```
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
7FF
```

L 2

```
**FILE**ID**FALACTINI
```

```
FFFFFFFFFF    AAAAAA    LL            AAAAAA      CCCCCCCC  TTTTTTTTTT    IIIIII    NN      NN    IIIIII
FFFFFFFFFF    AAAAAA    LL            AAAAAA      CCCCCCCC  TTTTTTTTTT    IIIIII    NN      NN    IIIIII
FF          AA      AA  LL          AA      AA  CC              TT          II      NN      NN      II
FF          AA      AA  LL          AA      AA  CC              TT          II      NN      NN      II
FF          AA      AA  LL          AA      AA  CC              TT          II      NNNN    NN      II
FF          AA      AA  LL          AA      AA  CC              TT          II      NNNN    NN      II
FFFFFFFF    AA      AA  LL          AA      AA  CC              TT          II      NN  NN  NN      II
FFFFFFFF    AA      AA  LL          AA      AA  CC              TT          II      NN  NN  NN      II
FF          AAAAAAAAAA  LL          AAAAAAAAAA  CC              TT          II      NN    NNNN      II
FF          AAAAAAAAAA  LL          AAAAAAAAAA  CC              TT          II      NN    NNNN      II
FF          AA      AA  LL          AA      AA  CC              TT          II      NN      NN      II           ....
FF          AA      AA  LL          AA      AA  CC              TT          II      NN      NN      II           ....
FF          AA      AA  LLLLLLLLLL  AA      AA    CCCCCCCC      TT        IIIIII    NN      NN    IIIIII         ....
FF          AA      AA  LLLLLLLLLL  AA      AA    CCCCCCCC      TT        IIIIII    NN      NN    IIIIII         ....

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            .TITLE  FALACTINI - STATE TABLE ACTION ROUTINES
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ; Facility: FAL (DECnet File Access Listener)
0000    ;
0000    ; Abstract:
0000    ;
0000    ;       This module contains action routines called by the state table manager.
0000    ;
0000    ; Environment: VAX/VMS, user mode
0000    ;
0000    ; Author: James A. Krycka,      Creation Date:  16-JUN-1977
0000    ;
0000    ; Modified By:
0000    ;
0000    ;       V03-002 JAK0136         J A Krycka      07-MAR-1984
0000    ;               Change FAL$T_NAMESPEC and FAL$K_NAMESPEC to FAL$T_FILESPEC2
0000    ;               and FAL$K_FILESPEC2.
0000    ;
0000    ;       V03-001 KRM0066         K Malik         23-NOV-1982
0000    ;               Add FAL$INIT_RENAME routine.
0000    ;               Change FAL$T_EXPANDED, FAL$K_EXPANDED, FAL$T_RESULTANT and
0000    ;               FAL$K_RESULTANT symbols to FAL$T_EXPAND, FAL$K_EXPAND,
0000    ;               FAL$T_RESULT and FAL$K_RESULT.
0000    ;
0000    ;--
```

```
0000          .SBTTL  DECLARATIONS
0000
0000     ;
0000     ; Include Files:
0000     ;
0000
0000          $DAPPLGDEF                          ; Define DAP prologue symbols
0000          $DAPHDRDEF                          ; Define DAP message header
0000          $DAPCTLDEF                          ; Define DAP Control message
0000          $FABDEF                             ; Define File Access Block symbols
0000          $FALWRKDEF                          ; Define FAL Work Area symbols
0000          $NAMDEF                             ; Define Name Block symbols
0000          $RABDEF                             ; Define Record Access Block sym**
0000          $XABDEF                             ; Define symbols common to all XABs
0000          $XABALLDEF                          ; Define Allocation XAB symbols
0000          $XABDATDEF                          ; Define Date and Time XAB symbols
0000          $XABFHCDEF                          ; Define File Header Char XAB symbols
0000          $XABKEYDEF                          ; Define Key Definition XAB symbols
0000          $XABPRODEF                          ; Define Protection XAB symbols
0000          $XABRDTDEF                          ; Define Revision Date and Time symbols
0000          $XABSUMDEF                          ; Define Summary XAB symobls
0000
0000     ;
0000     ; Macros:
0000     ;
0000     ;    None
0000     ;
0000     ; Equated Symbols:
0000     ;
0000
0000          ASSUME  DAP$Q_DCODE_FLG EQ 0
0000          ASSUME  FAL$Q_FLG EQ 0
0000
0000     ;
0000     ; Own Storage:
0000     ;
```

```
    0000          .SBTTL  ACTION ROUTINES
00000000          .PSECT  FAL$CODE        NOSHR,EXE,RD,NOWRT,BYTE
    0000
    0000    ;++
    0000    ; Functional Description:
    0000    ;
    0000    ;       This module contains action routines invoked by the state table
    0000    ;       manager (FAL$STATE).
    0000    ;
    0000    ;       The input parameters and completion codes listed below are applicable
    0000    ;       for all of these action routines. Note that an action routine may use
    0000    ;       R0-R7 and AP without restoring them on exit. R0 on exit, however, must
    0000    ;       represent a status code to indicate success/failure of the routine or
    0000    ;       a true/false condition, as appropriate. This status code is used by
    0000    ;       the state table manager to advance to the next state.
