```
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
```

```
NN       NN  TTTTTTTTTT    000000      GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT
NN       NN  TTTTTTTTTT    000000      GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT
NN       NN      TT       00      00  GG        EE              TT
NN       NN      TT       00      00  GG        EE              TT
NNNN     NN      TT       00    0000  GG        EE              TT
NNNN     NN      TT       00    0000  GG        EE              TT
NN  NN   NN      TT       00  00  00  GG        EEEEEEEE        TT
NN  NN   NN      TT       00  00  00  GG        EEEEEEEE        TT
NN    NNNN       TT       0000    00  GG  GGGGGG  EE              TT
NN    NNNN       TT       0000    00  GG  GGGGGG  EE              TT
NN       NN      TT       00      00  GG      GG  EE              TT          ....
NN       NN      TT       00      00  GG      GG  EE              TT          ....
NN       NN      TT         000000      GGGGGG    EEEEEEEEEE      TT          ....
NN       NN      TT         000000      GGGGGG    EEEEEEEEEE      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          $BEGIN  NTOGET,000,NF$NETWORK,<NETWORK GET/FIND RECORD>
0000     2
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 ;++
0000    30 ; Facility: RMS
0000    31 ;
0000    32 ; Abstract:
0000    33 ;
0000    34 ;       This module communicates with the File Access Listener (FAL) at the
0000    35 ;       remote node to perform the get and find operations.
0000    36 ;
0000    37 ; Environment: VAX/VMS, executive mode
0000    38 ;
0000    39 ; Author: James A. Krycka,      Creation Date:  09-DEC-1977
0000    40 ;
0000    41 ; Modified By:
0000    42 ;
0000    43 ;       V03-003 JAK0126         J A Krycka      08-SEP-1983
0000    44 ;               Add NT$RET_RRN routine to return relative record number in the
0000    45 ;               RAB$L_BKT field on sequential $GET, $FIND, and $PUT operations
0000    46 ;               operations to a relative file (per RMS documentation).
0000    47 ;
0000    48 ;       V03-002 JAK0121         J A Krycka      01-AUG-1983
0000    49 ;               Send key string in DAP Control message on $GET and $FIND for
0000    50 ;               sequential access of an indexed file with the key limit option
0000    51 ;               selected. Also, reorganize module a bit for clarity.
0000    52 ;
0000    53 ;       V03-001 KRM0090         Karl Malik      18-Mar-1983
0000    54 ;               Add support for STMLF and STMCR formats.
0000    55 ;
0000    56 ;--
```

```
                     0000     58          .SBTTL  DECLARATIONS
                     0000     59
                     0000     60 ;
                     0000     61 ; Include Files:
                     0000     62 ;
                     0000     63
                     0000     64          $BDBDEF                         ; Define BDB symbols
                     0000     65          $DAPPLGDEF                      ; Define DAP prologue symbols
                     0000     66          $DAPHDRDEF                      ; Define DAP message header
                     0000     67          $DAPCNFDEF                      ; Define DAP Configuration message
                     0000     68          $DAPCTLDEF                      ; Define DAP Control message
                     0000     69          $DAPDATDEF                      ; Define DAP Data message
                     0000     70          $DAPSTSDEF                      ; Define DAP Status message
                     0000     71          $FABDEF                         ; Define File Access Block symbols
                     0000     72          $IFBDEF                         ; Define IFAB symbols
                     0000     73          $IRBDEF                         ; Define IRAB symbols
                     0000     74          $NWADEF                         ; Define Network Work Area symbols
                     0000     75          $RABDEF                         ; Define Record Access Block symbols
                     0000     76          $RMSDEF                         ; Define RMS completion codes
                     0000     77
                     0000     78 ;
                     0000     79 ; Macros:
                     0000     80 ;
                     0000     81 ;        None
                     0000     82 ;
                     0000     83 ; Equated Symbols:
                     0000     84 ;
                     0000     85
00000A0D             0000     86 CRLF=^X0A0D                             ; ASCII codes for CR and LF
0000000A             0000     87 LF=^X0A                                 ; ASCII code for LF
0000000D             0000     88 CR=^X0D                                 ; ASCII code for CR
                     0000     89
                     0000     90          ASSUME  DAP$Q_DCODE_FLG EQ 0
                     0000     91          ASSUME  NWA$Q_FLG EQ 0
                     0000     92
                     0000     93 ;
                     0000     94 ; Own Storage:
                     0000     95 ;
                     0000     96 ;        None
                     0000     97 ;
```

```
0000     99          .SBTTL  NT$GET - PERFORM NETWORK GET RECORD FUNCTION
0000    100
0000    101 ;++
0000    102 ; NT$GET - engages in a DAP dialogue with the remote FAL to get (retrieve)
0000    103 ;       the specified record of a sequential, relative, or indexed file. This
0000    104 ;       routine supports both DAP file transfer and record transfer modes.
0000    105 ;
0000    106 ; Calling Sequence:
0000    107 ;
0000    108 ;       BSBW    NT$GET
0000    109 ;
0000    110 ; Input Parameters:
0000    111 ;
0000    112 ;       R4      BDB address
0000    113 ;       R8      RAB address
0000    114 ;       R9      IRAB address
0000    115 ;       R10     IFAB address
0000    116 ;       R11     Impure Area address
0000    117 ;
0000    118 ; Implicit Inputs:
0000    119 ;
0000    120 ;       BDB$L_ADDR
0000    121 ;       BDB$W_SIZE
0000    122 ;       DAP$V_DAPCRC
0000    123 ;       DAP$V_GEQ_V56
0000    124 ;       DAP$L_CRC_RSLT
0000    125 ;       IFB$L_NWA_PTR
0000    126 ;       IFB$V_SQO
0000    127 ;       IRB$B_MODE
0000    128 ;       NWA$V_FTM_EOF
0000    129 ;       NWA$V_FTM_INIT
0000    130 ;       NWA$V_FTM_STORE
0000    131 ;       NWA$B_ORG
0000    132 ;       NWA$B_RFM
0000    133 ;       RAB$B_KSZ
0000    134 ;       RAB$B_RAC
0000    135 ;       RAB$W_RFA
0000    136 ;       RAB$L_ROP
0000    137 ;       RAB$L_KBF
0000    138 ;       RAB$B_KRF
0000    139 ;
0000    140 ; Output Parameters:
0000    141 ;
0000    142 ;       R0      Status code (RMS)
0000    143 ;       R1-R3   Destroyed
0000    144 ;       AP      Destroyed
0000    145 ;
0000    146 ; Implicit Outputs:
0000    147 ;
0000    148 ;       BDB buffer contents
0000    149 ;       BDB$W_NUMB
0000    150 ;       DAP$L_CRC_RSLT
0000    151 ;       IRB$L_IOS and IRB$L_IOS4 zeroed
0000    152 ;       NWA$V_FTM_EOF
0000    153 ;       NWA$V_FTM_INIT cleared
0000    154 ;       NWA$V_FTM_RETRV
0000    155 ;       RAB$L_BKT
