGISTERS
0000    290          $PRTDEF                         ; DEFINE PAGE PROTECTION VALUES
0000    291          $PRVDEF                         ; PRIVILEGE BIT DEFINITIONS
0000    292          $PSLDEF                         ; DEFINE PSL FIELDS
0000    293          $RMSDEF                         ; DEFINE RMS ERROR STATUSES
0000    294          $SECDEF                         ; SECTION FLAGS
0000    295          $SGNDEF                         ; DEFINE SYSGEN CONSTANTS
0000    296          $SSDEF                          ; DEFINE SYSTEM STATUS CODES
0000    297          $STSDEF                         ; DEFINE STATUS CODE FIELDS
0000    298
0000    299 ;
0000    300 ; ASSUMPTIONS ABOUT THE STRUCTURE OF LOGICAL NAME AND OBJECT RIGHTS BLOCKS:
0000    301 ;
0000    302
0000    303          ASSUME  LNMB$L_FLINK,       EQ,  0
0000    304          ASSUME  LNMB$L_FLINK+4,     EQ,  LNMB$L_BLINK
0000    305          ASSUME  LNMB$L_BLINK+4,     EQ,  LNMB$W_SIZE
0000    306          ASSUME  LNMB$W_SIZE+2,      EQ,  LNMB$B_TYPE
0000    307          ASSUME  LNMB$B_TYPE+1,      EQ,  LNMB$B_ACMODE
0000    308          ASSUME  LNMB$B_ACMODE+1,    EQ,  LNMB$L_TABLE
0000    309          ASSUME  LNMB$L_TABLE+4,     EQ,  LNMB$B_FLAGS
0000    310          ASSUME  LNMB$B_FLAGS+1,     EQ,  LNMB$T_NAME
0000    311
0000    312          ASSUME  LNMX$B_FLAGS,       EQ,  0
0000    313          ASSUME  LNMX$B_FLAGS+1,     EQ,  LNMX$B_INDEX
0000    314          ASSUME  LNMX$B_INDEX+1,     EQ,  LNMX$W_HASH
0000    315          ASSUME  LNMX$W_HASH+2,      EQ,  LNMX$T_XLATION
0000    316
0000    317          ASSUME  LNMTH$B_FLAGS,      EQ,  0
0000    318          ASSUME  LNMTH$B_FLAGS+1,    EQ,  LNMTH$L_HASH
0000    319          ASSUME  LNMTH$L_HASH+4,     EQ,  LNMTH$L_ORB
0000    320          ASSUME  LNMTH$L_ORB+4,      EQ,  LNMTH$L_NAME
0000    321          ASSUME  LNMTH$L_NAME+4,     EQ,  LNMTH$L_PARENT
0000    322          ASSUME  LNMTH$L_PARENT+4,   EQ,  LNMTH$L_CHILD
0000    323          ASSUME  LNMTH$L_CHILD+4,    EQ,  LNMTH$L_SIBLING
```

```
                0000   324          ASSUME  LNMTH$L_SIBLING+4, EQ,  LNMTH$L_QTABLE
                0000   325          ASSUME  LNMTH$L_QTABLE+4,  EQ,  LNMTH$L_BYTESLM
                0000   326          ASSUME  LNMTH$L_BYTESLM+4, EQ,  LNMTH$L_BYTES
                0000   327
                0000   328          ASSUME  ORB$L_OWNER,        EQ,  0
                0000   329          ASSUME  ORB$L_OWNER+4,      EQ,  ORB$L_ACL_MUTEX
                0000   330          ASSUME  ORB$L_ACL_MUTEX+4,  EQ,  ORB$W_SIZE
                0000   331          ASSUME  ORB$W_SIZE+2,       EQ,  ORB$B_TYPE
                0000   332          ASSUME  ORB$B_TYPE+1,       EQ,  ORB$B_FLAGS
                0000   333          ASSUME  ORB$B_FLAGS+3,      EQ,  ORB$W_REFCOUNT
                0000   334          ASSUME  ORB$W_REFCOUNT+2,   EQ,  ORB$Q_MODE_PROT
                0000   335          ASSUME  ORB$Q_MODE_PROT+8,  EQ,  ORB$L_SYS_PROT
                0000   336          ASSUME  ORB$L_SYS_PROT+4,   EQ,  ORB$L_OWN_PROT
                0000   337          ASSUME  ORB$L_OWN_PROT+4,   EQ,  ORB$L_GRP_PROT
                0000   338          ASSUME  ORB$L_GRP_PROT+4,   EQ,  ORB$L_WOR_PROT
                0000   339          ASSUME  ORB$L_WOR_PROT+4,   EQ,  ORB$L_ACL_COUNT
                0000   340          ASSUME  ORB$L_ACL_COUNT+4,  EQ,  ORB$L_ACL_DESC
                0000   341          ASSUME  ORB$L_ACL_DESC+4,   EQ,  ORB$R_MIN_CLASS
                0000   342          ASSUME  ORB$R_MIN_CLASS+ORB$S_MIN_CLASS,-
                0000   343                                  EQ,  ORB$R_MAX_CLASS
                0000   344          ASSUME  ORB$R_MAX_CLASS+ORB$S_MAX_CLASS,-
                0000   345                                  EQ,  ORB$K_LENGTH
                0000   346
                0000   347 ;
                0000   348 ; MACROS:
                0000   349 ;
                0000   350
                0000   351          .MACRO  CRELNM,XLATION,XLATION_ATTR,LNMX,LNMB
                0000   352          BSBW    CRELNM
                0000   353          .WORD   <XLATION>
                0000   354          .WORD   <XLATION_ATTR>
                0000   355          .WORD   <LNMX>
                0000   356          .WORD   <LNMB>
                0000   357          .END    CRELNM
                0000   358
                0000   359 ;
                0000   360 ; EQUATED SYMBOLS:
                0000   361 ;
                0000   362
00000000        0000   363 NXTKVEC=0                       ;OFFSET TO NEXT FREE KERNEL VECTOR
00000100        0000   364 NXTEVEC=256                     ;OFFSET TO NEXT FREE EXEC VECTOR
00000200        0000   365 NXTRVEC=512                     ;OFFSET TO NEXT FREE RUNDWN VECTOR
00000300        0000   366 NXTMVEC=768                     ;OFFSET TO NEXT MESSAGE VECTOR
```

```
                                    0000   368
                                    0000   369 ;
                                    0000   370 ; OWN STORAGE:
                                    0000   371 ;
                                    0000   372
                                00000C00   373          .PSECT  YYPROCSTRT,5            ; PAGED PSECT
                                    0000   374
                                    0000   375 EXE$GQ_SYSDISK::                         ; DESCRIPTOR FO  SYS$DISK
49 44 24 53 59 53 00000008'010E0000'  0000   376          .ASCID  /SYS$DISK/
                                 4B 53  000E
                                    0010   377 DEFDESC:                                ; DEFAULT IMAGE FILE NAME
      45 58 45 2E 00000018'010E0000'  0010   378          .ASCID  /.EXE/
                                    001C   379
42 41 39 38 37 36 35 34 33 32 31 30  001C   380 CHARS:  .ASCII  /0123456789ABCDEF/      ; CHARS FOR OCTAL (HEX) -> ASCII CONVS
                     46 45 44 43  0028
                                    002C   381
                                    002C   382 ;
                                    002C   383 ; CATCH ALL HANDLER FATAL CONDITION MESSAGE SUFFIX.
                                    002C   384 ;
                                    002C   385
66 20 74 69 78 65 20 65 67 61 6D 69  002C   386 SUFFIX: .ASCIZ  /image exit forced./    ;
               00 2E 64 65 63 72 6F  0038
                                    003F   387
                                    003F   388 ;
                                    003F   389 ; STRINGS FOR IMAGE DUMP MERGE.
                                    003F   390 ;
                                    003F   391
                                    003F   392 DEFAULTNAMDSC:
49 4C 24 53 59 53 00000047'010E0000'  003F   393          .ASCID  /SYS$LIBRARY:.EXE/
      45 58 45 2E 3A 59 52 41 52 42  004D
                                    0057   394 IMGDMPNAM:
50 4D 44 47 4D 49 0000005F'010E0000'  0057   395          .ASCID  /IMGDMP/
                                    0065   396
                                    0065   397 ;
                                    0065   398 ; TEMPLATES FOR THE LOGICAL NAME TABLES AND NAMES CREATED WITHIN PROCSTRT.
                                    0065   399 ;
                                    0065   400
                                    0065   401          .ALIGN  QUAD
                                    0068   402 PROC_DIR:                               ; LNM$PROCESS_DIRECTORY TEMPLATE
                          00000000  0068   403          .LONG   0                      ; FORWARD LINK
                          00000000  006C   404          .LONG   0                      ; BACK LINK
                              0058' 0070   405          .WORD   PROC_DIR_SIZE          ; SIZE OF STRUCTURE
                                40  0072   406          .BYTE   DYN$C_LNM              ; TYPE OF STRUCTURE
                                00  0073   407          .BYTE   PSL$C_KERNEL           ; KERNEL ACCESS MODE
                          00000000  0074   408          .LONG   0                      ; CONTAINING TABLE HEADER ADDRESS
                                19  0078   409          .BYTE   LNMB$M_NO_ALIAS!-      ; NO ALIAS ALLOWED
                                    0079   410                  LNMB$M_TABLE!-         ; THIS IS A TABLE
                                    0079   411                  LNMB$M_NODELETE        ; ... THAT CANNOT BE DELETED
53 53 45 43 4F 52 50 24 4D 4E 4C 00' 0079   412          .ASCIC  "LNM$PROCESS_DIRECTORY" ; DIRECTORY NAME AS COUNTED STRING
      59 52 4F 54 43 45 52 49 44 5F  0085
                                15  0079
                                    008F   413
                                02  008F   414          .BYTE   LNMX$M_TERMINAL        ; TERMINAL TRANSLATION
                                82  0090   415          .BYTE   LNMX$C_TABLE           ; SPECIAL TABLE TRANSLATION INDEX
                              0000  0091   416          .WORD   0                      ; TRANSLATION HASH CODE
                                25  0093   417          .BYTE   LNMTH$K_LENGTH         ; SIZE OF TABLE HEADER BLOCK
                                    0094   418
```

```
                    0000002C  0094    419 PROC_DIR_LNMTH = . - PROC_DIR
                          02  0094    420         .BYTE   LNMTH$M_DIRECTORY       ; TABLE IS FOR A DIRECTORY
                    00000000  0095    421         .LONG   0                       ; ADDRESS OF HASH TABLE
                    00000000  0099    422         .LONG   0                       ; ADDRESS OF OBJECT RIGHTS BLOCK
                    00000000  009D    423         .LONG   0                       ; ADDRESS OF CONTAINING LNMB BLOCK
                    00000000  00A1    424         .LONG   0                       ; ADDRESS OF PARENT TABLE
                    00000000  00A5    425         .LONG   0                       ; ADDRESS OF CHILD TABLE
                    00000000  00A9    426         .LONG   0                       ; ADDRESS OF SIBLING TABLE
                    00000000  00AD    427         .LONG   0                       ; ADDRESS OF TABLE HOLDING QUOTA
                    7FFFFFFF  00B1    428         .LONG   ^X7FFFFFFF              ; INITIAL QUOTA    ( POSITIVE INFINITY )
                    7FFFFFFF  00B5    429         .LONG   ^X7FFFFFFF              ; REMAINING QUOTA ( POSITIVE INFINITY )
                              00B9    430
                          04  00B9    431         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
                              00BA    432         .ALIGN  QUAD
                    00000058  00C0    433 PROC_DIR_SIZE = . - PROC_DIR
                              00C0    434
                    00000058  00C0    435 PROC_TABLE = . - PROC_DIR               ; LNM$PROCESS_TABLE TEMPLATE
                    00000000  00C0    436         .LONG   0                       ; FORWARD LINK
                    00000000  00C4    437         .LONG   0                       ; BACK LINK
                        0050' 00C8    438         .WORD   PROC_TABLE_SIZE         ; SIZE OF STRUCTURE
                          40  00CA    439         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                          00  00CB    440         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
                    00000000  00CC    441         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                          09  00D0    442         .BYTE   LNMB$M_NO_ALIAS!-       ; NON-ALIASABLE
                              00D1    443                 LNMB$M_TABLE            ; A TABLE
53 53 45 43 4F 52 50 24 4D 4E 4C 00'  00D1    444         .ASCIC  "LNM$PROCESS_TABLE"     ; TABLE NAME AS COUNTED STRING
                 45 4C 42 41 54 5F    00DD
                          11  00D1
                              00E3    445
                          02  00E3    446         .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
                          82  00E4    447         .BYTE   LNMX$C_TABLE            ; SPECIAL TABLE TRANSLATION INDEX
                        0000  00E5    448         .WORD   0                       ; TRANSLATION HASH CODE
                          25  00E7    449         .BYTE   LNMTH$K_LENGTH          ; SIZE OF TABLE HEADER BLOCK
                              00E8    450
                    00000080  00E8    451 PROC_TABLE_LNMTH = . - PROC_DIR
                          00  00E8    452         .BYTE   0                       ; FLAGS BYTE
                    00000000  00E9    453         .LONG   0                       ; ADDRESS OF HASH TABLE
                    00000000  00ED    454         .LONG   0                       ; ADDRESS OF OBJECT RIGHTS BLOCK
                    00000000  00F1    455         .LONG   0                       ; ADDRESS OF CONTAINING LNMB BLOCK
                    00000000  00F5    456         .LONG   0                       ; ADDRESS OF PARENT TABLE
                    00000000  00F9    457         .LONG   0                       ; ADDRESS OF CHILD TABLE
                    00000000  00FD    458         .LONG   0                       ; ADDRESS OF SIBLING TABLE
                    00000000  0101    459         .LONG   0                       ; ADDRESS OF TABLE HOLDING QUOTA
                    00000000  0105    460         .LONG   0                       ; INITIAL QUOTA    ( POOLED )
                    00000000  0109    461         .LONG   0                       ; REMAINING QUOTA ( POOLED )
                              010D    462
                          04  010D    463         .BYTE   LNMX$M_xEND             ; LAST TRANSLATION
                              010E    464         .ALIGN  QUAD
                    00000050  0110    465 PROC_TABLE_SIZE = . - PROC_DIR - PROC_TABLE
                              0110    466
                              0110    467
                    000000A8  0110    468 PROCESS = . - PROC_DIR                  ; LNM$PROCESS TEMPLATE
                    00000000  0110    469         .LONG   0                       ; FORWARD LINK
                    00000000  0114    470         .LONG   0                       ; BACK LINK
                        0038' 0118    471         .WORD   PROCESS_SIZE            ; SIZE OF STRUCTURE
                          40  011A    472         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                          00  011B    473         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
```

```
                          00000000  011C   474          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                                00  0120   475          .BYTE   0                       ; FLAGS BYTE
53 53 45 43 4F 52 50 24 4D 4E 4C 00'  0121   476          .ASCIC  "LNM$PROCESS"           ; LOGICAL NAME AS COUNTED STRING
                                0B  0121
                                    012D   477
                                02  012D   478          .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
                                00  012E   479          .BYTE   0                       ; TRANSLATION INDEX IS 0
                              0000  012F   480          .WORD   0                       ; TRANSLATION HASH CODE
53 53 45 43 4F 52 50 24 4D 4E 4C 00'  0131   481          .ASCIC  "LNM$PROCESS_TABLE"     ; TRANSLATION AS COUNTED STRING
                   45 4C 42 41 54 5F  013D
                                11  0131
                                    0143   482
                                04  0143   483          .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
                                    0144   484          .ALIGN  QUAD
                          00000038  0148   485 PROCESS_SIZE = . - PROC_DIR - PROCESS
                                    0148   486
                          000000E0  0148   487 GROUP = . - PROC_DIR                     ; LNM$GROUP TEMPLATE
                          00000000  0148   488          .LONG   0                       ; FORWARD LINK
                          00000000  014C   489          .LONG   0                       ; BACK LINK
                              0038' 0150   490          .WORD   GROUP_SIZE              ; SIZE OF STRUCTURE
                                40  0152   491          .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                                00  0153   492          .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
                          00000000  0154   493          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                                00  0158   494          .BYTE   0                       ; FLAGS BYTE
      50 55 4F 52 47 24 4D 4E 4C 00'  0159   495          .ASCIC  "LNM$GROUP"             ; LOGICAL NAME AS COUNTED STRING
                                09  0159
                                    0163   496
                                02  0163   497          .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
                                00  0164   498          .BYTE   0                       ; TRANSLATION INDEX IS 0
                              0000  0165   499          .WORD   0                       ; TRANSLATION HASH CODE
78 5F 50 55 4F 52 47 24 4D 4E 4C 00'  0167   500          .ASCIC  "LNM$GROUP_xxxxxx"      ; TRANSLATION AS COUNTED STRING
                78 78 78 78 78 78  0173
                                10  0167
                                    0178   501
                                04  0178   502          .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
                          00000031  0179   503 GROUP_XEND_SIZE = . - PROC_DIR - GROUP
                                    0179   504          .ALIGN  QUAD
                          00000038  0180   505 GROUP_SIZE = . - PROC_DIR - GROUP
                                    0180   506
                          00000118  0180   507 JOB = . - PROC_DIR                      ; LNM$JOB TEMPLATE
                          00000000  0180   508          .LONG   0                       ; FORWARD LINK
                          00000000  0184   509          .LONG   0                       ; BACK LINK
                              0030' 0188   510          .WORD   JOB_SIZE                ; SIZE OF STRUCTURE
                                40  018A   511          .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                                00  018B   512          .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
                          00000000  018C   513          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                                00  0190   514          .BYTE   0                       ; FLAGS BYTE
            42 4F 4A 24 4D 4E 4C 00'  0191   515          .ASCIC  "LNM$JOB"               ; LOGICAL NAME AS COUNTED STRING
                                07  0191
                                    0199   516
                                02  0199   517          .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
                                00  019A   518          .BYTE   0                       ; TRANSLATION INDEX IS 0
                              0000  019B   519          .WORD   0                       ; TRANSLATION HASH CODE
78 78 78 5F 42 4F 4A 24 4D 4E 4C 00'  019D   520          .ASCIC  "LNM$JOB_xxxxxxxx"      ; TRANSLATION AS COUNTED STRING
                78 78 78 78 78 78  01A9
                                10  019D
                                    01AE   521
```

```
                                                     D 10
                          04  01AE    522          .BYTE   LNMX$M_XEND                ; LAST TRANSLATION
                    0000002F  01AF    523 JOB_XEND_SIZE = . - PROC_DIR - JOB
                              01AF    524          .ALIGN  QUAD
                    00000030  01B0    525 JOB_SIZE = . - PROC_DIR - JOB
                              01B0    526
                    00000148  01B0    527 SYS$INPUT = . - PROC_DIR                    ; SYS$INPUT TEMPLATE
                    00000000  01B0    528          .LONG   0                         ; FORWARD LINK
                    00000000  01B4    529          .LONG   0                         ; BACK LINK
                        0120' 01B8    530          .WORD   SYS$INPUT_SIZE            ; SIZE OF STRUCTURE
                          40  01BA    531          .BYTE   DYN$C_LNM                 ; TYPE OF STRUCTURE
                          01  01BB    532          .BYTE   PSL$C_EXEC                ; EXECUTIVE ACCESS MODE
                    00000000  01BC    533          .LONG   0                         ; CONTAINING TABLE HEADER ADDRESS
                          00  01C0    534          .BYTE   0                         ; FLAGS BYTE
 54 55 50 4E 49 24 53 59 53 00' 01C1    535          .ASCIC  "SYS$INPUT"               ; LOGICAL NAME AS COUNTED STRING
                          09  01C1
                              01CB    536
                    00000163  01CB    537 SYS$INPUT_LNMX = . - PROC_DIR
                          00  01CB    538          .BYTE   0                         ; TRANSLATION ATTRIBUTES
                          00  01CC    539          .BYTE   0                         ; TRANSLATION INDEX IS 0
                        0000  01CD    540          .WORD   0                         ; TRANSLATION HASH CODE
04'04'04'04'04'04'04'04'04'04'04'04' 01CF    541          .BYTE   LNMX$M_XEND[PQB$S_INPUT]; WORST CASE TRANSLATION AS COUNTED STRING
04'04'04'04'04'04'04'04'04'04'04'04' 01DB
04'04'04'04'04'04'04'04'04'04'04'04' 01E7
04'04'04'04'04'04'04'04'04'04'04'04' 01F3
04'04'04'04'04'04'04'04'04'04'04'04' 01FF
04'04'04'04'04'04'04'04'04'04'04'04' 020B
04'04'04'04'04'04'04'04'04'04'04'04' 0217
04'04'04'04'04'04'04'04'04'04'04'04' 0223
04'04'04'04'04'04'04'04'04'04'04'04' 022F
04'04'04'04'04'04'04'04'04'04'04'04' 023B
04'04'04'04'04'04'04'04'04'04'04'04' 0247
04'04'04'04'04'04'04'04'04'04'04'04' 0253
04'04'04'04'04'04'04'04'04'04'04'04' 025F
04'04'04'04'04'04'04'04'04'04'04'04' 026B
04'04'04'04'04'04'04'04'04'04'04'04' 0277
04'04'04'04'04'04'04'04'04'04'04'04' 0283
04'04'04'04'04'04'04'04'04'04'04'04' 028F
04'04'04'04'04'04'04'04'04'04'04'04' 029B
04'04'04'04'04'04'04'04'04'04'04'04' 02A7
04'04'04'04'04'04'04'04'04'04'04'04' 02B3
04'04'04'04'04'04'04'04'04'04'04'04' 02BF
                    04'04'04'04'04' 02CB
                              02CF    542
                          04  02CF    543          .BYTE   LNMX$M_XEND               ; LAST TRANSLATION
                              02D0    544          .ALIGN  QUAD
                    00000120  02D0    545 SYS$INPUT_SIZE = . - PROC_DIR - SYS$INPUT
                              02D0    546
                    00000268  02D0    547 TT = . - PROC_DIR                           ; TT TEMPLATE
                    00000000  02D0    548          .LONG   0                         ; FORWARD LINK
                    00000000  02D4    549          .LONG   0                         ; BACK LINK
                        0120' 02D8    550          .WORD   TT_SIZE                   ; SIZE OF STRUCTURE
                          40  02DA    551          .BYTE   DYN$C_LNM                 ; TYPE OF STRUCTURE
                          01  02DB    552          .BYTE   PSL$C_EXEC                ; EXECUTIVE ACCESS MODE
                    00000000  02DC    553          .LONG   0                         ; CONTAINING TABLE HEADER ADDRESS
                          00  02E0    554          .BYTE   0                         ; FLAGS BYTE
                    54 54 00' 02E1    555          .ASCIC  "TT"                      ; LOGICAL NAME AS COUNTED STRING
                          02  02E1
```

