```
RRRRRRRRRRR      MMM          MMM      SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRRRRRRRRRR      MMM          MMM      SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSS
RRR    RRR       MMM          MMM               SSS
RRR    RRR       MMM          MMM               SSS
RRR    RRR       MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR       RRR    MMM          MMM               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
```

FILEID**NTOCREATE

Table of contents

```
0000     1          $BEGIN  NTOCREATE,000,NF$NETWORK,<NETWORK CREATE FILE>
0000     2
0000     3  ;
0000     4  ;*****************************************************************************
0000     5  ;*                                                                           *
0000     6  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8  ;*  ALL RIGHTS RESERVED.                                                     *
0000     9  ;*                                                                           *
0000    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    11  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    13  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    14  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    15  ;*  TRANSFERRED.                                                              *
0000    16  ;*                                                                           *
0000    17  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    18  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    19  ;*  CORPORATION.                                                              *
0000    20  ;*                                                                           *
0000    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
0000    23  ;*                                                                           *
0000    24  ;*                                                                           *
0000    25  ;*****************************************************************************
0000    26  ;
```

```
0000     28 ;++
0000     29 ; Facility: RMS
0000     30 ;
0000     31 ; Abstract:
0000     32 ;
0000     33 ;       This module communicates with the File Access Listener (FAL) at the
0000     34 ;       remote node to create the specified file.
0000     35 ;
0000     36 ; Environment: VAX/VMS, executive mode
0000     37 ;
0000     38 ; Author: James A. Krycka,      Creation Date:  09-DEC-1977
0000     39 ;
0000     40 ; Modified By:
0000     41 ;
0000     42 ;       V03-011 JAK0142         J A Krycka      10-APR-1984
0000     43 ;               Fix inconsistency in handling MRS value when dealing with a
0000     44 ;               remote FAL that uses a stream-based file system.
0000     45 ;
0000     46 ;       V03-011 JAK0138         J A Krycka      28-MAR-1984
0000     47 ;               Call modified NT$EXCH_CNF routine with a parameter.
0000     48 ;               General cleanup.
0000     49 ;
0000     50 ;       V03-010 JAK0124         J A Krycka      06-SEP-1983
0000     51 ;               Make corresponding source code change for VMS V3.5 patch in
0000     52 ;               support of VAXELAN.
0000     53 ;
0000     54 ;       V03-009 JAK0119         J A Krycka      16-JUL-1983
0000     55 ;               Return an error if any journaling options are requested.
0000     56 ;
0000     57 ;       V03-008 KPL0001         Peter Lieberwirth       23-May-1983
0000     58 ;               Fix assembly error.
0000     59 ;
0000     60 ;       V03-007 KRM0088         Karl Malik      18-Mar-1983
0000     61 ;               Add support for STMCR and STMLF file formats. Also, removed
0000     62 ;               use of DAP$V_STM bit using NWA$B_RFM field instead.
0000     63 ;
0000     64 ;       V03-006 KRM0078         Karl Malik      10-Dec-1982
0000     65 ;               Add status check after call to RM$SETALLOC in NT$CREATE.
0000     66 ;
0000     67 ;       V03-005 KBT0410         Keith B. Thompson       30-Nov-1982
0000     68 ;               Change IFB$W_DEVBUFSIZ to IFB$L_DEVBUFSIZ.
0000     69 ;
0000     70 ;--
```

```
0000     72          .SBTTL  DECLARATIONS
0000     73
0000     74 ;
0000     75 ; Include Files:
0000     76 ;
0000     77
0000     78          $DAPPLGDEF                      ; Define DAP prologue symbols
0000     79          $DAPHDRDEF                      ; Define DAP message header
0000     80          $DAPCNFDEF                      ; Define DAP Configuration message
0000     81          $DAPATTDEF                      ; Define DAP Attributes message
0000     82          $DAPACCDEF                      ; Define DAP Access message
0000     83          $FABDEF                         ; Define File Access Block symbols
0000     84          $IFBDEF                         ; Define IFAB symbols
0000     85          $NWADEF                         ; Define Network Work Area symbols
0000     86          $RMSDEF                         ; Define RMS completion codes
0000     87          $XABDEF                         ; Define symbols common to all XABs
0000     88          $XABALLDEF                      ; Define Allocation XAB symbols
0000     89          $XABFHCDEF                      ; Define File Header Char XAB symbols
0000     90
0000     91 ;
0000     92 ; Macros:
0000     93 ;
0000     94 ;        None
0000     95 ;
0000     96 ; Equated Symbols:
0000     97 ;
0000     98
0000     99          ASSUME  DAP$Q_DCODE_FLG EQ 0
0000    100          ASSUME  NWA$Q_FLG EQ 0
0000    101
0000    102 ;
0000    103 ; Own Storage:
0000    104 ;
0000    105 ;        None
0000    106 ;
```

```
0000    108          .SBTTL  NT$CREATE - PERFORM NETWORK CREATE FUNCTION
0000    109
0000    110 ;++
0000    111 ; NT$CREATE - engages in a DAP dialogue with the remote FAL to create the
0000    112 ;       specified sequential, relative, or indexed file.
0000    113 ;
0000    114 ; Calling Sequence:
0000    115 ;
0000    116 ;       BSBW    NT$CREATE
0000    117 ;
0000    118 ; Input Parameters:
0000    119 ;
0000    120 ;       R8      FAB address
0000    121 ;       R9      IFAB address
0000    122 ;       R10     FWA address
0000    123 ;       R11     Impure Area address
0000    124 ;
0000    125 ; Implicit Inputs:
0000    126 ;
0000    127 ;       User FAB
0000    128 ;       User ALL, DAT, FHC, JNL, KEY, PRO, and RDT XABs
0000    129 ;       DAP$V_GEQ_V54, DAP$V_GEQ_V56
0000    130 ;       DAP$V_VAXVMS, DAP$V_VAXELAN
0000    131 ;       DAP fields
0000    132 ;       IFB$B_FAC
0000    133 ;       IFB$L_NWA_PTR
0000    134 ;       IFB$L_DEVBUFSIZ
0000    135 ;       IFB$V_CREATE
0000    136 ;       IFB fields
0000    137 ;       NWA$B_RFM
0000    138 ;       NWA$V_CVT_STM
0000    139 ;
0000    140 ; Output Parameters:
0000    141 ;
0000    142 ;       R0      Status code (RMS)
0000    143 ;       R1-R7   Destroyed
0000    144 ;       AP      Destroyed
0000    145 ;
0000    146 ; Implicit Outputs:
0000    147 ;
0000    148 ;       User FAB
0000    149 ;       User ALLXABs
0000    150 ;       IFB$B_BKS
0000    151 ;       IFB$L_HBK
0000    152 ;       IFB$V_DAP_OPEN
0000    153 ;       Resultant name string stored in FWA
0000    154 ;
0000    155 ; Completion Codes:
0000    156 ;
0000    157 ;       Standard RMS completion codes
0000    158 ;
0000    159 ; Side Effects:
0000    160 ;
0000    161 ;       None
0000    162 ;
0000    163 ;--
0000    164
```

```
                                0000   165 NT$CREATE::                                 ; Entry point
                                0000   166         $TSTPT  NTCREATE                    ;
                     57   3C A9  D0  0006   167         MOVL    IFB$L_NWA_PTR(R9),R7    ; Get address of NWA (and DAP)
                                000A   168
                                000A   169 ;+
                                000A   170 ; Perform the following FOP field processing:
                                000A   171 ;   (1) disallow the FOP options NFS and UFO.
                                000A   172 ;   (2) ignore the FOP options CIF and DFW without returning an error as CIF is
                                000A   173 ;       simulated via open/create and DFW is an unsupported performance option.
                                000A   174 ;-
                                000A   175
           04 A8  00030000 8F   D3  000A   176         BITL    #<<FAB$M_NFS>!-         ; Branch if none are specified;
                                0012   177                   <FAB$M_UFO>!-           ;  otherwise declare these options
                                0012   178                   0>,FAB$L_FOP(R8)        ;  unsupported over the network
                          03    13  0012   179         BEQL    10$                     ;
                        FFE9'   31  0014   180         BRW     NT$LCL_FOP              ; Return RMS$_SUPPORT error and
                                0017   181                                           ;  exit with RMS code in R0
                                0017   182
                                0017   183 ;+
                                0017   184 ; Exchange DAP Configuration messages with FAL and determine DAP buffer size.
                                0017   185 ;-
                                0017   186
                     50   02    D0  0017   187 10$:    MOVL    #DAP$K_CREATE,R0        ; Denote type of file access
                        FFE3'   30  001A   188         BSBW    NT$EXCH_CNF             ; Exchange Configuration messages
                      06 50     E9  001D   189         BLBC    R0,FAIL1                ; Branch on failure
                       02EF     30  0020   190         BSBW    NT$CHK_ORG              ; Check for supported organization
                      01 50     E8  0023   191         BLBS    R0,BUILD_MASK           ; Branch on success
                                05  0026   192 FAIL1:  RSB                             ; Exit with RMS code in R0
                                0027   193
                                0027   194 ;+
                                0027   195 ; First, build a mask (NWA$W_BUILD) that will determine which optional DAP
                                0027   196 ; messages to send to FAL. For $CREATE, these are the ALL, KEY, TIM  and PRO
                                0027   197 ; messages.
                                0027   198 ;
                                0027   199 ; Second, build a request mask (NWA$W_DISPLAY) that will be used in the Access
                                0027   200 ; message to request that optional DAP messages be returned by FAL. For $CREATE,
                                0027   201 ; these are the ALL and NAM messages. (Note that the Attributes message is
                                0027   202 ; required which will supply information to update the FAB.)
                                0027   203 ;
                                0027   204 ; Third, return an 'unsupported over network' error if any journaling options
                                0027   205 ; are requested.
