```
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    EEEEEEEEEE  XX      XX  TTTTTTTTTT  EEEEEEEEEE  NN      NN  DDDDDDDD
NN      NN  TTTTTTTTTT    000000    EEEEEEEEEE  XX      XX  TTTTTTTTTT  EEEEEEEEEE  NN      NN  DDDDDDDD
NN      NN      TT      00      00  EE          XX      XX      TT      EE          NN      NN  DD      DD
NN      NN      TT      00      00  EE          XX      XX      TT      EE          NN      NN  DD      DD
NNNN    NN      TT      00    0000  EE            XX  XX        TT      EE          NNNN    NN  DD      DD
NNNN    NN      TT      00    0000  EE            XX  XX        TT      EE          NNNN    NN  DD      DD
NN  NN  NN      TT      00  00  00  EEEEEEEE        XX          TT      EEEEEEEE    NN  NN  NN  DD      DD
NN  NN  NN      TT      00  00  00  EEEEEEEE        XX          TT      EEEEEEEE    NN  NN  NN  DD      DD
NN    NNNN      TT      0000    00  EE            XX  XX        TT      EE          NN    NNNN  DD      DD
NN    NNNN      TT      0000    00  EE            XX  XX        TT      EE          NN    NNNN  DD      DD
NN      NN      TT      00      00  EE          XX      XX      TT      EE          NN      NN  DD      DD      ....
NN      NN      TT      00      00  EE          XX      XX      TT      EE          NN      NN  DD      DD      ....
NN      NN      TT        000000    EEEEEEEEEE  XX      XX      TT      EEEEEEEEEE  NN      NN  DDDDDDDD        ....
NN      NN      TT        000000    EEEEEEEEEE  XX      XX      TT      EEEEEEEEEE  NN      NN  DDDDDDDD        ....

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          $BEGIN  NTOEXTEND,000,NF$NETWORK,<NETWORK EXTEND>
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 at the remote
0000    35  ;       node to extend the size of the specified file.
0000    36  ;
0000    37  ; Environment: VAX/VMS, executive mode
0000    38  ;
0000    39  ; Author: James A. Krycka,      Creation Date:  17-AUG-1981
0000    40  ;
0000    41  ; Modified By:
0000    42  ;
0000    43  ;       V03-004 RAS0315         Ron Schaefer    22-Jun-1984
0000    44  ;               Make local definitions of the RMS symbols.
0000    45  ;
0000    46  ;       V03-003 KRM0078         K R Malik       06-Jan-1983
0000    47  ;               Use FAB's ALQ instead of XAB's when no XAB
0000    48  ;               is present. (& fix bug in processing XAB
0000    49  ;               ALQ returned from FAL)
0000    50  ;
0000    51  ;       V03-002 KRM0049         K R Malik       20-Apr-1982
0000    52  ;               Fix bug in way SEND_ALL does XAB probe.
0000    53  ;
0000    54  ;       V03-001 KRM0044         K R Malik       05-Apr-1982
0000    55  ;               Probe each Allocation XAB before sending to FAL.
0000    56  ;
0000    57  ;       V02-002 JAK0066         J A Krycka      07-OCT-1981
```

```
0000     58 ;
0000     59 ;
0000     60 ;--
```