```
                           02E4   556
                0000027C   02E4   557 TT_LNMX = . - PROC_DIR
                     00    02E4   558          .BYTE   0                        ; TRANSLATION ATTRIBUTES
                     00    02E5   559          .BYTE   0                        ; TRANSLATION INDEX IS 0
                   0000    02E6   560          .WORD   0                        ; TRANSLATION HASH CODE
04'04'04'04'04'04'04'04'04'04'04'04'  02E8   561          .BYTE   LNMX$M_XEND[PQB$S_INPUT]; WORST CASE TRANSLATION AS COUNTED STRING
04'04'04'04'04'04'04'04'04'04'04'04'  02F4
04'04'04'04'04'04'04'04'04'04'04'04'  0300
04'04'04'04'04'04'04'04'04'04'04'04'  030C
04'04'04'04'04'04'04'04'04'04'04'04'  0318
04'04'04'04'04'04'04'04'04'04'04'04'  0324
04'04'04'04'04'04'04'04'04'04'04'04'  0330
04'04'04'04'04'04'04'04'04'04'04'04'  033C
04'04'04'04'04'04'04'04'04'04'04'04'  0348
04'04'04'04'04'04'04'04'04'04'04'04'  0354
04'04'04'04'04'04'04'04'04'04'04'04'  0360
04'04'04'04'04'04'04'04'04'04'04'04'  036C
04'04'04'04'04'04'04'04'04'04'04'04'  0378
04'04'04'04'04'04'04'04'04'04'04'04'  0384
04'04'04'04'04'04'04'04'04'04'04'04'  0390
04'04'04'04'04'04'04'04'04'04'04'04'  039C
04'04'04'04'04'04'04'04'04'04'04'04'  03A8
04'04'04'04'04'04'04'04'04'04'04'04'  03B4
04'04'04'04'04'04'04'04'04'04'04'04'  03C0
04'04'04'04'04'04'04'04'04'04'04'04'  03CC
04'04'04'04'04'04'04'04'04'04'04'04'  03D8
                04'04'04'04'  03E4
                           03E8   562
                     04    03E8   563          .BYTE   LNMX$M_XEND              ; LAST TRANSLATION
                           03E9   564          .ALIGN  QUAD
                00000120   03F0   565 TT_SIZE = . - PROC_DIR - TT
                           03F0   566
                00000388   03F0   567 SYS$OUTPUT = . - PROC_DIR                 ; SYS$OUTPUT TEMPLATE
                00000000   03F0   568          .LONG   0                       ; FORWARD LINK
                00000000   03F4   569          .LONG   0                       ; BACK LINK
                    0128'  03F8   570          .WORD   SYS$OUTPUT_SIZE         ; SIZE OF STRUCTURE
                      40   03FA   571          .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                      01   03FB   572          .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
                00000000   03FC   573          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                      00   0400   574          .BYTE   0                       ; FLAGS BYTE
   54 55 50 54 55 4F 24 53 59 53 00'  0401   575          .ASCIC  "SYS$OUTPUT"            ; LOGICAL NAME AS COUNTED STRING
                      0A   0401
                           040C   576
                000003A4   040C   577 SYS$OUTPUT_LNMX = . - PROC_DIR
                      00   040C   578          .BYTE   0                       ; TRANSLATION ATTRIBUTES
                      00   040D   579          .BYTE   0                       ; TRANSLATION INDEX IS 0
                    0000   040E   580          .WORD   0                       ; TRANSLATION HASH CODE
04'04'04'04'04'04'04'04'04'04'04'04'  0410   581          .BYTE   LNMX$M_XEND[PQB$S_OUTPUT]; WORST CASE TRANSLATION AS COUNTED STRING
04'04'04'04'04'04'04'04'04'04'04'04'  041C
04'04'04'04'04'04'04'04'04'04'04'04'  0428
04'04'04'04'04'04'04'04'04'04'04'04'  0434
04'04'04'04'04'04'04'04'04'04'04'04'  0440
04'04'04'04'04'04'04'04'04'04'04'04'  044C
04'04'04'04'04'04'04'04'04'04'04'04'  0458
04'04'04'04'04'04'04'04'04'04'04'04'  0464
04'04'04'04'04'04'04'04'04'04'04'04'  0470
04'04'04'04'04'04'04'04'04'04'04'04'  047C
```

```
04'04'04'04'04'04'04'04'04'04'04'04'  0488
04'04'04'04'04'04'04'04'04'04'04'04'  0494
04'04'04'04'04'04'04'04'04'04'04'04'  04A0
04'04'04'04'04'04'04'04'04'04'04'04'  04AC
04'04'04'04'04'04'04'04'04'04'04'04'  04B8
04'04'04'04'04'04'04'04'04'04'04'04'  04C4
04'04'04'04'04'04'04'04'04'04'04'04'  04D0
04'04'04'04'04'04'04'04'04'04'04'04'  04DC
04'04'04'04'04'04'04'04'04'04'04'04'  04E8
04'04'04'04'04'04'04'04'04'04'04'04'  04F4
04'04'04'04'04'04'04'04'04'04'04'04'  0500
                    04'04'04'04'  050C
                              0510  582
                           04 0510  583          .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
                              0511  584          .ALIGN  QUAD
                     00000128 0518  585 SYS$OUTPUT_SIZE = . - PROC_DIR - SYS$OUTPUT
                              0518  586
                     000004B0 0518  587 SYS$ERROR = . - PROC_DIR                   ; SYS$ERROR TEMPLATE
                     00000000 0518  588          .LONG   0                       ; FORWARD LINK
                     00000000 051C  589          .LONG   0                       ; BACK LINK
                         0120' 0520  590          .WORD   SYS$ERROR_SIZE          ; SIZE OF STRUCTURE
                           40 0522  591          .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                           01 0523  592          .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
                     00000000 0524  593          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                           00 0528  594          .BYTE   0                       ; FLAGS BYTE
      52 4F 52 52 45 24 53 59 53 00' 0529  595          .ASCIC  "SYS$ERROR"             ; LOGICAL NAME AS COUNTED STRING
                           09 0529
                              0533  596
                     000004CB 0533  597 SYS$ERROR_LNMX = . - PROC_DIR
                           00 0533  598          .BYTE   0                       ; TRANSLATION ATTRIBUTES
                           00 0534  599          .BYTE   0                       ; TRANSLATION INDEX IS 0
                         0000 0535  600          .WORD   0                       ; TRANSLATION HASH CODE
04'04'04'04'04'04'04'04'04'04'04'04'  0537  601          .BYTE   LNMX$M_XEND[PQB$S_ERROR]; WORST CASE TRANSLATION AS COUNTED STRING
04'04'04'04'04'04'04'04'04'04'04'04'  0543
04'04'04'04'04'04'04'04'04'04'04'04'  054F
04'04'04'04'04'04'04'04'04'04'04'04'  055B
04'04'04'04'04'04'04'04'04'04'04'04'  0567
04'04'04'04'04'04'04'04'04'04'04'04'  0573
04'04'04'04'04'04'04'04'04'04'04'04'  057F
04'04'04'04'04'04'04'04'04'04'04'04'  058B
04'04'04'04'04'04'04'04'04'04'04'04'  0597
04'04'04'04'04'04'04'04'04'04'04'04'  05A3
04'04'04'04'04'04'04'04'04'04'04'04'  05AF
04'04'04'04'04'04'04'04'04'04'04'04'  05BB
04'04'04'04'04'04'04'04'04'04'04'04'  05C7
04'04'04'04'04'04'04'04'04'04'04'04'  05D3
04'04'04'04'04'04'04'04'04'04'04'04'  05DF
04'04'04'04'04'04'04'04'04'04'04'04'  05EB
04'04'04'04'04'04'04'04'04'04'04'04'  05F7
04'04'04'04'04'04'04'04'04'04'04'04'  0603
04'04'04'04'04'04'04'04'04'04'04'04'  060F
04'04'04'04'04'04'04'04'04'04'04'04'  061B
04'04'04'04'04'04'04'04'04'04'04'04'  0627
                    04'04'04'04'  0633
                              0637  602
                           04 0637  603          .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
                              0638  604          .ALIGN  QUAD
```

```
                     00000120  0638    605 SYS$ERROR_SIZE = . - PROC_DIR - SYS$ERROR
                               0638    606
                     000005D0  0638    607 SYS$DISK = . - PROC_DIR                 ; SYS$DISK TEMPLATE
                     00000000  0638    608         .LONG   0                       ; FORWARD LINK
                     00000000  063C    609         .LONG   0                       ; BACK LINK
                         0120' 0640    610         .WORD   SYS$DISK_SIZE           ; SIZE OF STRUCTURE
                           40  0642    611         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                           01  0643    612         .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
                     00000000  0644    613         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
                           00  0648    614         .BYTE   0                       ; FLAGS BYTE
        4B 53 49 44 24 53 59 53 00'    615         .ASCIC  "SYS$DISK"              ; LOGICAL NAME AS COUNTED STRING
                           08  0649
                               0652    616
                     000005EA  0652    617 SYS$DISK_LNMX = . - PROC_DIR
                           00  0652    618         .BYTE   0                       ; TRANSLATION ATTRIBUTES
                           00  0653    619         .BYTE   0                       ; TRANSLATION INDEX IS 0
                         0000  0654    620         .WORD   0                       ; TRANSLATION HASH CODE
04'04'04'04'04'04'04'04'04'04'04'04'  0656    621         .BYTE   LNMX$M_XEND[PQB$S_DISK] ; WORST CASE TRANSLATION AS COUNTED STRING
04'04'04'04'04'04'04'04'04'04'04'04'  0662
04'04'04'04'04'04'04'04'04'04'04'04'  066E
04'04'04'04'04'04'04'04'04'04'04'04'  067A
04'04'04'04'04'04'04'04'04'04'04'04'  0686
04'04'04'04'04'04'04'04'04'04'04'04'  0692
04'04'04'04'04'04'04'04'04'04'04'04'  069E
04'04'04'04'04'04'04'04'04'04'04'04'  06AA
04'04'04'04'04'04'04'04'04'04'04'04'  06B6
04'04'04'04'04'04'04'04'04'04'04'04'  06C2
04'04'04'04'04'04'04'04'04'04'04'04'  06CE
04'04'04'04'04'04'04'04'04'04'04'04'  06DA
04'04'04'04'04'04'04'04'04'04'04'04'  06E6
04'04'04'04'04'04'04'04'04'04'04'04'  06F2
04'04'04'04'04'04'04'04'04'04'04'04'  06FE
04'04'04'04'04'04'04'04'04'04'04'04'  070A
04'04'04'04'04'04'04'04'04'04'04'04'  0716
04'04'04'04'04'04'04'04'04'04'04'04'  0722
04'04'04'04'04'04'04'04'04'04'04'04'  072E
04'04'04'04'04'04'04'04'04'04'04'04'  073A
04'04'04'04'04'04'04'04'04'04'04'04'  0746
                     04'04'04'04'    0752
                               0756    622
                           04  0756    623         .BYTE   LNMX$M_XEND              ; LAST TRANSLATION
                               0757    624         .ALIGN  QUAD
                     00000120  0758    625 SYS$DISK_SIZE = . - PROC_DIR - SYS$DISK
                     000006F0  0758    626 P1_ALLOC_SIZE = . - PROC_DIR
                               0758    627
                               0758    628     .ALIGN  5
                               0760    629 GROUP_TABLE:                            ; LNM$GROUP_xxxxxx TEMPLATE
                     00000000  0760    630         .LONG   0                       ; FORWARD LINK
                     00000000  0764    631         .LONG   0                       ; BACK LINK
                         00C0' 0768    632         .WORD   GROUP_TABLE_SIZE        ; SIZE OF STRUCTURE
                           40  076A    633         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
                           00  076B    634         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
                     00000000' 076C    635         .ADDRESS LNM_SYSTEM_DIR_LNMTH   ; CONTAINING TABLE HEADER ADDRESS
                           09  0770    636         .BYTE   LNMB$M_NO_ALIAS!-       ; NON-ALIASABLE
                               0771    637                 LNMB$M_TABLE            ; A TABLE
78 5F 50 55 4F 52 47 24 4D 4E 4C 00'  0771    638         .ASCIC  "LNM$GROUP_xxxxxx"  ; TABLE NAME AS COUNTED STRING
                  78 78 78 78 78 78   077D
```

```
                              10 0771
                                 0782
                              02 0782          .BYTE   LNMX$M_TERMINAL          ; TERMINAL TRANSLATION
                              82 0783          .BYTE   LNMX$C_TABLE             ; SPECIAL TABLE TRANSLATION INDEX
                            0000 0784          .WORD   0                        ; TRANSLATION HASH CODE
                              25 0786          .BYTE   LNMTH$K_LENGTH           ; SIZE OF TABLE HEADER BLOCK
                                 0787
                        00000027 0787  GROUP_TABLE_LNMTH = . - GROUP_TABLE
                              05 0787          .BYTE   LNMTH$M_SHAREABLE!-      ; TABLE IS SHAREABLE
                                 0788                  LNMTH$M_GROUP            ; A GROUP TABLE
                        00000000 0788          .LONG   0                        ; ADDRESS OF HASH TABLE
                        00000000 078C          .LONG   0                        ; ADDRESS OF OBJECT RIGHTS BLOCK
                        00000000 0790          .LONG   0                        ; ADDRESS OF CONTAINING LNMB BLOCK
                       00000000' 0794          .ADDRESS LNM_SYSTEM_DIR_LNMTH    ; ADDRESS OF PARENT TABLE
                        00000000 0798          .LONG   0                        ; ADDRESS OF CHILD TABLE
                        00000000 079C          .LONG   0                        ; ADDRESS OF SIBLING TABLE
                       00000000' 07A0          .ADDRESS LNM_SYSTEM_DIR_LNMTH    ; ADDRESS OF TABLE HOLDING QUOTA
                        00000000 07A4          .LONG   0                        ; INITIAL QUOTA   ( POOLED )
                        00000000 07A8          .LONG   0                        ; REMAINING QUOTA ( POOLED )
                                 07AC
                              04 07AC          .BYTE   LNMX$M_XEND              ; LAST TRANSLATION
                                 07AD
                                 07AD          .ALIGN  QUAD
                        00000050 07B0  GROUP_TABLE_ORB = . - GROUP_TABLE
                        00000000 07B0          .LONG   0                        ; GROUP NUMBER + 0 MEMBER NUMBER
                       0000 FFFF 07B4          .WORD   -1, 0                    ; INITIALIZED ACL MUTEX
                           0070' 07B8          .WORD   GROUP_TABLE_ORB_SIZ      ; SIZE OF OBJECT RIGHTS BLOCK
                              49 07BA          .BYTE   DYN$C_ORB                ; BLOCK TYPE
                              00 07BB          .BYTE   0                        ; NOTE NO ACL AS YET
                        00000000 07BC          .LONG   0                        ; ZERO RESERVED WORD & REF COUNT
               00000000 00000000 07C0          .QUAD   0                        ; OBJECT DOES NOT HAVE AN ACCESS MODE
                        00000000 07C8          .LONG   ^X00000000               ; SYSTEM PROTECTION IS RWED
                        0000000F 07CC          .LONG   ^X0000000F               ; OWNER PROTECTION
                        0000000E 07D0          .LONG   ^X0000000E               ; GROUP PROTECTION IS R
                        0000000F 07D4          .LONG   ^X0000000F               ; WORLD PROTECTION
               00000000 00000000 07D8          .LONG   0,0                      ; NULL INITIAL ACL
00'00'00'00'00'00'00'00'00'00'00'00' 07E0      .BYTE   0[ORB$S_MIN_CLASS]       ; MINIMUM CLASSIFICATION MASK
             00'00'00'00'00'00'00'00' 07EC
00'00'00'00'00'00'00'00'00'00'00'00' 07F4      .BYTE   0[ORB$S_MAX_CLASS]       ; MAXIMUM CLASSIFICATION MASK
             00'00'00'00'00'00'00'00' 0800
                                 0808          .ALIGN  5
                        00000070 0820  GROUP_TABLE_ORB_SIZ = . - GROUP_TABLE - GROUP_TABLE_ORB
                        000000C0 0820  GROUP_TABLE_SIZE = . - GROUP_TABLE
                                 0820
                        000000C0 0820  JOB_TABLE = . - GROUP_TABLE              ; LNM$JOB_xxxxxxxx TEMPLATE
                        00000000 0820          .LONG   0                        ; FORWARD LINK
                        00000000 0824          .LONG   0                        ; BACK LINK
                           00C0' 0828          .WORD   JOB_TABLE_SIZE           ; SIZE OF STRUCTURE
                              40 082A          .BYTE   DYN$C_LNM                ; TYPE OF STRUCTURE
                              00 082B          .BYTE   PSL$C_KERNEL             ; KERNEL ACCESS MODE
                       00000000' 082C          .ADDRESS LNM_SYSTEM_DIR_LNMTH    ; CONTAINING TABLE HEADER ADDRESS
                              09 0830          .BYTE   LNMB$M_NO_ALIAS!-        ; NON-ALIASABLE
                                 0831                  LNMB$M_TABLE             ; A TABLE
78 78 78 5F 42 4F 4A 24 4D 4E 4C 00' 0831      .ASCIC  "LNM$JOB_xxxxxxxx"       ; TABLE NAME AS COUNTED STRING
                  78 78 78 78 78 0830
                              10 0831
                                 0842
```

```
                     02  0842    691          .BYTE   LNMX$M_TERMINAL          ; TERMINAL TRANSLATION
                     82  0843    692          .BYTE   LNMX$C_TABLE             ; SPECIAL TABLE TRANSLATION INDEX
                   0000  0844    693          .WORD   0                        ; TRANSLATION HASH CODE
                     25  0846    694          .BYTE   LNMTH$K_LENGTH           ; SIZE OF TABLE HEADER BLOCK
                         0847    695
                 000000E7  0847    696 JOB_TABLE_LNMTH = . - GROUP_TABLE
                     01  0847    697          .BYTE   LNMTH$M_SHAREABLE        ; SHAREABLE TABLE
                 00000000  0848    698          .LONG   0                        ; ADDRESS OF HASH TABLE
                 00000000  084C    699          .LONG   0                        ; ADDRESS OF OBJECT RIGHTS BLOCK
                 00000000  0850    700          .LONG   0                        ; ADDRESS OF CONTAINING LNMB BLOCK
                 00000000' 0854    701          .ADDRESS LNM_SYSTEM_DIR_LNMTH    ; ADDRESS OF PARENT TABLE
                 00000000  0858    702          .LONG   0                        ; ADDRESS OF CHILD TABLE
                 00000000  085C    703          .LONG   0                        ; ADDRESS OF SIBLING TABLE
                 00000000' 0860    704          .ADDRESS LNM_SYSTEM_DIR_LNMTH    ; ADDRESS OF TABLE HOLDING QUOTA
                 00000000  0864    705          .LONG   0                        ; INITIAL QUOTA
                 00000000  0868    706          .LONG   0                        ; REMAINING QUOTA
                           086C    707
                     04    086C    708          .BYTE   LNMX$M_XEND              ; LAST TRANSLATION
                           086D    709
                           086D    710          .ALIGN  QUAD
                 00000110  0870    711 JOB_TABLE_ORB = . - GROUP_TABLE
                 00000000  0870    712          .LONG   0                        ; OWNER OF JOB TABLE
                0000 FFFF  0874    713          .WORD   -1, 0                    ; INITIALIZED ACL MUTEX
                     0070' 0878    714          .WORD   JOB_TABLE_ORB_SIZ        ; SIZE OF OBJECT RIGHTS BLOCK
                     49    087A    715          .BYTE   DYN$C_ORB                ; BLOCK TYPE
                     00    087B    716          .BYTE   0                        ; NOTE NO ACL AS YET
                 00000000  087C    717          .LONG   0                        ; ZERO RESERVED WORD & REF COUNT
        00000000 00000000  0880    718          .QUAD   0                        ; OBJECT DOES NOT HAVE AN ACCESS MODE
                 00000000  0888    719          .LONG   ^X00000000               ; SYSTEM PROTECTION IS RWED
                 00000000  088C    720          .LONG   ^X00000000               ; OWNER PROTECTION IS RWED
                 0000000F  0890    721          .LONG   ^X0000000F               ; GROUP PROTECTION
                 0000000F  0894    722          .LONG   ^X0000000F               ; WORLD PROTECTION
        00000000 00000000  0898    723          .LONG   0,0                      ; NULL INITIAL ACL
00'00'00'00'00'00'00'00'00'00'00'00'00'00'00'00  08A0    724          .BYTE   0[ORB$S_MIN_CLASS]       ; MINIMUM CLASSIFICATION MASK
                                   08AC
00'00'00'00'00'00'00'00'00'00'00'00'00'00'00'00  08B4    725          .BYTE   0[ORB$S_MAX_CLASS]       ; MAXIMUM CLASSIFICATION MASK
                                   08C0
                                   08C8    726          .ALIGN  5
                 00000070  08E0    727 JOB_TABLE_ORB_SIZ = . - GROUP_TABLE - JOB_TABLE_ORB
                 000000C0  08E0    728 JOB_TABLE_SIZE = . - GROUP_TABLE - JOB_TABLE
                 00000180  08E0    729 SO_ALLOC_SIZE = . - GROUP_TABLE
                           08E0    730
                           08E0    731 ;
                           08E0    732 ; OFFSETS FROM R6 USED BY THE IMAGE DUMP MERGE.
                           08E0    733 ;
                           08E0    734
                           08E0    735          $OFFSET 0,POSITIVE,<-
                           08E0    736                  <,<<IMGACT$_NARGS+1>*4>>,-  ; ARGUMENT LIST FOR $IMGACT CALL
                           08E0    737                  <IMGACT_INADR,8>,-          ; INPUT ADDRESS RANGE TO MAP IMAGE AT
                           08E0    738                  <IMGACT_RETADR,8>,-         ; RET ADR RANGE WHERE IMAGE WAS MAPPED
                           08E0    739                  <HDRBUF,512>,-              ; IMAGE HEADER BUFFER FOR $IMGACT
                           08E0    740                  <PROCPRIV,8>,-              ; PROCESS DEFAULT PRIVILEGES
                           08E0    741                  <IMAGPRIV,8>,-              ; IMAGE PRIVILEGES
                           08E0    742                  <PHD_FLAGS,4>,-             ; PROCESS HEADER FLAGS
                           08E0    743                  <JPI_PROC,12>,-             ; GETJPI LIST ITEM FOR PROCESS PRIVILEGES
                           08E0    744                  <JPI_IMAG,12>,-             ; GETJPI LIST ITEM FOR IMAGE PRIVILEGES
                           08E0    745                  <JPI_FLAG,12>,-             ; GETJPI LIST ITEM FOR FLAGS
```