                                0027   206 ;-
                                0027   207
                                0027   208 BUILD_MASK:                                 ; Build NWA$W_DISPLAY
                          56    D4  0027   209         CLRL    R6                      ; Indicate this is not a  'ose operation
                        FFD4'   30  0029   210         BSBW    NT$SCAN_XABCHN          ; Scan user XAB chain and check FAL's
                                002C   211                                           ;  capabilities; build request mask
                      F7 50     E9  002C   212         BLBC    R0,FAIL1                ; Branch on failure to complete scan
                   52   08     AA  002F   213         BICW2   #DAP$M_DSP_SUM,R2       ; Ignore Summary XAB in chain
                20   1D A8     91  0032   214         CMPB    FAB$B_ORG(R8),#FAB$C_IDX; Branch if IDX organization
                          03    13  0036   215         BEQL    10$                     ;
                   52   02     AA  0038   216         BICW2   #DAP$M_DSP_KEY,R2       ; Ignore Key Definition XAB in chain
              00D2 C7   52     B0  003B   217 10$:    MOVW    R2,NWA$W_BUILD(R7)      ; Save build mask
                                0040   218
                   52   32     AA  0040   219         BICW2   #<<DAP$M_DSP_PRO>!-     ; Ignore these XABs in chain because
                                0043   220                   <DAP$M_DSP_TIM>!-       ;  they are not updated on create
                                0043   221                   <DAP$M_DSP_KEY>!-       ;
```

```
                     0043   222                     0>,R2                           ;
          FFBA'  30  0043   223          BSBW     NT$SCAN_NAMBLK                  ; Scan user Name Block and check FAL's
                     0046   224                                                  ;  capabilities; update request mask
        DD 50  E9    0046   225          BLBC     R0,FAIL1                        ; Branch on failure to complete scan
    52  01  A8    0049   226          BISW2    #DAP$M_DSP_ATT,R2               ; Request main Attributes message
00D0 C7  52  B0    004C   227          MOVW     R2,NWA$W_DISPLAY(R7)            ; Save request mask
                     0051   228
     011E C7  B5    0051   229          TSTW     NWA$W_JNLXABJOP(R7)             ; Ok, if no journaling options have
          03  13    0055   230          BEQL     OVERRIDE_FAB                    ;  been requested
          FFA6'  31  0057   231          BRW      NT$LCL_JOP                      ; Declare RMS$_SUPPORT error and exit
                     005A   232
                     005A   233 ;+
                     005A   234 ; Override the ALQ, BKS, DEQ, and FOP fields of the FAB with corresponding
                     005A   235 ; ALQ, BKZ, DEQ, and AOP fields of the Allocation XAB (area 0 for index files),
                     005A   236 ; if one is present.
                     005A   237 ;-
                     005A   238
                     005A   239 OVERRIDE_FAB:                                    ; Update FAB from ALLXAB
 00000000'EF    16   005A   240          JSB      RM$SETALLOC                     ; Update FAB with ALQ, BKS, and DEQ
                     0060   241                                                  ;  fields from Allocation XAB (if
                     0060   242                                                  ;  present) and set-up default DEQ
        C3 50  E9    0060   243          BLBC     R0,FAIL1                        ; Branch on failure to update FAB
  20    1D A8  91    0063   244          CMPB     FAB$B_ORG(R8),#FAB$C_IDX        ; Branch if IDX organization
          29  13    0067   245          BEQL     SEND_ATT                        ;
56   0100 C7  D0    0069   246          MOVL     NWA$L_ALLXABADR(R7),R6          ; Get address of ALLXAB
          22  13    006E   247          BEQL     SEND_ATT                        ; Branch if not found
     08 A6  9A    0070   248          MOVZBL   XAB$B_AOP(R6),R1                ; Get AOP bits
          52  D4    0074   249          CLRL     R2                              ; Clear resultant FOP bits
                     0076   250          $MAPBIT  XAB$V_CTG,FAB$V_CTG             ; Map CTG bit
                     007E   251          $MAPBIT  XAB$V_CBT,FAB$V_CBT             ; Map CBT bit
04 A8    00300000 8F    CA   0086   252          BICL2    #<<FAB$M_CTG>!-         ; Clear affected FOP bits
                     008E   253                            <FAB$M_CBT>!-         ;
                     008E   254                            0>,FAB$L_FOP(R8)      ;
     04 A8    52    C8   008E   255          BISL2    R2,FAB$L_FOP(R8)            ; Update FOP from AOP
```

```
                     0092   257 ;+
                     0092   258 ; Build and send DAP Attributes message to partner.
                     0092   259 ;
                     0092   260 ; If the remote node is not a 'stream only' machine then create a file with
                     0092   261 ; the file attributes specified by the user.
                     0092   262 ; If, however, the remote node is a 'stream only' machine then:
                     0092   263 ;   (1) for block I/O mode accept any sequential file but specify to FAL that
                     0092   264 ;       ORG = SEQ, RFM = FIX, RAT = 0, MRS = 512, and ALQ = user value.
                     0092   265 ;   (2) for record I/O mode accept a sequential file having stream-embedded
                     0092   266 ;       format, or convert a sequential file in fixed-implied, variable-implied,
                     0092   267 ;       or stream-implied format to stream-embedded format by specifying that
                     0092   268 ;       ORG = SEQ, RFM = STM, RAT = EMBEDDED, MRS = 514, and ALQ = 0.
                     0092   269 ;-
                     0092   270
                     0092   271 SEND_ATT:                                    ; (required message)
50   02   D0        0092   272         MOVL    #DAP$K_ATT_MSG,R0             ; Get message type value
   FF68'  30        0095   273         BSBW    NT$BUILD_HEAD                 ; Construct message header
                     0098   274
                     0098   275         ASSUME  DAP$K_SEQ EQ FAB$C_SEQ
                     0098   276         ASSUME  DAP$K_REL EQ FAB$C_REL
                     0098   277         ASSUME  DAP$K_IDX EQ FAB$C_IDX
                     0098   278
                     0098   279         ASSUME  DAP$K_UDF EQ 0
                     0098   280         ASSUME  DAP$K_FIX EQ 1
                     0098   281         ASSUME  DAP$K_VAR EQ 2
                     0098   282         ASSUME  DAP$K_VFC EQ 3
                     0098   283         ASSUME  DAP$K_STM EQ 4
                     0098   284
                     0098   285         ASSUME  DAP$K_UDF EQ FAB$C_UDF
                     0098   286         ASSUME  DAP$K_FIX EQ FAB$C_FIX
                     0098   287         ASSUME  DAP$K_VAR EQ FAB$C_VAR
                     0098   288         ASSUME  DAP$K_VFC EQ FAB$C_VFC
                     0098   289         ASSUME  DAP$K_STM EQ FAB$C_STM
                     0098   290         ASSUME  DAP$K_STMLF EQ FAB$C_STMLF
                     0098   291         ASSUME  DAP$K_STMCR EQ FAB$C_STMCR
                     0098   292
                     0098   293         ASSUME  DAP$V_FTN EQ FAB$V_FTN
                     0098   294         ASSUME  DAP$V_CR  EQ FAB$V_CR
                     0098   295         ASSUME  DAP$V_PRN EQ FAB$V_PRN
                     0098   296         ASSUME  DAP$V_BLK EQ FAB$V_BLK
                     0098   297
                     0098   298 ;
                     0098   299 ; Construct attributes menu mask.
                     0098   300 ;
                     0098   301
51   0000106F 8F     D0    0098   302 PART1:  MOVL    #<<DAP$M_DATATYPE>!-          ; Always include DATATYPE, ORG, RFM,
                     009F   303                   <DAP$M_ORG>!-                 ;  RAT, MRS, ALQ, and FOP fields in
                     009F   304                   <DAP$M_RFM>!-                 ;  mask
                     009F   305                   <DAP$M_RAT>!-                 ;
                     009F   306                   <DAP$M_MRS>!-                 ;
                     009F   307                   <DAP$M_ALQ1>!-                ;
                     009F   308                   <DAP$M_FOP1>!-                ;
                     009F   309                   0>,R1                         ;
     00   1D A8   91    009F   310         CMPB    FAB$B_ORG(R8),#FAB$C_SEQ      ; Branch if SEQ organization
          0E   13    00A3   311         BEQL    10$                           ; If not, it's REL or IDX organization
                     00A5   312         $SETBIT #DAP$V_BKS,R1                 ; Add BKS fiels to mask
     20   1D A8   91    00A9   313         CMPB    FAB$B_ORG(R8),#FAB$C_IDX      ; Branch if IDX organization
```

```
                04   13  00AD   314          BEQL    10$                     ; If not, it's REL organization
                         00AF   315          $SETBIT #DAP$V_MRN,R1           ; Add MRN field to mask
                         00B3   316 10$:      $CASEB  SELECTOR=FAB$B_RFM(R8)- ; Record format:
                         00B3   317                  DISPL=<-                ;
                         00B3   318                      40$-                ;  UDF
                         00B3   319                      50$-                ;  FIX
                         00B3   320                      40$-                ;  VAR
                         00B3   321                      30$-                ;  VFC
                         00B3   322                      40$-                ;  STM
                         00B3   323                      20$-                ;  SRMLF (first defined in DAP V7.0)
                         00B3   324                      20$-                ;  STMCR (first defined in DAP V7.0)
                         00B3   325                  >                       ;
            FF37'        31   00C6   326          BRW     NT$LCL_RFM          ; Declare RMS$_SUPPORT error and exit
      07 67   26         E0   00C9   327 20$:      BBS     #DAP$V_GEQ_V70,(R7),40$ ; Branch if partner uses DAP since V7.0
            FF30'        31   00CD   328          BRW     NT$RMT_RFM          ; Declare RMS$_SUPPORT error and exit
                         00D0   329 30$:      $SETBIT #DAP$V_FSZ,R1           ; Add FSZ field to mask
   0A 67   23         E1   00D4   330 40$:      BBC     #DAP$V_GEQ_V54,(R7),50$ ; Branch if partner uses DAP before V5.4
         0108 C7      D5   00D8   331          TSTL    NWA$L_FHCXABADR(R7)     ; Branch if user did not specify a
                04   13  00DC   332          BEQL    50$                     ;  FHCXAB
                         00DE   333          $SETBIT #DAP$V_LRL,R1           ; Add LRL field to mask
              14 A8      B5   00E2   334 50$:      TSTW    FAB$W_DEQ(R8)       ; Branch if DEQ = 0
                04   13  00E5   335          BEQL    60$                     ;
                         00E7   336          $SETBIT #DAP$V_DEQ1,R1          ; Add DEQ field to mask
           56    51      D0   00EB   337 60$:      MOVL    R1,R6               ; Save attributes menu
              FF0F'        30   00EE   338          BSBW    NT$CVT_BN4_EXT      ; Store attmenu as an extensible field
                         00F1   339
                         00F1   340 ;
                         00F1   341 ; Store remaining attributes fields per attmenu mask.
