```
FFFFFFFFFFFFFFF     AAAAAAAAA     LLL
FFFFFFFFFFFFFFF     AAAAAAAAA     LLL
FFFFFFFFFFFFFFF     AAAAAAAAA     LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFFFFFFFFFFF     AAA         AAA  LLL
FFFFFFFFFFFF     AAA         AAA  LLL
FFFFFFFFFFFF     AAA         AAA  LLL
FFF              AAAAAAAAAAAAAAA  LLL
FFF              AAAAAAAAAAAAAAA  LLL
FFF              AAAAAAAAAAAAAAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLLLLLLLLLLLLLL
FFF              AAA         AAA  LLLLLLLLLLLLLLL
FFF              AAA         AAA  LLLLLLLLLLLLLLL
```

```
_$a

Va
---
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
```

```
FFFFFFFFFF    AAAAAA    LL          BBBBBBBB    LL          DDDDDDDD    XX      XX    AAAAAA    BBBBBBBB
FFFFFFFFFF    AAAAAA    LL          BBBBBBBB    LL          DDDDDDDD    XX      XX    AAAAAA    BBBBBBBB
FF           AA    AA   LL          BB      BB  LL          DD      DD  XX      XX   AA    AA   BB      BB
FF           AA    AA   LL          BB      BB  LL          DD      DD  XX      XX   AA    AA   BB      BB
FF           AA    AA   LL          BB      BB  LL          DD      DD    XX  XX     AA    AA   BB      BB
FF           AA    AA   LL          BB      BB  LL          DD      DD    XX  XX     AA    AA   BB      BB
FFFFFFF      AA    AA   LL          BBBBBBBB    LL          DD      DD      XX       AA    AA   BBBBBBBB
FFFFFFF      AA    AA   LL          BBBBBBBB    LL          DD      DD      XX       AA    AA   BBBBBBBB
FF           AAAAAAAAAA LL          BB      BB  LL          DD      DD    XX  XX     AAAAAAAAAA BB      BB
FF           AAAAAAAAAA LL          BB      BB  LL          DD      DD    XX  XX     AAAAAAAAAA BB      BB
FF           AA    AA   LL          BB      BB  LL          DD      DD  XX      XX   AA    AA   BB      BB      ....
FF           AA    AA   LL          BB      BB  LL          DD      DD  XX      XX   AA    AA   BB      BB      ....
FF           AA    AA   LLLLLLLLLL  BBBBBBBB    LLLLLLLLLL  DDDDDDDD    XX      XX   AA    AA   BBBBBBBB      ....
FF           AA    AA   LLLLLLLLLL  BBBBBBBB    LLLLLLLLLL  DDDDDDDD    XX      XX   AA    AA   BBBBBBBB      ....

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
```

```
0000     1          .TITLE  FALBLDXAB - BUILD DAP EXT ATT MESSAGES
0000     2          .IDENT  'V04-000'
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: FAL (DECnet File Access Listener)
0000    31  ;
0000    32  ; Abstract: This module builds the DAP extended Attributes messages.
0000    33  ;
0000    34  ; Environment: VAX/VMS, user mode
0000    35  ;
0000    36  ; Author: James A. Krycka,      Creation Date:  22-MAY-1979
0000    37  ;
0000    38  ; Modified By:
0000    39  ;
0000    40  ;       V03-002 JAK0136         J A Krycka      07-MAR-1984
0000    41  ;                Cleanup.
0000    42  ;
0000    43  ;--
```

```
0000     45          .SBTTL  DECLARATIONS
0000     46
0000     47 ;
0000     48 ; Include Files:
0000     49 ;
0000     50
0000     51          $DAPPLGDEF                      ; Define DAP prologue symbols
0000     52          $DAPHDRDEF                      ; Define DAP message header
0000     53          $DAPATTDEF                      ; Define DAP Attributes message
0000     54          $DAPKEYDEF                      ; Define DAP Key Definition message
0000     55          $DAPALLDEF                      ; Define DAP Allocation message
0000     56          $DAPSUMDEF                      ; Define DAP Summary message
0000     57          $DAPTIMDEF                      ; Define DAP Date and Time message
0000     58          $DAPPRODEF                      ; Define DAP Protection message
0000     59          $FALWRKDEF                      ; Define FAL Work Area symbols
0000     60          $FABDEF                         ; Define File Access Block symbols
0000     61          $XABDEF                         ; Define symbols common to all XABs
0000     62          $XABALLDEF                      ; Define Allocation XAB symbols
0000     63          $XABDATDEF                      ; Define Date and Time XAB symbols
0000     64          $XABKEYDEF                      ; Define Key Definition XAB symbols
0000     65          $XABPRODEF                      ; Define Protection XAB symbols
0000     66          $XABSUMDEF                      ; Define Summary XAB symobls
0000     67
0000     68 ;
0000     69 ; Macros:
0000     70 ;
0000     71 ;        None
0000     72 ;
0000     73 ; Equated Symbols:
0000     74 ;
0000     75
0000     76          ASSUME  DAP$Q_DCODE_FLG EQ 0
0000     77          ASSUME  FAL$Q_FLG EQ 0
0000     78
0000     79 ;
0000     80 ; Own Storage:
0000     81 ;
```

```
                     0000     83          .SBTTL  FAL$ENCODE_KEY
                00000000     84          .PSECT  FAL$CODE        NOSHR,EXE,RD,NOWRT,BYTE
                     0000     85
                     0000     86 ;++
                     0000     87 ; Functional Description:
                     0000     88 ;
                     0000     89 ;       FAL$ENCODE_KEY builds the specified DAP Key Definition message.