Reject $EXTEND request while in file transfer mode.

```
0000     62          .SBTTL  DECLARATIONS
0000     63
0000     64 ;
0000     65 ; Include Files:
0000     66 ;
0000     67
0000     68          $DAPPLGDEF                      ; Define DAP prologue symbols
0000     69          $DAPHDRDEF                      ; Define DAP message header
0000     70          $DAPCNFDEF                      ; Define DAP Configuration message
0000     71          $DAPACCDEF                      ; Define DAP Access message
0000     72          $DAPCTLDEF                      ; Define DAP Control message
0000     73          $DAPALLDEF                      ; Define DAP Allocation message
0000     74          $FABDEF                         ; Define FAB symbols
0000     75          $IFBDEF                         ; Define IFAB symbols
0000     76          $NWADEF                         ; Define Network Work Area symbols
0000     77          $RMSDEF                         ; Define RMS error message symbols
0000     78          $XABDEF                         ; Define XAB symbols
0000     79
0000     80 ;
0000     81 ; Macros:
0000     82 ;
0000     83 ;            None
0000     84 ;
0000     85 ; Equated Symbols:
0000     86 ;
0000     87
0000     88          ASSUME  DAP$Q_DCODE_FLG EQ 0
0000     89
0000     90 ;
0000     91 ; Own Storage:
0000     92 ;
0000     93 ;            None
0000     94 ;
```

```
                     0000    96          .SBTTL  NT$EXTEND -
                     0000    97
                     0000    98 ;++
                     0000    99 ; NT$EXTEND - engages in a DAP dialogue with the remote FAL to extend the
                     0000   100 ;       specified file according to the Allocation XABs present (or according
                     0000   101 ;       to the FAB if no Allocation XABs are present).
                     0000   102 ;
                     0000   103 ; Calling Sequence:
                     0000   104 ;
                     0000   105 ;       BSBW    NT$EXTEND
                     0000   106 ;
                     0000   107 ; Input Parameters:
                     0000   108 ;
                     0000   109 ;       R8      FAB address
                     0000   110 ;       R9      IFAB address
                     0000   111 ;       R10     IFAB address
                     0000   112 ;       R11     Impure Area address
                     0000   113 ;
                     0000   114 ; Implicit Inputs:
                     0000   115 ;
                     0000   116 ;       User ALLXABs
                     0000   117 ;       User FAB
                     0000   118 ;       DAP$Q_SYSCAP
                     0000   119 ;       NWA$V_FTM_RETRV
                     0000   120 ;       NWA$V_FTM_STORE
                     0000   121 ;
                     0000   122 ; Output Parameters:
                     0000   123 ;
                     0000   124 ;       R0      Status code (RMS)
                     0000   125 ;       R1-R7   Destroyed
                     0000   126 ;       AP      Destroyed
                     0000   127 ;
                     0000   128 ; Implicit Outputs:
                     0000   129 ;
                     0000   130 ;       User ALLXABs
                     0000   131 ;
                     0000   132 ; Completion Codes:
                     0000   133 ;
                     0000   134 ;       Standard RMS completion codes
                     0000   135 ;
                     0000   136 ; Side Effects:
                     0000   137 ;
                     0000   138 ;       None
                     0000   139 ;
                     0000   140 ;--
                     0000   141
                     0000   142 NT$EXTEND::                             ; Entry point
                     0000   143          $TSTPT  NTEXTEND              ;
 57   3C A9   D0    0006   144          MOVL    IFB$L_NWA_PTR(R9),R7     ; Get address of NWA (and DAP)
       04   E1    000A   145          BBC     #DAP$V_EXTEND,-          ; Branch if remote FAL does not support
  0A 28 A7          000C   146                  DAP$Q_SYSCAP(R7),ERRSUP  ;  extend file allocation function
   67   1A   E0    000F   147          BBS     #NWA$V_FTM_RETRV,(R7),-  ; Branch if file transfer mode retrieval