                         00F1   342 ;
                         00F1   343
         67    32      E1   00F1   344          BBC     #DAP$V_STM_ONLY,(R7),-  ; Branch if not stream-based remote
                63         00F4   345                  PART2B                  ;  file system
                         00F5   346
                         00F5   347 ;
                         00F5   348 ; The remote node is using a stream-based file system.
                         00F5   349 ; Assume that ORG = SEQ since the NT$CHK_ORG test was passed.
                         00F5   350 ;
                         00F5   351 ; This section deals with the DATATYPE, ORG, RFM, RAT, MRS, and ALQ fields.
                         00F5   352 ;
                         00F5   353
                05   E0  00F5   354 PART2A: BBS      #FAB$V_BIO,-            ; Branch if block I/O mode
      45 16 A8           00F7   355                  FAB$B_FAC(R8),50$       ;
                         00FA   356
                         00FA   357 ;
                         00FA   358 ; Based on RFM and RAT values, perform the following record format conversions
                         00FA   359 ; when creating a file in record I/O mode on a 'stream-only' machine:
                         00FA   360 ;
                         00FA   361 ;       FIX-implied     becomes STM-embedded,
                         00FA   362 ;       VAR-implied     becomes STM-embedded,
                         00FA   363 ;       STM-implied     becomes STM-embedded,
                         00FA   364 ;       STM-embedded    remains STM-embedded,
                         00FA   365 ;       and all other formats will be rejected.
                         00FA   366 ;
                         00FA   367 ; Note that implied means RAT = CR and embedded means RAT = no carriage control
                         00FA   368 ; bits are set.
                         00FA   369 ;
                         00FA   370
```

```
                     00FA   371          $CASEB  SELECTOR=FAB$B_RFM(R8)- ; Record format:
                     00FA   372                  DISPL=<-                ;
                     00FA   373                      10$-                ;  UDF
                     00FA   374                      30$-                ;  FIX
                     00FA   375                      30$-                ;  VAR
                     00FA   376                      10$-                ;  VFC
                     00FA   377                      20$-                ;  STM
                     00FA   378                      10$-                ;  STMLF
                     00FA   379                      10$-                ;  STMCR
                     00FA   380                  >                       ;
        FEF0'  31    010D   381 10$:     BRW     NT$RMT_RFM              ; Declare RMS$_SUPPORT error and exit
1E A8    05   93    0110   382 20$:     BITB    #<<FAB$M_FTN>!-         ; Disallow Fortran and print file
                    0114   383                      <FAB$M_PRN>!-       ;  carriage control; implied and none
                    0114   384                  0>,FAB$B_RAT(R8)        ;  are allowed
          0F   13   0114   385          BEQL    40$                     ;
        FEE7'  31   0116   386          BRW     NT$RMT_RAT              ; Declare RMS$_SUPPORT error and exit
                    0119   387 30$:     $SETBIT #NWA$V_CVT_STM,(R7)     ; Denote that file format will be
                    011D   388                                          ;  converted to stream at remote node
          01   E0   011D   389          BBS     #FAB$V_CR,-             ; Require implied carriage control
03 1E A8            011F   390                  FAB$B_RAT(R8),40$       ;  (i.e., <LF-record-CR>)
        FEDB'  31   0122   391          BRW     NT$RMT_RAT              ; Declare RMS$_SUPPORT error and exit
    85   01   90   0125   392 40$:     MOVB    #DAP$M_ASCII,(R5)+      ; Store DATATYPE field
    85   00   90   0128   393          MOVB    #DAP$K_SEQ,(R5)+        ; Store ORG field
    85   04   90   012B   394          MOVB    #DAP$K_STM,(R5)+        ; Store RFM field
    85   10   90   012E   395          MOVB    #DAP$M_EMBEDDED,(R5)+   ; Store RAT field
85  0202 8F   B0   0131   396          MOVW    #<512+2>,(R5)+          ; Store MRS field (allow for CRLF which
                    0136   397                                          ;  partner may consider part of record)
          85   94   0136   398          CLRB    (R5)+                   ; Store ALQ field as zero
00C7 C7   04   90   0138   399          MOVB    #DAP$K_STM,NWA$B_RFM(R7); Overwrite saved RFM field with new
                    013D   400                                          ;  converted value
          57   11   013D   401          BRB     PART3                   ; Join common code
                    013F   402
    85   02   90   013F   403 50$:     MOVB    #DAP$M_IMAGE,(R5)+      ; Store DATATYPE field
    85   00   90   0142   404          MOVB    #DAP$K_SEQ,(R5)+        ; Store ORG field
    85   01   90   0145   405          MOVB    #DAP$K_FIX,(R5)+        ; Store RFM field
          85   94   0148   406          CLRB    (R5)+                   ; Store RAT field
85  0200 8F   B0   014A   407          MOVW    #512,(R5)+              ; Store MRS field
  51  10 A8   D0   014F   408          MOVL    FAB$L_ALQ(R8),R1        ; Get allocation quantity value
      FEAA'  30   0153   409          BSBW    NT$CVT_BN4_IMG          ; Store ALQ as an image field
        3E   11   0156   410          BRB     PART3                   ; Join common code
                    0158   411
                    0158   412 ;
                    0158   413 ; The remote node is NOT using a stream-based file system.
                    0158   414 ;
                    0158   415 ; This section deals with the DATATYPE, ORG, RFM, RAT, MRS, and ALQ fields.
                    0158   416 ;
                    0158   417
    85   02   90   0158   418 PART2B: MOVB    #DAP$M_IMAGE,(R5)+      ; Store IMAGE mode in DATATYPE field
          05   E0   015B   419          BBS     #FAB$V_BIO,-            ;  for block I/O access
  04 16 A8            015D   420                  FAB$B_FAC(R8),10$       ;  and ASCII mode
FF A5    01   90   0160   421          MOVB    #DAP$M_ASCII,-1(R5)     ;  for record I/O access
85   1D A8   90   0164   422 10$:     MOVB    FAB$B_ORG(R8),(R5)+     ; Store ORG field
85   1F A8   90   0168   423          MOVB    FAB$B_RFM(R8),(R5)+     ; Store RFM field
04   1F A8   91   016C   424          CMPB    FAB$B_RFM(R8),#FAB$C_STM; Branch if not stream format
          0A   12   0170   425          BNEQ    20$                     ;
1E A8    07   93   0172   426          BITB    #<<FAB$M_FTN>!-         ; Force implied carriage control if
                    0176   427                      <FAB$M_CR>!-        ;  user specified no explicit carriage
```

```
                     0176   428                      <FAB$M_PRN>!-          ;  control
                     0176   429                  0>,FAB$B_RAT(R8)           ;
                     0176   430          BNEQ    20$                        ;
          1E A8  02  90  0178   431          MOVB    #FAB$M_CR,FAB$B_RAT(R8)    ;
       85   1E A8  90  017C   432 20$:      MOVB    FAB$B_RAT(R8),(R5)+        ; Store RAT field
       85   36 A8  B0  0180   433           MOVW    FAB$W_MRS(R8),(R5)+        ; Store MRS field
                04  12  0184   434           BNEQ    30$                        ; Branch if non-zero
    05 67   31  E1  0186   435           BBC     #DAP$V_FCS,(R7),30$        ; Branch if partner is not FCS based
FE A5   48 A9  B0  018A   436           MOVW    IFB$L_DEVBUFSIZ(R9),-2(R5); Specify maximum value
    51   10 A8  D0  018F   437 30$:      MOVL    FAB$L_ALQ(R8),R1           ; Get allocation quantity value
          FE6A'  30  0193   438           BSBW    NT$CVT_BN4_IMG             ; Store ALQ as an image field
                     0196   439
                     0196   440 ;
                     0196   441 ; This section deals with the BKS, FSZ, MRN, DEQ, FOP, and LRL fields.
                     0196   442 ;
                     0196   443
    04 56   07  E1   0196   444 PART3:   BBC     #DAP$V_BKS,R6,10$          ; Used only if ORG = REL or IDX
       85   3E A8  90  019A   445           MOVB    FAB$B_BKS(R8),(R5)+        ; Store BKS field
    0A 56   08  E1   019E   446 10$:      BBC     #DAP$V_FSZ,R6,20$          ; Used only if RFM = VFC
       85   3F A8  90  01A2   447           MOVB    FAB$B_FSZ(R8),(R5)+        ; Store FSZ field
                04  12  01A6   448           BNEQ    20$                        ; Branch if non-zero
    FF A5   02  90  01A8   449           MOVB    #2,-1(R5)                  ; Use default FSZ value
    07 56   09  E1   01AC   450 20$:      BBC     #DAP$V_MRN,R6,30$          ; Used only if ORG = REL
    51   38 A8  D0  01B0   451           MOVL    FAB$L_MRN(R8),R1           ; Get maximum record number value
          FE49'  30  01B4   452           BSBW    NT$CVT_BN4_IMG             ; Store MRN as an image field
    04 56   0B  E1   01B7   453 30$:      BBC     #DAP$V_DEQT,R6,40$         ; Used only if DEQ > 0
       85   14 A8  B0  01BB   454           MOVW    FAB$W_DEQ(R8),(R5)+        ; Store DEQ field
          0205   30  01BF   455 40$:      BSBW    NT$MAP_FOP                 ; Store FOP field
    09 56   10  E1   01C2   456           BBC     #DAP$V_LRL,R6,50$          ; Used only if RFM not = FIX
 51    0108 C7  D0  01C6   457           MOVL    NWA$L_FHCXABADR(R7),R1     ; Get address of FHCXAB
       85   0A A1  B0  01CB   458           MOVW    XAB$W_LRL(R1),(R5)+        ; Store LRL field
          FE2E'  30  01CF   459 50$:      BSBW    NT$BUILD_TAIL              ; Finish building message
          FE2B'  30  01D2   460           BSBW    NT$TRANSMIT                ; Send Attributes message to FAL
          03 50  E8  01D5   461           BLBS    R0,SEND_KEY                ; Branch on success
           0080  31  01D8   462           BRW     FAIL2                      ; Failure
```

```
                     01DB   464 ;+
                     01DB   465 ; Build and send DAP Key Definition message(s) to partner.