```
08E0    746                 <JPI_END,4>,-           ; GETJPI LIST TERMINATOR
08E0    747                 <SCRATCHSIZE,0>,-       ; SIZE OF AREA ADDRESS OFF OF FP
08E0    748                 >
0024        IMGACT_INADR:
002C        IMGACT_RETADR:
0034        HDRBUF:
0234        PROCPRIV:
023C        IMAGPRIV:
0244        PHD_FLAGS:
0248        JPI_PROC:
0254        JPI_IMAG:
0260        JPI_FLAG:
026C        JPI_END:
0270        SCRATCHSIZE:
08E0    749
```

```
                                      08E0          .SBTTL  EXE$PROCSTRT - STARTUP NEW PROCESS
                                      08E0
                                      08E0    ;++
                                      08E0    ; FUNCTIONAL DESCRIPTION:
                                      08E0    ;
                                      08E0    ; CALLING SEQUENCE:
                                      08E0    ;       NONE
                                      08E0    ;
                                      08E0    ; INPUT PARAMETERS:
                                      08E0    ;       SCH$GL_CURPCB - POINTS TO PCB OF CURRENT PROCESS
                                      08E0    ;       PCB$L_PQB - POINTER TO PROCESS QUOTA BLOCK
                                      08E0    ;
                                      08E0    ; IMPLICIT INPUTS:
                                      08E0    ;       IPL = IPL$_ASTDEL
                                      08E0    ;
                                      08E0    ; OUTPUT PARAMETERS:
                                      08E0    ;       NONE
                                      08E0    ;
                                      08E0    ; IMPLICIT OUTPUTS:
                                      08E0    ;       LOGICAL NAMES ARE DEFINED FOR 'SYS$INPUT', 'SYS$OUTPUT', AND 'SYS$ERROR'
                                      08E0    ;       BASED ON THE STRINGS PASSED IN THE PROCESS QUOTA BLOCK.
                                      08E0    ;
                                      08E0    ; COMPLETION CODES:
                                      08E0    ;       NONE
                                      08E0    ;
                                      08E0    ; SIDE EFFECTS:
                                      08E0    ;       NONE
                                      08E0    ;
                                      08E0    ;--
                                      08E0
                                      08E0
                                      08E0    ;
                                      08E0    ;  The PQB address must be stored before any instruction that can cause a page
                                      08E0    ;  fault. If a page fault occurs and the process is put into a resource wait
                                      08E0    ;  state, then the PQB address will be lost because the EFWM field, used to
                                      08E0    ;  store the resource number, overlaps PCB$L_PQB. This forces the first
                                      08E0    ;  two instructions into a nonpaged program section.
                                      08E0    ;
                                  00000000          .PSECT  AEXENONPAGED
                                      0000
                                      0000    EXE$PROCSTRT::                                  ; STARTUP NEW PROCESS
          54     00000000'EF      D0  0000          MOVL    SCH$GL_CURPCB,R4                ; GET POINTER TO CURRENT PCB
                  56    4C A4      D0  0007          MOVL    PCB$L_PQB(R4),R6                ; GET POINTER TO PROCESS QUOTA BLOCK
                 000008E0'GF      17  000B          JMP     G^EXE_PROCSTRT                  ; CONTINUE IN PAGEABLE EXEC
                                      0011
                                  000008E0          .PSECT  YYPROCSTRT
                                      08E0
                                      08E0    EXE_PROCSTRT:
                                      08E0    ;
                                      08E0    ;       N O T E :   THERE CAN BE NO I/O TO A PROCESS CHANNEL BETWEEN HERE
                                      08E0    ;                   AND THE END OF THE NEW CHANNEL CREATION CODE.
                                      08E0    ;
00000000'GF      00000000'GF      D0  08E0          MOVL    G^MMG$GL_RMSBASE,G^CTL$GL_RMSBASE ; SET RMS DISPATCHER BASE
00000000'GF      00000000'GF      D0  08EB          MOVL    G^MMG$GL_CTLBASVA,G^CTL$GL_CTLBASVA ; SET CTL BASE ADDRESS
                                      08F6    ;
                                      08F6    ; INITIALIZE THE DISPATCH VECTORS.
```

```
                                     08F6  ;
          55   00000000'9F     9E    08F6          MOVAB   @#CTL$A_DISPVEC,R5        ; GET ADR OF 1ST VECTOR PAGE
                    65   04     9A    08FD          MOVZBL  #04,NXTKVEC(R5)           ; SET OFFSET TO 1ST FREE KERNEL VECTOR
               0100 C5   04     9A    0900          MOVZBL  #04,NXTEVEC(R5)           ; SET OFFSET TO 1ST FREE EXEC VECTOR
               0200 C5   04     9A    0905          MOVZBL  #04,NXTRVEC(R5)           ; SET OFFSET TO 1ST FREE RUNDWN VECTOR
               0300 C5   04     9A    090A          MOVZBL  #04,NXTMVEC(R5)           ; SET OFFSET TO 1ST FREE MESSAGE VECTOR
                 04 A5   05     9A    090F          MOVZBL  #OP$_RSB,NXTKVEC+4(R5)    ; SET AN "RSB" INTO THE 1ST FREE VECTOR
               0104 C5   05     9A    0913          MOVZBL  #OP$_RSB,NXTEVEC+4(R5)    ; SET AN "RSB" INTO THE 1ST FREE VECTOR
               0204 C5   05     9A    0918          MOVZBL  #OP$_RSB,NXTRVEC+4(R5)    ; SET AN "RSB" INTO THE 1ST FREE VECTOR
               0304 C5   05     9A    091D          MOVZBL  #OP$_RSB,NXTMVEC+4(R5)    ; SET AN "RSB" INTO THE 1ST FREE VECTOR
       00000000'GF    04 A5     9E    0922          MOVAB   NXTKVEC+4(R5),G^CTL$GL_USRCHMK ; SET POINTER TO START OF VECTOR
     00000000'GF    0104 C5     9E    092A          MOVAB   NXTEVEC+4(R5),G^CTL$GL_USRCHME ; SET POINTER TO START OF VECTOR
     00000000'GF    0204 C5     9E    0933          MOVAB   NXTRVEC+4(R5),G^CTL$GL_USRUNDWN ; SET POINTER TO START OF VECTOR
     00000000'GF    0304 C5     9E    093C          MOVAB   NXTMVEC+4(R5),G^CTL$GL_GETMSG ; SET POINTER TO START OF VECTOR
          00000000'GF    54     D0    0945          MOVL    R4,G^CTL$GL_PCB           ; SET POINTER TO PCB
                                      094C          ASSUME PHD$Q_PRIVMSK EQ 0
          55   00000000'9F     D0    094C          MOVL    @#CTL$GL_PHD,R5           ; GET SAFE POINTER TO PROCESS HEADER WINDOW
             04 44 A6   00     E1    0953          BBC     #PQB$V_IMGDMP,PQB$W_FLAGS(R6),10$
                36 A5   20     A8    0958          BISW    #PHD$M_IMGDMP,PHD$W_FLAGS(R5) ; IMAGE DUMP WAS REQUESTED
                                      095C
                                      095C  ; SET UP P1 SPACE LOOKASIDE LIST FOR KERNEL MODE BUFFERS
                                      095C
          52   00000000'GF     9E    095C  10$:    MOVAB   G^CTL$GL_KRPFL,R2         ; GET LISTHEAD ADDRESS
          51   00000000'GF     9E    0963          MOVAB   G^CTL$GL_KRP,R1           ; GET
                   50   00'     D0    096A          MOVL    S^#CTL$C_KRP_COUNT,R0
                        0E     15    096D          BLEQ    30$
             04 B2      61     0E    096F  20$:    INSQUE  (R1),@4(R2)
          51   00000000'8F     C0    0973          ADDL    #CTL$C_KRP_SIZE,R1
                     F2 50     F5    097A          SOBGTR  R0,20$
                                      097D  30$:
                                      097D          .ENABL  LSB
                                      097D                                            ; SET PROPER QUOTAS FROM PQB
          5C A5     18 A6      D0    097D          MOVL    PQB$L_CPULM(R6),PHD$L_CPULIM(R5) ; SET CPU TIME LIMIT
          40 A5     0C A6      F7    0982          CVTLW   PQB$L_ASTLM(R6),PHD$W_ASTLM(R5)  ; SET AST LIMIT
          51   00000000'EF     D0    0987          MOVL    SGN$GL_MAXWSCNT,R1        ; GET MAXIMUM WORKING SET LIST LENGTH
50  00000000'EF   00000000'EF  C3    098E          SUBL3   SCH$GL_FREELIM,PFN$GL_PHYPGCNT,R0 ; GET AVAILABLE PAGES
                   51   50     D1    099A          CMPL    R0,R1                     ; MINIMIZE WITH SPECIFIED QUOTA
                        03     15    099D          BLEQ    10$                       ; USE QUOTA
                   50   51     D0    099F          MOVL    R1,R0                     ; USE MAXIMUM WORKING SET COUNT
          51     3C A6         3C    09A2  10$:    MOVZWL  PQB$L_WSEXTENT(R6),R1     ; GET MAXIMUM PAGES FOR WORKING SET
          52     30 A6         3C    09A6          MOVZWL  PQB$L_WSQUOTA(R6),R2      ; GET MAXIMUM QUOTA FOR WORKING SET
          53     34 A6         3C    09AA          MOVZWL  PQB$L_WSDEFAULT(R6),R3    ; GET DESIRED DEFAULT
                   52   51     D1    09AE          CMPL    R1,R2                     ; EXTENT MUST BE BIGGER THAN QUOTA
                        03     18    09B1          BGEQ    20$                       ; YES, USE IT AS IS
                   51   52     D0    09B3          MOVL    R2,R1                     ; FORCE TO QUOTA (EXTENT MAY BE 0)
                   50   51     D1    09B6  20$:    CMPL    R1,R0                     ; EXTENT MUST BE LESS THAN MAX PAGES
                        03     15    09B9          BLEQ    30$                       ; BRANCH IF OK AS IS
                   51   50     D0    09BB          MOVL    R0,R1                     ; SET EXTENT TO MAX MEMORY
                   51   52     D1    09BE  30$:    CMPL    R2,R1                     ; QUOTA MUST BE LESS THAN EXTENT
                        03     15    09C1          BLEQ    40$                       ; BRANCH IF OK AS IS
                   52   51     D0    09C3          MOVL    R1,R2                     ; SET QUOTA TO EXTENT
                   52   53     D1    09C6  40$:    CMPL    R3,R2                     ; DEFAULT MUST BE LESS THAN QUOTA
                        03     15    09C9          BLEQ    50$                       ; BRANCH IF OK AS IS
                   53   52     D0    09CB          MOVL    R2,R3                     ; SET DEFAULT TO QUOTA
          50     08 A5   01    A3    09CE  50$:    SUBW3   #1,PHD$W_WSLIST(R5),R0    ; GET BASE OFFSET TO WORKING SET LIST
                   51   50     A0    09D3          ADDW    R0,R1                     ; GET EXTENT
                16 A5   51     B0    09D6          MOVW    R1,PHD$W_WSEXTENT(R5)     ; SET EXTENT
```

```
                                                M 10
                14 A5    51    B0  09DA    866          MOVW    R1,PHD$W_WSAUTHEXT(R5)  ; SET AUTHORIZED EXTENT
                   52    50    A0  09DE    867          ADDW    R0,R2                   ; GET QUOTA
                18 A5    52    B0  09E1    868          MOVW    R2,PHD$W_WSQUOTA(R5)    ; QUOTA VALUE
                0A A5    52    B0  09E5    869          MOVW    R2,PHD$W_WSAUTH(R5)     ; AUTHORIZED VALUE
             1A A5       53    50  A1  09E9    870          ADDW3   R0,R3,PHD$W_DFWSCNT(R5) ; SAVE DEFAULT WORKING SET SIZE
                                       09EE    871
                                       09EE    872 ; THE AUTHPRI CELL EXISTS IN TWO PLACES. THE $SETPRI SYSTEM SERVICE USES
                                       09EE    873 ; THE PCB CELL BUT THE PHD CELL MUST EXIST FOREVER BECAUSE THAT IS WHERE
                                       09EE    874 ; THE JPI ITEM CODE BELIEVES THAT AUTHPRI IS LOCATED.
                                       09EE    875
              2B A4      2F A4    90  09EE    876          MOVB    PCB$B_PRIB(R4),PCB$B_AUTHPRI(R4)    ; SET INITIAL PROCESS PRIORITY
            010C C5      2F A4    90  09F3    877          MOVB    PCB$B_PRIB(R4),PHD$B_AUTHPRI(R5)    ; ... IN BOTH PCB AND PHD
                6C B4      66    7D  09F9    878          MOVQ    PQB$Q_PRVMSK(R6),@PCB$L_PHD(R4)     ; SET PRIVILEGES FOR PROCESS
           00000000'9F     66    7D  09FD    879          MOVQ    PQB$Q_PRVMSK(R6),@#CTL$GQ_PROCPRIV  ; BOTH PERMANENT AND CURRENT
              00E0 C5      66    7D  0A04    880          MOVQ    PQB$Q_PRVMSK(R6),PHD$Q_AUTHPRIV(R5) ; AND AUTHORIZED MASKS
         00000000'9F    46 A6    90  0A09    881          MOVB    PQB$B_MSGMASK(R6),@#CTL$GB_MSGMASK   ; GET DEFAULT MESSAGE FLAGS
     00000000'9F    00000000'EF    7D  0A11    882          MOVQ    EXE$GQ_SYSTIME,@#CTL$GQ_LOGIN        ; SAVE LOGIN TIME
                   7E       54    7D  0A1C    883          MOVQ    R4,-(SP)                             ; SAVE PCB AND PHD POINTERS
                                       0A1F    884
                                       0A1F    885 ; MOVE MINIMUM AND MAXIMUM AUTHORIZED SECURITY CLEARANCE RECORDS INTO THE PHD.
                                       0A1F    886 ; THE FOLLOWING ASSUME STATEMENTS GUARANTEE THAT WE CAN SAFELY PERFORM THIS
                                       0A1F    887 ; WITH A SINGLE MOVC3 INSTRUCTION.
                                       0A1F    888
                                       0A1F    889          ASSUME PQB$S_MIN_CLASS EQ PHD$S_MIN_CLASS
                                       0A1F    890          ASSUME PQB$S_MAX_CLASS EQ PHD$S_MAX_CLASS
                                       0A1F    891          ASSUME PQB$R_MAX_CLASS EQ <PQB$R_MIN_CLASS + PQB$S_MIN_CLASS>
                                       0A1F    892          ASSUME PHD$R_MAX_CLASS EQ <PHD$R_MIN_CLASS + PHD$S_MIN_CLASS>
                                       0A1F    893
                   28       28    0A1F    894          MOVC3   #<PQB$S_MIN_CLASS+PQB$S_MAX_CLASS>,-
                50 A6               0A21    895                  PQB$R_MIN_CLASS(R6),-
              0114 C5               0A23    896                  PHD$R_MIN_CLASS(R5)
                                       0A26    897
                                       0A26    898 ; INITIALIZE LISTHEADS FOR DOUBLY LINKED LISTS USED BY IMAGE ACTIVATOR
                                       0A26    899
50         00000000'9F       9E    0A26    900          MOVAB   @#IAC$GL_IMAGE_LIST,R0  ; LIST OF ACTIVATED IMAGES
              60    50       D0    0A2D    901          MOVL    R0,(R0)                 ; INITIALIZE FLINK
            04 A0   50       D0    0A30    902          MOVL    R0,4(R0)                ; ... AND BLINK
                                       0A34    903
50         00000000'9F       9E    0A34    904          MOVAB   @#IAC$GL_WORK_LIST,R0   ; LIST OF WORK IN PROGRESS
              60    50       D0    0A3B    905          MOVL    R0,(R0)                 ; INITIALIZE FLINK
            04 A0   50       D0    0A3E    906          MOVL    R0,4(R0)                ; ... AND BLINK
                                       0A42    907
50         00000000'9F       9E    0A42    908          MOVAB   @#IAC$GL_ICBFL,R0       ; ADDRESS OF ICB LOOKASIDE LIST
              60    50       D0    0A49    909          MOVL    R0,(R0)                 ; INITIALIZE FLINK
            04 A0   50       D0    0A4C    910          MOVL    R0,4(R0)                ; ... AND BLINK
                                       0A50    911
                                       0A50    912 ;
                                       0A50    913 ; CREATE THE PAGES FOR THE CCB TABLE, PROCESS ALLOCATION REGION, AND DEFAULT
                                       0A50    914 ;       IMAGE I/O SEGMENT
                                       0A50    915 ;
                                       0A50    916
53         00000000'GF       3C    0A50    917          MOVZWL  G^SGN$GW_PCHANCNT,R3    ; PICK UP SYSGEN PARAM FOR # CHANS
                   53       D6    0A57    918          INCL    R3                      ; ALLOW FOR WASTED CCB
              53    10       C4    0A59    919          MULL    #CCB$C_LENGTH,R3        ; CONVERT TO # BYTES
53         000001FF 8F       C0    0A5C    920          ADDL    #511,R3                 ; ROUND UP TO EVEN PAGES
53         000001FF 8F       CA    0A63    921          BICL    #511,R3
54         00000000'GF       3C    0A6A    922          MOVZWL  G^SGN$GW_CTLPAGES,R4    ; GET # PAGES FOR PROCESS ALL REGION
```