```

```
                     0000   156 ;       RAB$W_RFA
                     0000   157 ;       RAB$L_STV
                     0000   158 ;
                     0000   159 ; Completion Codes:
                     0000   160 ;
                     0000   161 ;       Standard RMS completion codes
                     0000   162 ;
                     0000   163 ; Side Effects:
                     0000   164 ;
                     0000   165 ;       None
                     0000   166 ;
                     0000   167 ;--
                     0000   168
                     0000   169 NT$GET::                                ; Entry point
                     0000   170          $TSTPT  NTGET                 ;
    00F0 8F    BB    0006   171          PUSHR   #^M<R4,R5,R6,R7>      ; Save registers
    56   54    D0    000A   172          MOVL    R4,R6                 ; Copy BDB address
57    3C AA    D0    000D   173          MOVL    IFB$L_NWA_PTR(R10),R7 ; Get address of NWA (and DAP)
      14 A6    B4    0011   174          CLRW    BDB$W_NUMB(R6)        ; Zero byte count in BDB
       FFE9'   30    0014   175          BSBW    NT$CHK_RAC            ; Validate record access mode
      12 50    E9    0017   176          BLBC    R0,10$                ; Branch on failure
    67   1B    E0    001A   177          BBS     #NWA$V_FTM_STORE,(R7),- ; $GET after $PUT not allowed in
           11          001D   178                  ERRFTMT               ;  file transfer mode
                     001E   179
                     001E   180          ASSUME  RAB$C_SEQ EQ 0
                     001E   181          ASSUME  RAB$C_KEY EQ 1
                     001E   182          ASSUME  RAB$C_RFA EQ 2
                     001E   183
                     001E   184          $CASEB  SELECTOR=RAB$B_RAC(R8)- ; Dispatch on access mode:
                     001E   185                  DISPL=<-              ;
                     001E   186                      GET_SEQ-          ;  Sequential record access
                     001E   187                      GET_KEY-          ;  Access by key value
                     001E   188                      GET_RFA-          ;  Access by record file address
                     001E   189                  >                     ; Value out-of-range
    0156   31    0029   190          BRW     ERRRAC                    ; Branch aid
    0166   31    002C   191 10$:     BRW     EXIT                      ; Branch aid
    0157   31    002F   192 ERRFTM1:BRW     ERRFTM                    ; Branch aid
                     0032   193
                     0032   194 ;+
                     0032   195 ; This support routine builds the first part of a Control message thru the
                     0032   196 ; CTLFUNC field for the get operation.
                     0032   197 ;-
                     0032   198
                     0032   199 BUILD_GET_CTL:                          ; Entry point
                     0032   200          $SETBIT #NWA$V_LAST_MSG,(R7)  ; Declare this last message to block
50    04    D0    0036   201          MOVL    #DAP$K_CTL_MSG,R0         ; Get message type value
   FFC4'   30    0039   202          BSBW    NT$BUILD_HEAD             ; Construct message header
85    01    90    003C   203          MOVB    #DAP$K_GET_READ,(R5)+     ; Store CTLFUNC field
           05    003F   204          RSB                               ; Exit
                     0040   205
                     0040   206 ;+
                     0040   207 ; Sequential access has been requested.
                     0040   208 ;
                     0040   209 ; Note: In record transfer mode a Control message must be sent for each record
                     0040   210 ;       retrieved, whereas, in file transfer mode only an initial Control
                     0040   211 ;       message is required.
                     0040   212 ;-
```

```
                     0040   213
                     0040   214 GET_SEQ:                                    ; Get operation with RAC = SEQ
05 6A    2D    E0    0040   215          BBS     #IFB$V_SQO,(R10),10$       ; Branch if in file transfer mode
   51    00    9A    0044   216          MOVZBL  #DAP$K_SEQ_ACC,R1          ; Set RAC for DAP message
         12    11    0047   217          BRB     GET_SEQ_COMMON             ; Join common code
03 67    1D    E1    0049   218 10$:     BBC     #NWA$V_FTM_EOF,(R7),20$    ; Check for EOF from a previous read
      0140    31    004D   219          BRW     ERREOF                     ; Branch aid
   67    19    E5    0050   220 20$:     BBCC    #NWA$V_FTM_INIT,(R7),-     ; Branch if no Control message required
         78          0053   221          GET_RECV_DAT                       ;  and turn off single-shot flag
                     0054   222          $SETBIT #NWA$V_FTM_RETRV,(R7)      ; Set file transfer mode retrieval flag
   51    03    9A    0058   223          MOVZBL  #DAP$K_SEQ_FILE,R1         ; Set RAC for DAP message
                     005B   224
                     005B   225 GET_SEQ_COMMON:                             ; Common code for get operation
      FFD4    30    005B   226          BSBW    BUILD_GET_CTL              ; Start building Control message
12 04 A8    0E    E1    005E   227          BBC     #RAB$V_LIM,RAB$L_ROP(R8),10$
20    00C6  C7    91    0063   228          CMPB    NWA$B_ORG(R7),#NWA$K_IDX   ; Fall thru if key limit option selected
         0B    12    0068   229          BNEQ    10$                        ;  for an indexed file
   85    0B    90    006A   230          MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     006D   231                    <DAP$M_KEY>!-            ;  control menu
                     006D   232                    <DAP$M_ROP>!-            ;
                     006D   233                    0>,(R5)+                 ;
   85    51    90    006D   234          MOVB    R1,(R5)+                   ; Store RAC field
      012A    30    0070   235          BSBW    NT$OBTAIN_KEY              ; Store key string
         48    11    0073   236          BRB     GET_SEND_CTL_COMMON        ; Join common code
   85    09    90    0075   237 10$:     MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     0078   238                    <DAP$M_ROP>!-            ;  control menu
                     0078   239                    0>,(R5)+                 ;
   85    51    90    0078   240          MOVB    R1,(R5)+                   ; Store RAC field
         40    11    007B   241          BRB     GET_SEND_CTL_COMMON        ; Join common code
                     007D   242
                     007D   243 ;+
                     007D   244 ; Access by relative record number or by key value has been requested.
                     007D   245 ; This operation must be performed in DAP record transfer mode.
                     007D   246 ;-
                     007D   247
                     007D   248 GET_KEY:                                    ; Get operation with RAC = KEY
AE 6A    2D    E0    007D   249          BBS     #IFB$V_SQO,(R10),ERRFTM1   ; Disallow file transfer mode