                     0000     90 ;
                     0000     91 ; Calling Sequence:
                     0000     92 ;
                     0000     93 ;       BSBW    FAL$ENCODE_KEY
                     0000     94 ;
                     0000     95 ; Input Parameters:
                     0000     96 ;
                     0000     97 ;       R6      Key of reference value
                     0000     98 ;       R8      Address of FAL work area
                     0000     99 ;       R9      Address of DAP control block
                     0000    100 ;       R10     Address of FAB
                     0000    101 ;       R11     Address of RAB
                     0000    102 ;
                     0000    103 ; Implicit Inputs:
                     0000    104 ;
                     0000    105 ;       FAB$B_ORG
                     0000    106 ;
                     0000    107 ; Output Parameters:
                     0000    108 ;
                     0000    109 ;       R0-R6   Destroyed
                     0000    110 ;       R7      Address of XAB
                     0000    111 ;
                     0000    112 ; Implicit Outputs:
                     0000    113 ;
                     0000    114 ;       None
                     0000    115 ;
                     0000    116 ; Completion Codes:
                     0000    117 ;
                     0000    118 ;       None
                     0000    119 ;
                     0000    120 ; Side Effects:
                     0000    121 ;
                     0000    122 ;       None
                     0000    123 ;
                     0000    124 ;--
                     0000    125
                     0000    126 FAL$ENCODE_KEY::                            ; Entry point
56   0000004C 8F     C4      0000    127         MULL2   #FAL$K_KEYXAB,R6    ; Using REF as an index, compute
  57    1000 C846    9E      0007    128         MOVAB   FAL$L_KEYXAB(R8)[R6],R7 ;  address of Key Definition XAB to use
        50    0A     D0      000D    129         MOVL    #DAP$K_KEY_MSG,R0   ; Get message type value
            FFED'    30      0010    130         BSBW    FAL$BUILD_HEAD      ; Construct message header
      20    1D AA    91      0013    131         CMPB    FAB$B_ORG(R10),#FAB$C_IDX ; Build dummy message (all fields
              03     13      0017    132         BEQL    5$                  ;  defaulted) if file ORG is not IDX
            0097     31      0019    133         BRW     40$                 ; Branch aid
51   0007EFFF 8F     D0      001C    134 5$:     MOVL    #<DAP$M_FLG!-       ; Get key menu value
                             0023    135                   DAP$M_DFL!-       ;
                             0023    136                   DAP$M_IFL!-       ;
                             0023    137                   DAP$M_NSG!-       ;
                             0023    138                   DAP$M_REF!-       ;
                             0023    139                   DAP$M_KNM!-       ;
```

```
                                                M 14

                     0023   140                      DAP$M_NUL!-            ;
                     0023   141                      DAP$M_IAN!-            ;
                     0023   142                      DAP$M_LAN!-            ;
                     0023   143                      DAP$M_DAN!-            ;
                     0023   144                      DAP$M_DTP!-            ;
                     0023   145                      DAP$M_RVB!-            ;
                     0023   146                      DAP$M_DVB!-            ;
                     0023   147                      DAP$M_DBS!-            ;
                     0023   148                      DAP$M_IBS!-            ;
                     0023   149                      DAP$M_LVL!-            ;
                     0023   150                      DAP$M_TKS!-            ;
                     0023   151                      DAP$M_MRL!-            ;
                     0023   152                    0>,R1                    ;
          FFDA'  30  0023   153          BSBW    FAL$CVT_BN4_EXT            ; Store KEYMENU as an extensible field
                     0026   154
                     0026   155 ;
                     0026   156 ; Include the FLG, DFL, and IFL fields in the message.
                     0026   157 ;
                     0026   158
     51  12 A7   9A  0026   159          MOVZBL  XAB$B_FLG(R7),R1           ; Get FLG bits returned by RMS
         52      D4  002A   160          CLRL    R2                         ; Clear corresponding DAP bits
                     002C   161          $MAPBIT XAB$V_DUP,DAP$V_DUP        ; Map DUP bit
                     0034   162          $MAPBIT XAB$V_CHG,DAP$V_CHG        ; Map CHG bit
                     003C   163          $MAPBIT XAB$V_NUL,DAP$V_NUL_CHR    ; Map NUL bit
   83    52   90   0044   164          MOVB    R2,(R3)+                   ; Store key options as extensible field
   83  1C A7   B0   0047   165          MOVW    XAB$W_DFL(R7),(R3)+        ; Store data bucket fill quantity field
   83  1A A7   B0   004B   166          MOVW    XAB$W_IFL(R7),(R3)+        ; Store index bucket fill quantity field
                     004F   167
                     004F   168 ;
                     004F   169 ; Include the NSG, POS, and SIZ fields in the message.
                     004F   170 ;
                     004F   171
     50  14 A7   90  004F   172          MOVB    XAB$B_NSG(R7),R0           ; Get loop count
   83    50   90   0053   173          MOVB    R0,(R3)+                   ; Store number of key segments field
         11   13   0056   174          BEQL    20$                        ; Branch if zero
     51  1E A7   3E  0058   175          MOVAW   XAB$W_POS(R7),R1           ; Get address of POS array
     52  2E A7   9E  005C   176          MOVAB   XAB$B_SIZ(R7),R2           ; Get address of SIZ array
   83    81   B0   0060   177 10$:       MOVW    (R1)+,(R3)+                ; Store key segment position field
   83    82   90   0063   178          MOVB    (R2)+,(R3)+                ; Store key segment size field
     F7  50   F5   0066   179          SOBGTR  R0,10$                     ; Loop if more to go
                     0069   180
                     0069   181 ;
                     0069   182 ; Include the REF, KNM, NUL, IAN, LAN, DAN, and DTP fields in the message.
                     0069   183 ;
                     0069   184
     83  17 A7   90  0069   185 20$:       MOVB    XAB$B_REF(R7),(R3)+        ; Store key of reference field
         83   94     006D   186          CLRB    (R3)+                      ; Assume no key name buffer
         38 A7   D5  006F   187          TSTL    XAB$L_KNM(R7)              ; Branch if no key name buffer
         09   13     0072   188          BEQL    30$                        ;
     FF A3   20   90  0074   189          MOVB    #32,-1(R3)                 ; Store KNM as an image field
63   38 B7   20   28  0078   190          MOVC3   #32,@XAB$L_KNM(R7),(R3)    ; Copy 32-byte key name field into msg
     83  15 A7   90  007D   191 30$:       MOVB    XAB$B_NUL(R7),(R3)+        ; Store null key character field
     83  08 A7   90  0081   192          MOVB    XAB$B_IAN(R7),(R3)+        ; Store index area number field
     83  09 A7   90  0085   193          MOVB    XAB$B_LAN(R7),(R3)+        ; Store lowest level index area
                     0089   194                                             ;  number field
     83  0A A7   90  0089   195          MOVB    XAB$B_DAN(R7),(R3)+        ; Store data area number field
     83  13 A7   90  008D   196          MOVB    XAB$B_DTP(R7),(R3)+        ; Store key data type field
```

```
                          0091   197
                          0091   198 ;
                          0091   199 ; Include the RVB, DVB, DBS, IBS, LVL, TKS, and MRL fields in the message.
                          0091   200 ;
                          0091   201
51   0E A7    D0          0091   202          MOVL    XAB$L_RVB(R7),R1          ; Get root bucket start VBN value
      FF68'   30          0095   203          BSBW    FAL$CVT_BN4_IMG           ; Store RVB as an image field
51   3C A7    D0          0098   204          MOVL    XAB$L_DVB(R7),R1          ; Get first data bucket start VBN value
      FF61'   30          009C   205          BSBW    FAL$CVT_BN4_IMG           ; Store DVB as an image field
83   0D A7    90          009F   206          MOVB    XAB$B_DBS(R7),(R3)+       ; Store data bucket fill size field
83   0C A7    90          00A3   207          MOVB    XAB$B_IBS(R7),(R3)+       ; Store index bucket fill size field
83   0B A7    90          00A7   208          MOVB    XAB$B_LVL(R7),(R3)+       ; Store level of root buckets field
83   16 A7    90          00AB   209          MOVB    XAB$B_TKS(R7),(R3)+       ; Store total key size field
83   18 A7    B0          00AF   210          MOVW    XAB$W_MRL(R7),(R3)+       ; Store minimum record length to contain
                          00B3   211                                            ;  key field
      FF4A'   30          00B3   212 40$:     BSBW    FAL$BUILD_TAIL            ; Finish building message
                          05     00B6   213          RSB                       ; Exit
```

```
                     00B7   215          .SBTTL  FAL$ENCODE_ALL
                0000000B7   216          .PSECT  FAL$CODE         NOSHR,EXE,RD,NOWRT,BYTE
                     00B7   217
                     00B7   218 ;++
                     00B7   219 ; Functional Description:
                     00B7   220 ;
                     00B7   221 ;       FAL$ENCODE_ALL builds the specified DAP Allocation message.