```
                54    54    09    78   0A71    923          ASHL    #9,R4,R4                ; CONVERT TO # BYTES
                      53    54    C0   0A75    924          ADDL    R4,R3                   ; GET TOTAL # BYTES NEEDED SO FAR
            57   00000000'GF    3C   0A78    925          MOVZWL  G^SGN$GW_PIOPAGES,R7    ; GET # PAGES FOR PIO SEGMENT
                57    57    09    78   0A7F    926          ASHL    #9,R7,R7                ; CONVERT TO NUMBER OF BYTES
                      53    57    C0   0A83    927          ADDL    R7,R3                   ; GET TOTAL # BYTES NEEDED
            58   00000000'GF    3C   0A86    928          MOVZWL  G^SGN$GW_IMGIOCNT,R8    ; GET # PAGES FOR IIO SEGMENT
                58    58    09    78   0A8D    929          ASHL    #9,R8,R8                ; CONVERT TO NUMBER OF BYTES
                      53    58    C0   0A91    930          ADDL    R8,R3                   ; GET TOTAL # BYTES NEEDED
            55   00000000'EF    DE   0A94    931          MOVAL   CTL$GL_CTLBASVA,R5      ; GET POINTER TO 'TOP' OF P1
                7E    65    01    C3   0A9B    932          SUBL3   #1,(R5),-(SP)           ; 'LAST' PAGE IN P1
                7E    65    53    C3   0A9F    933          SUBL3   R3,(R5),-(SP)           ; 'TOP' OF CREATED REGION
                      52    7E    7E   0AA3    934          MOVAQ   -(SP),R2                ; SPACE FOR RETADR
                 00000D00    8F    DD   0AA6    935          PUSHL   #PSL$C_KERNEL+<PRT$C_UREW@8> ; ACCESS MODE AND PROTECTION
                      52    DD   0AAC    936          PUSHL   R2                      ; RETADR ARRAY
                      08    A2    9F   0AAE    937          PUSHAB  8(R2)                   ; INADR ARRAY
                      03    DD   0AB1    938          PUSHL   #3                      ; ARGUMENT COUNT
                50    5E    D0   0AB3    939          MOVL    SP,R0
                                 0AB6    940          $CMKRNL_S -                     ; CALL INTERNAL ENTRY POINT FOR $CRETVA
                                 0AB6    941                  ROUTIN = G^MMG$CRETVA,-
                                 0AB6    942                  ARGLST = (R0)
                   0D 50    E9   0AC5    943          BLBC    R0,VABUG                ; GET OUT ON ERROR
                08 A2    62    D1   0AC8    944          CMPL    (R2),8(R2)              ; DID WE GET FULL REQUEST?
                      07    12   0ACC    945          BNEQ    VABUG                   ; NO, ERROR OUT
             0C A2    04 A2    D1   0ACE    946          CMPL    4(R2),12(R2)            ; MAKE DOUBLY SURE
                      0D    13   0AD3    947          BEQL    DIVR                    ; NO, ERROR OUT
                                 0AD5    948
00000000'FF    66    0E   0AD5    949 VABUG:   INSQUE  (R6),@EXE$GL_PQBBL     ; DEALLOCATE PQB TO LOOKASIDE LIST
                                 0ADC    950          SETIPL  #0                      ; ALLOW PROCESS TO BE DELETED
                0391    31   0ADF    951          BRW     EXE$EXIT_IMAGE          ; DELETE THE PROCESS
                                 0AE2    952 ;
                                 0AE2    953 ; NOW DIVIDE THE CREATED SPACE INTO FOUR AREAS
                                 0AE2    954 ;
                                 0AE2    955
      00000004'EF    62    D0   0AE2    956 DIVR:    MOVL    (R2),PIO$GQ_IIODEFAULT+4                ; DEFAULT IMAGE I/O AREA
      00000000'EF    58    D0   0AE9    957          MOVL    R8,PIO$GQ_IIODEFAULT                    ; SIZE
      50   00000004'9F    DE   0AF0    958          MOVAL   @#PIO$GW_PIOIMPA+IMP$L_IOSEGADDR,R0     ; GET POINTER ADDRESS
                58    62    C0   0AF7    959          ADDL    (R2),R8                 ; START OF REMAINING SPACE
                80    58    D0   0AFA    960          MOVL    R8,(R0)+                ; SET UP THE PIO SEG ADDR
                60    57    D0   0AFD    961          MOVL    R7,(R0)                 ; SET LENGTH
             50    57    58    C1   0B00    962          ADDL3   R8,R7,R0                ; GET POINTER TO FREE SPACE
      00000000'9F    50    D0   0B04    963          MOVL    R0,@#CTL$GQ_ALLOCREG    ; SET UP PROCESS ALLOCATION
                      80    D4   0B08    964          CLRL    (R0)+                   ; NULL FORWARD POINTER
                60    54    D0   0B0D    965          MOVL    R4,(R0)                 ; SET SIZE OF REGION
          50   00000000'GF    3C   0B10    966          MOVZWL  G^SGN$GW_CTLIMGLIM,R0           ; GET IMAGE LIMIT
   00000000'9F    50    09    78   0B17    967          ASHL    #9,R0,@#CTL$GL_PRCALLCNT        ; CONVERT TO # BYTES
00000000'9F    04 A2    0F    C3   0B1F    968          SUBL3   #CCB$C_LENGTH-1,4(R2),@#CTL$GL_CCBBASE  ; STORE BASE OF CHANNEL TABL
                                 0B28    969
00000000'9F   00000000'GF    3C   0B28    970          MOVZWL  G^SGN$GW_PCHANCNT,@#CTL$GW_NMIOCH ; SET NUMBER OF CHANNELS
                                 0B33    971
                                 0B33    972 ;
                                 0B33    973 ; NOTE(!!!!):    THE ABOVE ASSIGNMENT MUST BE DONE AT THE VERY END OF THIS
                                 0B33    974 ;                SECTION OF CODE, AS THE CELL NMIOCH BEING NON-ZERO IS AN
                                 0B33    975 ;                INDICATOR TO IOC$FFCHAN THAT THERE IS ACTUALLY A REAL
                                 0B33    976 ;                CHANNEL TABLE TO LOOK AT.
                                 0B33    977 ;
                                 0B33    978
                65    62    D0   0B33    979          MOVL    (R2),(R5)               ; UPDATE BASE OF VA IN CTL REGION
```

```
                5E    20 AE    DE  0B36   980          MOVAL   32(SP),SP               ; POP $CRETVA ARGS
                                   0B3A   981
                                   0B3A   982 ;
                                   0B3A   983 ; ALLOCATE P1 SPACE FOR THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE, FOR
                                   0B3A   984 ; THE PROCESS DIRECTORY LOGICAL NAME TABLE, AND FOR ALL PROCESS-PRIVATE
                                   0B3A   985 ; LOGICAL NAMES AND LOGICAL NAME TABLES THAT NEED TO BE SETUP AT PROCESS
                                   0B3A   986 ; CREATION TIME. INITIALLY FORMAT THE LOGICAL NAMES AND LOGICAL NAME TABLES
                                   0B3A   987 ; BY COPYING THEIR TEMPLATES ONTO THE P1 SPACE ALLOCATED FOR THEM, AND THEN
                                   0B3A   988 ; FORMAT THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE.
                                   0B3A   989 ;
                                   0B3A   990
          51    00000000'GF    D0  0B3A   991          MOVL    G^LNM$GL_HTBLSIZP,R1    ; RETRIEVE NUMBER OF HASH TABLE ENTRIES
        51    0000000C 9F41    DE  0B41   992          MOVAL   @#LNMHSH$K_BUCKET[R1],R1; MULTIPLY BY 4 AND ADD OVERHEAD
                 57    51    D0  0B49   993          MOVL    R1,R7                   ; SAVE SIZE OF HASH TABLE
          51    000006F0 8F    C0  0B4C   994          ADDL2   #P1_ALLOC_SIZE,R1       ; ADD IN SIZE OF LOGICAL NAME BLOCKS
                00000000'GF    16  0B53   995          JSB     G^EXE$ALOP1PROC         ; ALLOCATE TOTAL AMOUNT OF SPACE NEEDED
                    58    52    D0  0B59   996          MOVL    R2,R8                   ; SAVE ADDRESS OF ALLOCATED SPACE
                                   0B5C   997
                 06F0 8F    2C  0B5C   998          MOVC5   #P1_ALLOC_SIZE,-        ; COPY TEMPLATE FOR ALL LOGICAL NAMES
      62    51    00    F505 CF       0B60   999                  PROC_DIR,#0,R1,(R2)     ; AND ZERO PROCESS-PRIVATE HASH TABLE
      53    58    000006F0 8F    C1  0B66  1000          ADDL3   #P1_ALLOC_SIZE,R8,R3    ; COMPUTE HASH TABLE ADDRESS
                                   0B6E  1001
          00000000'9F    53    D0  0B6E  1002          MOVL    R3,@#CTL$GL_LNMHASH     ; STORE ADDRESS OF HASH TABLE AWAY
      50  00000000'GF    01    C3  0B75  1003          SUBL3   #1,G^LNM$GL_HTBLSIZP,R0 ; CALCULATE UPPER BOUND OF HASH INDEX
                    63    50    D2  0B7D  1004          MCOML   R0,LNMHSH$L_MASK(R3)    ; STORE HASH INDEX MASK IN HASH TABLE
                 08 A3    57    B0  0B80  1005          MOVW    R7,LNMHSH$W_SIZE(R3)    ; STORE HASH TABLE SIZE IN HEADER
                          38    90  0B84  1006          MOVB    #DYN$C_RSHT,-           ; STORE HASH TABLE STRUCTURE TYPE IN
                       0A A3       0B86  1007                  LNMHSH$B_TYPE(R3)       ; HASH TABLE HEADER
                                   0B88  1008
                                   0B88  1009 ;
                                   0B88  1010 ; FIXUP THE PROCESS DIRECTORY LOGICAL NAME TABLE, LNM$PROCESS_DIRECTORY, AND
                                   0B88  1011 ; LINK IT INTO THE APPROPRIATE HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME
                                   0B88  1012 ; HASH TABLE.
                                   0B88  1013 ;
                                   0B88  1014
                 57    2C A8    9E  0B88  1015          MOVAB   PROC_DIR_LNMTH(R8),R7   ; COMPUTE DIRECTORY'S TABLE HEADER ADDR
                 01 A7    53    D0  0B8C  1016          MOVL    R3,LNMTH$L_HASH(R7)     ; STORE HASH TABLE ADDR IN TABLE HEADER
                 0C A8    57    D0  0B90  1017          MOVL    R7,LNMB$L_TABLE(R8)     ; DIRECTORIES ALWAYS CONTAIN THEMSELVES
                 09 A7    58    D0  0B94  1018          MOVL    R8,LNMTH$L_NAME(R7)     ; STORE LNMB ADDRESS IN TABLE HEADER
                 19 A7    57    D0  0B98  1019          MOVL    R7,LNMTH$L_QTABLE(R7)   ; DIRECTORIES ARE QUOTA HOLDERS
      00000000'9F    58    D0  0B9C  1020          MOVL    R8,@#CTL$GL_LNMDIRECT   ; STORE ADDR OF PROCESS DIRECTORY AWAY
                                   0BA3  1021
                 51    11 A8    9E  0BA3  1022          MOVAB   LNMB$T_NAME(R8),R1      ; RETRIEVE THE SIZE AND ADDRESS OF THE
                    50    81    9A  0BA7  1023          MOVZBL  (R1)+,R0                ; PROCESS DIRECTORY'S NAME
                00000000'GF    16  0BAA  1024          JSB     G^LNM$HASH              ; HASH THE DIRECTORY NAME
                                   0BB0  1025
                    50    63    CA  0BB0  1026          BICL2   LNMHSH$L_MASK(R3),R0    ; MODIFY THE HASH INDEX TO BE IN RANGE
              0C A340    58    D0  0BB3  1027          MOVL    R8,LNMHSH$C_BUCKET(R3)[R0] ; INSERT THE PROCESS DIRECTORY TABLE
           04 A8    0C A340    DE  0BB8  1028          MOVAL   LNMHSH$C_BUCKET(R3)[R0],-  ; INTO THE APPROPRIATE HASH BUCKET
                                   0BBE  1029                  LNMB$L_BLINK(R8)
                                   0BBE  1030
                                   0BBE  1031 ;
                                   0BBE  1032 ; FIXUP THE PROCESS LOGICAL NAME TABLE, LNM$PROCESS_TABLE, AND INSERT IT INTO
                                   0BBE  1033 ; THE APPROPRIATE HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE.
                                   0BBE  1034 ;
                                   0BBE  1035
                 51    58 A8    9E  0BBE  1036          MOVAB   PROC_TABLE(R8),R1       ; COMPUTE ADDRESS OF LNM$PROCESS_TABLE
```

```
                59   0080 C8    9E  0BC2  1037          MOVAB   PROC_TABLE_LNMTH(R8),R9 ; COMPUTE AND SAVE ADDRESS OF LNMTH
                  0C A1    57    D0  0BC7  1038          MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
                  01 A9    53    D0  0BCB  1039          MOVL    R3,LNMTH$L_HASH(R9)     ; STORE HASH TABLE ADDR IN TABLE HEADER
                  09 A9    51    D0  0BCF  1040          MOVL    R1,LNMTH$L_NAME(R9)     ; STORE LNMB ADDRESS IN TABLE HEADER
                  0D A9    57    D0  0BD3  1041          MOVL    R7,LNMTH$L_PARENT(R9)   ; LNM$PROCESS_DIRECTORY IS PARENT AND
                  19 A9    57    D0  0BD7  1042          MOVL    R7,LNMTH$L_QTABLE(R9)   ; QUOTA HOLDER OF LNM$PROCESS_TABLE
                           52    D4  0BDB  1043          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
                 00000000'GF    16  0BDD  1044          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$PROCESS_TABLE
                                    0BE3  1045
                                    0BE3  1046 ;
                                    0BE3  1047 ; FIXUP LNM$PROCESS LNM$GROUP AND LNM$JOB AND INSERT THEM INTO THE APPROPRIATE
                                    0BE3  1048 ; HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE. LNM$GROUP AND
                                    0BE3  1049 ; LNM$JOB REQUIRE THAT THEIR EQUIVALENCE STRINGS BE CONSTRUCTED FROM THE UIC
                                    0BE3  1050 ; AND JIB ADDRESS OF THE NEW PROCESS RESPECTIVELY.
                                    0BE3  1051 ;
                                    0BE3  1052
                   54    6E    D0  0BE3  1053          MOVL    (SP),R4                 ; RESTORE PCB ADDRESS TO R4
                51   00A8 C8    9E  0BE6  1054          MOVAB   PROCESS(R8),R1          ; COMPUTE ADDRESS OF LNM$PROCESS
                  0C A1    57    D0  0BEB  1055          MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
                           52    D4  0BEF  1056          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
                 00000000'GF    16  0BF1  1057          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$PROCESS
                                    0BF7  1058
                51   0118 C8    9E  0BF7  1059          MOVAB   JOB(R8),R1              ; COMPUTE ADDRESS OF LNM$JOB
                  0C A1    57    D0  0BFC  1060          MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
                  53   2E A1    9E  0C00  1061          MOVAB   JOB_XEND_SIZE-1(R1),R3  ; COMPUTE ADDRESS OF LAST LNMX
                           52    D4  0C04  1062          CLRL    R2                      ; CLEAR INDEX REGISTER
        50 0080 C4    04    52    EF  0C06  1063 60$:     EXTZV   R2,#4,PCB$L_JIB(R4),R0  ; EXTRACT OUT HEX BITS AND TRANSFORM
              73  F40A CF40    90  0C0D  1064          MOVB    CHARS[R0],-(R3)         ; THEM INTO THEIR ASCII EQUIVALENT
           FFED 52    04    1F    9D  0C13  1065          ACBB    #31,#4,R2,60$           ; CONTINUE FROM RIGHT -> LEFT UNTIL DONE
                   5A    53    D0  0C19  1066          MOVL    R3,R10                  ; SAVE THE ADDRESS OF THE ASCII JIB ADDR
                           52    D4  0C1C  1067          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
                 00000000'GF    16  0C1E  1068          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$JOB
                                    0C24  1069
                51   00E0 C8    9E  0C24  1070          MOVAB   GROUP(R8),R1            ; COMPUTE ADDRESS OF LNM$GROUP
                  0C A1    57    D0  0C29  1071          MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
                  53   30 A1    9E  0C2D  1072          MOVAB   GROUP_XEND_SIZE-1(R1),R3; COMPUTE ADDRESS OF LAST LNMX
                           52    D4  0C31  1073          CLRL    R2                      ; CLEAR INDEX REGISTER
        50 00BE C4    03    52    EF  0C33  1074 61$:     EXTZV   R2,#3,PCB$W_GRP(R4),R0  ; EXTRACT OUT OCTAL BITS AND TRANSFORM
              73  F3DD CF40    90  0C3A  1075          MOVB    CHARS[R0],-(R3)         ; THEM INTO THEIR ASCII EQUIVALENT
           FFED 52    03    0E    9D  0C40  1076          ACBB    #14,#3,R2,61$           ; CONTINUE FROM RIGHT -> LEFT UNTIL DONE
                   73    30    90  0C46  1077          MOVB    #^A/0/,-(R3)            ; ASSUME HIGH ORDER BIT IS 0
                03 00BE C4    0F    E1  0C49  1078          BBC     #15,PCB$W_GRP(R4),62$   ; IF SO THEN GO INSERT LNM$GROUP
                   63    31    90  0C4F  1079          MOVB    #^A/1/,(R3)             ; OTHERWISE INSERT A 1
                   5B    53    D0  0C52  1080 62$:     MOVL    R3,R11                  ; SAVE THE ADDRESS OF THE ASCII GROUP
                           52    D4  0C55  1081          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
                 00000000'GF    16  0C57  1082          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$GROUP
                                    0C5D  1083
                                    0C5D  1084 ;
                                    0C5D  1085 ; FIXUP THE LOGICAL NAME BLOCKS FOR SYS$INPUT, TT, SYS$OUTPUT, SYS$ERROR, AND
                                    0C5D  1086 ; SYS$DISK, AND INSERT THEM INTO THE APPROPRIATE HASH BUCKET OF THE
                                    0C5D  1087 ; PROCESS-PRIVATE LOGICAL NAME HASH TABLE.
                                    0C5D  1088 ;
                                    0C5D  1089
                                    0C5D  1090          CRELNM  -                       ; FIXUP AND INSERT SYS$INPUT
                                    0C5D  1091                  PQB$T_INPUT,-
                                    0C5D  1092                  PQB$L_INPUT_ATT,-
                                    0C5D  1093                  SYS$INPUT_LNMX,-
```

```
                                                     D 11
                     0C5D  1094                      SYS$INPUT
                     0C68  1095
                     0C68  1096              CRELNM  -                               ; FIXUP AND INSERT SYS$OUTPUT
                     0C68  1097                      PQB$T_OUTPUT,-
                     0C68  1098                      PQB$L_OUTPUT_ATT,-
                     0C68  1099                      SYS$OUTPUT_LNMX,-
                     0C68  1100                      SYS$OUTPUT
                     0C73  1101
                     0C73  1102              CRELNM  -                               ; FIXUP AND INSERT SYS$ERROR
                     0C73  1103                      PQB$T_ERROR,-
                     0C73  1104                      PQB$L_ERROR_ATT,-
                     0C73  1105                      SYS$ERROR_LNMX,-
                     0C73  1106                      SYS$ERROR
                     0C7E  1107
                     0C7E  1108              CRELNM  -                               ; FIXUP AND INSERT TT
                     0C7E  1109                      PQB$T_INPUT,-
                     0C7E  1110                      PQB$L_INPUT_ATT,-
                     0C7E  1111                      TT_LNMX,-
                     0C7E  1112                      TT
                     0C89  1113
                     0C89  1114              CRELNM  -                               ; FIXUP AND INSERT SYS$DISK
                     0C89  1115                      PQB$T_DISK,-
                     0C89  1116                      PQB$L_DISK_ATT,-
                     0C89  1117                      SYS$DISK_LNMX,-
                     0C89  1118                      SYS$DISK
                     0C94  1119
                     0C94  1120 ;
                     0C94  1121 ; IF THE PROCESS BEING CREATED IS NOT A SUB-PROCESS THEN CREATE THE JOB AND
                     0C94  1122 ; GROUP LOGICAL NAME TABLES.
                     0C94  1123 ;
                     0C94  1124
          54   8E   D0  0C94  1125              MOVL    (SP)+,R4                        ; RETRIEVE PCB ADDRESS
            1C A4   D5  0C97  1126              TSTL    PCB$L_OWNER(R4)                 ; SUB-PROCESS?
              0D    12  0C9A  1127              BNEQ    65$                             ; IF YES THEN SKIP TABLE CREATION
      57   40 A6   D0  0C9C  1128              MOVL    PQB$L_JTQUOTA(R6),R7            ; RETRIEVE JOB TABLE CREATION QUOTA
            04EA   30  0CA0  1129              BSBW    EXE$CRE_JGTABLE                 ; CREATE JOB AND GROUP TABLES
           03 50   E8  0CA3  1130              BLBS    R0,65$                          ; CONTINUE IF SUCCESS
            FE2C   31  0CA6  1131 64$:          BRW     VABUG                           ; OTHERWISE, TAKE COMMON EXIT PATH
                     0CA9  1132
                     0CA9  1133 ;
                     0CA9  1134 ; ALLOCATE P1 SPACE FOR THE PROCESS-PRIVATE LOGICAL NAME TABLE NAME CACHE
                     0CA9  1135 ;
                     0CA9  1136
51   00000000'GF    08   C5  0CA9  1137 65$:      MULL3   #8,G^LNM$GL_HTBLSIZP,R1 ; ALLOCATE TWICE HASH TABLE SIZE
58   51   00000080 8F   C7  0CB1  1138          DIVL3   #LNMC$K_LENGTH,R1,R8    ; COMPUTE # OF ENTRIES
                 23   13  0CB9  1139          BEQL    67$                             ; IF ANY
          00000000'GF   16  0CBB  1140          JSB     G^EXE$ALOP1PROC                 ; ALLOCATE TOTAL AMOUNT OF SPACE NEEDED
                 E2   50  E9  0CC1  1141          BLBC    R0,64$                          ; IF POSSIBLE
       08 A2   0080 8F   B0  0CC4  1142 66$:      MOVW    #LNMC$K_LENGTH,LNMC$W_SIZE(R2)  ; SET SIZE
                 0C A2   D4  0CCA  1143          CLRL    LNMC$L_TBLADDR(R2)              ; MARK EMPTY
      00000000'9F    62   0E  0CCD  1144          INSQUE  (R2),@#CTL$GQ_LNMTBLCACHE       ; INSERT IN QUEUE
      52   00000080 8F   C0  0CD4  1145          ADDL2   #LNMC$K_LENGTH,R2               ; POINT TO NEXT
                 E6   58  F5  0CDB  1146          SOBGTR  R8,66$                  ; LOOP
                     0CDE  1147
                     0CDE  1148 ;
                     0CDE  1149 ; RESTORE PCB AND PHD ADDRESS, SET IPL TO 0 TO ALLOW FOR PROCESS DELETION
                     0CDE  1150 ; (IF DESIRED), RESET ADDRESS SPACE, AND SET WSLAST.
```