      FFAE    30    0081   250          BSBW    BUILD_GET_CTL              ; Start building Control message
20    00C6  C7    91    0084   251          CMPB    NWA$B_ORG(R7),#NWA$K_IDX   ; Branch if IDX organization
         0B    13    0089   252          BEQL    10$                        ;
   85    0B    90    008B   253          MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     008E   254                    <DAP$M_KEY>!-            ;  control menu
                     008E   255                    <DAP$M_ROP>!-            ;
                     008E   256                    0>,(R5)+                 ;
   85    01    90    008E   257          MOVB    #DAP$K_KEY_ACC,(R5)+       ; Store RAC field
      0109    30    0091   258          BSBW    NT$OBTAIN_KEY              ; Store key value
         27    11    0094   259          BRB     GET_SEND_CTL_COMMON        ; Join common code
   85    0F    90    0096   260 10$:     MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     0099   261                    <DAP$M_KEY>!-            ;  control menu
                     0099   262                    <DAP$M_KRF>!-            ;
                     0099   263                    <DAP$M_ROP>!-            ;
                     0099   264                    0>,(R5)+                 ;
   85    01    90    0099   265          MOVB    #DAP$K_KEY_ACC,(R5)+       ; Store RAC field
      00FE    30    009C   266          BSBW    NT$OBTAIN_KEY              ; Store key string
85    35 A8    90    009F   267          MOVB    RAB$B_KRF(R8),(R5)+        ; Store KRF field
         18    11    00A3   268          BRB     GET_SEND_CTL_COMMON        ; Join common code
                     00A5   269
```

```
                     00A5   270 ;+
                     00A5   271 ; Access by record file address has been requested.
                     00A5   272 ; This operation must be performed in DAP record transfer mode.
                     00A5   273 ;-
                     00A5   274
                     00A5   275 GET_RFA:                                ; Get operation with RAC = RFA
86 6A    2D   E0    00A5   276         BBS     #IFB$V_SQO,(R10),ERRFTM1; Disallow file transfer mode
     FF86     30    00A9   277         BSBW    BUILD_GET_CTL           ; Start building Control message
   85   0B   90    00AC   278         MOVB    #<<DAP$M_RAC>!-         ; Message must include these fields in
                    00AF   279                   <DAP$M_KEY>!-         ;  control menu
                    00AF   280                   <DAP$M_ROP>!-         ;
                    00AF   281                   0>,(R5)+              ;
   85   02   90    00AF   282         MOVB    #DAP$K_RFA_ACC,(R5)+    ; Store RAC field
   85   06   90    00B2   283         MOVB    #6,(R5)+                ; Store RFA in KEY field
85   10 A8   D0    00B5   284         MOVL    RAB$L_RFA0(R8),(R5)+    ;
85   14 A8   B0    00B9   285         MOVW    RAB$W_RFA4(R8),(R5)+    ;
                    00BD   286
                    00BD   287 ;+
                    00BD   288 ; Common code to build rest of the Control message and send it to partner.
                    00BD   289 ;-
                    00BD   290
                    00BD   291 GET_SEND_CTL_COMMON:                    ; Complete and send Control message
     FF40'   30     00BD   292         BSBW    NT$MAP_ROP              ; Store ROP as an extensible field
     FF3D'   30     00C0   293         BSBW    NT$BUILD_TAIL           ; Finish building message
     FF3A'   30     00C3   294         BSBW    NT$TRANSMIT             ; Send Control message to FAL
    03 50    E8     00C6   295         BLBS    R0,GET_RECV_DAT         ; Branch on success
     00C9    31     00C9   296         BRW     EXIT                    ; Branch aid
                    00CC   297
                    00CC   298 ;+
                    00CC   299 ; Receive DAP Data message from partner containing user record.
                    00CC   300 ;-
                    00CC   301
                    00CC   302 GET_RECV_DAT:                           ;
                    00CC   303         $SETBIT #DAP$K_DAT_MSG,DAP$L_MSG_MASK(R7)
                    00D1   304                                         ; Expect response of Data message
     FF2C'   30     00D1   305         BSBW    NT$RECEIVE              ; Get record
    7D 50    E9     00D4   306         BLBC    R0,CHKEOF               ; Branch on failure
       15    E1     00D7   307         BBC     #DAP$V_DAPCRC,-         ; Branch if partner does not support
 10 28 A7           00D9   308                 DAP$Q_SYSCAP(R7),10$    ;  file level CRC checksum
52   44 A7   7D     00DC   309         MOVQ    DAP$Q_FILEDATA(R7),R2   ; Put descriptor of record in <R2,R3>
   0000'CF   0B     00E0   310         CRC     W^NT$CRC_TABLE,-        ; Compute CRC (destroying R0-R3)
     20 A7          00E4   311                 DAP$L_CRC_RSLT(R7),-    ;  using result of previous CRC
   63   52          00E6   312                 R2,(R3)                 ;  calculation as initial CRC value
20 A7   50   D0     00E8   313         MOVL    R0,DAP$L_CRC_RSLT(R7)   ; Store CRC resultant value
52   44 A7   7D     00EC   314 10$:    MOVQ    DAP$Q_FILEDATA(R7),R2   ; Put descriptor of record in <R2,R3>
                    00F0   315
                    00F0   316 ;
                    00F0   317 ; Remove standard record terminator from record of a stream formatted file.
                    00F0   318 ;
                    00F0   319
                    00F0   320         ASSUME  FAB$C_UDF EQ 0
                    00F0   321         ASSUME  FAB$C_FIX EQ 1
                    00F0   322         ASSUME  FAB$C_VAR EQ 2
                    00F0   323         ASSUME  FAB$C_VFC EQ 3
                    00F0   324         ASSUME  FAB$C_STM EQ 4
                    00F0   325         ASSUME  FAB$C_STMLF EQ 5
                    00F0   326         ASSUME  FAB$C_STMCR EQ 6
```

```
                        00F0   327
                        00F0   328          $CASEB  SELECTOR=NWA$B_RFM(R7)- ; Dispatch on record format:
                        00F0   329                  BASE=#FAB$C_STM-       ;
                        00F0   330                  DISPL=<-               ;
                        00F0   331                      20$-               ;  STM
                        00F0   332                      30$-               ;  STMLF
                        00F0   333                      40$-               ;  STMCR
                        00F0   334                  >                      ;  UDF, FIX, VAR, VFC
                 2E 11  00FC   335          BRB     60$                    ; Bypass stream specific processing
                        00FE   336
                        00FE   337 ;
                        00FE   338 ; We've received a record in stream format--strip trailing CRLF if present.
                        00FE   339 ;
                        00FE   340
           02 52 B1     00FE   341 20$:    CMPW    R2,#2                  ; Check record length
              29 1F     0101   342         BLSSU   60$                    ; Branch if less than two