                     01DB   466 ;
                     01DB   467 ; Note: NT$SCAN_XABCHN determined that Key Definition XABs were linked together.
                     01DB   468 ;-
                     01DB   469
                     01DB   470 SEND_KEY:                                    ; (optional message)
          01   E1    01DB   471          BBC     #DAP$V_DSP_KEY,-           ; Branch if Key Definition message
     00D2 C7         01DD   472                  NWA$W_BUILD(R7),-          ;  should not be sent
          20         01E0   473                  SEND_ALL                   ;
56   010C C7   D0    01E1   474          MOVL    NWA$L_KEYXABADR(R7),R6     ; Get address of first user KEYXAB
       FE17'  30     01E6   475 10$:     BSBW    NT$SCAN_KEYXAB             ; Scan the Key Definition XAB
        6F 50 E9     01E9   476          BLBC    R0,FAIL2                   ; Branch on failure
       FE11'  30     01EC   477          BSBW    NT$ENCODE_KEY              ; Build message
       FE0E'  30     01EF   478          BSBW    NT$TRANSMIT                ; Send Key Definition message to FAL
        66 50 E9     01F2   479          BLBC    R0,FAIL2                   ; Branch on failure
     011D C7   97    01F5   480          DECB    NWA$B_KEYXABCNT(R7)        ; Any more KEYXABs to process?
          06   1B    01F9   481          BLEQU   SEND_ALL                   ; Branch if not
  56   04 A6   D0    01FB   482          MOVL    XAB$L_NXT(R6),R6           ; Get address on next XAB in chain
          E5   11    01FF   483          BRB     10$                        ; Continue with next XAB
                     0201   484
                     0201   485 ;+
                     0201   486 ; Build and send DAP Allocation message(s) to partner.
                     0201   487 ;
                     0201   488 ; Note: NT$SCAN_XABCHN determined that Allocation XABs were chained together.
                     0201   489 ;-
                     0201   490
                     0201   491 SEND_ALL:                                    ; (optional message)
          02   E1    0201   492          BBC     #DAP$V_DSP_ALL,-           ; Branch if Allocation message
     00D2 C7         0203   493                  NWA$W_BUILD(R7),-          ;  should not be sent
          20         0206   494                  SEND_TIM                   ;
56   0100 C7   D0    0207   495          MOVL    NWA$L_ALLXABADR(R7),R6     ; Get address of first user ALLXAB
       FDF1'  30     020C   496 10$:     BSBW    NT$SCAN_ALLXAB             ; Scan the Allocation XAB
        49 50 E9     020F   497          BLBC    R0,FAIL2                   ; Branch on failure
       FDEB'  30     0212   498          BSBW    NT$ENCODE_ALL              ; Build message
       FDE8'  30     0215   499          BSBW    NT$TRANSMIT                ; Send Allocation message to FAL
        40 50 E9     0218   500          BLBC    R0,FAIL2                   ; Branch on failure
     011C C7   97    021B   501          DECB    NWA$B_ALLXABCNT(R7)        ; Any more Allocation XABs to process?
          06   1B    021F   502          BLEQU   SEND_TIM                   ; Branch if not
  56   04 A6   D0    0221   503          MOVL    XAB$L_NXT(R6),R6           ; Get address on next XAB in chain
          E5   11    0225   504          BRB     10$                        ; Continue with next XAB
                     0227   505
                     0227   506 ;+
                     0227   507 ; Build and send DAP Date and Time message to partner.
                     C227   508 ;
                     0227   509 ; Note: If the user supplied a Date and Time XAB it will be used to build the
                     0227   510 ;       message, else the user supplied Revision Date and Time XAB will be used.
                     0227   511 ;-
                     0227   512
                     0227   513 SEND_TIM:                                    ; (optional message)
          04   E1    0227   514          BBC     #DAP$V_DSP_TIM,-           ; Branch if Date and Time message
     00D2 C7         0229   515                  NWA$W_BUILD(R7),-          ;  should not be sent
          1A         022C   516                  SEND_PRO                   ;
56   0104 C7   D0    022D   517          MOVL    NWA$L_DATXABADR(R7),R6     ; Get address of user DATXAB
          05   13    0232   518          BEQL    10$                        ; Branch if none
       FDC9'  30     0234   519          BSBW    NT$ENCODE_TIM_D            ; Build message
          08   11    0237   520          BRB     20$                        ; Join common code
```

```
                56  0114 C7   D0  0239   521 10$:    MOVL    NWA$L_RDTXABADR(R7),R6  ; Get address of user RDTXAB instead
                    FDBF'     30  023E   522         BSBW    NT$ENCODE_TIM_R         ; Build message
                    FDBC'     30  0241   523 20$:    BSBW    NT$TRANSMIT             ; Send Date and Time message to FAL
                      14 50   E9  0244   524         BLBC    R0,FAIL2                ; Branch on failure
                              0247   525
                              0247   526 ;+
                              0247   527 ; Build and send DAP Protection message to partner.
                              0247   528 ;-
                              0247   529
                              0247   530 SEND_PRO:                                  ; (optional message)
                         05   E1  0247   531         BBC     #DAP$V_DSP_PRO,-        ; Branch if Protection message
                    00D2 C7       0249   532                 NWA$W_BUILD(R7),-        ;  should not be sent
                         0F       024C   533                 SEND_ACC                ;
                56  0110 C7   D0  024D   534         MOVL    NWA$L_PROXABADR(R7),R6  ; Get address of user PROXAB
                    FDAB'     30  0252   535         BSBW    NT$ENCODE_PRO           ; Build message
                    FDA8'     30  0255   536         BSBW    NT$TRANSMIT             ; Send Protection message to FAL
                      01 50   E8  0258   537         BLBS    R0,SEND_ACC             ; Branch on success
                              05  025B   538 FAIL2:   RSB                             ; Exit with RMS code in R0
                              025C   539
                              025C   540 ;+
                              025C   541 ; Build and send DAP Access message to partner.
                              025C   542 ;-
                              025C   543
                              025C   544 SEND_ACC:                                  ; (required message)
                              025C   545         $SETBIT #NWA$V_LAST_MSG,(R7)     ; Declare this last message to block
                50   03   D0  0260   546         MOVL    #DAP$K_ACC_MSG,R0       ; Get message type value
                    FD9A'     30  0263   547         BSBW    NT$BUILD_HEAD           ; Construct message header
                85   02   90  0266   548         MOVB    #DAP$K_CREATE,(R5)+     ; Store ACCFUNC field
                85   01   90  0269   549         MOVB    #DAP$M_NONFATAL,(R5)+   ; Store ACCOPT field
                    FD91'     30  026C   550         BSBW    NT$CRC_INIT             ; Initialize CRC value if both parties
                              026F   551                                             ;  support file level CRC computation
                  04 50   E9  026F   552         BLBC    R0,10$                  ; Branch if CRC checking disabled
              FF A5   08   88  0272   553         BISB2   #DAP$M_RET_CRC,-1(R5)   ; Request checksum option
                    FD87'     30  0276   554 10$:    BSBW    NT$GET_FILESPEC         ; Store FILESPEC as a counted
                              0279   555                                             ;  ASCII string
                    00C2      30  0279   556         BSBW    NT$GET_FAC_SHR          ; Store FAC and SHR fields
            51  00D0 C7   3C  027C   557         MOVZWL  NWA$W_DISPLAY(R7),R1    ; Get request mask
                51   01   B1  0281   558         CMPW    #DAP$M_DSP_ATT,R1       ; Omit DISPLAY field from message if
                     03   13  0284   559         BEQL    20$                     ;  only Attributes message specified
                              0286   560                                             ;  (because some older FALs do not
                              0286   561                                             ;  support this field nor Ext Att msgs)
                    FD77'     30  0286   562         BSBW    NT$CVT_BN4_EXT          ; Store DISPLAY as an extensible field
                    FD74'     30  0289   563 20$:    BSBW    NT$BUILD_TAIL           ; Finish building message
                    FD71'     30  028C   564         BSBW    NT$TRANSMIT             ; Send Access message to FAL
                  C9 50   E9  028F   565         BLBC    R0,FAIL2                ; Branch on failure
```

```
                     0292   567 ;+
                     0292   568 ; Receive DAP Attributes message from partner and update the user FAB.
                     0292   569 ; Also update the user ALLXAB if an Allocation message will not be returned
                     0292   570 ; by FAL.
                     0292   571 ;
                     0292   572 ; Note: The user XAB chain is scanned again to probe all user XABs to protect
                     0292   573 ;       RMS from a user who deletes the address space where an XAB was
                     0292   574 ;       previously found.
                     0292   575 ;-
                     0292   576
                     0292   577 RECV_ATT:                                    ; (required message)
                     0292   578         $SETBIT #DAP$K_ATT_MSG,DAP$L_MSG_MASK(R7)
                     0297   579                                              ; Expect response of Attributes message
      FD66'  30      0297   580         BSBW    NT$RECEIVE                   ; Get reply from FAL
   5E 50     E9      029A   581         BLBC    R0,FAIL3                     ; Branch on failure
   45 A7     90      029D   582         MOVB    DAP$B_ORG(R7),-              ; Save file organization type
00C6 C7              02A0   583                 NWA$B_ORG(R7)                ;
   46 A7     90      02A3   584         MOVB    DAP$B_RFM(R7),-              ; Save file format type
00C7 C7              02A6   585                 NWA$B_RFM(R7)                ;
      FD54'  30      02A9   586         BSBW    NT$MAP_DEV_CHAR              ; Process device characteristics
                     02AC   587
                     02AC   588 ;
                     02AC   589 ; Update user control blocks.