                     00B7   222 ;
                     00B7   223 ; Calling Sequence:
                     00B7   224 ;
                     00B7   225 ;       BSBW    FAL$ENCODE_ALL
                     00B7   226 ;
                     00B7   227 ; Input Parameters:
                     00B7   228 ;
                     00B7   229 ;       R6      Area ID value
                     00B7   230 ;       R8      Address of FAL work area
                     00B7   231 ;       R9      Address of DAP control block
                     00B7   232 ;       R10     Address of FAB
                     00B7   233 ;       R11     Address of RAB
                     00B7   234 ;
                     00B7   235 ; Implicit Inputs:
                     00B7   236 ;
                     00B7   237 ;       DAP$V_VAXVMS
                     00B7   238 ;
                     00B7   239 ; Output Parameters:
                     00B7   240 ;
                     00B7   241 ;       R0-R6   Destroyed
                     00B7   242 ;       R7      Address of XAB
                     00B7   243 ;
                     00B7   244 ; Implicit Outputs:
                     00B7   245 ;
                     00B7   246 ;       None
                     00B7   247 ;
                     00B7   248 ; Completion Codes:
                     00B7   249 ;
                     00B7   250 ;       None
                     00B7   251 ;
                     00B7   252 ; Side Effects:
                     00B7   253 ;
                     00B7   254 ;       None
                     00B7   255 ;
                     00B7   256 ;--
                     00B7   257
                     00B7   258 FAL$ENCODE_ALL::                             ; Entry point
           56    20  C4   00B7   259          MULL2   #FAL$K_ALLXAB,R6        ; Using AID as an index, compute
     57    0C00 C846  9E   00BA   260          MOVAB   FAL$L_ALLXAB(R8)[R6],R7 ;  address of Allocation XAB to use
           50    0B   D0   00C0   261          MOVL    #DAP$K_ALL_MSG,R0       ; Get message type value
              FF3A'   30   00C3   262          BSBW    FAL$BUILD_HEAD          ; Construct message header
       51   01E5 8F   3C   00C6   263          MOVZWL  #<DAP$M_VOL!-           ; Get allocation menu value
                            00CB   264                    DAP$M_AOP!-
                            00CB   265                    DAP$M_ALQ2!-
                            00CB   266                    DAP$M_AID!-
                            00CB   267                    DAP$M_BKZ!-
                            00CB   268                    DAP$M_DEQ2!-
                            00CB   269                    0>,R1
        03 69    34   E1   00CB   270          BBC     #DAP$V_VAXVMS,(R9),10$  ; Branch if partner is not VAX/VMS
           51    0A   A8   00CF   271          BISW2   #<DAP$M_ALN!DAP$M_LOC>,R1 ; Add to menu
```

```
                FF2B'   30   00D2    272 10$:    BSBW    FAL$CVT_BN4_EXT         ; Store ALLMENU as an extensible field
                        00D5    273
                        00D5    274 ;
                        00D5    275 ; Include the VOL, ALN, and AOP fields in the message.
                        00D5    276 ;
                        00D5    277
          83   0A A7    B0   00D5    278         MOVW    XAB$W_VOL(R7),(R3)+     ; Store relative volume number field
          04 69   34    E1   00D9    279         BBC     #DAP$V_VAXVMS,(R9),20$  ; Branch if partner is not VAX/VMS
                        00DD    280
                        00DD    281         ASSUME  DAP$K_ANY EQ 0
                        00DD    282         ASSUME  DAP$K_CYL EQ XAB$C_CYL
                        00DD    283         ASSUME  DAP$K_LBN EQ XAB$C_LBN
                        00DD    284         ASSUME  DAP$K_VBN EQ XAB$C_VBN
                        00DD    285
          83   09 A7    90   00DD    286         MOVB    XAB$B_ALN(R7),(R3)+     ; Store alignment options field
          51   08 A7    9A   00E1    287 20$:    MOVZBL  XAB$B_AOP(R7),R1        ; Get AOP bits returned by RMS
                  52    D4   00E5    288         CLRL    R2                      ; Clear corresponding DAP bits
                        00E7    289         $MAPBIT XAB$V_CBT,DAP$V_CBT2    ; Map CBT bit
                        00EF    290         $MAPBIT XAB$V_CTG,DAP$V_CTG2    ; Map CTG bit
          10 69   34    E1   00F7    291         BBC     #DAP$V_VAXVMS,(R9),30$  ; Branch if partner is not VAX/VMS
                        00FB    292         $MAPBIT XAB$V_HRD,DAP$V_HRD     ; Map HRD bit
                        0103    293         $MAPBIT XAB$V_ONC,DAP$V_ONC     ; Map ONC bit
             51   52    D0   010B    294 30$:    MOVL    R2,R1                   ; Move data to correct register
                FEEF'   30   010E    295         BSBW    FAL$CVT_BN4_EXT         ; Store AOP as an extensible field
                        0111    296
                        0111    297 ;
                        0111    298 ; Include the LOC, ALQ, AID, BKZ, and DEQ fields in the message.
                        0111    299 ;
                        0111    300
          07 69   34    E1   0111    301         BBC     #DAP$V_VAXVMS,(R9),40$  ; Branch if partner is not VAX/VMS
          51   0C A7    D0   0115    302         MOVL    XAB$L_LOC(R7),R1        ; Get starting location value
                FEE4'   30   0119    303         BSBW    FAL$CVT_BN4_IMG         ; Store LOC as an image field
          51   10 A7    D0   011C    304 40$:    MOVL    XAB$L_ALQ(R7),R1        ; Get allocation quantity value
                FEDD'   30   0120    305         BSBW    FAL$CVT_BN4_IMG         ; Store ALQ as an image field
          83   17 A7    90   0123    306         MOVB    XAB$B_AID(R7),(R3)+     ; Store area identification field
          83   16 A7    90   0127    307         MOVB    XAB$B_BKZ(R7),(R3)+     ; Store bucket size field
          83   14 A7    B0   012B    308         MOVW    XAB$W_DEQ(R7),(R3)+     ; Store default extension quantity field
                FECE'   30   012F    309         BSBW    FAL$BUILD_TAIL          ; Finish building message
                        05   0132    310         RSB                             ; Exit
```

```
                     0133   312          .SBTTL  FAL$ENCODE_SUM
                     0133   313          .PSECT  FAL$CODE         NOSHR,EXE,RD,NOWRT,BYTE
                     0133   314
                     0133   315 ;++
                     0133   316 ; Functional Description:
                     0133   317 ;
                     0133   318 ;       FAL$ENCODE_SUM builds the DAP Summary message.