     51    53 52 C1     0103   343         ADDL3   R2,R3,R1               ; Get address past record
  FE A1    0A0D 8F B1   0107   344         CMPW    #CRLF,-2(R1)           ; If record terminates with CRLF
              1D 12     010D   345         BNEQ    60$                    ;  then remove the characters
           52 02 C2     010F   346         SUBL2   #2,R2                  ;  and reduce the record length
              18 11     0112   347         BRB     60$                    ; Branch to common code
                        0114   348
                        0114   349 ;
                        0114   350 ; We've received a record in STMLF or STMCR format--strip last character.
                        0114   351 ;
                        0114   352
           50 0A 9A     0114   353 30$:    MOVZBL  #LF,R0                 ; Make LF terminator to check against
              03 11     0117   354         BRB     50$                    ;
           50 0D 9A     0119   355 40$:    MOVZBL  #CR,R0                 ; Make  CR terminator to check against
              52 B5     011C   356 50$:    TSTW    R2                     ; Check record length
              0C 13     011E   357         BEQL    60$                    ; Branch if null record
     51    53 52 C1     0120   358         ADDL3   R2,R3,R1               ; Get address past record
  FF A1    50 91        0124   359         CMPB    R0,-1(R1)              ; Check for terminator present
              02 12     0128   360         BNEQ    60$                    ; Branch if no terminator to strip
              52 D7     012A   361         DECL    R2                     ; Strip terminator
                        012C   362
                        012C   363 ;
                        012C   364 ; Copy record to BDB buffer.
                        012C   365 ;
                        012C   366
  16 A6    52 B1        012C   367 60$:    CMPW    R2,BDB$W_SIZE(R6)      ; Is record too big for BDB buffer?
              45 1A     0130   368         BGTRU   ERRRTB                 ; Branch if so
  14 A6    52 B0        0132   369         MOVW    R2,BDB$W_NUMB(R6)      ; Update byte count in BDB
18 B6  63  52 28        0136   370         MOVC3   R2,(R3),@BDB$L_ADDR(R6) ; Move record to BDB buffer
                        013E   371
                        013B   372 ;+
                        013B   373 ; Receive DAP Status message from partner if we are in record transfer mode
                        013B   374 ; and return record file address of the first block accessed.
                        013B   375 ;-
                        013B   376
                        013B   377 GET_RECV_STS:                           ;
                        013B   378          RMSSUC                         ; Anticipate success
  53 6A    2D E0        013E   379          BBS     #IFB$V_SQO,(R10),EXIT  ; Branch if in file transfer mode
  4F 67    24 E1        0142   380          BBC     #DAP$V_GEQ_V56,(R7),EXIT; Branch if partner uses DAP before V5.6
                        0146   381 ; ***** $SETBIT #DAP$K_STS_MSG,DAP$L_MSG_MASK(R7); Implied for receive
        FEB7'   30      0146   382          BSBW    NT$RECEIVE             ; Obtain status of get request
     49 50      E9      0149   383          BLBC    R0,EXIT                ; Branch on failure
```

```
                 008D  30  014C   384          BSBW    NT$RET_RFA              ; Return RFA value to user RAB
                 0097  30  014F   385          BSBW    NT$RET_RRN              ; Return relative record number in BKT
                       0152   386                                              ;  field (if appropriate)
                   41  11  0152   387          BRB     EXIT                    ; Branch aid
                       0154   388
                       0154   389 ;
                       0154   390 ; Check for end-of-file.
                       0154   391 ;
                       0154   392
               3D 6A   2D  51  0154   393 CHKEOF: BBC     #IFB$V_SQO,(R10),EXIT   ; Branch if record transfer mode
            827A 8F    50  B1  0158   394          CMPW    R0,#<RMS$_EOF&^XFFFF>   ; Is it an end-of-file?
                   36  12  015D   395          BNEQ    EXIT                    ; Branch if not
                       015F   396          $SETBIT #NWA$V_FTM_EOF,(R7)     ; Denote that end-of-file has been
                   30  11  0163   397          BRB     EXIT                    ;  reached so that EOF status can be
                       0165   398                                              ;  returned on next read attempt
                       0165   399 ;+
                       0165   400 ; Common exit code for $GET and $FIND.
                       0165   401 ;-
                       0165   402
                   01  BA  0165   403 ERRKBF: POPR    #^M<R0>                 ; Discard return address on stack
                       0167   404          RMSERR  KBF                     ; Key buffer error
                   27  11  016C   405          BRB     EXIT                    ;
                   01  BA  016E   406 ERRKSZ: POPR    #^M<R0>                 ; Discard return address on stack
                       0170   407          RMSERR  KSZ                     ; Key buffer size error
                   1E  11  0175   408          BRB     EXIT                    ;
                       0177   409 ERRRTB: RMSERR  RTB                     ; Record too big
                 0C A8   52  3C  017C   410          MOVZWL  R2,RAB$L_STV(R8)        ; Return record size in STV
                   13  11  0180   411          BRB     EXIT                    ;
                       0182   412 ERRRAC: RMSERR  RAC                     ; Invalid record access value
                   0C  11  0187   413          BRB     EXIT                    ;
                       0189   414 ERRFTM: RMSERR  FTM                     ; File transfer mode error
                   05  11  018E   415          BRB     EXIT                    ;
                       0190   416 ERREOF: RMSERR  EOF                     ; Declare end-of-file
                 0C A9   7C  0195   417 EXIT:   CLRQ    IRB$L_IOS(R9)           ; Zero I/O status block
                       0198   418                                              ;  (pertinent status info is already in
                       0198   419                                              ;  R0, EDB$W_NUMB, and RAB$L_STV)
                00F0 8F    BA  0198   420          POPR    #^M<R4,R5,R6,R7>        ; Restore registers
                       05  019C   421          RSB                             ; Exit with RMS code in R0
```

```
                                                C 14
                                019D    423          .SBTTL  NT$OBTAIN_KEY - STORE KEY VALUE
                                019D    424
                                019D    425 ;++
                                019D    426 ; NT$OBTAIN_KEY - stores the key value as an image field in the DAP message.
                                019D    427 ; For sequential and relative files, the key is a relative record number.