```
                                                E 11
                                    0CDE  1151 ;
                                    0CDE  1152
                         55   8E    D0  0CDE  1153 67$:    MOVL    (SP)+,R5                ; RESTORE PHD ADDRESS
                                    0CE1  1154
                   5C  00000000'9F   DE  0CE1  1155        MOVAL   @#MMG$IMGHDRBUF,AP      ; IMAGE HEADER BUFFER ADDRESS
                       00000000'GF   16  0CE8  1156        JSB     G^MMG$IMGRESET          ; RESET ADDRESS SPACE AND SET WSLAST
                                    0CEE  1157
                                    0CEE  1158 ; THE FOLLOWING MOVC SEQUENCES DESTROY R0 THROUGH R5
                                    0CEE  1159
                       6C   07C8 C6   9A  0CEE  1160 IMGNAM: MOVZBL  PQB$T_IMAGE(R6),(AP)    ; SIZE OF IMAGE NAME STRING
                     04 AC    08 AC   DE  0CF3  1161        MOVAL   8(AP),4(AP)             ; ADDRESS OF IMAGE NAME STRING
                08 AC  07C9 C6   6C   28  0CF8  1162        MOVC3   (AP),PQB$T_IMAGE+1(R6),8(AP) ; MOVE THE NAME STRING
                                    0CFF  1163
                         06C8 C6   95  0CFF  1164        TSTB    PQB$T_DDSTRING(R6)      ; CHECK FOR NULL STRING
                              0C   13  0D03  1165        BEQL    70$                     ; YES, DONT MOVE ANYTHING
                         0100 8F   28  0D05  1166        MOVC3   #PQB$S_DDSTRING,-
               00000000'9F   06C8 C6       0D09  1167                PQB$T_DDSTRING(R6),@#PIO$GT_DDSTRING ; AND DEFAULT DIRECTORY
                                    0D11  1168 70$:                                     ; CONTINUE
                                    0D11  1169
                                    0D11  1170 ; Move CLI and CLI table information to P1 space in one fell swoop:
                                    0D11  1171 ;       PQB$T_CLI_NAME    -> CTL$GT_CLINAME
                                    0D11  1172 ;       PQB$T_CLI_TABLE   -> CTL$GT_TABLENAME
                                    0D11  1173 ;       PQB$T_SPAWN_CLI   -> CTL$GT_SPAWNCLI
                                    0D11  1174 ;       PQB$T_SPAWN_TABLE -> CTL$GT_SPAWNTABLE
                                    0D11  1175
                                    0D11  1176        ASSUME PQB$T_CLI_TABLE   EQ <PQB$T_CLI_NAME  + PQB$S_CLI_NAME>
                                    0D11  1177        ASSUME PQB$T_SPAWN_CLI   EQ <PQB$T_CLI_TABLE + PQB$S_CLI_TABLE>
                                    0D11  1178        ASSUME PQB$T_SPAWN_TABLE EQ <PQB$T_SPAWN_CLI + PQB$S_SPAWN_CLI>
                                    0D11  1179
                              28    0D11  1180        MOVC3   #<PQB$S_CLI_NAME+-
                                    0D12  1181                  PQB$S_CLI_TABLE+-
                                    0D12  1182                  PQB$S_SPAWN_CLI+-
                                    0D12  1183                  PQB$S_SPAWN_TABLE>,-
                 0088 C6   0240 8F      0D12  1184                PQB$T_CLI_NAME(R6),@#CTL$GT_CLINAME
                       00000000'9F      0D18
                                    0D1D  1185
                                    0D1D  1186 ; STORE EVERYTHING ELSE OF INTEREST BEFORE WE GET RID OF THE PQB
                                    0D1D  1187
           00000000'9F   4C A6   D0  0D1D  1188        MOVL    PQB$L_CREPRC_FLAGS(R6),@#CTL$GL_CREPRC_FLAGS
           00000000'9F   48 A6   D0  0D25  1189        MOVL    PQB$L_UAF_FLAGS(R6),@#CTL$GL_UAF_FLAGS
                                    0D2D  1190
                                    0D2D  1191 ; ****************************** TEMP ******************************
                                    0D2D  1192 ;
                                    0D2D  1193 ; THE FOLLOWING CODE WILL BE REMOVED WHEN WE DECIDE WHAT TO DO WITH THE
                                    0D2D  1194 ; ACCOUNT AND USERNAME FIELDS IN THE P1 POINTER PAGE.
                                    0D2D  1195 ;
                                    0D2D  1196        assume jib$t_account eq <jib$t_username + jib$s_username>
                                    0D2D  1197
              50   00000000'GF   D0  0D2D  1198        movl    g^ctl$gl_pcb,r0         ; get pcb address ...
                   50   0080 C0   D0  0D34  1199        movl    pcb$l_jib(r0),r0        ; so that we can get jib address
                             14   28  0D39  1200        movc3   #<jib$s_username + jib$s_account>,-
                          0C A0       0D3B  1201                jib$t_username(r0),-    ; move username and account
                       00000000'9F      0D3D  1202                @#ctl$t_username        ; in one instruction
                                    0D42  1203 ;
                                    0D42  1204 ; ****************************** END TEMP ******************************
                                    0D42  1205
           00000000'FF   66   0E    0D42  1206        INSQUE  (R6),@EXE$GL_PQBBL      ; DEALLOCATE PQB TO LOOKASIDE LIST
```

```
                                 0D49  1207          SETIPL  #0                      ; DROP IPL AND ALLOW PROCESS DELETION
                                 0D4C  1208
                                 0D4C  1209 ;
                                 0D4C  1210 ; INITIALIZE FIXUP VECTOR LINKED LISTS TO CONTAIN A SINGLE DUMMY ENTRY
                                 0D4C  1211 ;
                                 0D4C  1212
00000000'9F   00000000'9F   DE   0D4C  1213          MOVAL   a#CTL$GL_IAFPERM,a#CTL$GL_IAFLINK
00000000'9F   00000000'9F   DE   0D57  1214          MOVAL   a#CTL$GL_IAFPERM,a#CTL$GL_IAFLAST
                                 0D62  1215
                                 0D62  1216 ;
                                 0D62  1217 ; INITIALIZE ARRAYS THAT DETERMINE HOW PRIVILEGED VECTORS ARE RESET
                                 0D62  1218 ;
        50    00000000'9F   3E   0D62  1219          MOVAW   a#IAC$AW_VECRESET,R0    ; STORE RESET ARRAY ADDRESS
              80    04      B0   0D69  1220          MOVW    #4,(R0)+                ; KERNEL VECTOR
              80    04      B0   0D6C  1221          MOVW    #4,(R0)+                ; EXEC VECTOR
              80    04      B0   0D6F  1222          MOVW    #4,(R0)+                ; RUNDOWN VECTOR
              80    04      B0   0D72  1223          MOVW    #4,(R0)+                ; MESSAGE VECTOR
                                 0D75  1224
        50    00000000'9F   3E   0D75  1225          MOVAW   a#IAC$AW_VECSET,R0      ; STORE START ARRAY ADDRESS
              80    04      B0   0D7C  1226          MOVW    #4,(R0)+                ; KERNEL VECTOR
              80    04      B0   0D7F  1227          MOVW    #4,(R0)+                ; EXEC VECTOR
              80    04      B0   0D82  1228          MOVW    #4,(R0)+                ; RUNDOWN VECTOR
              80    04      B0   0D85  1229          MOVW    #4,(R0)+                ; MESSAGE VECTOR
                                 0D88  1230
                                 0D88  1231 EXE$PROCIMGACT::                       ; ENTRY POINT FOR STAND-ALONE SYSGEN
        54    00000000'9F   D0   0D88  1232          MOVL    a#CTL$GL_PCB,R4         ; GET PCB ADDRESS
     58   24 A4    01      13   EF   0D8F  1233          EXTZV   #PCB$V_HIBER,#1,PCB$L_STS(R4),R8; SAVE HIBERNATE CONTROL
          00000F4C'EF    00   FB   0D95  1234          CALLS   #0, XQPMERGE            ; MERGE XQP INTO PROCESS
       03 00000000'GF    00'  E1   0D9C  1235          BBC     S^#EXE$V_INIT, G^EXE$GL_FLAGS, 72$ ; DON'T MERGE IF NOT INIT
                   5A    50   E9   0DA4  1236          BLBC    R0, 75$                 ; EXIT IF MERGE FAILS
              7E    05    16   9C   0DA7  1237 72$:     ROTL    #PSL$V_PRVMOD,#<PSL$C_EXEC@2+PSL$C_EXEC>,-(SP) ; FORM EXEC PSL
                          6C   10   0DAB  1238          BSBB    80$                     ; CHANGE MODE TO EXECUTIVE
                                 0DAD  1239
                                 0DAD  1240 ; ********** THE FOLLOWING CODE EXECUTES IN EXEC MODE **********
                                 0DAD  1241
                    52    6C   9A   0DAD  1242          MOVZBL  (AP),R2                 ; GET ADR OF FILENAME STRING DESC
                    52    03   C0   0DB0  1243          ADDL    #3,R2                   ; ROUND THE NUMBER OF BYTES IN
                    52    03   CA   0DB3  1244          BICL    #3,R2                   ; THE NAME UP TO A LONGWORD BOUNDRY
                    5E    52   C2   0DB6  1245          SUBL    R2,SP                   ; ALLOCATE SPACE FOR NAME ON STACK
                          6E   9F   0DB9  1246          PUSHAB  (SP)                    ; BUILD STRING DESCRIPTOR FOR
                    7E    6C   9A   0DBB  1247          MOVZBL  (AP),-(SP)              ; FILENAME ON THE STACK
                    51    5E   D0   0DBE  1248          MOVL    SP,R1                   ; GET ADR OF STRING DESCRIPTOR
                          3E   BB   0DC1  1249          PUSHR   #^M<R1,R2,R3,R4,R5>     ; SAVE REGISTERS
     04 B1    04 BC    52   28   0DC3  1250          MOVC3   R2,@4(AP),@4(R1)        ; MOVE FILENAME TO STACK
                          3E   BA   0DC9  1251          POPR    #^M<R1,R2,R3,R4,R5>     ; RESTORE REGISTERS
                                 0DCB  1252          $IMGACT_S -                     ; ACTIVATE THE IMAGE
                                 0DCB  1253                  NAME  =(AP),-           ; DESCRIPTOR FOR IMAGE NAME
                                 0DCB  1254                  DFLNAM=DEFDESC,-        ; DEFAULT NAME DESCRIPTOR
                                 0DCB  1255                  HDRBUF=(AP)             ; ADDRESS IF IMAGE HEADER BUFFER
                    52    08   C0   0DE2  1256          ADDL    #8,R2                   ; CALCULATE # OF BYTES ON STACK
                    5E    52   C0   0DE5  1257          ADDL    R2,SP                   ; AND CLEAN THEM OFF
                      16  50   E9   0DE8  1258          BLBC    R0,75$                  ; BRANCH IF IMGACT FAILED
        50    00000000'9F   9E   0DEB  1259          MOVAB   a#PIO$AL_RMSEXH,R0      ; GET ADDRESS OF EXIT HANDLER CONTROL BLOCK
         04 A0    0F25'CF   9E   0DF2  1260          MOVAB   W^EXE$RMSEXH,4(R0)      ; SET ADDRESS OF RMS EXIT HANDLER
                            0DF8  1261          $DCLEXH_S (R0)                  ; DECLARE EXEC MODE EXIT HANDLER
                      63  50   E9   0E01  1262 75$:     BLBC    R0,120$                 ; IF LBC ERROR
00000000'GF   0E6B'CF   9E   0E04  1263          MOVAB   W^EXE$CLI_UTILSRV+2,G^CTL$AL_CLICALBK ; SET CLI CALL BACK ADDRESS
```

```
                7E   0F    16    9C  0E0D  1264          ROTL    #PSL$V_PRVMOD,#<PSL$C_USER@2+PSL$C_USER>,-(SP) ; FORM USER PSL
                           06    10  0E11  1265          BSBB    80$                             ; CHANGE TO USER MODE
                                     0E13  1266
                                     0E13  1267 ; ********** THE FOLLOWING CODE EXECUTES IN USER MODE **********
                                     0E13  1268
                           5D    D4  0E13  1269          CLRL    FP                              ; TERMINATE CALL FRAME CHAIN
                     1A'AF       6C  0E15  1270          CALLG   (AP),B^90$                      ; CREATE TOP FRAME
                                 02  0E19  1271 80$:     REI                                     ; CHANGE TO NEW MODE
                               0000  0E1A  1272 90$:     .WORD   0                               ; ENTRY MASK
                     6D   80'AF    9E  0E1C  1273          MOVAB   B^EXE$CATCH_ALL,(FP)            ; SET EXCEPTION HANDLER ADDRESS
                                     0E20  1274          $SETEXV_S #2,B^EXE$CATCH_ALL            ; DECLARE LAST CHANCE HANDLER
                                     0E30  1275          $IMGFIX_S                               ; PERFORM ADDRESS RELOCATION
                        2D 50    E9  0E37  1276          BLBC    R0,120$                         ; QUIT IF ERROR OCCURS
                           58    DD  0E3A  1277          PUSHL   R8                              ; SAVE HIBERNATE FLAG
                        52    6C  7D  0E3C  1278 100$:    MOVQ    (AP),R2                         ; GET IMAGE HEADER BLOCK DESCRIPTOR
                              7E  D4  0E3F  1279          CLRL    -(SP)                           ; CLEAR COMMAND INTERPRETER FLAGS
                        20 A2    DD  0E41  1280          PUSHL   IHD$L_LNKFLAGS(R2)              ; PUSH LINKER FLAGS
                           7E    52  7D  0E44  1281          MOVQ    R2,-(SP)                        ; THIRD AND FOURTH ARGUMENTS TO PROG
                             69'AF  9F  0E47  1282          PUSHAB  B^EXE$CLI_UTILSRV               ; PUSH ADDRESS OF CLI CALL BACK ROUTINE
                        51   02 A2  3C  0E4A  1283          MOVZWL  IHD$W_ACTIVOFF(R2),R1           ; OFFSET TO TRANSFER VECTOR
                           52    51  C0  0E4E  1284          ADDL    R1,R2                           ; FORM ADDRESS OF START VECTOR
                                 62  DF  0E51  1285          PUSHAL  (R2)                            ; MOVE TO ARGUMENT LIST
                           07 18 AE  E9  0E53  1286          BLBC    24(SP),110$                     ; BR IF NO HIBERNATE
                                     0E57  1287          $HIBER_S                                ; SET, HIBERNATE UNTIL SOME WAKE
                           92    06  FB  0E5E  1288 110$:    CALLS   #6,@(R2)+                       ; CALL IMAGE
                              03 50  E9  0E61  1289          BLBC    R0,120$                         ; EXIT IF NOT SUCCESS
                              D5 6E  E8  0E64  1290          BLBS    (SP),100$                       ; CHECK FOR HIBERNATE AGAIN
                                 0A  11  0E67  1291 120$:    BRB     EXE$EXIT_IMAGE                  ;
                                     0E69  1292
                                     0E69  1293 ;
                                     0E69  1294 ; DUMMY COMMAND INTERPRETER CALL BACK ROUTINE
                                     0E69  1295 ;
                                     0E69  1296
                               0000  0E69  1297          .ENTRY  EXE$CLI_UTILSRV,^M<>
                50   00038822 8F    D0  0E6B  1298          MOVL    #CLI$_INVREQTYP,R0              ; SET INVALID REQUEST TYPE STATUS
                                 04  0E72  1299          RET                                     ;
                                     0E73  1300          .DSABL  LSB
```

```
                        0E73  1303          .SBTTL  EXIT IMAGE AND RUN DOWN FILES
                        0E73  1304 ;+
                        0E73  1305 ;
                        0E73  1306 ; EXE$EXIT_IMAGE - EXIT IMAGE AND RUN DOWN FILES
                        0E73  1307 ;
                        0E73  1308 ; THIS ROUTINE IS JUMPED TO AT THE CONCLUSION OF IMAGE EXECUTION TO RUN DOWN
                        0E73  1309 ; RMS FILES AND TO RETURN THE FINAL IMAGE STATUS.
                        0E73  1310 ;
                        0E73  1311 ; INPUTS:
                        0E73  1312 ;
                        0E73  1313 ;       R0 = FINAL IMAGE STATUS.
                        0E73  1314 ;
                        0E73  1315 ; OUTPUTS:
                        0E73  1316 ;
                        0E73  1317 ;       IMAGE EXIT IS EXECUTED.
                        0E73  1318 ;-
                        0E73  1319
                        0E73  1320 EXE$EXIT_IMAGE::                             ; EXIT IMAGE
              50    DD  0E73  1321          PUSHL   R0                        ; SAVE FINAL IMAGE STATUS
              01    DD  0E75  1322          PUSHL   #1                        ; SET NUMBER OF ARGUMENTS
00000000'9F     6E    FA  0E77  1323 10$:   CALLG   (SP),@#SYS$EXIT           ; EXIT IMAGE
              F7    11  0E7E  1324          BRB     10$                       ;
```

```
                     0E80  1327          .SBTTL  CATCH ALL CONDITION HANDLER
                     0E80  1328 ;+
                     0E80  1329 ; EXE$CATCH_ALL - CATCH ALL CONDITION HANDLER
                     0E80  1330 ;
                     0E80  1331 ; THIS ROUTINE IS ENTERED AS THE RESULT OF AN UNFIELDED OR IMPROPERLY HANDLED
                     0E80  1332 ; EXCEPTION CONDITION OR SOFTWARE SIGNAL.
                     0E80  1333 ;
                     0E80  1334 ; INPUTS:
                     0E80  1335 ;
                     0E80  1336 ;       CHF$L_MCHARGLST(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
                     0E80  1337 ;       CHF$L_SIGARGLST(AP) = ADDRESS OF CONDITION ARGUMENT LIST.
                     0E80  1338 ;
                     0E80  1339 ; OUTPUTS:
                     0E80  1340 ;
                     0E80  1341 ;       A MESSAGE IS ISSUED USING THE SYS$PUTMSG SYSTEM SERVICE AND A TEST IS
                     0E80  1342 ;       MADE ON THE CONDITION NAME TO DETERMINE IF THE IMAGE SHOULD BE ALLOWED
                     0E80  1343 ;       TO CONTINUE EXECUTION. THE FOLLOWING CONDITIONS CAUSE A FORCED IMAGE
                     0E80  1344 ;       EXIT:
                     0E80  1345 ;
                     0E80  1346 ;               1. ANY ENTRY TO THIS ROUTINE VIA THE LAST CHANCE VECTOR.
                     0E80  1347 ;
                     0E80  1348 ;               2. THE CONDITION NAME HAS A SEVERITY OF ''SEVERE ERROR''.
                     0E80  1349 ;
                     0E80  1350 ;       IF A FORCED IMAGE EXIT IS PERFORMED, THEN A SUMMARY OF THE CONDITION
                     0E80  1351 ;       ARGUMENTS AND FINAL REGISTERS ARE WRITTEN TO SYS$OUTPUT.
                     0E80  1352 ;-
                     0E80  1353
                0FFC 0E80  1354          .ENTRY  EXE$CATCH_ALL,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
             00   DD 0E82  1355          PUSHL   #0                              ; SET EXCEPTION NAME FLAG FALSE
             52   DD 0E84  1356          PUSHL   R2                              ; SAVE REGISTER
       52    04 AC  D0 0E86  1357          MOVL    CHF$L_SIGARGLST(AP),R2          ; GET ADDRESS OF SIGNAL ARGUMENTS
             62   DD 0E8A  1358          PUSHL   (R2)                            ; SAVE NUMBER OF ARGUMENTS
 045C 8F    04 A2  B1 0E8C  1359          CMPW    CHF$L_SIG_NAME(R2),#SS$_SSFAIL  ; IS EXCEPTION SYS. SERV. FAIL.?
             09  12 0E92  1360          BNEQ    5$                              ; NO
                    0E94  1361          $SETSFM_S       #0                      ; YES, TURN OFF SYS. SERV. FAIL. EXCEP.
          06 A2  B5 0E9D  1362 5$:       TSTW    CHF$L_SIG_NAME+2(R2)            ; POSSIBLY SYSTEM EXCEPTION NAME?
             20  12 0EA0  1363          BNEQ    20$                             ; IF NEQ NO
          08 AE  D6 0EA2  1364          INCL    8(SP)                           ; SET EXCEPTION NAME FLAG TRUE
51  00000000'EF  9E 0EA5  1365          MOVAB   L^EXE$EXCEPTABLE,R1             ; GET ADDRESS OF EXCEPTION TABLE
          50   81 9A 0EAC  1366          MOVZBL  (R1)+,R0                        ; SET LOOP COUNT
             81  95 0EAF  1367 10$:      TSTB    (R1)+                           ; SKIP NUMBER OF ARGUMENTS
          7E   81 3C 0EB1  1368          MOVZWL  (R1)+,-(SP)                     ; GET NEXT HARDWARE EXCEPTION CODE
          0C   03 ED 0EB4  1369          CMPZV   #STS$V_CODE,#STS$S_CODE,-       ; CONDITION VALUE HARDWARE CODE?
           8E  04 A2      0EB7  1370                  CHF$L_SIG_NAME(R2),(SP)+ ;
             09  13 0EBA  1371          BEQL    30$                             ; IF EQL YES
          F0 50  F5 0EBC  1372          SOBGTR  R0,10$                          ; ANY MORE TO COMPARE?
          08 AE  D4 0EBF  1373          CLRL    8(SP)                           ; SET EXCEPTION NAME FALG FALSE
       62   02  C2 0EC2  1374 20$:      SUBL    #2,(R2)                         ; ADJUST LENGTH OF ARGUMENT LIST
 00000000'9F  95 0EC5  1375 30$:      TSTB    @#CTL$GB_SSFILTER               ; SYSTEM SERVICE INHIBITED NOW?
             0D  12 0ECB  1376          BNEQ    35$                             ; YES, DO NOT TRY TO PRINT ANYTHING
             00  DD 0ECD  1377          PUSHL   #0                              ; CLEAR ADDRESS OF FACILITY NAME DESCRIPTOR
             00  DD 0ECF  1378          PUSHL   #0                              ; CLEAR ADDRESS OF ACTION ROUTINE
             62  9F 0ED1  1379          PUSHAB  (R2)                            ; SET ADDRESS OF MESSAGE VECTOR
00000000'9F  03  FB 0ED3  1380          CALLS   #3,@#SYS$PUTMSG                 ; OUTPUT MESSAGE
             62 8ED0 0EDA  1381 35$:      POPL    (R2)                            ; RESTORE ARGUMENT COUNT
       50    04 A2  D0 0EDD  1382          MOVL    CHF$L_SIG_NAME(R2),R0           ; GET CONDITION NAME
             52 8ED0 0EE1  1383          POPL    R2                              ; RESTORE REGISTER
```