       09          0012   148                  ERRFTM                   ;  in progress
   67   1B   E0    0013   149          BBS     #NWA$V_FTM_STORE,(R7),-  ; Branch if file transfer mode storage
       05          0016   150                  ERRFTM                   ;  in progress
       09   11     0017   151          BRB     BUILD_MASK               ; Proceed
     FFE4'  31     0019   152 ERRSUP:  BRW     NT$SUP_CTLFUNC           ; Assemble error codes in R0 and STV
```

```
                     001C   153                                    ;  and exit
                     001C   154 ERRFTM: RMSERR  FTM                 ; Declare file transfer mode error
                  05 0021   155         RSB                        ; Exit with RMS code in R0
                     0022   156
                     0022   157 ;+
                     0022   158 ; Build display field request mask which will be used in the Control message
                     0022   159 ; to request that optional DAP messages be returned by FAL.
                     0022   160 ; For $EXTEND these are only the ALL messages.
                     0022   161 ;-
                     0022   162
                     0022   163 BUILD_MASK:                         ; Build NWA$W_DISPLAY
                  56 D4   0022   164         CLRL    R6              ; Indicate this is not a close operation
                FFD9' 30   0024   165         BSBW    NT$SCAN_XABCHN  ; Scan user XAB chain and check FAL's
                     0027   166                                    ;  capabilities; request mask put in R2
                03 50 E8   0027   167         BLBS    R0,10$          ; Branch on success
                 00DA 31   002A   168         BRW     FAIL            ; Branch (failure to complete scan)
          52 FFFB 8F AA    002D   169 10$:     BICW2   #<^C<DAP$M_DSP_ALL>>,R2 ; Ignore all but ALLXABs in chain
        00D0 C7   52 B0    0032   170         MOVW    R2,NWA$W_DISPLAY(R7) ; Save request mask
                  06 12    0037   171         BNEQ    SEND_CTL_BEGIN  ; Continue (ALLXAB bit set)
                     0039   172         $SETBIT #DAP$V_DSP_ALL,-       ; If no ALLXAB present, then set the
                     0039   173                 NWA$W_DISPLAY(R7)      ;  bit manually
                     003F   174 ;+
                     003F   175 ; Build and send DAP Control (begin) message to partner.
                     003F   176 ;-
                     003F   177
                     003F   178 SEND_CTL_BEGIN:                     ; (required message sequence)
                50   04 D0   003F   179         MOVL    #DAP$K_CTL_MSG,R0 ; Get message type value
                   FFBB' 30   0042   180         BSBW    NT$BUILD_HEAD   ; Construct message header
                85   0B 90   0045   181         MOVB    #DAP$K_EXTEND_B,(R5)+ ; Store CTLFUNC field
                   FFB5' 30   0048   182         BSBW    NT$BUILD_TAIL   ; Finish building message
                   FFB2' 30   004B   183         BSBW    NT$TRANSMIT     ; Send Control message to FAL
                03 50 E8   004E   184         BLBS    R0,SEND_ALL     ; Branch on success
                 0083 31   0051   185         BRW     FAIL            ; Branch (failure)
                     0054   186
                     0054   187 ;+
                     0054   188 ; Build and send DAP Allocation message(s) to partner.
                     0054   189 ;
                     0054   190 ; Note: NT$SCAN_XABCHN determined that Allocation XABs were chained together.
                     0054   191 ;-
                     0054   192
                     0054   193 SEND_ALL:                           ; (one or more messages required)
          56 0100 C7 D0    0054   194         MOVL    NWA$L_ALLXABADR(R7),R6 ; Get address of first user ALLXAB
                   21 13   0059   195         BEQL    SEND_ALL2       ; Branch if no ALLXABs present
                  FFA2' 30 005B   196 10$:     BSBW    NT$SCAN_ALLXAB  ; Scan the XAB
                03 50 E8   005E   197         BLBS    R0,20$          ; Branch on success
                 00A3 31   0061   198         BRW     FAIL            ; Branch (failure)