                                019D    428 ; For indexed files, the key is a string (text or packed decimal) or a binary
                                019D    429 ; number (signed or unsigned).
                                019D    430 ;--
                                019D    431
                                019D    432 NT$OBTAIN_KEY::                             ; Entry point
          50    34 A8    9A     019D    433          MOVZBL  RAB$B_KSZ(R8),R0           ; Get length of key
          51    30 A8    D0     01A1    434          MOVL    RAB$L_KBF(R8),R1           ; Get address of key buffer
      20  00C6 C7    91         01A5    435          CMPB    NWA$B_ORG(R7),#NWA$K_IDX   ; Branch if IDX organization
                17    13         01AA    436          BEQL    20$                        ;
                                01AC    437
                                01AC    438 ;
                                01AC    439 ; This is a sequential or relative file.
                                01AC    440 ;
                                01AC    441
    61  04    0A A9    0C         01AC    442          PROBER  IRB$B_MODE(R9),#4,(R1)     ; Branch if key buffer is not readable
                B2    13         01B1    443          BEQL    ERRKBF                     ;
                50    95         01B3    444          TSTB    R0                         ; Check key size
                05    13         01B5    445          BEQL    10$                        ; Ok if default of zero specified
            04  50    91         01B7    446          CMPB    R0,#4                      ; Check key size again
                B2    12         01BA    447          BNEQ    ERRKSZ                     ; Branch if not = 4
            51  61    D0         01BC    448 10$:     MOVL    (R1),R1                    ; Get key value
               FE3E'  30         01BF    449          BSBW    NT$CVT_BN4_IMG             ; Store KEY as an image field
                05    05         01C2    450          RSB                                ; Exit
                                01C3    451
                                01C3    452 ;
                                01C3    453 ; This is an indexed file.
                                01C3    454 ;
                                01C3    455 ; Note: At this point RMS does not know the type of key being used (DTP value);
                                01C3    456 ;       FAL (actually the file system at the remote node) has this information,
                                01C3    457 ;       but this has not be communicated back to RMS (at the local node).
                                01C3    458 ;       Consequently, RMS will package the key value for remote access to an
                                01C3    459 ;       indexed file as follows:
                                01C3    460 ;       (1) If the key size is explicilty stated (required for string keys
                                01C3    461 ;           anyway), RMS will use the specified size to obtain the key value
                                01C3    462 ;           from the key buffer.
                                01C3    463 ;       (2) If the key size in zero (normally allowed for binary keys), RMS
                                01C3    464 ;           will default the size to 4 and obtain the key value from the key
                                01C3    465 ;           buffer. This will result in a restriction: for 2-byte and 8-byte
                                01C3    466 ;           binary keys (DTP = BN2, IN2, BN8, and IN8) the user will have to
                                01C3    467 ;           explicitly specify the correct key size (2 or 8) for a network
                                01C3    468 ;           request. A more general solution should be considered for a future
                                01C3    469 ;           release to remove this restriction.
                                01C3    470 ;
                                01C3    471
                50    D5         01C3    472 20$:     TSTL    R0                         ; Branch if key size is explicitly
                03    12         01C5    473          BNEQ    30$                        ;  stated
            50  04    D0         01C7    474          MOVL    #4,R0                      ; Default to key size of 4 bytes
    61  50    0A A9    0C         01CA    475 30$:     PROBER  IRB$B_MODE(R9),R0,(R1)     ; Branch if key buffer is not readable
                94    13         01CF    476          BEQL    ERRKBF                     ;
            85  50    90         01D1    477          MOVB    R0,(R5)+                   ; Store KEY as an image field but do not
        65  61  50    28         01D4    478          MOVC3   R0,(R1),(R5)               ;  suppress leading zero bytes (as is
                                01D8    479                                             ;  done for relative record numbers)
```

```
                55   53   D0  01D8   480          MOVL    R3,R5                   ; Restore pointer to correct register
                          05  01DB   481          RSB                             ; Exit
```

```
                     01DC   483          .SBTTL  NT$RET_RFA - RETURN RFA VALUE
                     01DC   484
                     01DC   485 ;++
                     01DC   486 ; NT$RET_RFA - returns the record file address value to the RFA field of the
                     01DC   487 ; user RAB.
                     01DC   488 ;
                     01DC   489 ; R1 is destroyed on output.
                     01DC   490 ;--
                     01DC   491
                     01DC   492 NT$RET_RFA::                                 ; Entry point
51    42 A7    DE    01DC   493          MOVAL   DAP$W_RFA(R7),R1             ; Get address of RFA returned by FAL
10 A8    81    D0    01E0   494          MOVL    (R1)+,RAB$L_RFA0(R8)         ; Move 6-byte RFA value to user RAB
14 A8    61    B0    01E4   495          MOVW    (R1),RAB$W_RFA4(R8)          ;
                     05    01E8   496          RSB                          ; Exit
```

```
                     01E9   498          .SBTTL  NT$RET_RRN - RETURN RRN VALUE
                     01E9   499
                     01E9   500 ;++
                     01E9   501 ; NT$RET_RRN - returns the relative record number value to the BKT field of the
                     01E9   502 ; user RAB on sequential access to a relative file.
                     01E9   503 ;--
                     01E9   504
                     01E9   505 NT$RET_RRN::                                 ; Entry point
10   00C6 C7   91   01E9   506          CMPB    NWA$B_ORG(R7),#NWA$K_REL; Branch if this is not a relative
          0B   12   01EE   507          BNEQ    10$                         ;  file
  00    1E A8   91   01F0   508          CMPB    RAB$B_RAC(R8),#RAB$C_SEQ; Branch if this is not sequential
          05   12   01F4   509          BNEQ    10$                         ;  access to the file