```
                                                                J 11
                51    08 AC    D0  0EE4  1384          MOVL    CHF$L_MCHARGLST(AP),R1  ; GET ADDRESS OF MECHANISM ARRAY
             7E 08 A1    03    C1  0EE8  1385          ADDL3   #3,CHF$L_MCH_DEPTH(R1),-(SP) ; LAST CHANCE ENTRY?
                         0E    13  0EED  1386          BEQL    50$                     ; IF EQL YES
                      07 50    E8  0EEF  1387          BLBS    R0,40$                  ; IF LBS SUCCESS CODE
                   03    00    ED  0EF2  1388          CMPZV   #STS$V_SEVERITY,#STS$S_SEVERITY,- ; SEVERE ERROR OR GREATER?
                   04    50        0EF5  1389                  R0,#STS$K_SEVERE        ;
                         04    18  0EF7  1390          BGEQ    50$                     ; IF GEQ YES
                   50    01    3C  0EF9  1391 40$:     MOVZWL  #SS$_CONTINUE,R0        ; SET CONTINUATION CODE
                         04    0EFC  1392              RET                             ;
                               0EFD  1393
                         50    DD  0EFD  1394 50$:     PUSHL   R0                      ; SAVE EXCEPTION NAME
                00000000'9F    95  0EFF  1395          TSTB    @#CTL$GB_SSFILTER       ; SYSTEM SERVICES INHIBITED NOW?
                         14    12  0F05  1396          BNEQ    70$                     ; YES, DON'T TRY TO PRINT ANYTHING
                   0D 08 AE    E9  0F07  1397          BLBC    8(SP),60$               ; IF LBC NOT EXCEPTION
                         6C    9F  0F0B  1398          PUSHAB  (AP)                    ; SET ADDRESS OF SIGNAL ARGUMENTS
                   F11B CF     9F  0F0D  1399          PUSHAB  SUFFIX                  ; SET ADDRESS OF MESSAGE SUFFIX
             00000000'EF       02  FB  0F11  1400          CALLS   #2,EXE$EXCMSG           ; OUTPUT EXCEPTION SUMMARY
                      00D2     30  0F18  1401 60$:     BSBW    EXE$IMGDMP_MERGE        ; TRY TO TAKE A DUMP
                         50 8ED0  0F1B  1402 70$:     POPL    R0                      ; RESTORE EXCEPTION NAME
                00 50    1C    E2  0F1E  1403          BBSS    #STS$V_INHIB_MSG,R0,80$ ; SET INHIBIT MESSAGE BIT
                      FF4E    31  0F22  1404 80$:     BRW     EXE$EXIT_IMAGE          ;
```

```
                                0F25  1407          .SBTTL EXE$RMSEXH - EXEC Mode Exit Handler
                                0F25  1408 ;+
                                0F25  1409 ; EXE$RMSEXH - Executive mode exit handler
                                0F25  1410 ;
                                0F25  1411 ; This routine is called as the result of an attempt to exit from exec mode.
                                0F25  1412 ; It's function is to run down all RMS files.
                                0F25  1413 ;
                                0F25  1414 ; INPUTS:
                                0F25  1415 ;
                                0F25  1416 ;       NONE.
                                0F25  1417 ;
                                0F25  1418 ; OUTPUTS:
                                0F25  1419 ;
                                0F25  1420 ;       NONE.
                                0F25  1421 ;
                                0F25  1422 ; SIDE EFFECTS:
                                0F25  1423 ;
                                0F25  1424 ;       RMS files are run down.
                                0F25  1425 ;-
                                0F25  1426
                          0000  0F25  1427          .ENTRY  EXE$RMSEXH,^M<>
          5E    80 AE     9E  0F27  1428          MOVAB   -128(SP),SP             ; ALLOCATE STRING BUFFER
                6E  9F      0F2B  1429          PUSHAB  (SP)                    ; BUILD BUFFER DESCRIPTOR
                00  DD      0F2D  1430          PUSHL   #0                      ;
          6E    80 8F     9A  0F2F  1431 10$:    MOVZBL  #128,(SP)               ; SET LENGTH OF STRING BUFFER
                01  DD      0F33  1432          PUSHL   #1                      ; RUN DOWN IMAGE AND ALL PPFS
             04 AE  9F      0F35  1433          PUSHAB  4(SP)                   ; PUSH ADDRESS OF BUFFER DESCRIPTOR
00000000'9F     02  FB      0F38  1434          CALLS   #2,@#SYS$RMSRUNDWN      ; RUN DOWN THE NEXT FILE
0001848C 8F     50  D1      0F3F  1435          CMPL    R0,#RMS$_BUSY           ; BUSY ERROR IMPLIES DON'T TRY
                03  13      0F46  1436          BEQL    20$                     ;  TO DO RUNDOWN AT ALL
             E4 50  E9      0F48  1437          BLBC    R0,10$                  ; IF LBC, THEN MORE TO GO
                04  0F4B  1438 20$:    RET
                    0F4C  1439
```

```
                                                L 11
                                .SBTTL XQPMERGE - Merge the XQP into P1 Space
                        ;++
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ; This routine merges the XQP into P1 space.
                        ;
                        ; The number of global sections specified by XQP$GL_SECTIONS is mapped into
                        ; the end of P1 space.  The sections have names of the form SYSXQP_nnn where
                        ; nnn ranges from zero to XQP$W_SECTIONS-1.  The section is mapped writeable-CRF
                        ; if the corresponding bit in XQP$GL_SECPROT is set.
                        ;
                        ; CALLING SEQUENCE:
                        ;
                        ;       CALLS   #0,XQPMERGE
                        ;
                        ; INPUT PARAMETERS:
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT INPUT:
                        ;
                        ;       none
                        ;
                        ; OUTPUT PARAMETERS:
                        ;
                        ;       none
                        ;
                        ; IMPLICIT OUTPUT:
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES:
                        ;
                        ;       R0 low bit set => XQP successfully merged
                        ;
                        ;               SS$_NORMAL
                        ;
                        ;       R0 low bit clear => Error occurred while merging XQP
                        ;
                        ;               Various errors returned by $IMGACT and $MGBLSC
                        ;
                        ; SIDE EFFECTS:
                        ;
                        ;       The permanent portion of P1 space is
                        ;       expanded to accommodate the merged image.
                        ;
                        ;--
                        ;
                        XQPMERGE:
                  00FC  ;       .WORD   ^M<R2,R3,R4,R5,R6,R7>   ;REGISTER SAVE MASK
                        ;
00000000'EF     D5    TSTL    XQP$GL_DZRO             ;IS THERE ANY DZRO
          16    13    BEQL    5$                      ;NO
                      $EXPREG_S -                     ;CREATE THE XQP OWN STORAGE
                              PAGCNT = XQP$GL_DZRO,-
                              REGION = #1,-
                              ACMODE = #PSL$C_EXEC
```

```
                     70 50    E9  0F69  1499          BLBC    R0,20$
                                 0F6C  1500 ;
          5E    0000000C'8F    C2  0F6C  1501 5$:      SUBL    #<XQP_NAMSIZ+3>&^C3,SP  ;RESERVE SPACE FOR GSD NAME
                        56    5E  0F73  1502          MOVL    SP,R6                   ;SAVE ADDRESS OF GSD NAME
    66  00000FDD'EF    000A'8F  28  0F76  1503          MOVC3   #XQP_NAMSIZ,XQP_NAM,(R6) ;PUT GSD NAME IN WRITEABLE STORAGE
          53    00000000'EF    D0  0F80  1504          MOVL    XQP$GL_SECTIONS,R3      ;COUNT OF SECTIONS TO MAP
          00000009'E6    53    80  0F87  1505          ADDB    R3,XQP_NAMSIZ-1(R6)     ;START WITH LAST GSD NAME
                        56    DD  0F8E  1506          PUSHL   R6                      ;BUILD DESCRIPTOR FOR GSD NAME
                0000000A'8F    DD  0F90  1507          PUSHL   #XQP_NAMSIZ
                        52    5E  0F96  1508          MOVL    SP,R2                   ;ADDRESS OF DESCRIPTOR
          7E    7FFFFFFF 8F    D0  0F99  1509          MOVL    #^X7FFFFFFF,-(SP)       ;END VA FOR BLUEPRINT P0 VA RANGE
                     7E    6E  D0  0FA0  1510          MOVL    (SP),-(SP)              ;START VA FOR BLUEPRINT P0 VA RANGE
                     54    5E  D0  0FA3  1511          MOVL    SP,R4                   ;ADR OF INPUT VA RANGE
                           7E  7C  0FA6  1512          CLRQ    -(SP)                   ;RETURN VA RANGE
                     55    5E  D0  0FA8  1513          MOVL    SP,R5                   ;ADR OF RETURN VA RANGE
                           53  D7  0FAB  1514          DECL    R3                      ;MAKE COUNT ZERO-BASED
                00000009'E6    97  0FAD  1515 10$:     DECB    XQP_NAMSIZ-1(R6)        ;NEXT GSD NAME
                                   0FB3  1516          $MGBLSC_S -
                                   0FB3  1517                  INADR  = (R4),-
                                   0FB3  1518                  RETADR = (R5),-
                                   0FB3  1519                  FLAGS  = #<SEC$M_EXPREG!SEC$M_SYSGBL>,-
                                   0FB3  1520                  GSDNAM = (R2),-
                                   0FB3  1521                  ACMODE = #PSL$C_EXEC
                     0D 50    E9  0FCC  1522          BLBC    R0,20$
                     DB 53    F4  0FCF  1523          SOBGEQ  R3,10$
                                 0FD2  1524 ;
          00000000'GF    65    D0  0FD2  1525          MOVL    (R5),G^CTL$GL_CTLBASVA  ;SET A NEW CONTROL REGION BASE
                     00 B5    17  0FD9  1526          JMP     @(R5)                   ;XQP SELF-INITIALIZATION
                              04  0FDC  1527 20$:      RET                             ; AND RETURN TO CALLER
                                 0FDD  1528
                                 0FDD  1529 XQP_NAM:
30 30 30 5F 50 51 58 53 59 53    0FDD  1530          .ASCII  /SYSXQP_000/
                          0000000A  0FE7  1531          XQP_NAMSIZ = .-XQP_NAM
```

```
                                0FE7  1534          .SBTTL  IMAGE DUMP MERGE
                                0FE7  1535 ;+
                                0FE7  1536 ; EXE$IMGDMP_MERGE - MERGE IN THE IMAGE DUMP FACILITY AND CALL IT
                                0FE7  1537 ;
                                0FE7  1538 ; THIS ROUTINE IS ENTERED AS THE RESULT OF A FATAL CONDITION WHICH WILL FORCE
                                0FE7  1539 ; IMAGE EXIT
                                0FE7  1540 ;
                                0FE7  1541 ; INPUTS:
                                0FE7  1542 ;
                                0FE7  1543 ;       CHF$L_MCHARGLST(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
                                0FE7  1544 ;       CHF$L_SIGARGLST(AP) = ADDRESS OF CONDITION ARGUMENT LIST.
                                0FE7  1545 ;
                                0FE7  1546 ; OUTPUTS:
                                0FE7  1547 ;
                                0FE7  1548 ;       AFTER PRIVILEGE CHECKS, THE IMAGE DUMP FACILITY IS MERGED INTO THE
                                0FE7  1549 ;       ADDRESS SPACE AND CALLED.
                                0FE7  1550 ;
                                0FE7  1551 ;-
                                0FE7  1552
                                0FE7  1553 EXE$IMGDMP_EXEC::                    ; EXEC MODE ENTRY POINT
                     0FFC 8F    BB 0FE7  1554          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                          10    11 0FEB  1555          BRB     EXEC_M
                                0FED  1556
                                0FED  1557          .ENABL  LSB
                                0FED  1558
                                0FED  1559 EXE$IMGDMP_MERGE::
                     0FFC 8F    BB 0FED  1560          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                          50    DC 0FF1  1561          MOVPSL  R0                      ; GET CURRENT PSL
           50   50    02    18 EF 0FF3  1562          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R0,R0
                     03    50 D1 0FF8  1563          CMPL    R0,#PSL$C_USER          ; IS IT USER MODE
                           5F 12 0FFB  1564          BNEQ    5$                      ; NO - DUMP NOT ALLOWED
                     5E    FD90 CE 0FFD  1565 EXEC_M: MOVAL   -SCRATCHSIZE(SP),SP     ; RESERVE SCRATCH SPACE ON STACK
                           56   5E D0 1002  1566          MOVL    SP,R6
6E   0270 8F    00    6E    00 2C 1005  1567          MOVC5   #0,(SP),#0,#SCRATCHSIZE,(SP) ; ZERO IT
      0248 C6     02040004 8F D0 100D  1568          MOVL    #<JPI$_PROCPRIV@16>+4,JPI_PROC(R6) ;INITIALIZE TO GET PROCESS PRIV
          024C C6    0234 C6 9E 1016  1569          MOVAB   PROCPRIV(R6),JPI_PROC+4(R6)
      0254 C6     04130004 8F D0 101D  1570          MOVL    #<JPI$_IMAGPRIV@16>+4,JPI_IMAG(R6) ;INITIALIZE TO GET IMAGE PRIV
          0258 C6    023C C6 9E 1026  1571          MOVAB   IMAGPRIV(R6),JPI_IMAG+4(R6)
      0260 C6     041B0004 8F D0 102D  1572          MOVL    #<JPI$_PHDFLAGS@16>+4,JPI_FLAG(R6) ;INITIALIZE TO GET PHD FLAGS
          0264 C6    0244 C6 9E 1036  1573          MOVAB   PHD_FLAGS(R6),JPI_FLAG+4(R6)
               50    0248 C6 9E 103D  1574          MOVAB   JPI_PROC(R6),R0 ; ADDRESS OF ITEM LIST
                           1042  1575          $GETJPI_S EFN = #EXE$C_SYSEFN,-
                           1042  1576                    ITMLST = (R0)
                     03    50 E8 1059  1577          BLBS    R0,10$
                      0084 31 105C  1578 5$:        BRW     30$                     ; ERROR - GIVE UP
          F7 0244 C6    05 E1 105F  1579 10$:       BBC     #PHD$V_IMGDMP,PHD_FLAGS(R6),5$ ; NO DUMP REQUESTED
      023C C6    0234 C6 CA 1065  1580          BICL    PROCPRIV(R6),IMAGPRIV(R6) ; TEST THAT IMAGE PRIVILEGES AREN'T GREATE
      0240 C6    0238 C6 CA 106C  1581          BICL    PROCPRIV+4(R6),IMAGPRIV+4(R6)
      023C C6    0240 C6 C8 1073  1582          BISL    IMAGPRIV+4(R6),IMAGPRIV(R6)
                      0E 13 107A  1583          BEQL    20$                     ; NO EXCESS IMAGE PRIVILEGES
        08 0234 C6    00 E0 107C  1584          BBS     #PRV$V_CMKRNL,PROCPRIV(R6),20$
        02 0234 C6    0E E0 1082  1585          BBS     #PRV$V_SETPRV,PROCPRIV(R6),20$
                      D2 11 1088  1586          BRB     5$                      ; INSUFFICIENT PRIVILEGES
                66    08 D0 108A  1587 20$:       MOVL    #IMGACT$_NARGS,(R6)     ; SET ARGUMENT COUNT FOR $IMGACT CALL
        04 A6   EFC6  CF 9E 108D  1588          MOVAB   IMGDMPNAM,IMGACT$_NAME(R6) ; SET ADR OF INPUT FILE NAME DESC
        08 A6   EFA8  CF 9E 1093  1589          MOVAB   DEFAULTNAMDSC,IMGACT$_DFLNAM(R6) ; SET ADR OF DEFAULT NAME STR
             10 A6    30 D0 1099  1590          MOVL    #<IAC$M_MERGE ! IAC$M_EXPREG>,IMGACT$_IMGCTL(R6) ; SET CTL FLAGS
```

```
                                                     B 12
                0C A6    34 A6    9E  109D  1591          MOVAB   HDRBUF(R6),IMGACT$_HDRBUF(R6) ; SET ADR OF IMAGE HEADER BUFFER
                14 A6    24 A6    9E  10A2  1592          MOVAB   IMGACT_INADR(R6),IMGACT$_INADR(R6) ; SET ADR OF INPUT VA RANGE
                18 A6    2C A6    9E  10A7  1593          MOVAB   IMGACT_RETADR(R6),IMGACT$_RETADR(R6) ; SET ADR OF RETURN RANGE
                         1C A6    D4  10AC  1594          CLRL    IMGACT$_IDENT(R6)           ; NO MATCH IDENT SPECIFIED
             24 A6    0200 8F    3C  10AF  1595          MOVZWL  #^X200,IMGACT_INADR(R6) ; SET A BLUEPRINT P0 ADDRESS RANGE FOR
        28 A6   3FFFFFFF 8F    D0  10B5  1596          MOVL    #1@30-1,IMGACT_INADR+4(R6) ; MAPPING TO FIRST FREE VA SPACE
                                    10BD  1597          $IMGACT_G (R6)                      ; MAP IN THE DUMP IMAGE
                         1C 50    E9  10C4  1598          BLBC    R0,30$                      ; ERROR - GIVE UP
                                    10C7  1599          $IMGFIX_S
                         12 50    E9  10CE  1600          BLBC    R0,30$
                   51    2C A6    D0  10D1  1601          MOVL    IMGACT_RETADR(R6),R1        ; START OF THE MERGED IN CODE
                51 51    08 A1    C1  10D5  1602          ADDL3   8(R1),R1,R1                 ; START ADDRESS OF THE DUMP ROUTINE
                   5E    0270 C6    DE  10DA  1603          MOVAL   SCRATCHSIZE(R6),SP          ; GET RID OF SCRATCH STORAGE
                         61   16    10DF  1604          JSB     (R1)
                         05   11    10E1  1605          BRB     40$
                                    10E3  1606 ;
                   5E    0270 C6    DE  10E3  1607 30$:    MOVAL   SCRATCHSIZE(R6),SP          ; GET RID OF SCRATCH STORAGE
                         0FFC 8F    BA  10E8  1608 40$:    POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                         05   10    10EC  1609          RSB
                                    10ED  1610          .DSABL  LSB
                                    10ED  1611
                                    10ED  1612          .SBTTL CRELNM - FIXUP AND INSERT A LOGICAL NAME BLOCK
                                    10ED  1613 ;++
                                    10ED  1614 ; FUNCTIONAL DESCRIPTION:
                                    10ED  1615 ;
                                    10ED  1616 ; THE PURPOSE OF THIS ROUTINE IS TO FIXUP A LNMB FOR A LOGICAL NAME AND INSERT
                                    10ED  1617 ; IT INTO THE APPROPRIATE HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME HASH
                                    10ED  1618 ; TABLE. THE LOGICAL NAME BEING FIXED REQUIRES THAT ITS EQUIVALENCE STRING BE
                                    10ED  1619 ; MOVED FROM THE PQB INTO THE STORAGE ALLOCATED FOR IT. IF THE LENGTH OF THE
                                    10ED  1620 ; EQUIVALENCE STRING IS 0 THEN THE BLOCK OF STORAGE FOR THIS LOGICAL NAME IS
                                    10ED  1621 ; DEALLOCATED AND THE ROUTINE EXITS.
                                    10ED  1622 ;
                                    10ED  1623 ; CALLING SEQUENCE:
                                    10ED  1624 ;
                                    10ED  1625 ;       BSBW    CRELNM
                                    10ED  1626 ;       .WORD   PQB$T_<OFFSET>
                                    10ED  1627 ;       .WORD   PQB$T_<OFFSET>_ATT
                                    10ED  1628 ;       .WORD   <NAME>_LNMX - PROC_DIR
                                    10ED  1629 ;       .WORD   <NAME> - PROC_DIR
                                    10ED  1630 ;
                                    10ED  1631 ; INPUT PARAMETERS:
                                    10ED  1632 ;
                                    10ED  1633 ;       R6      - ADDRESS OF PQB
                                    10ED  1634 ;       R7      - ADDRESS OF PROCESS DIRECTORY'S TABLE HEADER
                                    10ED  1635 ;       R8      - ADDRESS OF ALLOCATED P1 STORAGE
                                    10ED  1636 ;       R9      - ADDRESS OF PROCESS TABLE'S TABLE HEADER
                                    10ED  1637 ;
                                    10ED  1638 ; IMPLICIT INPUT:
                                    10ED  1639 ;
                                    10ED  1640 ;       IT IS ASSUMED THAT THE LOGICAL NAME BEING CREATED HAS ALREADY BEEN
                                    10ED  1641 ;       FORMATTED WITH THE EXCEPTION OF:
                                    10ED  1642 ;
                                    10ED  1643 ;       1. THE CONTAINING TABLE ADDRESS WITHIN ITS LNMB.
                                    10ED  1644 ;       2. THE TRANSLATION STRING AND ATTRIBUTES WITHIN ITS LNMX.
                                    10ED  1645 ;
                                    10ED  1646 ; OUTPUT PARAMETERS:
                                    10ED  1647 ;       NONE
```