                  FF99' 30 0064   199 20$:     BSBW    NT$ENCODE_ALL   ; Build message
                  FF96' 30 0067   200         BSBW    NT$TRANSMIT     ; Send Allocation message to FAL
                03 50 E8   006A   201         BLBS    R0,30$          ; Branch on success
                 0097 31   006D   202         BRW     FAIL            ; Branch (failure)
              011C C7 97   0070   203 30$:     DECB    NWA$B_ALLXABCNT(R7) ; Any more Allocation XABs to process?
                   3E 1B   0074   204         BLEQU   SEND_CTL_END    ; Branch if not
           56   04 A6 D0   0076   205         MOVL    XAB$L_NXT(R6),R6 ; Get address on next XAB in chain
                   DF 11   007A   206         BRB     10$             ; Continue with next XAB
                     007C   207 ;+
                     007C   208 ; No ALLXAB is present. Construct a DAP allocation attributes msg based on
                     007C   209 ; FAB information.
```

```
                     007C   210 ;-
                     007C   211
                     007C   212 SEND_ALL2:
     50    08   D0   007C   213         MOVL    #DAP$K_ALL_MSG,R0        ; Get msg type value
        FF7E'   30   007F   214         BSBW    NT$BUILD_HEAD            ; Construct message header
     51    24   9A   0082   215         MOVZBL  #<DAP$M_AOP!-            ; Get allocation msg value
                     0085   216                   DAP$M_ALQ2!-           ;
                     0085   217                   0>,R1                  ;
        FF78'   30   0085   218         BSBW    NT$CVT_BN4_EXT           ; Store ALLMENU as an extensible
                     0088   219                                          ;  field
  51   04 A8   D0   0088   220         MOVL    FAB$L_FOP(R8),R1         ; Get FAB FOP field
          52   D4   008C   221         CLRL    R2                       ; Clear RMS AOP bits
                     008E   222         $MAPBIT FAB$V_CTG,DAP$V_CTG2     ; Map CTG bit
                     0096   223         $MAPBIT FAB$V_CBT,DAP$V_CBT2     ; Map CBT bit
     51    52   D0   009E   224         MOVL    R2,R1                    ; Move data to correct register
        FF5C'   30   00A1   225         BSBW    NT$CVT_BN4_EXT           ; Store AOP as an extensible field
  51   10 A8   D0   00A4   226         MOVL    FAB$L_ALQ(R8),R1         ; Get FAB's ALQ value
        FF55'   30   00A8   227         BSBW    NT$CVT_BN4_IMG           ; Store ALQ as an image field
        FF52'   30   00AB   228         BSBW    NT$BUILD_TAIL            ; Finish building message
        FF4F'   30   00AE   229         BSBW    NT$TRANSMIT              ; Send Allocation msg to FAL
      53 50   E9   00B1   230         BLBC    R0,FAIL                  ; Branch on failure
                     00B4   231
                     00B4   232 ;+
                     00B4   233 ; Build and send DAP Control (end) message to partner.
                     00B4   234 ;-
                     00B4   235
                     00B4   236 SEND_CTL_END:                          ; (required message sequence)
                     00B4   237         $SETBIT #NWA$V_LAST_MSG,(R7)     ; Declare this last message to block
     50    04   D0   00B8   238         MOVL    #DAP$K_CTL_MSG,R0        ; Get message type value
        FF42'   30   00BB   239         BSBW    NT$BUILD_HEAD            ; Construct message header
     85    0F   90   00BE   240         MOVB    #DAP$K_EXTEND_E,(R5)+    ; Store CTLFUNC field
     85    20   90   00C1   241         MOVB    #DAP$M_DISPLAY2,(R5)+    ; Store CTLMENU field
  51 00D0 C7   3C   00C4   242         MOVZWL  NWA$W_DISPLAY(R7),R1     ; Get request mask
        FF34'   30   00C9   243         BSBW    NT$CVT_BN4_EXT           ; Store DISPLAY as an extensible field
        FF31'   30   00CC   244         BSBW    NT$BUILD_TAIL            ; Finish building message
        FF2E'   30   00CF   245         BSBW    NT$TRANSMIT              ; Send Control message to FAL
      32 50   E9   00D2   246         BLBC    R0,FAIL                  ; Branch on failure
     0100 C7   D5   00D5   247         TSTL    NWA$L_ALLXABADR(R7)      ; Any ALLXABs present?