        48 A7   D0   01F6   510          MOVL    DAP$L_RECNUM2(R7),-         ; Store relative record number from
        38 A8        01F9   511                  RAB$L_BKT(R8)               ;  Status message in BKT field of RAB
                05   01FB   512 10$:     RSB                                 ; Exit
```

```
                     514          .SBTTL  NT$FIND - PERFORM NETWORK FIND RECORD FUNCTION
                     515
                     516 ;++
                     517 ; NT$FIND - engages in a DAP dialogue with the remote FAL to find (locate)
                     518 ;       the specified record of a sequential, relative, or indexed file.
                     519 ;
                     520 ; Calling Sequence:
                     521 ;
                     522 ;       BSBW    NT$FIND
                     523 ;
                     524 ; Input Parameters:
                     525 ;
                     526 ;       R4      BDB address
                     527 ;       R8      RAB address
                     528 ;       R9      IRAB address
                     529 ;       R10     IFAB address
                     530 ;       R11     Impure Area address
                     531 ;
                     532 ; Implicit Inputs:
                     533 ;
                     534 ;       DAP$V_GEQ_V56
                     535 ;       IFB$L_NWA_PTR
                     536 ;       IFB$V_SQO
                     537 ;       IRB$B_MODE
                     538 ;       NWA$B_ORG
                     539 ;       RAB$B_KSZ
                     540 ;       RAB$B_RAC
                     541 ;       RAB$W_RFA
                     542 ;       RAB$L_ROP
                     543 ;       RAB$L_KBF
                     544 ;       RAB$B_KRF
                     545 ;
                     546 ; Output Parameters:
                     547 ;
                     548 ;       R0      Status code (RMS)
                     549 ;       R1-R3   Destroyed
                     550 ;       AP      Destroyed
                     551 ;
                     552 ; Implicit Outputs:
                     553 ;
                     554 ;       IRB$L_IOS and IRB$L_IOS4 zeroed
                     555 ;       RAB$L_BKT
                     556 ;       RAB$W_RFA
                     557 ;
                     558 ; Completion Codes:
                     559 ;
                     560 ;       Standard RMS completion codes
                     561 ;
                     562 ; Side Effects:
                     563 ;
                     564 ;       None
                     565 ;
                     566 ;--
                     567
                     568 NT$FIND::                                    ; Entry point
                     569          $TSTPT  NTFIND                     ;
00F0 8F    BB  0202  570          PUSHR   #^M<R4,R5,R6,R7>           ; Save registers
```

```
                1F 6A    2D    E0  0206   571          BBS     #IFB$V_SQO,(R10),ERRFTM2; Network find function not allowed
                                   020A   572                                         ;  if file transfer mode selected
                57   3C AA    D0  020A   573          MOVL    IFB$L_NWA_PTR(R10),R7   ; Get address of NWA (and DAP)
                     14 A4    B4  020E   574          CLRW    BDB$W_NUMB(R4)          ; Zero byte count in BDB
                      FDEC'   30  0211   575          BSBW    NT$CHK_RAC              ; Validate record access mode
                     0E 50    E9  0214   576          BLBC    R0,10$                  ; Branch on failure
                                 0217   577
                                 0217   578          ASSUME  RAB$C_SEQ EQ 0
                                 0217   579          ASSUME  RAB$C_KEY EQ 1
                                 0217   580          ASSUME  RAB$C_RFA EQ 2
                                 0217   581
                                 0217   582          $CASEB  SELECTOR=RAB$B_RAC(R8)- ; Dispatch on access mode:
                                 0217   583                  DISPL=<-               ;
                                 0217   584                      FIND_SEQ-          ;  Sequential record access
                                 0217   585                      FIND_KEY-          ;  Access by relative record #
                                 0217   586                      FIND_RFA-          ;  Access by record file address
                                 0217   587                  >                      ; Value out-of-range
                       FF5D      31  0222   588          BRW     ERRRAC                  ; Branch aid
                       FF6D      31  0225   589 10$:     BRW     EXIT                    ; Branch aid
                       FF5E      31  0228   590 ERRFTM2:BRW     ERRFTM                  ; Branch aid
                                 022B   591
                                 022B   592 ;+
                                 022B   593 ; This support routine builds the first part of a Control message thru the
                                 022B   594 ; CTLFUNC field for the find operation.
                                 022B   595 ;-
                                 022B   596
                                 022B   597 BUILD_FIND_CTL:                          ; Entry point
                                 022B   598          $SETBIT #NWA$V_LAST_MSG,(R7)    ; Declare this last message to block
                     50   04    D0  022F   599          MOVL    #DAP$K_CTL_MSG,R0       ; Get message type value
                        FDCB'   30  0232   600          BSBW    NT$BUILD_HEAD           ; Construct message header
                     85   0E    90  0235   601          MOVB    #DAP$K_FIND,(R5)+       ; Store CTLFUNC field
                               05  0238   602          RSB                             ; Exit
                                   0239   603
                                   0239   604 ;+
                                   0239   605 ; Sequential access has been requested.
                                   0239   606 ;-
                                   0239   607
                                   0239   608 FIND_SEQ:                             ; Find operation with RAC = SEQ
                          FFEF     30  0239   609          BSBW    BUILD_FIND_CTL          ; Start building Control message
                12 04 A8    0E    E1  023C   610          BBC     #RAB$V_LIM,RAB$L_ROP(R8),10$
                 20   00C6 C7      91  0241   611          CMPB    NWA$B_ORG(R7),#NWA$K_IDX; Fall thru if key limit option selected
                           0B      12  0246   612          BNEQ    10$                     ;  for an indexed file
                     85    0B      90  0248   613          MOVB    #<<DAP$M_RAC>!-         ; Message must include these fields in
                                       024B   614                    <DAP$M_KEY>!-       ;  control menu
                                       024B   615                    <DAP$M_ROP>!-       ;
                                       024B   616                    0>,(R5)+            ;