```
                                10ED  1648 ;
                                10ED  1649 ; IMPLICIT OUTPUT:
                                10ED  1650 ;       NONE
                                10ED  1651 ;
                                10ED  1652 ; COMPLETION CODES:
                                10ED  1653 ;       NONE
                                10ED  1654 ;
                                10ED  1655 ; SIDE EFFECTS:
                                10ED  1656 ;
                                10ED  1657 ;       R0 - R3, R5, AND AP ARE DESTROYED.
                                10ED  1658 ;--
                                10ED  1659
                                10ED  1660 CRELNM:                                      ; FIXUP AND INSERT THE LOGICAL NAME
                   5C   6E   D0 10ED  1661          MOVL    (SP),AP                    ; RETRIEVE ARGUEMENT POINTER
                   6E   08   C0 10F0  1662          ADDL2   #8,(SP)                    ; CORRECT RETURN PC VALUE
                   51   8C   3C 10F3  1663          MOVZWL  (AP)+,R1                   ; RETRIEVE OFFSET TO TRANSLATION
                50   6641   9A 10F6  1664          MOVZBL  (R6)[R1],R0                ; RETRIEVE THE SIZE OF THE TRANSLATION
                       13   12 10FA  1665          BNEQ    10$                        ; IF ITS NOT 0 THEN CONTINUE
                50   04 AC   3C 10FC  1666          MOVZWL  4(AP),R0                   ; RETRIEVE OFFSET TO LNMB
                   50   58   C0 1100  1667          ADDL2   R8,R0                      ; COMPUTE ADDRESS OF LNMB
                51   08 A0   3C 1103  1668          MOVZWL  LNMB$W_SIZE(R0),R1         ; RETRIEVE SIZE OF BLOCK TO DEALLOCATE
             00000000'EF   16 1107  1669          JSB     EXE$DEAP1                  ; DEALLOCATE THE LNMB
                       27   11 110D  1670          BRB     20$                        ; GO RETURN
                            110F  1671
                   52   8C   3C 110F  1672 10$:     MOVZWL  (AP)+,R2                   ; RETRIEVE OFFSET TO TRANSLATION ATTRIBUTES
                   53   8C   3C 1112  1673          MOVZWL  (AP)+,R3                   ; RETRIEVE OFFSET TO LNMX
            6843   01 A642   90 1115  1674          MOVB    1(R6)[R2],-                ; STORE THE TRANSLATION ATTRIBUTES FROM
                                1118  1675                  LNMX$B_FLAGS(R8)[R3]       ; THE PQB INTO THEN LNMX FLAG FIELD
                   50   D6   11B  1676          INCL    R0                         ; MOVE COUNT ALONG WITH TRANSLATION
         04 A843   6641   50   28 111D  1677          MOVC3   R0,(R6)[R1],-              ; MOVE TRANSLATION COUNT AND STRING FROM
                                1124  1678                  LNMX$T_XLATION(R8)[R3]     ; THE PQB INTO THE APPROPRIATE LNMX FIELD
                   51   8C   3C 1124  1679          MOVZWL  (AP)+,R1                   ; RETRIEVE OFFSET TO LNMB
                   51   58   C0 1127  1680          ADDL2   R8,R1                      ; COMPUTE ADDRESS OF LNMB
                0C A1   59   D0 112A  1681          MOVL    R9,LNMB$L_TABLE(R1)        ; STORE CONTAINING TABLE HEADER'S ADDR
                   52   D4 112E  1682          CLRL    R2                         ; NO SPECIAL INSERTION ATTRIBUTES
             00000000'EF   16 1130  1683          JSB     LNM$INSLOGTAB              ; APPROPRIATELY INSERT SYS$ERROR
                       05   1136  1684 20$:     RSB                                ; RETURN
```

```
1137  1686          .SBTTL EXE$CRE_JGTABLE - CREATE GROUP AND JOB-WIDE LOGICAL NAME TABLES
1137  1687 ;++
1137  1688 ; FUNCTIONAL DESCRIPTION:
1137  1689 ;
1137  1690 ; THE PURPOSE OF THIS ROUTINE IS TO HANDCRAFT GROUP AND JOB-WIDE LOGICAL
1137  1691 ; NAME TABLES AND DIRECT THEIR INSERTION INTO THE APPROPRIATE HASH BUCKET
1137  1692 ; OF THE SYSTEM LOGICAL NAME HASH TABLE. GROUP LOGICAL NAME TABLES ARE INSERTED
1137  1693 ; SUCH THAT IF THERE IS AN EXISTING GROUP TABLE FOR THAT GROUP, THE CALLER OF
1137  1694 ; THIS ROUTINE IS MAPPED TO IT.
1137  1695 ;
1137  1696 ; CALLING SEQUENCE:
1137  1697 ;
1137  1698 ;       BSBW    EXE$CRE_JGTABLE
1137  1699 ;
1137  1700 ; INPUT PARAMETERS:
1137  1701 ;
1137  1702 ;       R7      - JOB TABLE QUOTA
1137  1703 ;       R10     - ADDRESS OF ASCII EQUIVALENT OF JIB ADDRESS
1137  1704 ;       R11     - ADDRESS OF ASCII EQUIVALENT OF GROUP NUMBER
1137  1705 ;
1137  1706 ; IMPLICIT INPUT:
1137  1707 ;
1137  1708 ;       LNM_SYSTEM_DIR_LNMTH    - ADDRESS OF SYSTEM DIRECTORY TABLE HEADER
1137  1709 ;
1137  1710 ;       LNM$AL_HASHTBL          - ADDRESS OF POINTER TO SYSTEM HASH TABLE
1137  1711 ;
1137  1712 ;       SCH$GL_CURPCB           - ADDRESS OF PCB
1137  1713 ;
1137  1714 ; OUTPUT PARAMETERS:
1137  1715 ;       NONE
1137  1716 ;
1137  1717 ; IMPLICIT OUTPUT:
1137  1718 ;
1137  1719 ;       R4                      - ADDRESS OF PCB
1137  1720 ;
1137  1721 ; COMPLETION CODES:
1137  1722 ;
1137  1723 ;       1               - SUCCESS
1137  1724 ;       SS$_EXLNMQUOTA  - INSUFFICIENT QUOTA IN SYSTEM DIRECTORY TABLE
1137  1725 ;       SS$_INSFMEM     - INSUFFICIENT PAGED POOL TO ALLOCATE LNMBS
1137  1726 ;
1137  1727 ; SIDE EFFECTS:
1137  1728 ;
1137  1729 ;       R0 - R5 AND R8 ARE DESTROYED.
1137  1730 ;
1137  1731 ;       THE JOB-WIDE LOGICAL NAME TABLE WILL HAVE BEEN CREATED POTENTIALLY
1137  1732 ;       RESULTING IN THE DELETION OF ANY SHAREABLE TABLE WITH THE SAME NAME.
1137  1733 ;
1137  1734 ;       THE GROUP LOGICAL NAME TABLE WILL HAVE BEEN CREATED PROVIDED A GROUP
1137  1735 ;       TABLE WITH THAT NAME DOES NOT ALRADY EXIST IN WHICH CASE NOTHING IS
1137  1736 ;       DONE.
1137  1737 ;
1137  1738 ;--
1137  1739
1137  1740          .ENABL  LSB
1137  1741
1137  1742 EXE$CRE_GTABLE::
```

```
                          1137  1743
                          1137  1744 ;
                          1137  1745 ; THIS ROUTINE EXE$CRE_GTABLE IS IDENTICAL TO THE ROUTINE EXE$CRE_JGTABLE
                          1137  1746 ; WITH EXCEPTION THAT THE JOB LOGICAL NAME TABLE IS NOT CREATED. THUS THE
                          1137  1747 ; ONLY INPUT PARAMETER IS R11, WHICH HAS THE ADDRESS OF ASCII EQUIVALENT
                          1137  1748 ; OF GROUP NUMBER.
                          1137  1749 ;
                          1137  1750
      51   00C0 8F   3C   1137  1751          MOVZWL  #GROUP_TABLE_SIZE,R1    ; SET SIZE OF GROUP TABLE TO BE CREATED
       00000000'GF   16   113C  1752          JSB     G^EXE$ALOPAGED          ; ALLOCATE REQUIRED AMOUNT OF PAGED POOL
             08 50   E8   1142  1753          BLBS    R0,2$                   ; CONTINUE IF ALLOCATION IS SUCCESSFUL
      50   0124 8F   3C   1145  1754          MOVZWL  #SS$_INSFMEM,R0         ; OTHERWISE SETUP R0 WITH ERROR CODE
                0126   31   114A  1755          BRW     40$                     ; AND EXIT
                          114D  1756
   54   00000000'GF   D0   114D  1757 2$:      MOVL    G^SCH$GL_CURPCB,R4      ; RETRIEVE PCB ADDRESS
       00000000'GF   16   1154  1758          JSB     G^LNM$LOCKW             ; LOCK LOGICAL NAME MUTEX FOR WRITING
                          115A  1759
00000021'EF   51   D1   115A  1760          CMPL    R1,LNMTH$L_BYTES+-      ; IS THERE ENOUGH QUOTA IN THE SYSTEM
                          1161  1761                  LNM_SYSTEM_DIR_LNMTH    ; DIRECTORY TABLE?
                17   15   1161  1762          BLEQ    4$                      ; CONTINUE IF ENOUGH QUOTA
            50   52   D0   1163  1763          MOVL    R2,R0                   ; SETUP TO DEALLOCATE THE PAGED POOL
       00000000'GF   16   1166  1764          JSB     G^EXE$DEAPGDSIZ         ; DEALLOCATE IT
       00000000'GF   16   116C  1765          JSB     G^LNM$UNLOCK            ; UNLOCK THE LOGICAL NAME MUTEX
      50   224C 8F   3C   1172  1766          MOVZWL  #SS$_EXLNMQUOTA,R0      ; SETUP REASON FOR PREMATURE TERMINATION
                00F9   31   1177  1767          BRW     40$                     ; AND GO RETURN TO CALLER
                          117A  1768
            58   52   D0   117A  1769 4$:      MOVL    R2,R8                   ; SAVE ADDRESS OF STORAGE ALLOCATED
   62   F5DE CF   51   28   117D  1770          MOVC3   R1,GROUP_TABLE,(R2)     ; FORMAT THE LOGICAL NAME TABLE(S)
    54   00000000'GF   D0   1183  1771          MOVL    G^SCH$GL_CURPCB,R4      ; RETRIEVE PCB ADDRESS
                009C   31   118A  1772          BRW     CREATE_GTABLE           ; GO CREATE GROUP TABLE
                          118D  1773
                          118D  1774 EXE$CRE_JGTABLE::
                          118D  1775
                          118D  1776 ;
                          118D  1777 ; ALLOCATE PAGED POOL FOR THE GROUP AND JOB-WIDE LOGICAL NAME TABLES. AFTER
                          118D  1778 ; ALLOCATING SUFFICIENT PAGED POOL, WRITE LOCK THE LOGICAL NAME MUTEX, AND MAKE
                          118D  1779 ; SURE THAT THE PARENT LOGICAL NAME TABLE, THE SYSTEM DIRECTORY TABLE, HAS
                          118D  1780 ; SUFFICIENT QUOTA FOR THE CREATION OF BOTH LOGICAL NAME TABLES AND FOR ANY
                          118D  1781 ; SEPARATE QUOTA THAT WILL BE RELEGATED TO THEM. IF THE SYSTEM DIRECTORY TABLE
                          118D  1782 ; DOES NOT CONTAIN SUFFICIENT QUOTA THEN EXIT IMMEDIATELY WITH THE APPROPRIATE
                          118D  1783 ; ERROR; OTHERWISE, THE BLOCK OF STORAGE THAT HAS BEEN ALLOCATED FOR THE LOGICAL
                          118D  1784 ; NAME TABLES IS FORMATED.
                          118D  1785 ;
                          118D  1786
      51   0180 8F   3C   118D  1787          MOVZWL  #SO_ALLOC_SIZE,R1       ; ASSUME BOTH TABLES WILL BE CREATED
       00000000'GF   16   1192  1788          JSB     G^EXE$ALOPAGED          ; ALLOCATE REQUIRED AMOUNT OF PAGED POOL
             07 50   E8   1198  1789          BLBS    R0,1$                   ; CONTINUE IF ALLOCATION IS SUCCESSFUL
      50   0124 8F   3C   119B  1790          MOVZWL  #SS$_INSFMEM,R0         ; OTHERWISE SETUP R0 WITH ERROR CODE
                2E   11   11A0  1791          BRB     10$                     ; AND EXIT
                          11A2  1792
   54   00000000'GF   D0   11A2  1793 1$:      MOVL    G^SCH$GL_CURPCB,R4      ; RETRIEVE PCB ADDRESS
       00000000'GF   16   11A9  1794          JSB     G^LNM$LOCKW             ; LOCK LOGICAL NAME MUTEX FOR WRITING
                          11AF  1795
      50   51   57   C1   11AF  1796          ADDL3   R7,R1,R0                ; DETERMINE TOTAL AMOUNT OF QUOTA
00000021'EF   50   D1   11B3  1797          CMPL    R0,LNMTH$L_BYTES+-      ; IS THERE ENOUGH QUOTA IN THE SYSTEM
                          11BA  1798                  LNM_SYSTEM_DIR_LNMTH    ; DIRECTORY TABLE?
                17   15   11BA  1799          BLEQ    20$                     ; CONTINUE IF ENOUGH QUOTA
```

```
                50    52    D0  11BC  1800          MOVL    R2,R0                           ; SETUP TO DEALLOCATE THE PAGED POOL
 00000000'GF          16    11BF  1801          JSB     G^EXE$DEAPGDSIZ                 ; DEALLOCATE IT
 00000000'GF          16    11C5  1802          JSB     G^LNM$UNLOCK                    ; UNLOCK THE LOGICAL NAME MUTEX
           50    224C  8F    3C  11CB  1803          MOVZWL  #SS$_EXLNMQUOTA,R0      ; SETUP REASON FOR PREMATURE TERMINATION
                 00A0  31    11D0  1804 10$:        BRW     40$                     ; AND GO RETURN TO CALLER
                             11D3  1805
                58    52    D0  11D3  1806 20$:     MOVL    R2,R8                   ; SAVE ADDRESS OF STORAGE ALLOCATED
        62    F585  CF    51  28  11D6  1807          MOVC3   R1,GROUP_TABLE,(R2)     ; FORMAT THE LOGICAL NAME TABLE(S)
        54    00000000'GF   D0  11DC  1808          MOVL    G^SCH$GL_CURPCB,R4      ; RETRIEVE PCB ADDRESS
                             11E3  1809
                             11E3  1810 ;
                             11E3  1811 ; FIXUP THE LOGICAL NAME BLOCK FOR THE JOB TABLE THAT IS BEING CREATED, AND
                             11E3  1812 ; THEN INSERT IT INTO THE APPROPRIATE HASH BUCKET OF THE SHAREABLE LOGICAL NAME
                             11E3  1813 ; HASH TABLE. THIS FIXING UP OF THE JOB TABLE'S LOGICAL NAME BLOCK INCLUDES
                             11E3  1814 ; APPENDING TO THE "LNM$JOB_" ASCII STRING ALREADY PRESENT WITHIN THE NAME FIELD
                             11E3  1815 ; OF THE LNMB, THE ASCII EQUIVALENT OF THE JIB'S HEXADECIMAL ADDRESS. A POINTER
                             11E3  1816 ; TO THIS ASCII EQUIVALENCE IS PASSED TO THIS ROUTINE IN R7.
                             11E3  1817 ;
                             11E3  1818
           51    00C0  C8    9E  11E3  1819          MOVAB   JOB_TABLE(R8),R1        ; RETRIEVE ADDRESS OF JOB TABLE'S LNMB
             1A  A1    6A    7D  11E8  1820          MOVQ    (R10),LNMB$T_NAME+9(R1) ; MOVE ASCII HEX JIB ADDR INTO NAME FIELD
 00000000'FF          D0  11EC  1821          MOVL    @LNM$AL_HASHTBL,-               ; MOVE THE ADDRESS OF THE SHAREABLE
                 00E8  C8    11F2  1822                  JOB_TABLE_LNMTH+-           ; LOGICAL NAME HASH TABLE INTO THE JOB
                             11F5  1823                  LNMTH$L_HASH(R8)            ; TABLE'S TABLE HEADER
                 0110  C8    9E  11F5  1824          MOVAB   JOB_TABLE_ORB(R8),-     ; MOVE THE ADDRESS OF THE JOB TABLE'S
                 00EC  C8    11F9  1825                  JOB_TABLE_LNMTH+-           ; OBJECT RIGHTS BLOCK INTO THE JOB
                             11FC  1826                  LNMTH$L_ORB(R8)             ; TABLE'S TABLE HEADER
           00F0  C8    51  D0  11FC  1827          MOVL    R1,JOB_TABLE_LNMTH+-    ; MOVE THE ADDRESS OF THE JOB TABLE'S
                             1201  1828                  LNMTH$L_NAME(R8)            ; LNMB INTO THE JOB TABLE'S TABLE HEADER
                             1201  1829
                 00BC  C4    D0  1201  1830          MOVL    PCB$L_UIC(R4),-         ; MOVE THE PROCESS'S UIC INTO THE
                 0110  C8    1205  1831                  JOB_TABLE_ORB+-             ; APPROPRIATE FIELD OF THE JOB TABLE'S
                             1208  1832                  ORB$L_OWNER(R8)             ; OBJECT RIGHTS BLOCK
                 0138  C8    7C  1208  1833          CLRQ    JOB_TABLE_ORB+-         ; SET INITIAL NULL ACL
                             120C  1834                  ORB$L_ACL_COUNT(R8)
                             120C  1835
                      57     D5  120C  1836          TSTL    R7                      ; IS JOB TABLE QUOTA POOLED?
                      11     13  120E  1837          BEQL    30$                     ; IF SO THEN GO INSERT LNMB
                 00E7  C8    9E  1210  1838          MOVAB   JOB_TABLE_LNMTH(R8),-   ; OTHERWISE SET UP THE JOB TABLE'S
                 0100  C8    1214  1839                  JOB_TABLE_LNMTH+-           ; TABLE HEADER AS THE QUOTA HOLDER FOR
                             1217  1840                  LNMTH$L_QTABLE(R8)          ; THE JOB TABLE
     0104  C8    57    D0  1217  1841          MOVL    R7,JOB_TABLE_LNMTH+-    ; SET THE BYTE LIMIT FIELD TO THE
                             121C  1842                  LNMTH$L_BYTESLM(R8)         ; INITIAL AMOUNT OF JOB TABLE QUOTA
     0108  C8    57    D0  121C  1843          MOVL    R7,JOB_TABLE_LNMTH+-    ; SET THE BYTE REMAINING FIELD TO THE
                             1221  1844                  LNMTH$L_BYTES(R8)           ; INITIAL AMOUNT OF JOB TABLE QUOTA
                      52     D4  1221  1845 30$:     CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
 00000000'GF          16    1223  1846          JSB     G^LNM$INSLOGTAB             ; APPROPRIATELY INSERT LNM$GROUP_xxxxxx
                             1229  1847
                             1229  1848 ;
                             1229  1849 ; FIXUP THE LOGICAL NAME BLOCK FOR THE GROUP TABLE THAT IS BEING CREATED, AND
                             1229  1850 ; THEN INSERT IT INTO THE APPROPRIATE HASH BUCKET OF THE SHAREABLE LOGICAL NAME
                             1229  1851 ; HASH TABLE PROVIDED A TABLE FOR THAT GROUP DOES NOT ALREADY EXIST. THIS
                             1229  1852 ; FIXING UP OF THE GROUP TABLE'S LOGICAL NAME BLOCK INCLUDES APPENDING TO THE
                             1229  1853 ; "LNM$GROUP_" ASCII STRING ALREADY PRESENT WITHIN THE NAME FIELD OF THE LNMB,
                             1229  1854 ; THE ASCII EQUIVALENT OF THE "OCTAL GROUP" THE PROCESS BELONGS TO. A POINTER
                             1229  1855 ; TO THIS ASCII EQUIVALENCE IS PASSED TO THIS ROUTINE IN R8.
                             1229  1856 ;
```