                     0133   319 ;
                     0133   320 ; Calling Sequence:
                     0133   321 ;
                     0133   322 ;       BSBW    FAL$ENCODE_SUM
                     0133   323 ;
                     0133   324 ; Input Parameters:
                     0133   325 ;
                     0133   326 ;       R8      Address of FAL work area
                     0133   327 ;       R9      Address of DAP control block
                     0133   328 ;       R10     Address of FAB
                     0133   329 ;       R11     Address of RAB
                     0133   330 ;
                     0133   331 ; Implicit Inputs:
                     0133   332 ;
                     0133   333 ;       FAB$B_ORG
                     0133   334 ;
                     0133   335 ; Output Parameters:
                     0133   336 ;
                     0133   337 ;       R0-R6   Destroyed
                     0133   338 ;       R7      Address of XAB
                     0133   339 ;
                     0133   340 ; Implicit Outputs:
                     0133   341 ;
                     0133   342 ;       None
                     0133   343 ;
                     0133   344 ; Completion Codes:
                     0133   345 ;
                     0133   346 ;       None
                     0133   347 ;
                     0133   348 ; Side Effects:
                     0133   349 ;
                     0133   350 ;       None
                     0133   351 ;
                     0133   352 ;--
                     0133   353
                     0133   354 FAL$ENCODE_SUM::                             ; Entry point
57   03A4 C8   DE   0133   355          MOVAL   FAL$L_SUMXAB(R8),R7          ; Get address of Summary XAB
     50   0C   D0   0138   356          MOVL    #DAP$K_SUM_MSG,R0            ; Get message type value
        FEC2'  30   013B   357          BSBW    FAL$BUILD_HEAD               ; Construct message header
20   1D AA   91   013E   358          CMPB    FAB$B_ORG(R10),#FAB$C_IDX ; Build dummy message (all fields
        0F   12   0142   359          BNEQ    10$                          ;  defaulted) if file ORG is not IDX
                     0144   360
                     0144   361          ASSUME  DAP$V_NOK LT 7
                     0144   362          ASSUME  DAP$V_NOA LT 7
                     0144   363          ASSUME  DAP$V_PVN LT 7
                     0144   364
     83   0B   90   0144   365          MOVB    #<DAP$M_NOK!-                ; Get summary menu value
                     0147   366                    DAP$M_NOA!-                ;
                     0147   367                    DAP$M_PVN!-                ;
                     0147   368                    0>,(R3)+                   ; Store sumenu as an extensible field
```

```
83   09 A7    90   0147   369          MOVB    XAB$B_NOK(R7),(R3)+     ; Store number of keys field
83   08 A7    90   014B   370          MOVB    XAB$B_NOA(R7),(R3)+     ; Store number of allocation areas field
83   0A A7    B0   014F   371          MOVW    XAB$W_PVN(R7),(R3)+     ; Store prologue version number field
      FEAA'    30   0153   372 10$:     BSBW    FAL$BUILD_TAIL          ; Finish building message
               05   0156   373          RSB                             ; Exit
```

```
                     0157   375          .SBTTL  FAL$ENCODE_TIM
                 00000157   376          .PSECT  FAL$CODE         NOSHR,EXE,RD,NOWRT,BYTE
                     0157   377
                     0157   378 ;++
                     0157   379 ; Functional Description:
                     0157   380 ;
                     0157   381 ;       FAL$ENCODE_TIM builds the DAP Date and Time message.
                     0157   382 ;
                     0157   383 ; Calling Sequence:
                     0157   384 ;
                     0157   385 ;       BSBW    FAL$ENCODE_TIM
                     0157   386 ;
                     0157   387 ; Input Parameters:
                     0157   388 ;
                     0157   389 ;       R8      Address of FAL work area
                     0157   390 ;       R9      Address of DAP control block
                     0157   391 ;       R10     Address of FAB
                     0157   392 ;       R11     Address of RAB
                     0157   393 ;
                     0157   394 ; Implicit Inputs:
                     0157   395 ;
                     0157   396 ;       DAP$V_GEQ_V60
                     0157   397 ;
                     0157   398 ; Output Parameters:
                     0157   399 ;
                     0157   400 ;       R0-R6   Destroyed
                     0157   401 ;       R7      Address of XAB
                     0157   402 ;
                     0157   403 ; Implicit Outputs:
                     0157   404 ;
                     0157   405 ;       None
                     0157   406 ;
                     0157   407 ; Completion Codes:
                     0157   408 ;
                     0157   409 ;       None
                     0157   410 ;
                     0157   411 ; Side Effects:
                     0157   412 ;
                     0157   413 ;       None
                     0157   414 ;
                     0157   415 ;--
                     0157   416
                     0157   417 FAL$ENCODE_TIM::                             ; Entry point
57   0320 C8   DE    0157   418          MOVAL   FAL$L_DATXAB(R8),R7         ; Get address of Date and Time XAB
     50    0D   D0    015C   419          MOVL    #DAP$K_TIM_MSG,R0           ; Get message type value
        FE9E'   30    015F   420          BSBW    FAL$BUILD_HEAD              ; Construct message header
                     0162   421
                     0162   422 ;
                     0162   423 ; Construct date and time menu value.
                     0162   424 ; Send only time fields that have a non-zero 64-bit time value as zero means
                     0162   425 ; the current date and time, not 17-NOV-1858! (actually only the upper 32-bits
                     0162   426 ; will be tested for zero, i.e., any time on 17-NOV-1858 will be considered
                     0162   427 ; as the default time.)
                     0162   428 ;
                     0162   429
                     0162   430          ASSUME  DAP$V_CDT EQ 0
                     0162   431          ASSUME  DAP$V_CDT+1 EQ DAP$V_RDT
```

```
                     0162   432          ASSUME  DAP$V_RDT+1 EQ DAP$V_EDT
                     0162   433          ASSUME  DAP$V_EDT+1 EQ DAP$V_RVN
                     0162   434          ASSUME  DAP$V_RVN+1 EQ DAP$V_BDT
                     0162   435
          54   D4    0162   436          CLRL    R4                          ; Initialize time menu field
    18 A7   D5    0164   437          TSTL    XAB$Q_CDT+4(R7)             ; Branch if creation date and time
       03   13    0167   438          BEQL    10$                         ;  is zero
    54 01   88    0169   439          BISB2   #DAP$M_CDT,R4               ; Otherwise, send field
    10 A7   D5    016C   440 10$:      TSTL    XAB$Q_RDT+4(R7)             ; Branch if revision date and time
       03   13    016F   441          BEQL    20$                         ;  is zero
    54 02   88    0171   442          BISB2   #DAP$M_RDT,R4               ; Otherwise, send field
    20 A7   D5    0174   443 20$:      TSTL    XAB$Q_EDT+4(R7)             ; Branch if expiration date and time
       03   13    0177   444          BEQL    30$                         ;  is zero
    54 04   88    0179   445          BISB2   #DAP$M_EDT,R4               ; Otherwise, send field
08 69 25   E1    017C   446 30$:      BBC     #DAP$V_GEQ_V60,(R9),40$     ; Branch if partner uses DAP before V6.0
    28 A7   D5    0180   447          TSTL    XAB$Q_BDT+4(R7)             ; Branch if backup date and time
       03   13    0183   448          BEQL    40$                         ;  is zero
    54 10   88    0185   449          BISB2   #DAP$M_BDT,R4               ; Otherwise, send field
    54 08   88    0188   450 40$:      BISB2   #DAP$M_RVN,R4               ; Send revision number field
    83 54   90    018B   451          MOVB    R4,(R3)+                    ; Store TIMENU as an extensible field
                     018E   452
                     018E   453 ;
                     018E   454 ; Now process each time field.