                     85    00      90  024B   617          MOVB    #DAP$K_SEQ_ACC,(R5)+    ; Store RAC field
                          FF4C     30  024E   618          BSBW    NT$OBTAIN_KEY           ; Store key string
                           40      11  0251   619          BRB     FIND_SEND_CTL_COMMON    ; Join common code
                     85    09      90  0253   620 10$:     MOVB    #<<DAP$M_RAC>!-         ; Message must include these fields in
                                       0256   621                    <DAP$M_ROP>!-       ;  control menu
                                       0256   622                    0>,(R5)+            ;
                     85    00      90  0256   623          MOVB    #DAP$K_SEQ_ACC,(R5)+    ; Store RAC field
                           38      11  0259   624          BRB     FIND_SEND_CTL_COMMON    ; Join common code
                                       025B   625
                                       025B   626 ;+
                                       025B   627 ; Access by relative record number or by key value has been requested.
```

```
                     025B   628 ; This operation must be performed in DAP record transfer mode.
                     025B   629 ;-
                     025B   630
                     025B   631 FIND_KEY:                                    ; Find operation with RAC = KEY
        FFCD   30    025B   632          BSBW    BUILD_FIND_CTL             ; Start building Control message
20  00C6 C7    91    025E   633          CMPB    NWA$B_ORG(R7),#NWA$K_IDX   ; Branch if IDX organization
          0B   13    0263   634          BEQL    10$                        ;
     85   0B   90    0265   635          MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     0268   636                    <DAP$M_KEY>!-            ;  control menu
                     0268   637                    <DAP$M_ROP>!-            ;
                     0268   638                    0>,(R5)+                 ;
     85   01   90    0268   639          MOVB    #DAP$K_KEY_ACC,(R5)+       ; Store RAC field
        FF2F   30    026B   640          BSBW    NT$OBTAIN_KEY              ; Store key value
          23   11    026E   641          BRB     FIND_SEND_CTL_COMMON       ; Join common code
     85   0F   90    0270   642 10$:     MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     0273   643                    <DAP$M_KEY>!-            ;  control menu
                     0273   644                    <DAP$M_KRF>!-            ;
                     0273   645                    <DAP$M_ROP>!-            ;
                     0273   646                    0>,(R5)+                 ;
     85   01   90    0273   647          MOVB    #DAP$K_KEY_ACC,(R5)+       ; Store RAC field
        FF24   30    0276   648          BSBW    NT$OBTAIN_KEY              ; Store key string
  85    35 A8   90    0279   649          MOVB    RAB$B_KRF(R8),(R5)+        ; Store KRF field
          14   11    027D   650          BRB     FIND_SEND_CTL_COMMON       ; Join common code
                     027F   651
                     027F   652 ;+
                     027F   653 ; Access by record file address has been requested.
                     027F   654 ; This operation must be performed in DAP record transfer mode.
                     027F   655 ;-
                     027F   656
                     027F   657 FIND_RFA:                                    ; Find operation with RAC = RFA
        FFA9   30    027F   658          BSBW    BUILD_FIND_CTL             ; Start building Control message
     85   0B   90    0282   659          MOVB    #<<DAP$M_RAC>!-            ; Message must include these fields in
                     0285   660                    <DAP$M_KEY>!-            ;  control menu
                     0285   661                    <DAP$M_ROP>!-            ;
                     0285   662                    0>,(R5)+                 ;
     85   02   90    0285   663          MOVB    #DAP$K_RFA_ACC,(R5)+       ; Store RAC field
     85   06   90    0288   664          MOVB    #6,(R5)+                   ; Store RFA in KEY field
  85    10 A8   D0    028B   665          MOVL    RAB$L_RFA0(R8),(R5)+       ;
  85    14 A8   B0    028F   666          MOVW    RAB$W_RFA4(R8),(R5)+       ;
                     0293   667
                     0293   668 ;+
                     0293   669 ; Common code to build rest of the Control message and send it to partner.
                     0293   670 ;-
                     0293   671
                     0293   672 FIND_SEND_CTL_COMMON:                        ; Complete and send Control message
      FD6A'  30      0293   673          BSBW    NT$MAP_ROP                 ; Store ROP as an extensible field
      FD67'  30      0296   674          BSBW    NT$BUILD_TAIL              ; Finish building message
      FD64'  30      0299   675          BSBW    NT$TRANSMIT                ; Send Control message to FAL
     10 50   E9      029C   676          BLBC    R0,EXIT1                   ; Branch on failure
                     029F   677
                     029F   678 ;+
                     029F   679 ; Receive DAP Status message from partner.
                     029F   680 ;-
                     029F   681
                     029F   682 FIND_RECV_STS:                               ;
0C 67    24   E1     029F   683          BBC     #DAP$V_GEQ_V56,(R7),EXIT1  ;Branch if partner uses DAP before V5.6
                     02A3   684 ; ***** $SETBIT #DAP$K_STS_MSG,DAP$L_MSG_MASK(R7); Implied for receive
```

```
 FD5A'   30   02A3    685          BSBW    NT$RECEIVE              ; Obtain status of find request
06 50    E9   02A6    686          BLBC    R0,EXIT1                ; Branch on failure
 FF30    30   02A9    687          BSBW    NT$RET_RFA              ; Return RFA value to user RAB
 FF3A    30   02AC    688          BSBW    NT$RET_RRN              ; Return relative record number in BKT
              02AF    689                                          ;  field (if appropriate)
 FEE3    31   02AF    690 EXIT1:   BRW     EXIT                    ; Branch aid
              02B2    691
              02B2    692          .END                            ; End of module
```

```
$$.PSECT_EP                     = 00000000
$$COUNT                         = 00000003
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
BDB$L_ADDR                      = 00000018
BDB$W_NUMB                      = 00000014
BDB$W_SIZE                      = 00000016
BUILD_FIND_CTL                    0000022B R      01
BUILD_GET_CTL                     00000032 R      01
CHKEOF                            00000154 R      01
CR                              = 0000000D
CRLF                            = 00000A0D
DAP$B_BITCNT                      00000035
DAP$B_BLKCNT                      00000056
DAP$B_CTLFUNC                     00000040
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_FILESYS                     00000043
DAP$B_FLAGS                       00000031
DAP$B_KRF                         00000047
DAP$B_LEN256                      00000034
DAP$B_LENGTH                      00000033