          0E   13   00D9   248         BEQL    RECV_EXT_ATT2            ; Branch if not (FAB is being used)
        FF22'   30   00DB   249         BSBW    NT$SCAN_XABCHN           ; Scan user XAB chain to reset
                     00DE   250                                          ;  ALLXABCNT
      26 50   E9   00DE   251         BLBC    R0,FAIL                  ; Branch on error
                     00E1   252
                     00E1   253 ;+
                     00E1   254 ; Receive DAP Allocation messages from partner and update the user ALLXABs.
                     00E1   255 ;-
                     00E1   256
                     00E1   257 RECV_EXT_ATT:                          ; (optional--must be requested)
        FF1C'   30   00E1   258         BSBW    NT$RECV_EXT_ATT          ; Process Extended Attributes messages
      20 50   E9   00E4   259         BLBC    R0,FAIL                  ; Branch on failure
          10   11   00E7   260         BRB     RECV_ACK                 ; Go receive ACK msg
                     00E9   261 ;+
                     00E9   262 ; Receive DAP Allocation message from partner and update user FAB.
                     00E9   263 ;-
                     00E9   264
                     00E9   265 RECV_EXT_ATT2:
                     00E9   266         $SETBIT #DAP$K_ALL_MSG,DAP$L_MSG_MASK(R7)
```

```
                00EE   267                                     ; Expect response of Allocation message
 FF0F'   30     00EE   268          BSBW    NT$RECEIVE           ; Get reply from FAL
13 50    E9     00F1   269          BLBC    R0,FAIL              ; Branch on failure
4C A7    D0     00F4   270          MOVL    DAP$L_ALQ2(R7),-     ; Move ALQ value to user FAB
10 A8           00F7   271                  FAB$L_ALQ(R8)        ;
                00F9   272 ;+
                00F9   273 ; Receive DAP Acknowledge message from partner.
                00F9   274 ;-
                00F9   275
                00F9   276 RECV_ACK:                             ; (required message)
                00F9   277          $SETBIT #DAP$K_ACK_MSG,DAP$L_MSG_MASK(R7)
                00FE   278                                      ; Expect response of Acknowledge message
 FEFF'   30     00FE   279          BSBW    NT$RECEIVE           ; Get reply from FAL
03 50    E9     0101   280          BLBC    R0,FAIL              ; Branch on failure
                0104   281 SUC:     RMSSUC                       ; Return success
         05     0107   282 FAIL:    RSB                          ; Exit with RMS code in R0
                0108   283
                0108   284          .END                         ; End of module
```

```
$$.PSECT_EP                     = 00000000
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
BUILD_MASK                        00000022 R      01
DAP$B_ACCFUNC                     00000040
DAP$B_ACCOPT                      00000041
DAP$B_AID                         00000050
DAP$B_ALN                         00000044
DAP$B_AOP                         00000045
DAP$B_BITCNT                      00000035
DAP$B_BKZ                         00000051
DAP$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$B_FAC                         00000042
DAP$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$B_SHR                         00000043
DAP$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$K_ACK_MSG                   = 00000006
DAP$K_ALL_MSG                   = 0000000B
DAP$K_BLN                         000000C0
DAP$K_CTL_MSG                   = 00000004
DAP$K_EXTEND_B                  = 0000000B
DAP$K_EXTEND_E                  = 0000000F
DAP$K_SEQ_ACC                   = 00000000
DAP$L_ALQ2                        0000004C
DAP$L_CHWA                        00000030
DAP$L_CRC_RSLT                    00000020
DAP$L_DCODE_STS                   00000018
DAP$L_LOC                         00000048
DAP$L_MSG_MASK                    0000001C
DAP$L_ROP                         00000050
DAP$L_SSPWA                       00000080
DAP$L_TEMP                        00000090
DAP$M_ALQ2                      = 00000020
DAP$M_AOP                       = 00000004
DAP$M_BITCNT                    = 00000008
DAP$M_BLKCNT                    = 00000040
DAP$M_DISPLAY2                  = 00000020