                     02AC   590 ;
                     02AC   591
        56   D4      02AC   592         CLRL    R6                           ; Indicate this is not a close operation
      FD4F'  30      02AE   593         BSBW    NT$SCAN_XABCHN               ; Scan user XAB list again
   47 50     E9      02B1   594         BLBC    R0,FAIL3                     ; Branch on failure to scan XABs
     0045    30      02B4   595         BSBW    CREATE_UPDATE_FAB            ; Update user FAB
       02    E0      02B7   596         BBS     #DAP$V_DSP_ALL,-             ; Branch if Allocation message
00D0 C7              02B9   597                 NWA$W_DISPLAY(R7),-          ;  was requested
       03            02BC   598                 RECV_EXT_ATT                 ;
      FD40'  30      02BD   599         BSBW    NT$DECODE_ALL_A              ; Update user ALLXAB from fields in
                     02C0   600                                              ;  Attributes message (unless ORG = IDX)
                     02C0   601
                     02C0   602 ;+
                     02C0   603 ; Receive DAP Extended Attributes messages from partner and update the user
                     02C0   604 ; ALLXABs.
                     02C0   605 ;
                     02C0   606 ; Note: For indexed files, multiple Allocation messages may be returned.
                     02C0   607 ;-
                     02C0   608
                     02C0   609 RECV_EXT_ATT:                                ; (optional--must be requested)
       02    E1      02C0   610         BBC     #DAP$V_DSP_ALL,-             ; Branch if Allocation message was not
00D0 C7              02C2   611                 NWA$W_DISPLAY(R7),-          ;  requested
       06            02C5   612                 RECV_NAM                     ;
      FD37'  30      02C6   613         BSBW    NT$RECV_EXT_ATT              ; Process Extended Attributes messages
   2F 50     E9      02C9   614         BLBC    R0,FAIL3                     ; Branch on failure
                     02CC   615
                     02CC   616 ;+
                     02CC   617 ; Receive DAP (resultant) Name message from partner.
                     02CC   618 ;-
                     02CC   619
                     02CC   620 RECV_NAM:                                    ; (optional--must be requested)
       08    E1      02CC   621         BBC     #DAP$V_DSP_NAM,-             ; Branch if Name message was not
00D0 C7              02CE   622                 NWA$W_DISPLAY(R7),-          ;  requested
       0E            02D1   623                 RECV_ACK                     ;
```

```
                     02D2   624          $SETBIT #DAP$K_NAM_MSG,DAP$L_MSG_MASK(R7)
                     02D7   625                                           ; Expect response of Name message
          FD26'  30  02D7   626          BSBW    NT$RECEIVE              ; Get reply from FAL
        1E 50    E9  02DA   627          BLBC    R0,FAIL3                ; Branch on failure
          FD20'  30  02DD   628          BSBW    NT$DECODE_NAM           ; Process resultant name string
                     02E0   629
                     02E0   630 ;+
                     02E0   631 ; Receive DAP Acknowledge message from partner.
                     02E0   632 ;-
                     02E0   633
                     02E0   634 RECV_ACK:                                 ; (required message)
                     02E0   635          $SETBIT #DAP$K_ACK_MSG,DAP$L_MSG_MASK(R7)
                     02E5   636                                           ; Expect response of Acknowledge message
          FD18'  30  02E5   637          BSBW    NT$RECEIVE              ; Get reply from FAL
        10 50    E9  02E8   638          BLBC    R0,FAIL3                ; Branch on failure
                     02EB   639          $SETBIT #IFB$V_DAP_OPEN,(R9)    ; Yes, denote FAL has opened file
                     02EF   640          RMSSUC                          ; Return success
  05 67    06    E1  02F2   641          BBC     #NWA$V_CVT_STM,(R7),FAIL3;Branch if no conversion will occur
                     02F6   642          RMSERR  CRE_STM                 ; Return alternate success to indicate
                     02FB   643                                           ;  that file format will become stream
                  05 02FB   644 FAIL3:   RSB                             ; Exit with RMS code in R0
```

```
                     02FC    646          .SBTTL  CREATE_UPDATE_FAB
                     02FC    647
                     02FC    648 ;++
                     02FC    649 ; Update the user FAB from the Attributes message.
                     02FC    650 ;
                     02FC    651 ; Note: BLS will be updated directly in the FAB, whereas, ALQ will be updated
                     02FC    652 ;       in the IFB and then returned to the FAB by RMSOOPEN exit code.
                     02FC    653 ;--
                     02FC    654
                     02FC    655 CREATE_UPDATE_FAB:                      ; Entry point
                     02FC    656
                     02FC    657 ;
                     02FC    658 ; Process the DAP BLS field.
                     02FC    659 ;
                     02FC    660
   00    45 A7   91  02FC    661          CMPB    DAP$B_ORG(R7),#DAP$K_SEQ; Branch if not SEQ organization
         05      12  0300    662          BNEQ    10$                     ;
3C A8    48 A7   B0  0302    663          MOVW    DAP$W_BLS(R7),FAB$W_BLS(R8)
                     0307    664
                     0307    665 ;
                     0307    666 ; Process the DAP ALQ and HBK fields.
                     0307    667 ;
                     0307    668 ; Note: ALQ and HBK are equivalent, but not all non-VAX nodes return HBK.
                     0307    669 ;
                     0307    670
70 A9    4C A7   D0  0307    671 10$:     MOVL    DAP$L_ALQ1(R7),IFB$L_HBK(R9)
                     030C    672
                     030C    673 ;
                     030C    674 ; Save FAB BKS field for use later by RMSOOPEN.
                     030C    675 ;
                     030C    676
5E A9    3E A8   90  030C    677 20$:     MOVB    FAB$B_BKS(R8),IFB$B_BKS(R9)
         05      05  0311    678          RSB                             ; Exit
```

```
                     0312   680          .SBTTL  NT$CHK_ORG - CHECK FILE ORGANIZATION
                     0312   681
                     0312   682 ;++
                     0312   683 ; NT$CHK_ORG checks the ORG value against that supported by the remote FAL.
                     0312   684 ;--
                     0312   685
                     0312   686 NT$CHK_ORG::                                ; Entry point
00    1D A8    91    0312   687          CMPB    FAB$B_ORG(R8),#FAB$C_SEQ; Check for sequential organization
            07    12    0316   688          BNEQ    10$                      ; Branch if not
            01    E0    0318   689          BBS     #DAP$V_SEQORG,-          ; Branch if supported by partner
      1D 28 A7          031A   690                  DAP$Q_SYSCAP(R7),40$     ;
            18    11    031D   691          BRB     30$                      ; Error if not
10    1D A8    91    031F   692 10$:       CMPB    FAB$B_ORG(R8),#FAB$C_REL; Check for relative organization
            07    12    0323   693          BNEQ    20$                      ; Branch if not
            02    E0    0325   694          BBS     #DAP$V_RELORG,-          ; Branch if supported by partner
      10 28 A7          0327   695                  DAP$Q_SYSCAP(R7),40$     ;
            0B    11    032A   696          BRB     30$                      ; Error if not
20    1D A8    91    032C   697 20$:       CMPB    FAB$B_ORG(R8),#FAB$C_IDX; Check for indexed organization
            05    12    0330   698          BNEQ    30$                      ; Error if not
            0B    E0    0332   699          BBS     #DAP$V_IDXORG,-          ; Branch if supported by partner
      03 28 A7          0334   700                  DAP$Q_SYSCAP(R7),40$     ;
         FCC6'    31    0337   701 30$:       BRW     NT$RMT_ORG               ; Declare RMS$_SUPPORT error and exit
                     033A   702 40$:       RMSSUC                           ; Return success
                  05 033D   703          RSB                              ; Exit with RMS code in R0
```

```
                     705          .SBTTL  NT$GET_FAC_SHR
                     706
                     707 ;++
                     708 ; This routine builds the DAP FAC and SHR fields and stores them as extensible
                     709 ; fields in the DAP message.
                     710 ;
                     711 ; Note: It is assumed that each of these fields can be encoded into one byte
                     712 ;       (i.e., that an extension byte is never needed).
                     713 ;
                     714 ; Note: The definitions of the bit offsets for the FAC field in the DAP and FAB
                     715 ;       control blocks are the same, whereas, the defintions of the bit offsets
                     716 ;       for the SHR field in these control blocks are slightly different.
                     717 ;--
                     718
                     719 NT$GET_FAC_SHR::                        ; Entry point
                     720
                     721          ASSUME DAP$V_GET EQ FAB$V_GET
                     722          ASSUME DAP$V_PUT EQ FAB$V_PUT
                     723          ASSUME DAP$V_DEL EQ FAB$V_DEL
                     724          ASSUME DAP$V_UPD EQ FAB$V_UPD
                     725          ASSUME DAP$V_TRN EQ FAB$V_TRN
                     726          ASSUME DAP$V_BIO EQ FAB$V_BIO
                     727          ASSUME DAP$V_BRO EQ FAB$V_BRO
                     728
                     729 ;+
                     730 ; First the FAC field.
                     731 ;
                     732 ; Note: If partner node is:
                     733 ;   (1) VAX/VMS since  DAP V5.6 then map <GET,PUT,DEL,UPD,TRN,BIO,BRO>.
                     734 ;   (2) VAX/VMS before DAP V5.6 then map <GET,PUT,DEL,UPD,TRN,BIO>.
                     735 ;   (3) RMS based but not VAX/VMS    map <GET,PUT,DEL,UPD,TRN,BIO>.
                     736 ;   (4) not RMS based then only      map <GET,PUT,BIO>.
                     737 ; Note: In the above definition, include VAXELAN in the VAX/VMS check.
                     738 ; In addition, if partner is RT-11, then mask out <GET> on $CREATE and
                     739 ; mask out <PUT> on $OPEN.