```
                        1229  1857
                        1229  1858 CREATE_GTABLE:                                  ;
            51    58    DO  1229  1859          MOVL    R8,R1                       ; SETUP TO INSERT THE GROUP TABLE'S LNMB
         1C A1    6B    DO  122C  1860          MOVL    (R11),LNMB$T_NAME+11(R1)    ; APPEND THE ASCII "OCTAL GROUP" TO THE
      20 A1    04 AB    B0  1230  1861          MOVW    4(R11),LNMB$T_NAME+15(R1)   ; "LNM$GROUP_" ALREADY IN THE NAME FIELD
        00000000'FF    DO  1235  1862          MOVL    @LNM$AL_HASHTBL,-           ; MOVE THE ADDRESS OF THE SHAREABLE
              28 A8         123B  1863                  GROUP_TABLE_LNMTH+-         ; LOGICAL NAME HASH TABLE INTO THE GROUP
                            123D  1864                  LNMTH$L_HASH(R8)            ; LOGICAL NAME TABLE'S VABLE HEADER
              50 A8    9E   123D  1865          MOVAB   GROUP_TABLE_ORB(R8),-       ; MOVE THE ADDRESS OF THE GROUP TABLE'S
              2C A8         1240  1866                  GROUP_TABLE_LNMTH+-         ; OBJECT RIGHTS BLOCK INTO THE GROUP
                            1242  1867                  LNMTH$L_ORB(R8)             ; TABLE'S TABLE HEADER
         30 A8    51    DO  1242  1868          MOVL    R1,GROUP_TABLE_LNMTH+-      ; MOVE THE ADDRESS OF THE LNMB INTO THE
                            1246  1869                  LNMTH$L_NAME(R8)            ; GROUP TABLE'S TABLE HEADER
                            1246  1870
            00BE C4    B0   1246  1871          MOVW    PCB$W_GRP(R4),-             ; MOVE THE PROCESS'S GROUP NUMBER
              52 A8         124A  1872                  GROUP_TABLE_ORB+-           ; INTO THE APPROPRIATE FIELD OF THE
                            124C  1873                  ORB$L_OWNER+2(R8)           ; GROUP TABLE'S OBJECT RIGHTS BLOCK
              78 A8    7C   124C  1874          CLRQ    GROUP_TABLE_ORB+-           ; SET INITIAL NULL ACL
                            124F  1875                  ORB$L_ACL_COUNT(R8)
                            124F  1876
   52   01000000 8F    DO  124F  1877          MOVL    #LNM$M_CREATE_IF,R2         ; GROUP TABLES ARE INSERTED CREATE_IF
        00000000'GF    16  1256  1878          JSB     G^LNM$INSLOGTAB             ; APPROPRIATELY INSERT LNM$GROUP_xxxxxx
                            125C  1879
            50    01    B1  125C  1880          CMPW    #SS$_NORMAL,R0              ; DID THE GROUP TABLE ALREADY EXIST?
                  09    12  125F  1881          BNEQ    35$                         ; GO UNLOCK THE MUTEX IF IT DIDN'T
            50    58    DO  1261  1882          MOVL    R8,R0                       ; DELETE THE LNMB FOR WHAT WOULD HAVE
        00000000'GF    16  1264  1883          JSB     G^EXE$DEAPAGED              ; BECOME A GROUP LOGICAL NAME TABLE
                            126A  1884 ;
                            126A  1885 ; UNLOCK THE LOGICAL NAME MUTEX AND RETURN STATUS.
                            126A  1886 ;
                            126A  1887
        00000000'GF    16  126A  1888 35$:    JSB     G^LNM$UNLOCK                ; UNLOCK THE LOGICAL NAME MUTEX
            50    01    9A  1270  1889         MOVZBL  #1,R0                       ; SUCCESS
                  05    05  1273  1890 40$:    RSB                                 ; RETURN
                            1274  1891
                            1274  1892         .DSABL  LSB
                            1274  1893
                            1274  1894         .END
```

```
$$ARGS                          = 00000008
$$T1                            = 00000001
CCB$C_LENGTH                    = 00000010
CHARS                             0000001C R      02
CHF$L_MCHARGLST                 = 00000008
CHF$L_MCH_DEPTH                 = 00000008
CHF$L_SIGARGLST                 = 00000004
CHF$L_SIG_NAME                  = 00000004
CLI$_INVREQTYP                  = 00038822
CREATE_GTABLE                     00001229 R      02
CRELNM                            000010ED R      02
CTL$AL_CLICALBK                   ******** X      02
CTL$A_DISPVEC                     ******** X      02
CTL$C_KRP_COUNT                   ******** X      02
CTL$C_KRP_SIZE                    ******** X      02
CTL$GB_MSGMASK                    ******** X      02
CTL$GB_SSFILTER                   ******** X      02
CTL$GL_CCBBASE                    ******** X      02
CTL$GL_CREPRC_FLAGS               ******** X      02
CTL$GL_CTLBASVA                   ******** X      02
CTL$GL_GETMSG                     ******** X      02
CTL$GL_IAFLAST                    ******** X      02
CTL$GL_IAFLINK                    ******** X      02
CTL$GL_IAFPERM                    ******** X      02
CTL$GL_KRP                        ******** X      02
CTL$GL_KRPFL                      ******** X      02
CTL$GL_LNMDIRECT                  ******** X      02
CTL$GL_LNMHASH                    ******** X      02
CTL$GL_PCB                        ******** X      02
CTL$GL_PHD                        ******** X      02
CTL$GL_PRCALLCNT                  ******** X      02
CTL$GL_RMSBASE                    ******** X      02
CTL$GL_UAF_FLAGS                  ******** X      02
CTL$GL_USRCHME                    ******** X      02
CTL$GL_USRCHMK                    ******** X      02
CTL$GL_USRUNDWN                   ******** X      02
CTL$GQ_ALLOCREG                   ******** X      02
CTL$GQ_LNMTBLCACHE                ******** X      02
CTL$GQ_LOGIN                      ******** X      02
CTL$GQ_PROCPRIV                   ******** X      02
CTL$GT_CLINAME                    ******** X      02
CTL$GW_NMIOCH                     ******** X      02
CTL$T_USERNAME                    ******** X      02
DEFAULTNAMDSC                     0000003F R      02
DEFDESC                           00000010 R      02
DIR...                          = 00000001
DIVR                              00000AE2 R      02
DYN$C_LNM                       = 00000040
DYN$C_ORB                       = 00000049
DYN$C_RSHT                      = 00000038
EXE$ALOP1PROC                     ******** X      02
EXE$ALOPAGED                      ******** X      02
EXE$CATCH_ALL                     00000E80 RG     02
EXE$CLI_UTILSRV                   00000E69 RG     02
EXE$CRE_GTABLE                    00001137 RG     02
EXE$CRE_JGTABLE                   0000118D RG     02
EXE$C_SYSEFN                      ******** X      02
EXE$DEAP1                         ******** X      02
EXE$DEAPAGED                      ******** X      02
EXE$DEAPGDSIZ                     ******** X      02
EXE$EXCEPTABLE                    ******** X      02
EXE$EXCMSG                        ******** X      02
EXE$EXIT_IMAGE                    00000E73 RG     02
EXE$GL_FLAGS                      ******** X      02
EXE$GL_PQBBL                      ******** X      02
EXE$GQ_SYSDISK                    00000000 RG     02
EXE$GQ_SYSTIME                    ******** X      02
EXE$IMGDMP_EXEC                   00000FE7 RG     02
EXE$IMGDMP_MERGE                  00000FED RG     02
EXE$PROCIMGACT                    00000D88 RG     02
EXE$PROCSTRT                      00000000 RG     03
EXE$RMSEXH                        00000F25 RG     02
EXE$V_INIT                        ******** X      02
EXEC_A                            00000FFD R      02
EXE_PROCSTRT                      000008E0 R      02
GROUP                           = 000000E0
GROUP_SIZE                      = 00000038
GROUP_TABLE                       00000760 R      02
GROUP_TABLE_LNMTH               = 00000027
GROUP_TABLE_ORB                 = 00000050
GROUP_TABLE_ORB_SIZ             = 00000070
GROUP_TABLE_SIZE                = 000000C0
GROUP_XEND_SIZE                 = 00000031
HDRBUF                            00000034
IAC$AW_VECRESET                   ******** X      02
IAC$AW_VECSET                     ******** X      02
IAC$GL_ICBFL                      ******** X      02
IAC$GL_IMAGE_LIST                 ******** X      02
IAC$GL_WORK_LIST                  ******** X      02
IAC$M_EXPREG                    = 00000020
IAC$M_MERGE                     = 00000010
IHD$L_LNKFLAGS                  = 00000020
IHD$W_ACTIVOFF                  = 00000002
IMAGPRIV                          0000023C
IMGACT$_ACMODE                  = 00000020
IMGACT$_DFLNAM                  = 00000008
IMGACT$_HDRBUF                  = 0000000C
IMGACT$_IDENT                   = 0000001C
IMGACT$_IMGCTL                  = 00000010
IMGACT$_INADR                   = 00000014
IMGACT$_NAME                    = 00000004
IMGACT$_NARGS                   = 00000008
IMGACT$_RETADR                  = 00000018
IMGACT_INADR                      00000024
IMGACT_RETADR                     0000002C
IMGDMPNAM                         00000057 R      02
IMGNAM                            00000CEE R      02
IMP$L_IOSEGADDR                 = 00000004
JIB$S_ACCOUNT                   = 00000008
JIB$S_USERNAME                  = 0000000C
JIB$T_ACCOUNT                   = 00000018
JIB$T_USERNAME                  = 0000000C
JOB                             = 00000118
JOB_SIZE                        = 00000030
```

```
JOB_TABLE                       = 000000C0
JOB_TABLE_LNMTH                 = 000000E7
JOB_TABLE_ORB                   = 00000110
JOB_TABLE_ORB_SIZ               = 00000070
JOB_TABLE_SIZE                  = 000000C0
JOB_XEND_SIZE                   = 0000002F
JPI$_IMAGPRIV                   = 00000413
JPI$_PHDFLAGS                   = 0000041B
JPI$_PROCPRIV                   = 00000204
JPI_END                           0000026C
JPI_FLAG                          00000260
JPI_IMAG                          00000254
JPI_PROC                          00000248
LNM$AL_HASHTBL                    ********    X   02
LNM$GL_HTBLSIZP                   ********    X   02
LNM$HASH                          ********    X   02
LNM$INSLOGTAB                     ********    X   02
LNM$LOCKW                         ********    X   02
LNM$M_CREATE_IF                 = 01000000
LNM$UNLOCK                        ********    X   02
LNMB$B_ACMODE                   = 0000000B
LNMB$B_FLAGS                    = 00000010
LNMB$B_TYPE                     = 0000000A
LNMB$L_BLINK                    = 00000004
LNMB$L_FLINK                    = 00000000
LNMB$L_TABLE                    = 0000000C
LNMB$M_NODELETE                 = 00000010
LNMB$M_NO_ALIAS                 = 00000001
LNMB$M_TABLE                    = 00000008
LNMB$T_NAME                     = 00000011
LNMB$W_SIZE                     = 00000008
LNMC$K_LENGTH                   = 00000080
LNMC$L_TBLADDR                  = 0000000C
LNMC$W_SIZE                     = 00000008
LNMHSH$B_TYPE                   = 0000000A
LNMHSH$C_BUCKET                 = 0000000C
LNMHSH$K_BUCKET                 = 0000000C
LNMHSH$L_MASK                   = 00000000
LNMHSH$W_SIZE                   = 00000008
LNMTH$B_FLAGS                   = 00000000
LNMTH$K_LENGTH                  = 00000025
LNMTH$L_BYTES                   = 00000021
LNMTH$L_BYTESLM                 = 0000001D
LNMTH$L_CHILD                   = 00000011
LNMTH$L_HASH                    = 00000001
LNMTH$L_NAME                    = 00000009
LNMTH$L_ORB                     = 00000005
LNMTH$L_PARENT                  = 0000000D
LNMTH$L_QTABLE                  = 00000019
LNMTH$L_SIBLING                 = 00000015
LNMTH$M_DIRECTORY               = 00000002
LNMTH$M_GROUP                   = 00000004
LNMTH$M_SHAREABLE               = 00000001
LNMX$B_FLAGS                    = 00000000
LNMX$B_INDEX                    = 00000001
LNMX$C_TABLE                    = FFFFFF82
LNMX$M_TERMINAL                 = 00000002
LNMX$M_XEND                     = 00000004
LNMX$T_XLATION                  = 00000004
LNMX$W_HASH                     = 00000002
LNM_SYSTEM_DIR_LNMTH              ********    X   02
MMG$CRETVA                        ********    X   02
MMG$GL_CTLBASVA                   ********    X   02
MMG$GL_RMSBASE                    ********    X   02
MMG$IMGHDRBUF                     ********    X   02
MMG$IMGRESET                      ********    X   02
NXTEVEC                         = 00000100
NXTKVEC                         = 00000000
NXTMVEC                         = 00000300
NXTRVEC                         = 00000200
OP$_RSB                         = 00000005
ORB$B_FLAGS                     = 0000000B
ORB$B_TYPE                      = 0000000A
ORB$K_LENGTH                    = 00000058
ORB$L_ACL_COUNT                 = 00000028
ORB$L_ACL_DESC                  = 0000002C
ORB$L_ACL_MUTEX                 = 00000004
ORB$L_GRP_PROT                  = 00000020
ORB$L_OWNER                     = 00000000
ORB$L_OWN_PROT                  = 0000001C
ORB$L_SYS_PROT                  = 00000018
ORB$L_WOR_PROT                  = 00000024
ORB$Q_MODE_PROT                 = 00000010
ORB$R_MAX_CLASS                 = 00000044
ORB$R_MIN_CLASS                 = 00000030
ORB$S_MAX_CLASS                 = 00000014
ORB$S_MIN_CLASS                 = 00000014
ORB$W_REFCOUNT                  = 0000000E
ORB$W_SIZE                      = 00000008
P1_ALLOC_SIZE                   = 000006F0
PCB$B_AUTHPRI                   = 0000002B
PCB$B_PRIB                      = 0000002F
PCB$L_JIB                       = 00000080
PCB$L_OWNER                     = 0000001C
PCB$L_PHD                       = 0000006C
PCB$L_PQB                       = 0000004C
PCB$L_STS                       = 00000024
PCB$L_UIC                       = 000000BC
PCB$V_HIBER                     = 00000013
PCB$W_GRP                       = 000000BE
PFN$GL_PHYPGCNT                   ********    X   02
PHD$B_AUTHPRI                   = 0000010C
PHD$L_CPULIM                    = 0000005C
PHD$M_IMGDMP                    = 00000020
PHD$Q_AUTHPRIV                  = 000000E0
PHD$Q_PRIVMSK                   = 00000000
PHD$R_MAX_CLASS                 = 00000128
PHD$R_MIN_CLASS                 = 00000114
PHD$S_MAX_CLASS                 = 00000014
PHD$S_MIN_CLASS                 = 00000014
PHD$V_IMGDMP                    = 00000005
PHD$W_ASTLM                     = 00000040
PHD$W_DFWSCNT                   = 0000001A
PHD$W_FLAGS                     = 00000036
```

```
PHD$W_WSAUTH                    =  0000000A
PHD$W_WSAUTHEXT                 =  00000014
PHD$W_WSEXTENT                  =  00000016
PHD$W_WSLIST                    =  00000008
PHD$W_WSQUOTA                   =  00000018
PHD_FLAGS                          00000244
PIO$AL_RMSEXH                      ********    X   02
PIO$GQ_IIODEFAULT                  ********    X   02
PIO$GT_DDSTRING                    ********    X   02
PIO$GW_PIOIMPA                     ********    X   02
PQB$B_MSGMASK                   =  00000046
PQB$L_ASTLM                     =  0000000C
PQB$L_CPULM                     =  00000018
PQB$L_CREPRC_FLAGS              =  0000004C
PQB$L_DISK_ATT                  =  00000084
PQB$L_ERROR_ATT                 =  00000080
PQB$L_INPUT_ATT                 =  00000078
PQB$L_JTQUOTA                   =  00000040
PQB$L_OUTPUT_ATT                =  0000007C
PQB$L_UAF_FLAGS                 =  00000048
PQB$L_WSDEFAULT                 =  00000034
PQB$L_WSEXTENT                  =  0000003C
PQB$L_WSQUOTA                   =  00000030
PQB$Q_PRVMSK                    =  00000000
PQB$R_MAX_CLASS                 =  00000064
PQB$R_MIN_CLASS                 =  00000050
PQB$S_CLI_NAME                  =  00000020
PQB$S_CLI_TABLE                 =  00000100
PQB$S_DDSTRING                  =  00000100
PQB$S_DISK                      =  00000100
PQB$S_ERROR                     =  00000100
PQB$S_INPUT                     =  00000100
PQB$S_MAX_CLASS                 =  00000014
PQB$S_MIN_CLASS                 =  00000014
PQB$S_OUTPUT                    =  00000100
PQB$S_SPAWN_CLI                 =  00000020
PQB$S_SPAWN_TABLE               =  00000100
PQB$T_CLI_NAME                  =  00000088
PQB$T_CLI_TABLE                 =  000000A8
PQB$T_DDSTRING                  =  000006C8
PQB$T_DISK                      =  000005C8
PQB$T_ERROR                     =  000004C8
PQB$T_IMAGE                     =  000007C8
PQB$T_INPUT                     =  000002C8
PQB$T_OUTPUT                    =  000003C8
PQB$T_SPAWN_CLI                 =  000001A8
PQB$T_SPAWN_TABLE               =  000001C8
PQB$V_IMGDMP                    =  00000000
PQB$W_FLAGS                     =  00000044
PR$_IPL                         =  00000012
PROCESS                         =  000000A8
PROCESS_SIZE                    =  00000038
PROCPRIV                           00000234
PROC_DIR                           00000068 R      02
PROC_DIR_LNMTH                  =  0000002C
PROC_DIR_SIZE                   =  00000058
PROC_TABLE                      =  00000058
PROC_TABLE_LNMTH                =  00000080
PROC_TABLE_SIZE                 =  00000050
PRT$C_UREW                      =  0000000D
PRV$V_CMKRNL                    =  00000000
PRV$V_SETPRV                    =  0000000E
PSL$C_EXEC                      =  00000001
PSL$C_KERNEL                    =  00000000
PSL$C_USER                      =  00000003
PSL$S_CURMOD                    =  00000002
PSL$V_CURMOD                    =  00000018
PSL$V_PRVMOD                    =  00000016
RMS$_BUSY                       =  0001848C
SO_ALLOC_SIZE                   =  00000180
SAVABS...                       =  00000270
SCH$GL_CURPCB                      ********    X   03
SCH$GL_FREELIM                     ********    X   02
SCRATCHSIZE                        00000270
SEC$M_EXPREG                    =  00020000
SEC$M_SYSGBL                    =  00008000
SGN$GL_MAXWSCNT                    ********    X   02
SGN$GW_CTLIMGLIM                   ********    X   02
SGN$GW_CTLPAGES                    ********    X   02
SGN$GW_IMGIOCNT                    ********    X   02
SGN$GW_PCHANCNT                    ********    X   02
SGN$GW_PIOPAGES                    ********    X   02
SS$_CONTINUE                    =  00000001
SS$_EXLNMQUOTA                  =  0000224C
SS$_INSFMEM                     =  00000124
SS$_NORMAL                      =  00000001
SS$_SSFAIL                      =  0000045C
STS$K_SEVERE                    =  00000004
STS$S_CODE                      =  0000000C
STS$S_SEVERITY                  =  00000003
STS$V_CODE                      =  00000003
STS$V_INHIB_MSG                 =  0000001C
STS$V_SEVERITY                  =  00000000
SUFFIX                             0000002C R      02
SYS$CMKRNL                         ********  GX    02
SYS$DCLEXH                         ********  GX    02
SYS$DISK                        =  000005D0
SYS$DISK_LNMX                   =  000005EA
SYS$DISK_SIZE                   =  00000120
SYS$ERROR                       =  000004B0
SYS$ERROR_LNMX                  =  000004CB
SYS$ERROR_SIZE                  =  00000120
SYS$EXIT                           ********   X    02
SYS$EXPREG                         ********  GX    02
SYS$GETJPI                         ********  GX    02
SYS$HIBER                          ********  GX    02
SYS$IMGACT                         ********  GX    02
SYS$IMGFIX                         ********  GX    02
SYS$INPUT                       =  00000148
SYS$INPUT_LNMX                  =  00000163
SYS$INPUT_SIZE                  =  00000120
SYS$MGBLSC                         ********  GX    02
SYS$OUTPUT                      =  00000388
SYS$OUTPUT_LNMX                 =  000003A4
```

```
SYS$OUTPUT_SIZE                 = 00000128
SYS$PUTMSG                        ********   X   02
SYS$RMSRUNDWN                     ********   X   02
SYS$SETEXV                        ********  GX   02
SYS$SETSFM                        ********  GX   02
TT                              = 00000268
TT_LNMX                         = 0000027C
TT_SIZE                         = 00000120
VABUG                             00000AD5 R     02
XQP$GL_DZRO                       ********   X   02
XQP$GL_SECTIONS                   ********   X   02
XQPMERGE                          00000F4C R     02
XQP_NAM                           00000FDD R     02
XQP_NAMSIZ                      = 0000000A
```

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	00000270 (624.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
YYPROCSTRT	00001274 (4724.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC 21
AEXENONPAGED	00000011 (17.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.06	00:00:00.70
Command processing	112	00:00:00.50	00:00:03.78
Pass 1	601	00:00:27.11	00:01:35.27
Symbol table sort	0	00:00:03.87	00:00:10.29
Pass 2	363	00:00:06.72	00:00:23.84
Symbol table output	42	00:00:00.31	00:00:01.35
Psect synopsis output	2	00:00:00.03	00:00:00.18
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1152	00:00:38.61	00:02:15.41

```
The working set limit was 2250 pages.
154901 bytes (303 pages) of virtual memory were used to buffer the intermediate code.
There were 130 pages of symbol table space allocated to hold 2571 non-local and 54 local symbols.
1894 source lines were read in Pass 1, producing 37 object records in Pass 2.
54 pages of virtual memory were used to define 52 macros.
```

```
                                    +----------------------------+
                                    ! Macro library statistics !
                                    +----------------------------+

Macro library name                   Macros defined
------------------                   --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                12
_$255$DUA28:[SYSLIB]STARLET.MLB;2             36
TOTALS (all libraries)                        48

2738 GETS were required to define 48 macros.

There were no errors, warnings or information messages.

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

0379 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RELOCDRV
LIS

RMSRESET
LIS

PROCSTRT
LIS

PHDUTL
LIS

PMSDAT
LIS

POSTEF
LIS

POWERFAIL
LIS

PRDEF
LIS

PTEDUMP
LIS

PDAT
LIS

RMSVECTOR
LIS