DAP$M_DSP_3NAM                  = 00000200
DAP$M_DSP_ALL                   = 00000004
DAP$M_GET                       = 00000002
DAP$M_GO_NOGO                   = 00000010
DAP$M_MSE                       = 00000010
DAP$M_SEGMENT                   = 00000040
DAP$M_TMP1$                     = 000000F0
DAP$M_TMP2$                     = 0000FE00
DAP$Q_DCODE_FLG                   00000000
DAP$Q_FILESPEC                    00000044
DAP$Q_KEY                         00000048
DAP$Q_MSG_BUF1                    00000008
DAP$Q_MSG_BUF2                    00000010
DAP$Q_PASSWORD                    00000050
DAP$Q_SYSCAP                      00000028
DAP$Q_SYSPEC                      00000038
DAP$V_CBT2                      = 00000002
DAP$V_CTG2                      = 00000001
DAP$V_DSP_ALL                   = 00000002
DAP$V_EXTEND                    = 00000004
DAP$W_ALLMENU                     00000040
DAP$W_BUFSIZ                      00000040
DAP$W_CTLMENU                     00000044
DAP$W_DEQ2                        00000052
DAP$W_DISPLAY1                    0000004C
DAP$W_DISPLAY2                    00000054
DAP$W_PARTNER                     00000006
DAP$W_VERSION                     00000004
DAP$W_VOL                         00000042
ERRFTM                            0000001C R      01
ERRSUP                            00000019 R      01
FAB$L_ALQ                       = 00000010
FAB$L_FOP                       = 00000004
FAB$V_CBT                       = 00000015
FAB$V_CTG                       = 00000014
FAIL                              00000107 R      01
IFB$L_NWA_PTR                   = 0000003C
NT$BUILD_READ                     ********    X   01
NT$BUILD_TAIL                     ********    X   01
NT$CVT_BN4_EXT                    ********    X   01
NT$CVT_BN4_IMG                    ********    X   01
NT$ENCODE_ALL                     ********    X   01
NT$EXTEND                         00000000 RG     01
NT$RECEIVE                        ********    X   01
NT$RECV_EXT_ATT                   ********    X   01
NT$SCAN_ALLXAB                    ********    X   01
NT$SCAN_XABCHN                    ********    X   01
NT$SUP_CTLFUNC                    ********    X   01
NT$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$L_ALLXABADR                 0000010C
NWA$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_RETRV               = 0000001A
NWA$V_FTM_STORE               = 0000001B
NWA$V_LAST_MSG                = 00C00000
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$A_TRACE                     ********    X   01
RECV_ACK                        000000F9  R     01
RECV_EXT_AT                     000000E1  R     01
RECV_EXT_AT2                    000000E9  R     01
RMS$_FTM                      = 000187C4
SEND_ALL                        00000054  R     01
SEND_ALL2                       0000007C  R     01
SEND_CTL_BEGIN                  0000003F  R     01
SEND_CTL_END                    000000B4  R     01
SUC                             00000104  R     01
TPT$L_NTEXTEND                  ********    X   01
XAB$L_NXT                     = 00000004
```

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	00000108 (264.)	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	00:00:00.07	00:00:00.32
Command processing	125	00:00:00.60	00:00:03.64
Pass 1	310	00:00:11.58	00:00:30.68
Symbol table sort	0	00:00:01.45	00:00:02.47
Pass 2	65	00:00:01.95	00:00:04.23
Symbol table output	23	00:00:00.16	00:00:00.31
Psect synopsis output	2	00:00:00.03	00:00:00.10
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	559	00:00:15.86	00:00:41.77

The working set limit was 1350 pages.
60705 bytes (119 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1119 non-local and 10 local symbols.
284 source lines were read in Pass 1, producing 14 object records in Pass 2.
27 pages of virtual memory were used to define 26 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	18
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	22

1395 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

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

NT0DAPRMS
LIS

NT0GET
LIS

NT0NWASET
LIS

NT0EXTEND
LIS

NT0DECODE
LIS

NT0ENCODE
LIS

NT0ERASE
LIS

NT0DISCON
LIS

NT0DISPLY
LIS

NT0MISC
LIS