                     740 ;-
                     741
                     742 MAP_FAC:                                ;
65   22 A9   90  033E 743          MOVB    IFB$B_FAC(R9),(R5)      ; Store FAC field
65   80 8F   8A  0342 744          BICB2   #FAB$M_EXE,(R5)         ; Mask out this option but don't
                 0346 745                                          ;  complain about it
16 67   30   E0  0346 746          BBS     #DAP$V_RMS,(R7),20$     ; Branch if partner is RMS based
65   5C 8F   8A  034A 747          BICB2   #<<DAP$M_DEL>!-         ; Mask out more bits
                 034E 748                    <DAP$M_UPD>!-         ;
                 034E 749                    <DAP$M_TRN>!-         ;
                 034E 750                    <DAP$M_BRO>!-         ;
                 034E 751                    0>,(R5)               ;
1E 67   38   E1  034E 752          BBC     #DAP$V_RT11,(R7),40$    ; Branch if remote node is not RT-11
05 69   32   E0  0352 753          BBS     #IFB$V_CREATE,(R9),10$  ; Branch on $CREATE
   65   01   8A  0356 754          BICB2   #DAP$M_PUT,(R5)         ; Mask out <PUT> access on $OPEN
        15   11  0359 755          BRB     40$                     ;
   65   02   8A  035B 756 10$:     BICB2   #DAP$M_GET,(R5)         ; Mask out <GET> access on $CREATE
        10   11  035E 757          BRB     40$                     ;
08 67   24   E1  0360 758 20$:     BBC     #DAP$V_GEQ_V56,(R7),30$ ; Branch if partner uses DAP before V5.6
08 67   34   E0  0364 759          BBS     #DAP$V_VAXVMS,(R7),40$  ; Branch if partner is VAX/VMS
04 67   35   E0  0368 760          BBS     #DAP$V_VAXELAN,(R7),40$ ; Branch if partner is VAXELAN
65   40 8F   8A  036C 761 30$:     BICB2   #DAP$M_BRO,(R5)         ; Mask out this option but don't
```

```
                     0370   762                                     ;  complain about it
          55   D6    0370   763 40$:    INCL    R5                  ; Advance to  ext field in message
                     0372   764
                     0372   765 ;+
                     0372   766 ; Next the SHR field.
                     0372   767 ;
                     0372   768 ; Note: If partner node is:
                     0372   769 ;   (1) VAX/VMS since  DAP V5.6 then map <GET,PUT,DEL,UPD,UPI,NIL>.
                     0372   770 ;   (2) VAX/VMS before DAP V5.6 then map <GET,PUT,DEL,UPD>.
                     0372   771 ;   (3) not VAX/VMS then map <GET,PUT> only.
                     0372   772 ;-
                     0372   773
                     0372   774 MAP_SHR:                            ;
51   17 A8   9A      0372   775         MOVZBL  FAB$B_SHR(R8),R1     ; Get SHR bits
          52   D4    0376   776         CLRL    R2                   ; Clear resultant SHR bits
                     0378   777         $MAPBIT FAB$V_SHRPUT,DAP$V_SHRPUT; Map PUT bit
                     0380   778         $MAPBIT FAB$V_SHRGET,DAP$V_SHRGET; Map GET bit
24 67    34   E1     0388   779         BBC     #DAP$V_VAXVMS,(R7),10$  ; Branch if partner is not VAX/VMS
                     038C   780         $MAPBIT FAB$V_SHRDEL,DAP$V_SHRDEL; Map DEL bit
                     0394   781         $MAPBIT FAB$V_SHRUPD,DAP$V_SHRUPD; Map UPD bit
10 67    24   E1     039C   782         BBC     #DAP$V_GEQ_V56,(R7),10$ ; Branch if partner uses DAP before V5.6
                     03A0   783         $MAPBIT FAB$V_UPI,DAP$V_UPI  ;  Map UPI bit
                     03A8   784         $MAPBIT FAB$V_NIL,DAP$V_NIL  ;  Map NIL bit
     85   52   90     03B0   785 10$:    MOVB    R2,(R5)+             ; Store SHR field
                     03B3   786
                     03B3   787 ;
                     03B3   788 ; The following is special code to handle an incompatibility in the processing
                     03B3   789 ; of the FAC/SHR fields that exists between VAX-11 RMS V3.0 and RMS-11 V1.8
                     03B3   790 ; (and possibly with other file systems).
                     03B3   791 ;
                     03B3   792 ; VAX-11 RMS allows FAC = <GET> and SHR = <GET!PUT> but RMS-11 objects to <PUT>
                     03B3   793 ; in the SHR field unless <PUT> in the FAC field is also set.
                     03B3   794 ;
                     03B3   795 ; The temporary solution is to mask out <PUT> in the SHR field in this
                     03B3   796 ; circumstance if the remote node is not VAX/VMS.
                     03B3   797 ;
                     03B3   798
   0F 67    34   E0     03B3   799         BBS     #DAP$V_VAXVMS,(R7),20$  ; Branch if partner is VAX/VMS
0A FF A5    00   E1     03B7   800         BBC     #DAP$V_SHRPUT,-1(R5),20$; Branch if PUT bitnot set in SHR
05 FE A5    00   E0     03BC   801         BBS     #DAP$V_PUT,-2(R5),20$   ; Branch if PUT bitset in FAC
                     03C1   802         $CLRBIT #DAP$V_SHRPUT,-1(R5)    ; Clear PUT bit in SHR field
               05    03C6   803 20$:    RSB                          ; Exit
```

```
                     03C7   805          .SBTTL  NT$MAP_FOP - MAP FOP OPTIONS
                     03C7   806
                     03C7   807 ;++
                     03C7   808 ; This routine builds the DAP FOP field and stores it as an extensible field
                     03C7   809 ; in the DAP message.
                     03C7   810 ;--
                     03C7   811
                     03C7   812 NT$MAP_FOP::                             ; Entry point
51   04 A8   D0      03C7   813          MOVL    FAB$L_FOP(R8),R1        ; Get FOP bits
     52      D4      03CB   814          CLRL    R2                      ; Clear resultant FOP bits
                     03CD   815
                     03CD   816 ;
                     03CD   817 ; The following DAP bits are defined in the DAP V4.1 specification.
                     03CD   818 ; These will be mapped.
                     03CD   819 ;
                     03CD   820
                     03CD   821          $MAPBIT FAB$V_CTG,DAP$V_CTG     ; Map CTG bit
                     03D5   822          $MAPBIT FAB$V_SUP,DAP$V_SUP     ; Map SUP bit
                     03DD   823          $MAPBIT FAB$V_TMP,DAP$V_TMP     ; Map TMP bit
                     03E5   824          $MAPBIT FAB$V_TMD,DAP$V_TMD     ; Map TMD bit
                     03ED   825          $MAPBIT FAB$V_WCK,DAP$V_WCK     ; Map WCK bit
                     03F5   826          $MAPBIT FAB$V_RCK,DAP$V_RCK     ; Map RCK bit
                     03FD   827 ; ##### $MAPBIT FAB$V_DMO,DAP$V_DMO     ; Map DMO bit
                     03FD   828                                          ; Note: this is not implemented in RMS32
                     03FD   829
                     03FD   830 ;
                     03FD   831 ; The following DAP bits are defined in the DAP V4.1 specification.
                     03FD   832 ; These will be mapped iff partner is VAX/VMS.
                     03FD   833 ;
                     03FD   834
20   67   34   E1      03FD   835          BBC     #DAP$V_VAXVMS,(R7),10$  ; Branch if partner is not VAX/VMS
                     0401   836          $MAPBIT FAB$V_RWO,DAP$V_RWO     ; Map RWO bit
                     0409   837          $MAPBIT FAB$V_RWC,DAP$V_RWC     ; Map RWC bit
                     0411   838          $MAPBIT FAB$V_POS,DAP$V_POS     ; Map POS bit
                     0419   839          $MAPBIT FAB$V_NEF,DAP$V_NEF     ; Map NEF bit
                     0421   840 ; ##### $MAPBIT FAB$V_CIF,DAP$V_CIF     ; Map CIF bit
                     0421   841                                          ; Note: this option is simulated, thus
                     0421   842                                          ;  it is not requested via FOP field
                     0421   843
                     0421   844 ;
                     0421   845 ; The following DAP bits were defined in the DAP V4.2 specification.
                     0421   846 ; These will be mapped iff partner supports all of the requested options; else
                     0421   847 ; ignored for now--they will be re-examined on close for potential alternate
                     0421   848 ; processing using a separate DAP access function.
                     0421   849 ;
                     0421   850
                     0421   851          ASSUME  FAB$V_SPL+1 EQ FAB$V_SCF
                     0421   852          ASSUME  FAB$V_SCF+1 EQ FAB$V_DLT
                     0421   853
                     0421   854          ASSUME  DAP$V_FOPSPL+1 EQ DAP$V_FOPSCF
                     0421   855          ASSUME  DAP$V_FOPSCF+1 EQ DAP$V_FOPDLT
                     0421   856
54   51   03   0D    EF   0421   857 10$:     EXTZV   #FAB$V_SPL,#3,R1,R4     ; Extract the three FOP bits
               23    13   0426   858          BEQL    20$                     ; Branch if none set
          03   1D    EF   0428   859          EXTZV   #DAP$V_FOPSPL,#3,-      ; Extract the three corresponding
     50     28 A7         042B   860                  DAP$Q_SYSCAP(R7),R0     ;  SYSCAP bits
          54   50    CA   042E   861          BICL2   R0,R4                   ; Mask out supported requests
```

```
                 18   12  0431   862          BNEQ    20$                     ; Branch if any are not supported
                         0433   863          $MAPBIT FAB$V_SPL,DAP$V_SPL     ; Map SPL bit
                         043B   864          $MAPBIT FAB$V_SCF,DAP$V_SCF     ; Map SCF bit
                         0443   865          $MAPBIT FAB$V_DLT,DAP$V_DLT     ; Map DLT bit
                         044B   866
                         044B   867 ;
                         044B   868 ; The following DAP bits were defined in the DAP V4.2 specification or later.
                         044B   869 ; These will be mapped iff partner is VAX/VMS and for MXV additional checks
                         044B   870 ; will be made.