    0000    ;
    0000    ; Calling Sequence:
    0000    ;
    0000    ;       BSBW    FAL$INIT (only routine at present)
    0000    ;
    0000    ; Input Parameters:
    0000    ;
    0000    ;       R8      Address of FAL work area
    0000    ;       R9      Address of DAP control block
    0000    ;       R10     Address of FAB
    0000    ;       R11     Address of RAB
    0000    ;
    0000    ; Implicit Inputs:
    0000    ;
    0000    ;       None
    0000    ;
    0000    ; Output Parameters:
    0000    ;
    0000    ;       R0      Status code
    0000    ;       R1-R7   Destroyed
    0000    ;       AP      Destroyed
    0000    ;
    0000    ; Implicit Outputs:
    0000    ;
    0000    ;       None
    0000    ;
    0000    ; Completion Codes:
    0000    ;
    0000    ;       R0      1 = success; 0 = failure
    0000    ;
    0000    ; Side Effects:
    0000    ;
    0000    ;       None
    0000    ;
    0000    ;--
```

```
                           0000          .SBTTL  FAL$INIT
                           0000
                           0000  ;++
                           0000  ; This routine initializes the FAL work area in preparation for the next file
                           0000  ; access request. This includes setting up the FAB, RAB, NAM, and FHCXAB control
                           0000  ; blocks and linking them together.
                           0000  ;--
                           0000
                           0000  FAL$INIT::                              ; Entry point
  68   00000702 8F    CA   0000          BICL2   #<<FAL$M_ATT_MSG>!-     ; Clear flags
                           0007                    <FAL$M_WILD>!-        ;
                           0007                    <FAL$M_FTM>!-         ;
                           0007                    <FAL$M_BLK_IO>!-      ;
                           0007                  0>,(R8)                 ;
       0000EDBE 8F    D0   0007          MOVL    #DAP$K_VALID_R2F,-      ; Define which DAP messages are valid
             1C A9         000D                  DAP$L_MSG_MASK(R9)      ;  to receive from partner
             00       90   000F          MOVB    #DAP$K_RAC_D,-          ; Set previous RAC value to the DAP
           01F7 C8         0011                  FAL$B_RAC(R8)           ;  default RAC value
                           0014
                           0014  ;
                           0014  ; Initialize the FAB, RAB, NAM, and FHCXAB control blocks.
                           0014  ;
                           0014
                           0014          ASSUME  FAL$K_FAB GE FAB$C_BLN
                           0014          ASSUME  FAL$K_RAB GE RAB$C_BLN
                           0014          ASSUME  FAL$K_NAM GE NAM$C_BLN
                           0014          ASSUME  FAL$K_FHCXAB GE XAB$C_FHCLEN
                           0014
                           0014          ASSUME  FAL$L_FAB+FAL$K_FAB+FAL$K_RAB+FAL$K_NAM EQ FAL$L_FHCXAB
                           0014
                           0014          $ZERO_FILL-                     ; Zero FAB, RAB, NAM, FHCXAB blocks
                           0014                  DST=(R10)-              ;  which must be contiguous in memory
                           0014                  SIZE=#<FAL$K_FAB+FAL$K_RAB+FAL$K_NAM+FAL$K_FHCXAB>
                           001C
                           001C          ASSUME  FAB$B_BID+1 EQ FAB$B_BLN
                           001C          ASSUME  RAB$B_BID+1 EQ RAB$B_BLN
                           001C          ASSUME  NAM$B_BID+1 EQ NAM$B_BLN
                           001C          ASSUME  XAB$B_COD+1 EQ XAB$B_BLN
                           001C
           5003 8F    B0   001C          MOVW    #<FAB$C_BLN@8+FAB$C_BID>,-
                6A         0020                  FAB$B_BID(R10)          ; Insert FAB block ID and length
           0400 C8    9E   0021          MOVAB   FAL$T_FILESPEC(R8),-    ; Store address of file specification
             2C AA         0025                  FAB$L_FNA(R10)          ;  string buffer in FAB
                           0027          $SETBIT #FAB$V_NAM,FAB$L_FOP(R10);Denote NAM block is present
           4401 8F    B0   002C          MOVW    #<RAB$C_BLN@8+RAB$C_BID>,-
                6B         0030                  RAB$B_BID(R11)          ; Insert RAB block ID and length
  57       0294 C8    9E   0031          MOVAB   FAL$L_NAM(R8),R7        ; Get address of NAM block
           6002 8F    B0   0036          MOVW    #<NAM$C_BLN@8+NAM$C_BID>,-
                67         003A                  NAM$B_BID(R7)           ; Insert NAM block ID and length
           0500 C8    9E   003B          MOVAB   FAL$T_EXPAND(R8),-      ; Store address of expanded string
             0C A7         003F                  NAM$L_ESA(R7)           ;  buffer in NAM block
             FF 8F    90   0041          MOVB    #FAL$K_EXPAND,-         ; Store size of expanded string
             0A A7         0044                  NAM$B_ESS(R7)           ;  buffer in NAM block
           0600 C8    9E   0046          MOVAB   FAL$T_RESULT(R8),-      ; Store address of resultant string
             04 A7         004A                  NAM$L_RSA(R7)           ;  buffer in NAM block
             FF 8F    90   004C          MOVB    #FAL$K_RESULT,-         ; Store size of resultant string
             02 A7         004F                  NAM$B_RSS(R7)           ;  buffer in NAM block
           08 A7    01     90  0051          MOVB    #NAM$M_PWD,NAM$B_NOP(R7); Do not mask out password in node spec
              2C1D 8F      B0  0055          MOVW    #<XAB$C_FHCLEN@8+XAB$C_FHC>,-
              02F4 C8          0059                  FAL$L_FHCXAB+XAB$B_COD(R8) ; Insert FHCXAB ID and length
                               005C
                               005C    ;
                               005C    ; Link only the RAB, NAM, and FHCXAB to the FAB at this time.