                     018E   455 ;
                     018E   456
06 54   00   E1    018E   457          BBC     #DAP$V_CDT,R4,50$           ; Branch if CDT is not to be included
50   14 A7   7E    0192   458          MOVAQ   XAB$Q_CDT(R7),R0            ; Get address of 64-bit value for
                     0196   459                                              ;  creation date and time
       26   10    0196   460          BSBB    CONVERT_TIME                ; Store CDT as an image field
06 54   01   E1    0198   461 50$:      BBC     #DAP$V_RDT,R4,60$           ; Branch if RDT is not to be included
50   0C A7   7E    019C   462          MOVAQ   XAB$Q_RDT(R7),R0            ; Get address of 64-bit value for
                     01A0   463                                              ;  revision date and time
       1C   10    01A0   464          BSBB    CONVERT_TIME                ; Store RDT as an image field
06 54   02   E1    01A2   465 60$:      BBC     #DAP$V_EDT,R4,70$           ; Branch if EDT is not to be included
50   1C A7   7E    01A6   466          MOVAQ   XAB$Q_EDT(R7),R0            ; Get address of 64-bit value for
                     01AA   467                                              ;  expiration date and time
       12   10    01AA   468          BSBB    CONVERT_TIME                ; Store EDT as an image field
83   08 A7   B0    01AC   469 70$:      MOVW    XAB$W_RVN(R7),(R3)+         ; Store revision number field
06 54   04   E1    01B0   470          BBC     #DAP$V_BDT,R4,80$           ; Branch if BDT is not to be included
50   24 A7   7E    01B4   471          MOVAQ   XAB$Q_BDT(R7),R0            ; Get address of 64-bit value for
                     01B8   472                                              ;  backup date and time
       04   10    01B8   473          BSBB    CONVERT_TIME                ; Store BDT as an image field
       FE43'  30    01BA   474 80$:      BSBW    FAL$BUILD_TAIL              ; Finish building message
              05    01BD   475          RSB                                 ; Exit
                     01BE   476
                     01BE   477 ;
                     01BE   478 ; This routine converts a time value in 64-bit binary format to an ASCII string.
                     01BE   479 ; Then it stores the string as an 18-byte fixed length field in the DAP message
                     01BE   480 ; with the first two digits of the year removed (per DAP specification).
                     01BE   481 ;
                     01BE   482
                     01BE   483 CONVERT_TIME:                               ; Entry point
    5E   20   C2    01BE   484          SUBL2   #<20+12>,SP                 ; Allocate space from the stack
    52   5E   D0    01C1   485          MOVL    SP,R2                       ; Save address of work area
14 A2   14   D0    01C4   486          MOVL    #20,20(R2)                  ; Form descriptor of buffer to receive
18 A2   5E   D0    01C8   487          MOVL    SP,24(R2)                   ;  ASCII time string
                     01CC   488          $ASCTIM_S-                          ; Convert binary time to ASCII time
```

```
                                01CC    489                     TIMLEN=28(R2)-          ;  Address of word to return string size
                                01CC    490                     TIMBUF=20(R2)-          ;  Address of descriptor for buffer
                                01CC    491                     TIMADR=(R0)-            ;  Address of 64-bit time value
                                01CC    492                     CVTFLG=#0               ;  Flag set to request date and time
                                01DD    493             $CHECK_SS                       ; Check status code and exit on failure
                    62  20  91  01E0    494             CMPB    #^A\ \,(R2)             ; Convert leading space to zero in
                        03  12  01E3    495             BNEQ    10$                     ;  day-of-month field to conform to
                    62  30  90  01E5    496             MOVB    #^A\0\,(R2)             ;  the DAP V6.0 specification
                                01E8    497                                             ; Store time field omitting the two
                                01E8    498                                             ;  century digits
                        10  BB  01E8    499 10$:        PUSHR   #^M<R4>                 ; Save time menu mask
                63  62  07  28  01EA    500             MOVC3   #7,(R2),(R3)            ; Copy bytes 1-7 of input string
            63  02  A1  0B  28  01EE    501             MOVC3   #11,2(R1),(R3)          ; Copy bytes 9-20 of input string
                        10  BA  01F3    502             POPR    #^M<R4>                 ; Restore time menu mask
                    5E  20  C0  01F5    503             ADDL2   #<20+12>,SP             ; Deallocate space from the stack
                            05  01F8    504             RSB                             ; Exit
```

```
                     01F9   506          .SBTTL  FAL$ENCODE_PRO
                000001F9    507          .PSECT  FAL$CODE        NOSHR,EXE,RD,NOWRT,BYTE
                     01F9   508
                     01F9   509 ;++
                     01F9   510 ; Functional Description:
                     01F9   511 ;
                     01F9   512 ;       FAL$ENCODE_PRO builds the DAP Protection message.
                     01F9   513 ;
                     01F9   514 ; Calling Sequence:
                     01F9   515 ;
                     01F9   516 ;       BSBW    FAL$ENCODE_PRO
                     01F9   517 ;
                     01F9   518 ; Input Parameters:
                     01F9   519 ;
                     01F9   520 ;       R8      Address of FAL work area
                     01F9   521 ;       R9      Address of DAP control block
                     01F9   522 ;       R10     Address of FAB
                     01F9   523 ;       R11     Address of RAB
                     01F9   524 ;
                     01F9   525 ; Implicit Inputs:
                     01F9   526 ;
                     01F9   527 ;       None
                     01F9   528 ;
                     01F9   529 ; Output Parameters:
                     01F9   530 ;
                     01F9   531 ;       R0-R6   Destroyed
                     01F9   532 ;       R7      Address of XAB
                     01F9   533 ;
                     01F9   534 ; Implicit Outputs:
                     01F9   535 ;
                     01F9   536 ;       None
                     01F9   537 ;
                     01F9   538 ; Completion Codes:
                     01F9   539 ;
                     01F9   540 ;       None
                     01F9   541 ;
                     01F9   542 ; Side Effects:
                     01F9   543 ;
                     01F9   544 ;       None
                     01F9   545 ;
                     01F9   546 ;--
                     01F9   547
                     01F9   548 FAL$ENCODE_PRO::                             ; Entry point
 57  034C CB   DE   01F9   549          MOVAL   FAL$L_PROXAB(R8),R7         ; Get address of Protection XAB
 50    0E   D0   01FE   550          MOVL    #DAP$K_PRO_MSG,R0               ; Get message type value
    FDFC'  30   0201   551          BSBW    FAL$BUILD_HEAD                  ; Construct message header
                0204   552
                0204   553          ASSUME  DAP$V_OWNER LT 7
                0204   554          ASSUME  DAP$V_PROSYS LT 7
                0204   555          ASSUME  DAP$V_PROOWN LT 7
                0204   556          ASSUME  DAP$V_PROGRP LT 7
                0204   557          ASSUME  DAP$V_PROWLD LT 7
                0204   558
    83    1F   90   0204   559          MOVB    #<DAP$M_OWNER!-                 ; Get protection menu value
                0207   560                    DAP$M_PROSYS!-                  ;
                0207   561                    DAP$M_PROOWN!-                  ;
                0207   562                    DAP$M_PROGRP!-                  ;
```

```
                     0207   563                  DAP$M_PROWLD!-          ;
                     0207   564                 0>,(R3)+                 ; Store PROMENU as an extensible field
                     0207   565
                     0207   566 ;
                     0207   567 ; Include the OWNER field in the message.