DAP$B_OSTYPE                      00000042
DAP$B_RAC                         00000046
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_BLN                         000000C0
DAP$K_CTL_MSG                   = 00000004
DAP$K_DAT_MSG                   = 00000008
DAP$K_FIND                      = 0000000E
DAP$K_GET_READ                  = 00000001
DAP$K_KEY_ACC                   = 00000001
DAP$K_RFA_ACC                   = 00000002
DAP$K_SEQ_ACC                   = 00000000
DAP$K_SEQ_FILE                  = 00000003
DAP$L_CMWA                        00000030
DAP$L_CRC_RSLT                    00000020
DAP$L_DCODE_STS                   00000018
DAP$L_MSG_MASK                    0000001C
DAP$L_RECNUM1                     00000040
DAP$L_RECNUM2                     00000048
DAP$L_ROP                         00000050
DAP$L_SSPWA                       00000080
DAP$L_STV                         0000004C
DAP$L_TEMP                        00000090
DAP$M_BITCNT                    = 00000008
DAP$M_BLKCNT                    = 00000040
DAP$M_KEY                       = 00000002
DAP$M_KRF                       = 00000004
DAP$M_RAC                       = 00000001
DAP$M_ROP                       = 00000008
DAP$M_SEGMENT                   = 00000040
DAP$M_TMP1$                     = 00000008
DAP$M_TMP2$                     = FFF80000
DAP$Q_DCODE_FLG                   00000000
DAP$Q_FILEDATA                    00000044
DAP$Q_KEY                         00000048
DAP$Q_MSG_BUF1                    00000008
DAP$Q_MSG_BUF2                    00000010
DAP$Q_STX                         00000050
DAP$Q_SYSCAP                      00000028
DAP$Q_SYSPEC                      00000038
DAP$V_DAPCRC                    = 00000015
DAP$V_GEQ_V56                   = 00000024
DAP$W_BUFSIZ                      00000040
DAP$W_CTLMENU                     00000044
DAP$W_DISPLAY2                    00000054
DAP$W_PARTNER                     00000006
DAP$W_RFA                         00000042
DAP$W_STSCODE                     00000040
DAP$W_VERSION                     00000004
ERREOF                            00000190 R      01
ERRFTM                            00000189 R      01
ERRFTM1                           0000002F R      01
ERRFTM2                           00000228 R      01
ERRKBF                            00000165 R      01
ERRKSZ                            0000016E R      01
ERRRAC                            00000182 R      01
ERRRTB                            00000177 R      01
EXIT                              00000195 R      01
EXIT1                             000002AF R      01
FAB$C_FIX                       = 00000001
FAB$C_STM                       = 00000004
FAB$C_STMCR                     = 00000006
FAB$C_STMLF                     = 00000005
FAB$C_UDF                       = 00000000
FAB$C_VAR                       = 00000002
FAB$C_VFC                       = 00000003
FIND_KEY                          0000025B R      01
FIND_RECV_STS                     0000029F R      01
FIND_RFA                          0000027F R      01
FIND_SEND_CTL_COMMON              00000293 R      01
FIND_SEQ                          00000239 R      01
GET_KEY                           0000007D R      01
GET_RECV_DAT                      000000CC R      01
GET_RECV_STS                      0000013E R      01
GET_RFA                           000000A5 R      01
GET_SEND_CTL_COMMON               000000BD R      01
GET_SEQ                           00000040 R      01
GET_SEQ_COMMON                    0000005B R      01
IFB$L_NWA_PTR                   = 0000003C
IFB$V_SQO                       = 0000002D
IRB$B_MODE                      = 0000000A
IRB$L_IOS                       = 0000000C
```

```
LF                              = 0000000A
NT$BUILD_HEAD                     ********    X   01
NT$BUILD_TAIL                     ********    X   01
NT$CHK_RAC                        ********    X   01
NT$CRC_TABLE                      ********    X   01
NT$CVT_BN4_IMG                    ********    X   01
NT$FIND                           000001FC RG     01
NT$GET                            00000000 RG     01
NT$MAP_ROP                        ********      X 01
NT$OBTAIN_KEY                     0000019D RG     01
NT$RECEIVE                        ********    X   01
NT$RET_RFA                        000001DC RG     01
NT$RET_RRN                        000001E9 RG     01
NT$TRANSMIT                       ********    X   01
NWA$B_ALLXABCNT                   0000011C
NWA$B_DAP_RAC                     000000C9
NWA$B_FILESYS                     0000C0C5
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$K_BLN                         00000800
NWA$K_IDX                       = 00000020
NWA$K_REL                       = 00000010
NWA$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$V_FTM_EOF                   = 0000001D
NWA$V_FTM_INIT                  = 00000019
NWA$V_FTM_RETRV                 = 0000001A
NWA$V_FTM_STORE                 = 0000001B
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$W_JNLXABJOP                   0000011E
PIO$A_TRACE                       ********    X   01
RAB$B_KRF                       = 00000035
RAB$B_KSZ                       = 00000034
RAB$B_RAC                       = 0000001E
RAB$C_KEY                       = 00000001
RAB$C_RFA                       = 00000002
RAB$C_SEQ                       = 00000000
RAB$L_BKT                       = 00000038
RAB$L_KBF                       = 00000030
RAB$L_RFAO                      = 00000010
RAB$L_ROP                       = 00000004
RAB$L_STV                       = 0000000C
RAB$V_LIM                       = 0000000E
RAB$W_RFA4                      = 00000014
RMS$_EOF                        = 0001827A
RMS$_FTM                        = 000187C4
RMS$_KBF                        = 0001858C
RMS$_KSZ                        = 000185A4
RMS$_RAC                        = 00018644
RMS$_RTB                        = 000181A8
TPT$C_NTFIND                      ********    X   01
TPT$L_NTGET                       ********    X   01
```

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	000002B2 (690.)	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	29	00:00:00.10	00:00:00.50
Command processing	106	00:00:00.60	00:00:03.38
Pass 1	373	00:00:15.12	00:00:42.51
Symbol table sort	0	00:00:01.91	00:00:03.11
Pass 2	132	00:00:03.07	00:00:09.55
Symbol table output	26	00:00:00.20	00:00:00.74
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	670	00:00:21.03	00:00:59.82

The working set limit was 1500 pages.
79333 bytes (155 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1389 non-local and 26 local symbols.
692 source lines were read in Pass 1, producing 15 object records in Pass 2.
30 pages of virtual memory were used to define 29 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	21
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	25

1618 GETS were required to define 25 macros.

There were no errors, warnings or information messages.

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

NT0DAPRMS
LIS

NT0GET
LIS

NT0NWASET
LIS

NT0EXTEND
LIS

NT0DECODE
LIS

NT0ENCODE
LIS

NT0ERASE
LIS

NT0DISCON
LIS

NT0DISPLY
LIS

NT0MISC
LIS