                         044B   871 ;
                         044B   872
          16 67    34    E0   044B   873 20$:     BBS     #DAP$V_VAXVMS,(R7),30$  ; Branch if partner is VAX/VMS
          12 67    35    E0   044F   874          BBS     #DAP$V_VAXELAN,(R7),30$ ; Branch if partner is VAXELAN
          2E 67    30    E1   0453   875          BBC     #DAP$V_RMS,(R7),40$     ; Branch if partner is not RMS based
          2A 67    24    E1   0457   876          BBC     #DAP$V_GEQ_V56,(R7),40$ ; Branch if partner uses DAP before V5.6
                         045B   877          $MAPBIT FAB$V_MXV,DAP$V_MXV     ; Map MXV bit
                 20   11  0463   878          BRB     40$                     ;
                         0465   879 30$:     $MAPBIT FAB$V_SQO,DAP$V_SQO     ; Map SQO bit
                         046D   880          $MAPBIT FAB$V_MXV,DAP$V_MXV     ; Map MXV bit
                         0475   881          $MAPBIT FAB$V_CBT,DAP$V_CBT     ; Map CBT bit
                         047D   882 ; *****   $MAPBIT FAB$V_DFW,DAP$V_DFW     ; Map DFW bit
                         047D   883          $MAPBIT FAB$V_TEF,DAP$V_TEF     ; Map TEF bit
                         0485   884 ; #####   $MAPBIT FAB$V_OFP,DAP$V_OFP     ; Map OFP bit
                         0485   885                                          ; Note: it makes no sense to send this
                         0485   886                                          ;  bit since all parsing is done locally
              51   52    D0   0485   887 40$:     MOVL    R2,R1                   ; Move data to correct register
                 FB75'  30   0488   888          BSBW    NT$CVT_BN4_EXT          ; Store FOP as an extensible field
                        05   048B   889          RSB                             ; Exit with RMS code in R0
                         048C   890
                         048C   891          .END                            ; End of module
```

```
$$.PSECT_EP                     = 00000000
$$COUNT                         = 00000007
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
BUILD_MASK                        00000027 R      01
CREATE_UPDATE_FAB                 000002FC R      01
DAP$B_ACCFUNC                     00000040
DAP$B_ACCOPT                      00000041
DAP$B_BITCNT                      00000035
DAP$B_BKS                         00000050
DAP$B_BSZ                         00000052
DAP$B_DATATYPE                    00000044
DAP$B_DCODE_FID                   00000019
DAP$B_DCODE_MAC                   0000001B
DAP$B_DCODE_MSG                   0000001A
DAP$B_DECVER                      00000047
DAP$B_ECONUM                      00000045
DAP$B_FAC                         00000042
DAP$B_FILESYS                     00000043
DAP$B_FLAGS                       00000031
DAP$B_FSZ                         00000051
DAP$B_LEN256                      00000034
DAP$B_LENGTH                      00000033
DAP$B_ORG                         00000045
DAP$B_OSTYPE                      00000042
DAP$B_RAT                         00000047
DAP$B_RFM                         00000046
DAP$B_SHR                         00000043
DAP$B_STREAMID                    00000032
DAP$B_TYPE                        00000030
DAP$B_USRNUM                      00000046
DAP$B_USRVER                      00000048
DAP$B_VERNUM                      00000044
DAP$B_X_FIELD                     00000024
DAP$C_BLN                         000000C0
DAP$K_ACC_MSG                   = 00000003
DAP$K_ACK_MSG                   = 00000006
DAP$K_ATT_MSG                   = 00000002
DAP$K_BLN                         000000C0
DAP$K_CREATE                    = 00000002
DAP$K_FIX                       = 00000001
DAP$K_IDX                       = 00000020
DAP$K_NAM_MSG                   = 0000000F
DAP$K_REL                       = 00000010
DAP$K_SEQ                       = 00000000
DAP$K_STM                       = 00000004
DAP$K_STMCR                     = 00000006
DAP$K_STMLF                     = 00000005
DAP$K_UDF                       = 00000000
DAP$K_VAR                       = 00000002
DAP$K_VFC                       = 00000003
DAP$L_ALQ1                        0000004C
DAP$L_ATTMENU                     00000040
DAP$L_CMWA                        00000030
DAP$L_CRC_RSLT                    00000020
DAP$L_DCODE_STS                   00000018
DAP$L_DEV                         00000068
DAP$L_EBK                         00000078
DAP$L_FOP1                        00000064
DAP$L_HBK                         00000074
DAP$L_MRN                         00000058
DAP$L_MSG_MASK                    0000001C
DAP$L_SBN                         0000007C
DAP$L_SSPWA                       00000080
DAP$L_TEMP                        00000090
DAP$M_ALQ1                      = 00000040
DAP$M_ASCII                     = 00000001
DAP$M_BITCNT                    = 00000008
DAP$M_BRO                       = 00000040
DAP$M_CMPFMT                    = 00000008
DAP$M_DATATYPE                  = 00000001
DAP$M_DEL                       = 00000004
DAP$M_DMO                       = 00002000
DAP$M_DSP_3NAM                  = 00000200
DAP$M_DSP_ATT                   = 00000001
DAP$M_DSP_KEY                   = 00000002
DAP$M_DSP_PRO                   = 00000020
DAP$M_DSP_SUM                   = 00000008
DAP$M_DSP_TIM                   = 00000010
DAP$M_EMBEDDED                  = 00000010
DAP$M_FOP1                      = 00001000
DAP$M_GET                       = 00000002
DAP$M_GO_NOGO                   = 00000010
DAP$M_IMAGE                     = 00000002
DAP$M_LSA                       = 00000040
DAP$M_MACY11                    = 00000080
DAP$M_MRS                       = 00000020
DAP$M_MSE                       = 00000010
DAP$M_NONFATAL                  = 00000001
DAP$M_ORG                       = 00000002
DAP$M_PUT                       = 00000001
DAP$M_RAT                       = 00000008
DAP$M_RET_CRC                   = 00000008
DAP$M_RFM                       = 00000004
DAP$M_SEGMENT                   = 00000040
DAP$M_TMP1$                     = 000000C0
DAP$M_TMP2$                     = 0000FC00
DAP$M_TMP3$                     = 00020000
DAP$M_TMP4$                     = 01000000
DAP$M_TMP5$                     = F0000000
DAP$M_TRN                       = 00000010
DAP$M_UPD                       = 00000008
DAP$M_ZERO                      = 00000080
DAP$Q_DCODE_FLG                   00000000
DAP$Q_FI_ESPEC                    00000044
DAP$Q_MSG_BUF1                    00000008
DAP$Q_MSG_BUF2                    00000010
DAP$Q_PASSWORD                    00000050
DAP$Q_RUNSYS                      0000005C
DAP$Q_SYSCAP                      00000028
DAP$Q_SYSPEC                      00000038
DAP$V_BIO                       = 00000005
```

```
DAP$V_BKS                       =  00000007
DAP$V_BLK                       =  00000003
DAP$V_BRO                       =  00000006
DAP$V_CBT                       =  00000017
DAP$V_CR                        =  00000001
DAP$V_CTG                       =  00000007
DAP$V_DEL                       =  00000002
DAP$V_DEQ1                      =  0000000B
DAP$V_DLT                       =  00000016
DAP$V_DSP_ALL                   =  00000002
DAP$V_DSP_KEY                   =  00000001
DAP$V_DSP_NAM                   =  00000008
DAP$V_DSP_PRO                   =  00000005
DAP$V_DSP_TIM                   =  00000004
DAP$V_FCS                       =  00000031
DAP$V_FOPDLT                    =  0000001F
DAP$V_FOPSCF                    =  0000001E
DAP$V_FOPSPL                    =  0000001D
DAP$V_FSZ                       =  00000008
DAP$V_FTN                       =  00000000
DAP$V_GEQ_V54                   =  00000023
DAP$V_GEQ_V56                   =  00000024
DAP$V_GEQ_V70                   =  00000026
DAP$V_GET                       =  00000001
DAP$V_IDXORG                    =  0000000B
DAP$V_LRL                       =  00000010
DAP$V_MRN                       =  00000009
DAP$V_MXV                       =  00000013
DAP$V_NEF                       =  00000009
DAP$V_NIL                       =  00000006
DAP$V_POS                       =  00000003
DAP$V_PRN                       =  00000002
DAP$V_PUT                       =  00000000
DAP$V_RCK                       =  0000000F
DAP$V_RELORG                    =  00000002
DAP$V_RMS                       =  00000030
DAP$V_RT11                      =  00000038
DAP$V_RWC                       =  00000001
DAP$V_RWO                       =  00000000
DAP$V_SCF                       =  00000015
DAP$V_SEQORG                    =  00000001
DAP$V_SHRDEL                    =  00000002
DAP$V_SHRGET                    =  00000001
DAP$V_SHRPUT                    =  00000000
DAP$V_SHRUPD                    =  00000003
DAP$V_SPL                       =  00000014
DAP$V_SQO                       =  00000012
DAP$V_STM_ONLY                  =  00000032
DAP$V_SUP                       =  00000008
DAP$V_TEF                       =  0000001A
DAP$V_TMD                       =  0000000B
DAP$V_TMP                       =  0000C00A
DAP$V_TRN                       =  00000004
DAP$V_UPD                       =  00000003
DAP$V_UPI                       =  00000005
DAP$V_VAXELAN                   =  00000035
DAP$V_VAXVMS                    =  00000034
DAP$V_WCK                       =  0000000E
DAP$W_BLS                          00000048
DAP$W_BUFSIZ                       00000040
DAP$W_DEQ1                         00000054
DAP$W_DISPLAY1                     0000004C
DAP$W_FFB                          00000072
DAP$W_LRL                          00000070
DAP$W_MRS                          0000004A
DAP$W_PARTNER                      00000006
DAP$W_VERSION                      00000004
FAB$B_BKS                          0000003E
FAB$B_FAC                       =  00000016
FAB$B_FSZ                       =  0000003F
FAB$B_ORG                       =  0000001D
FAB$B_RAT                       =  0000001E
FAB$B_RFM                       =  0000001F
FAB$B_SHR                       =  00000017
FAB$C_FIX                       =  00000001