                     0207   568 ;
                     0207   569
      5E   1C   C2   0207   570          SUBL2   #<16+12>,SP              ; Allocate space from the stack
      52   5E   D0   020A   571          MOVL    SP,R2                    ; Save address of work area
   10 A2   10   D0   020D   572          MOVL    #16,16(R2)               ; Form descriptor of buffer to receive
   14 A2   5E   D0   0211   573          MOVL    SP,20(R2)                ;  ASCII string
   50   0E A7   3C   0215   574          MOVZWL  XAB$W_GRP(R7),R0         ; Get group UIC value
   51   0C A7   3C   0219   575          MOVZWL  XAB$W_MBM(R7),R1         ; Get member UIC value
                     021D   576          $FAO_S-                          ; Format the UIC string
                     021D   577                  CTRSTR=W^FAL$GQ_UIC-     ;  Address of FAO control string
                     021D   578                  OUTLEN=24(R2)-           ;  Address of receive string length
                     021D   579                  OUTBUF=16(R2)-           ;  Address of buffer descriptor
                     021D   580                  P1=R0-                   ;  Group number of file owner
                     021D   581                  P2=R1                    ;  Member number of file owner
                     0232   582          $CHECK_SS                        ; Check status code and exit on failure
   50   18 A2   3C   0235   583          MOVZWL  24(R2),R0                ; Get length of returned string
      83   50   90   0239   584          MOVB    R0,(R3)+                 ; Store owner as an image field
63    62   50   28   023C   585          MOVC3   R0,(R2),(R3)             ; Copy owner string to message
      5E   1C   C0   0240   586          ADDL2   #<16+12>,SP              ; Deallocate space from the stack
                     0243   587
                     0243   588 ;
                     0243   589 ; Construct the four protection fields: PROSYS, PROOWN, PROGRP, and PROWLD.
                     0243   590 ;
                     0243   591
                     0243   592          ASSUME  DAP$V_RED_ACC EQ XAB$V_NOREAD
                     0243   593          ASSUME  DAP$V_WRT_ACC EQ XAB$V_NOWRITE
                     0243   594          ASSUME  DAP$V_EXE_ACC EQ XAB$V_NOEXE
                     0243   595          ASSUME  DAP$V_DLT_ACC EQ XAB$V_NODEL
                     0243   596
                     0243   597          ASSUME  DAP$V_RED_ACC LT 7
                     0243   598          ASSUME  DAP$V_WRT_ACC LT 7
                     0243   599          ASSUME  DAP$V_EXE_ACC LT 7
                     0243   600          ASSUME  DAP$V_DLT_ACC LT 7
                     0243   601
      50   08 A7   3C   0243   602          MOVZWL  XAB$W_PRO(R7),R0         ; Get protection value
51    50   04   00   EF   0247   603          EXTZV   #XAB$V_SYS,#4,R0,R1      ; Store system protection field
      83   51   90   024C   604          MOVB    R1,(R3)+                 ;  as an extensible field
51    50   04   04   EF   024F   605          EXTZV   #XAB$V_OWN,#4,R0,R1      ; Store owner protection field
      83   51   90   0254   606          MOVB    R1,(R3)+                 ;  as an extensible field
51    50   04   08   EF   0257   607          EXTZV   #XAB$V_GRP,#4,R0,R1      ; Store group protection field
      83   51   90   025C   608          MOVB    R1,(R3)+                 ;  as an extensible field
51    50   04   0C   EF   025F   609          EXTZV   #XAB$V_WLD,#4,R0,R1      ; Store world protection field
      83   51   90   0264   610          MOVB    R1,(R3)+                 ;  as an extensible field
         FD96'   30   0267   611          BSBW    FAL$BUILD_TAIL           ; Finish building message
                 05   026A   612          RSB                              ; Exit
                     026B   613
                     026B   614          .END                             ; End of module
```

```
$$T2                    = 00000005
CONVERT_TIME              000001BE R      02
DAP$B_AID                 00000050
DAP$B_ALN                 00000044
DAP$B_AOP                 00000045
DAP$B_BITCNT              00000035
DAP$B_BKS                 00000050
DAP$B_BKZ                 00000051
DAP$B_BSZ                 00000052
DAP$B_DAN                 00000070
DAP$B_DATATYPE            00000044
DAP$B_DBS                 0000007C
DAP$B_DCODE_FID           00000019
DAP$B_DCODE_MAC           0000001B
DAP$B_DCODE_MSG           0000001A
DAP$B_DTP                 00000071
DAP$B_FLAGS               00000031
DAP$B_FLG                 00000048
DAP$B_FSZ                 00000051
DAP$B_IAN                 0000006E
DAP$B_IBS                 0000007D
DAP$B_LAN                 0000006F
DAP$B_LEN256              00000034
DAP$B_LENGTH              00000033
DAP$B_LVL                 0000007E
DAP$B_NOA                 00000045
DAP$B_NOK                 00000044
DAP$B_NOR                 00000046
DAP$B_NSG                 00000049
DAP$B_NUL                 0000006D
DAP$B_ORG                 00000045
DAP$B_RAT                 00000047
DAP$B_REF                 0000006C
DAP$B_RFM                 00000046
DAP$B_SIZ                 0000005C
DAP$B_SIZ_TMP             0000004A
DAP$B_STREAMID            00000032
DAP$B_TKS                 0000007F
DAP$B_TYPE                00000030
DAP$B_X_FIELD             00000024
DAP$C_BLN                 000000C0
DAP$K_ALL_MSG           = 0000000B
DAP$K_ANY               = 00000000
DAP$K_BLN                 000000C0
DAP$K_CTL               = 00000001
DAP$K_FIX               = 00000001
DAP$K_KEY_MSG           = 0000000A
DAP$K_LBN               = 00000002
DAP$K_PRO_MSG           = 0000000E
DAP$K_SEQ               = 00000000
DAP$K_STG               = 00000000
DAP$K_SUM_MSG           = 0000000C
DAP$K_TIM_MSG           = 0000000D
DAP$K_VBN               = 00000003
DAP$L_ALQ1                0000C04C
DAP$L_ALQ2                0000004C
DAP$L_ATTMENU             00000040
DAP$L_CMWA                00000030
DAP$L_CRC_RSLT            00000020
DAP$L_DCODE_STS           00000018
DAP$L_DEV                 00000068
DAP$L_DVB                 00000078
DAP$L_EBK                 00000078
DAP$L_FOP1                00000064
DAP$L_HBK                 00000074
DAP$L_KEYMENU             00000040
DAP$L_LOC                 00000048