FAB$C_IDX                       =  00000020
FAB$C_REL                       =  00000010
FAB$C_SEQ                       =  00000000
FAB$C_STM                       =  00000004
FAB$C_STMCR                     =  00000006
FAB$C_STMLF                     =  00000005
FAB$C_UDF                       =  00000000
FAB$C_VAR                       =  00000002
FAB$C_VFC                       =  00000003
FAB$L_ALQ                       =  00000010
FAB$L_FOP                       =  00000004
FAB$L_MRN                       =  00000038
FAB$M_CBT                       =  00200000
FAB$M_CR                        =  00000002
FAB$M_CTG                       =  00100000
FAB$M_EXE                       =  00000080
FAB$M_FTN                       =  00000001
FAB$M_NFS                       =  00010000
FAB$M_PRN                       =  00000004
FAB$M_UFO                       =  00020000
FAB$V_BIO                       =  00000005
FAB$V_BLK                       =  00000003
FAB$V_BRO                       =  00000006
FAB$V_CBT                       =  00000015
FAB$V_CR                        =  00000001
FAB$V_CTG                       =  00000014
FAB$V_DEL                       =  00000002
FAB$V_DLT                       =  0000000F
FAB$V_FTN                       =  00000000
FAB$V_GET                       =  00000001
FAB$V_MXV                       =  00000001
FAB$V_NEF                       =  0000000A
FAB$V_NIL                       =  00000005
FAB$V_POS                       =  00000008
FAB$V_PRN                       =  00000002
FAB$V_PUT                       =  00000000
FAB$V_RCK                       =  00000017
FAB$V_RWC                       =  0000000B
FAB$V_RWO                       =  00000007
```

```
FAB$V_SCF                       = 0000000E
FAB$V_SHRDEL                    = 00000002
FAB$V_SHRGET                    = 00000001
FAB$V_SHRPUT                    = 00000000
FAB$V_SHRUPD                    = 00000003
FAB$V_SPL                       = 0000000D
FAB$V_SQO                       = 00000006
FAB$V_SUP                       = 00000002
FAB$V_TEF                       = 0000001C
FAB$V_TMD                       = 00000004
FAB$V_TMP                       = 00000003
FAB$V_TRN                       = 00000004
FAB$V_UPD                       = 00000003
FAB$V_UPI                       = 00000006
FAB$V_WCK                       = 00000009
FAB$W_BLS                       = 0000003C
FAB$W_DEQ                       = 00000014
FAB$W_MRS                       = 00000036
FAIL1                             00000026 R       01
FAIL2                             0000025B R       01
FAIL3                             000002FB R       01
IFB$B_BKS                       = 0000005E
IFB$B_FAC                       = 00000022
IFB$L_DEVBUFSIZ                 = 00000048
IFB$L_HBK                       = 00000070
IFB$L_NWA_PTR                   = 0000003C
IFB$V_CREATE                    = 00000032
IFB$V_DAP_OPEN                  = 0000003D
MAP_FAC                           0000033E R       01
MAP_SHR                           00000372 R       01
NT$BUILD_HEAD                     ******** X       01
NT$BUILD_TAIL                     ******** X       01
NT$CHK_ORG                        00000312 RG      01
NT$CRC_INIT                       ******** X       01
NT$CREATE                         00000000 RG      01
NT$CVT_BN4_EXT                    ******** X       01
NT$CVT_BN4_IMG                    ******** X       01
NT$DECODE_ALL_A                   ******** X       01
NT$DECODE_NAM                     ******** X       01
NT$ENCODE_ALL                     ******** X       01
NT$ENCODE_KEY                     ******** X       01
NT$ENCODE_PRO                     ******** X       01
NT$ENCODE_TIM_D                   ******** X       01
NT$ENCODE_TIM_R                   ******** X       01
NT$EXCH_CNF                       ******** X       01
NT$GET_FAC_SHR                    0000033E RG      01
NT$GET_FILESPEC                   ******** X       01
NT$LCL_FOP                        ******** X       01
NT$LCL_JOP                        ******** X       01
NT$LCL_RFM                        ******** X       01
NT$MAP_DEV_CHAR                   ******** X       01
NT$MAP_FOP                        000003C7 RG      01
NT$RECEIVE                        ******** X       01
NT$RECV_EXT_ATT                   ******** X       01
NT$RMT_ORG                        ******** X       01
NT$RMT_RAT                        ******** X       01
NT$RMT_RFM                        ******** X       01
NT$SCAN_ALLXAB                    ******** X       01
NT$SCAN_KEYXAB                    ******** X       01
NT$SCAN_NAMBLK                    ******** X       01
NT$SCAN_XABCHN                    ******** X       01
NT$TRANSMIT                       ******** X       01
NWA$B_ALLXABCNT                 = 0000011C
NWA$B_DAP_RAC                   = 000000C9
NWA$B_FILESYS                   = 000000C5
NWA$B_KEYXABCNT                 = 0000011D
NWA$B_NETSTRSIZ                 = 0000016F
NWA$B_NODBUFSIZ                 = 00000168
NWA$B_ORG                       = 000000C6
NWA$B_OSTYPE                    = 000000C4
NWA$B_RFM                       = 000000C7
NWA$B_RMS_RAC                   = 000000C8
NWA$C_BLN                       = 00000800
NWA$K_BLN                       = 00000800
NWA$L_ALLXABADR                 = 00000100
NWA$L_DATXABADR                 = 00000104
NWA$L_DEV                       = 000000C0
NWA$L_FHCXABADR                 = 00000108
NWA$L_KEYXABADR                 = 0000010C
NWA$L_MSG_MASK                  = 000000D4
NWA$L_PROXABADR                 = 00000110
NWA$L_RDTXABADR                 = 00000114
NWA$L_SAVE_FLGS                 = 00000128
NWA$L_SUMXABADR                 = 00000118
NWA$L_THREAD                    = 000000FC
NWA$L_XLTATTR                   = 00000238
NWA$L_XLTBUFFLG                 = 0000022C
NWA$L_XLTCNT                    = 00000228
NWA$L_XLTMAXINDX                = 00000234
NWA$L_XLTSIZ                    = 00000230
NWA$Q_ACS                       = 00000244
NWA$Q_BIGBUF                    = 00000170
NWA$Q_BLD                       = 000000F0
NWA$Q_FLG                       = 00000000
NWA$Q_INODE                     = 0000025C
NWA$Q_IOSB                      = 000000D8
NWA$Q_LNODE                     = 00000160
NWA$Q_LOGNAME                   = 0000023C
NWA$Q_NCB                       = 00000264
NWA$Q_RCV                       = 000000E0
NWA$Q_SAVE_DESC                 = 00000120
NWA$Q_XLTBUF1                   = 0000024C
NWA$Q_XLTBUF2                   = 00000254
NWA$Q_XMT                       = 000000E8
NWA$T_ACSBUF                    = 0000026C
NWA$T_AUXBUF                    = 000005E0
NWA$T_DAP                       = 00000000
NWA$T_INODEBUF                  = 000004AC
NWA$T_ITM_ATTR                  = 00000200
NWA$T_ITM_END                   = 00000224
NWA$T_ITM_LST                   = 00000200
NWA$T_ITM_MAXINDX               = 00000218
NWA$T_ITM_STRING                = 0000020C
NWA$T_NCBBUF                    = 0000052C
```

```
NWA$T_NODEBUF                   00000169
NWA$T_RCVBUF                    000001A0
NWA$T_SCAN                      00000100
NWA$T_TEMP                      00000120
NWA$T_XLTBUF1                   000002AC
NWA$T_XLTBUF2                   000003AC
NWA$T_XMTBUF                    000003C0
NWA$V_CVT_STM                 = 00000006
NWA$V_LAST_MSG                = 00000000
NWA$W_BUILD                     000000D2
NWA$W_DAPBUFSIZ                 000000CA
NWA$W_DIR_OFF                   000000CC
NWA$W_DISPLAY                   000000D0
NWA$W_FIL_OFF                   000000CE
NWA$W_JNLXABJOP                 0000011E
OVERRIDE_FAB                    0000005A R      01
PART1                           00000098 R      01
PART2A                          000000F5 R      01
PART2B                          00000158 R      01
PART3                           00000196 R      01
PIO$A_TRACE                     ******** X      01
RECV_ACK                        000002E0 R      01
RECV_ATT                        00000292 R      01
RECV_EXT_ATT                    000002C0 R      01
RECV_NAM                        000002CC R      01
RMS$ETALLOC                     ******** X      01
RMS$_CRE_STM                  = 00018069
SEND_ACC                        0000025C R      01
SEND_ALL                        00000201 R      01
SEND_ATT                        00000092 R      01
SEND_KEY                        000001DB R      01
SEND_PRO                        00000247 R      01
SEND_TIM                        00000227 R      01
TPT$C_NTCREATE                  ******** X    01
XAB$B_AOP                     = 00000008
XAB$L_NXT                     = 00000004
XAB$V_CBT                     = 00000005
XAB$V_CTG                     = 00000007
XAB$W_LRL                     = 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
NF$NETWORK	0000048C (1164.)	01 (1.)	PIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC BYTE
ABS	00000800 (2048.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.09	00:00:00.61
Command processing	123	00:00:00.57	00:00:04.73
Pass 1	382	00:00:15.49	00:00:43.61
Symbol table sort	0	00:00:01.77	00:00:02.26
Pass 2	174	00:00:03.49	00:00:07.20
Symbol table output	44	00:00:00.36	00:00:01.66
Psect synopsis output	1	00:00:00.02	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	758	00:00:21.81	00:01:00.12

The working set limit was 1650 pages.
82442 bytes (162 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1277 non-local and 83 local symbols.
891 source lines were read in Pass 1, producing 18 object records in Pass 2.
31 pages of virtual memory were used to define 30 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	22
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	26

1555 GETS were required to define 26 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:NTOCREATE/OBJ=OBJ$:NTOCREATE MSRC$:NTOCREATE/UPDATE=(ENH$:NTOCREATE)+LIB$:RMS/LIB
```

NT0ACCESS
LIS

NT0CLOSE
LIS

NT0BLDXAB
LIS

NT0CONN
LIS

NT0CREATE
LIS

NT0DAPIO
LIS

NT0DAPCRC
LIS

NT0ACCFIL
LIS

NT0BLKIO
LIS