DAP$L_MRN                 00000058
DAP$L_MSG_MASK            0000001C
DAP$L_RVB                 00000074
DAP$L_SBN                 0000007C
DAP$L_SSPWA               00000080
DAP$L_TEMP                00000090
DAP$M_AID               = 00000040
DAP$M_ALN               = 00000002
DAP$M_ALQ2              = 00000020
DAP$M_AOP               = 00000004
DAP$M_BDT               = 00000010
DAP$M_BITCNT            = 00000008
DAP$M_BKZ               = 00000080
DAP$M_CDT               = 00000001
DAP$M_CMPFMT            = 00000008
DAP$M_DAN               = 00000200
DAP$M_DBS               = 00004000
DAP$M_DEQ2              = 00000100
DAP$M_DFL               = 00000002
DAP$M_DMO               = 00002000
DAP$M_DTP               = 00000400
DAP$M_DVB               = 00002000
DAP$M_EDT               = 00000004
DAP$M_EMBEDDED          = 00000010
DAP$M_FLG               = 00000001
DAP$M_IAN               = 00000080
DAP$M_IBS               = 00008000
DAP$M_IFL               = 00000004
DAP$M_IMAGE             = 00000002
DAP$M_KNM               = 00000020
DAP$M_LAN               = 00000100
DAP$M_LOC               = 00000008
DAP$M_LSA               = 00000040
DAP$M_LVL               = 00010000
DAP$M_MACY11            = 00000080
DAP$M_MRL               = 00040000
DAP$M_NOA               = 00000002
DAP$M_NOK               = 00000001
DAP$M_NSG               = 00000008
DAP$M_NUL               = 00000040
DAP$M_OWNER             = 00000001
DAP$M_PROGRP            = 00000008
DAP$M_PROOWN            = 00000004
DAP$M_PROSYS            = 00000002
DAP$M_PROWLD            = 00000010
DAP$M_PVN               = 00000008
DAP$M_RDT               = 00000002
```

```
DAP$M_REF                       = 00000010
DAP$M_RVB                       = 00000800
DAP$M_RVN                       = 00000008
DAP$M_SEGMENT                   = 00000040
DAP$M_TKS                       = 00020000
DAP$M_TMP1$                     = 0000FE00
DAP$M_TMP2$                     = 0000FE00
DAP$M_TMP3$                     = 00020000
DAP$M_TMP4$                     = 01000000
DAP$M_TMP5$                     = F0000000
DAP$M_VOL                       = 00000001
DAP$M_ZERO                      = 00000080
DAP$Q_ADT                         00000070
DAP$Q_BDT                         00000060
DAP$Q_CDT                         00000048
DAP$Q_DCODE_FLG                   00000000
DAP$Q_EDT                         00000058
DAP$Q_KNM                         00000064
DAP$Q_MSG_BUF1                    00000008
DAP$Q_MSG_BUF2                    00000010
DAP$Q_OWNER                       00000048
DAP$Q_PDT                         00000068
DAP$Q_RDT                         00000050
DAP$Q_RUNSYS                      0000005C
DAP$Q_SYSPEC                      00000038
DAP$V_BDT                       = 00000004
DAP$V_CBT2                      = 00000002
DAP$V_CDT                       = 00000000
DAP$V_CHG                       = 00000001
DAP$V_CTG2                      = 00000001
DAP$V_DLT_ACC                   = 00000003
DAP$V_DUP                       = 00000000
DAP$V_EDT                       = 00000002
DAP$V_EXE_ACC                   = 00000002
DAP$V_GEQ_V60                   = 00000025
DAP$V_HRD                       = 00000000
DAP$V_NOA                       = 00000001
DAP$V_NOK                       = 00000000
DAP$V_NUL_CHR                   = 00000002
DAP$V_ONC                       = 00000003
DAP$V_OWNER                     = 00000000
DAP$V_PROGRP                    = 00000003
DAP$V_PROOWN                    = 00000002
DAP$V_PROSYS                    = 00000001
DAP$V_PROWLD                    = 00000004
DAP$V_PVN                       = 00000003
DAP$V_RDT                       = 00000001
DAP$V_RED_ACC                   = 00000000
DAP$V_RVN                       = 00000003
DAP$V_VAXVMS                    = 00000034
DAP$V_WRT_ACC                   = 00000001
DAP$W_ALLMENU                     00000040
DAP$W_BLS                         00000048
DAP$W_DEQ1                        00000054
DAP$W_DEQ2                        00000052
DAP$W_DFL                         00000044
DAP$W_FFB                         00000072
DAP$W_IFL                         00000046
DAP$W_LRL                         00000070
DAP$W_MRL                         00000072
DAP$W_MRS                         0000004A
DAP$W_PARTNER                     00000006
DAP$W_POS                         0000004C
DAP$W_POS_TMP                     0000004A
DAP$W_PROGRP                      00000054
DAP$W_PROMENU                     00000040
DAP$W_PROOWN                      00000052
DAP$W_PROSYS                      00000050
DAP$W_PROWLD                      00000056
DAP$W_PVN                         00000042
DAP$W_RVN                         00000042
DAP$W_SUMENU                      00000040
DAP$W_TIMENU                      00000040
DAP$W_VERSION                     00000004
DAP$W_VOL                         00000042
FAB$B_ORG                       = 0000001D
FAB$C_IDX                       = 00000020
FAL$BUILD_HEAD                    ********    X   02
FAL$BUILD_TAIL                    ********    X   02
FAL$B_ACCFUNC                     000001F6
FAL$B_ACCOPT                      000001F5
FAL$B_DATATYPE                    000001F4
FAL$B_DISABLE                     00000006
FAL$B_ENABLE                      00000005
FAL$B_LOGGING                     00000004
FAL$B_MISCOPT                     00000007
FAL$B_RAC                         000001F7
FAL$B_RBK_CACHE                   00000012
FAL$B_RCVBUFIDX                   00000011
FAL$B_VALUE                       00000010
FAL$CHECK_SS                      ********      X   02
FAL$CVT_BN4_EXT                   ********      X   02
FAL$CVT_BN4_IMG                   ********      X   02
FAL$C_WRKBLN                      00002000
FAL$ENCODE_ALL                    000000B7 RG       02
FAL$ENCODE_KEY                    00000000 RG       02
FAL$ENCODE_PRO                    000001F9 RG       02
FAL$ENCODE_SUM                    00000133 RG       02
FAL$ENCODE_TIM                    00000157 RG       02
FAL$GQ_UIC                        ********     X    02
FAL$K_ALLXAB                    = 00000020
FAL$K_KEYXAB                    = 0000004C
FAL$K_WRKBLN                      00002000
FAL$L_ALLXAB                      00000C00
FAL$L_ALLXABINI                   00000074
FAL$L_CHAIN_NXT                   0000007C
FAL$L_DATXAB                      00000320
FAL$L_FAB                         00000200
FAL$L_FAB2                        00000800
FAL$L_FHCXAB                      000002F4
FAL$L_FOP                         000001F8
FAL$L_KEYNAM                      00001C00
FAL$L_KEYXAB                      00001000
FAL$L_KEYXABINI                   00000078
```

```
FAL$L_NAM                       00000294
FAL$L_NAM2                      00000850
FAL$L_NUMBER                    000001FC
FAL$L_PROXAB                    0000034C
FAL$L_RAB                       00000250
FAL$L_RCVBUF                    0000005C
FAL$L_RDTXAB                    000003B0
FAL$L_RMS_PTR                   0000006C
FAL$L_STB                       000000C0
FAL$L_SUMXAB                    000003A4
FAL$L_TEMP                      000003F4
FAL$L_USE_SC1                   000000A8
FAL$L_USE_SC2                   000000AC
FAL$L_USE_VER                   000000A4
FAL$Q_BLD                       00000050
FAL$Q_DIRNAME                   00000088
FAL$Q_FALLOG                    00000090
FAL$Q_FLG                       00000000
FAL$Q_MBX                       00000038
FAL$Q_MBXIOSB                   00000030
FAL$Q_RCV                       00000040
FAL$Q_RCVIOSB                   00000020
FAL$Q_RMS                       00000064
FAL$Q_STATE_CTX                 00000008
FAL$Q_SYSNET                    00000098
FAL$Q_TEMP                      000003F8
FAL$Q_VOLNAME                   00000080
FAL$Q_XMT                       00000048
FAL$Q_XMTIOSB                   00000028
FAL$T_DAP                       00000100
FAL$T_DIRNAME                   00001F00
FAL$T_EXPAND                    00000500
FAL$T_EXPAND2                   00000A00
FAL$T_FALLOG                    00001C00
FAL$T_FILESPEC                  00000400
FAL$T_FILESPEC2                 00000900
FAL$T_KEYBUF                    00000700
FAL$T_MBXBUF                    00001980
FAL$T_PRTBUF1                   00001A00
FAL$T_PRTBUF2                   00001B00
FAL$T_RESULT                    00000600
FAL$T_RESULT2                   00000B00
FAL$T_SYSNET                    00001D00
FAL$T_VOLNAME                   00001E00
FAL$W_DAPBUFSIZ                 0000001A
FAL$W_DISPLAY                   00000070
FAL$W_LNKCHN                    0000001C
FAL$W_MBXCHN                    0000001E
FAL$W_QIOBUFSIZ                 00000018
FAL$W_RECEIVED                  00000072
FAL$W_USE_DBS                   000000A0
FAL$W_USE_SYS                   000000A2
SYS$ASCTIM                      ********  GX    02
SYS$FAO                         ********   X    02
XAB$B_AID                   =   00000017
XAB$B_ALN                   =   00000009
XAB$B_AOP                   =   00000008
XAB$B_BKZ                       =  00000016
XAB$B_DAN                       =  0000000A
XAB$B_DBS                       =  0000000D
XAB$B_DTP                       =  00000013
XAB$B_FLG                       =  00000012
XAB$B_IAN                       =  00000008
XAB$B_IBS                       =  0000000C
XAB$B_LAN                       =  00000009
XAB$B_LVL                       =  0000000B
XAB$B_NOA                       =  00000008
XAB$B_NOK                       =  00000009
XAB$B_NSG                       =  00000014
XAB$B_NUL                       =  00000015
XAB$B_REF                       =  00000017
XAB$B_SIZ                       =  0000002E
XAB$B_TKS                       =  00000016
XAB$C_CYL                       =  000000C1
XAB$C_LBN                       =  00000002
XAB$C_VBN                       =  00000003
XAB$L_ALQ                       =  00000010
XAB$L_DVB                       =  0000003C
XAB$L_KNM                       =  00000038
XAB$L_LOC                       =  0000000C
XAB$L_RVB                       =  0000000E
XAB$Q_BDT                       =  00000024
XAB$Q_CDT                       =  00000014
XAB$Q_EDT                       =  0000001C
XAB$Q_RDT                       =  0000000C
XAB$V_CBT                       =  0000C005
XAB$V_CHG                       =  00000001
XAB$V_CTG                       =  00000007
XAB$V_DUP                       =  00000000
XAB$V_GRP                       =  00000008
XAB$V_HRD                       =  00000000
XAB$V_NODEL                     =  00000003
XAB$V_NOEXE                     =  00000002
XAB$V_NOREAD                    =  00000000
XAB$V_NOWRITE                   =  00000001
XAB$V_NUL                       =  00000002
XAB$V_ONC                       =  00000001
XAB$V_OWN                       =  00000004
XAB$V_SYS                       =  00000000
XAB$V_WLD                       =  0000000C
XAB$W_DEQ                       =  00000014
XAB$W_DFL                       =  0000001C
XAB$W_GRP                       =  0000000E
XAB$W_IFL                       =  0000001A
XAB$W_MBM                       =  0000000C
XAB$W_MRL                       =  00000018
XAB$W_POS                       =  0000001E
XAB$W_PRO                       =  00000008
XAB$W_PVN                       =  0000000A
XAB$W_RVN                       =  00000008
XAB$W_VOL                       =  0000000A
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00002000 (8192.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
FAL$CODE	0000026B (619.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.04	00:00:01.07
Command processing	139	00:00:00.41	00:00:03.29
Pass 1	342	00:00:09.19	00:00:31.10
Symbol table sort	0	00:00:01.02	00:00:05.62
Pass 2	117	00:00:01.80	00:00:06.69
Symbol table output	47	00:00:00.19	00:00:01.55
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	684	00:00:12.67	00:00:49.34

The working set limit was 1650 pages.
72669 bytes (142 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1145 non-local and 26 local symbols.
614 source lines were read in Pass 1, producing 15 object records in Pass 2.
30 pages of virtual memory were used to define 28 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[FAL.OBJ]FAL.MLB;1	11
_255DUA28:[SYSLIB]STARLET.MLB;2	14
TOTALS (all libraries)	25

1500 GETS were required to define 25 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:FALBLDXAB/OBJ=OBJ$:FALBLDXAB MSRC$:FALBLDXAB/UPDATE=(ENH$:FALBLDXAB)+LIB$:FAL/LIB
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

FALACTION
LIS

FALDAPCRC
LIS

FALMACROS
MAR

FALBLDXAB
LIS

FALBLDATT
LIS

FALBLDSTS
LIS

FALDEF
MDL

FALDAPIO
LIS

FALACTINI
LIS

FALACTMSG
LIS