                               005C    ; The other XABs will be chained in as required when the DAP Access message
                               005C    ; is processed.
                               005C    ;
                               005C
           3C AB    5A     D0  005C          MOVL    R10,RAB$L_FAB(R11)      ; Store FAB pointer in RAB
              0294 C8      DE  0060          MOVAL   FAL$L_NAM(R8),-         ; Store NAM pointer in FAB
                28 AA          0064                  FAB$L_NAM(R10)          ;
              02F4 C8      DE  0066          MOVAL   FAL$L_FHCXAB(R8),-      ; Store FHCXAB pointer in XAB chain
                24 AA          006A                  FAB$L_XAB(R10)          ;
              02F8 C8      DE  006C          MOVAL   FAL$L_FHCXAB+XAB$L_NXT(R8),-
                7C A8          0070                  FAL$L_CHAIN_NXT(R8)     ; Save address of next chain pointer
                               0072
                               0072    ;
                               0072    ; Initialize XAB related fields in the FAL work area.
                               0072    ;
                               0072
                72 A8      B4  0072          CLRW    FAL$W_RECEIVED(R8)      ; Zero mask of received XABs to chain
                74 A8      D4  0075          CLRL    FAL$L_ALLXABINI(R8)     ; Zero list of ALLXABs initialized
                78 A8      D4  0078          CLRL    FAL$L_KEYXABINI(R8)     ; Zero list of KEYXABs initialized
                               007B
                               007B    ;
                               007B    ; Initialize volume and directory name descriptors for possible wildcard
                               007B    ; processing.
                               007B    ;
                               007B
                               007B          ASSUME  FAL$Q_VOLNAME+8 EQ FAL$Q_DIRNAME
                               007B
        50    0080 C8      7E  007B          MOVAQ   FAL$Q_VOLNAME(R8),R0    ; Get address of first descriptor
                   80      D4  0080          CLRL    (R0)+                   ; Initialize volume name descriptor
        80    1E00 C8      9E  0082          MOVAB   FAL$T_VOLNAME(R8),(R0)+ ;
                   80      D4  0087          CLRL    (R0)+                   ; Initialize directory name descriptor
        80    1F00 C8      9E  0089          MOVAB   FAL$T_DIRNAME(R8),(R0)+ ;
                               008E
                               008E    ;
                               008E    ; Exit state with success if a valid Configuration message has been processed;
                               008E    ; otherwise, exit state with failure.
                               008E    ;
                               008E
   50   68    01    00     EF  008E          EXTZV   #FAL$V_CNF_MSG,#1,(R8),R0
                           05  0093          RSB                             ; Exit state with status code in R0
```

```
                        0094            .SBTTL  FAL$INIT_RENAME
                        0094
                        0094    ;++
                        0094    ; This routine initializes the secondary FAB and NAM control blocks in the FAL
                        0094    ; work area which are used by the rename operation.
                        0094    ;--
                        0094
                        0094    FAL$INIT_RENAME::                               ; Entry point
                        0094
                        0094            ASSUME  FAL$K_FAB2 GE FAB$C_BLN
                        0094            ASSUME  FAL$K_NAM2 GE NAM$C_BLN
                        0094
                        0094            ASSUME  FAL$L_FAB2+FAL$K_FAB2 EQ FAL$L_NAM2
                        0094
5A    0800 C8     DE    0094            MOVAL   FAL$L_FAB2(R8),R10      ; Get address of FAB2 in R10
                        0099            $ZERO_FILL-                     ; Zero FAB2 and NAM2 blocks which must
                        0099                    DST=(R10)-              ;  be contiguous in memory
                        0099                    SIZE=#<FAL$K_FAB2+FAL$K_NAM2>
                        00A1
      5003 8F     B0    00A1            MOVW    #<FAB$C_BLN@8+FAB$C_BID>,-
             6A         00A5                    FAB$B_BID(R10)          ; Insert FAB block ID and length
      0900 C8     9E    00A6            MOVAB   FAL$T_FILESPEC2(R8),-   ; Store address of file specification
        2C AA           00AA                    FAB$L_FNA(R10)          ;  string buffer in FAB
                        00AC            $SETBIT #FAB$V_NAM,FAB$L_FOP(R10);Denote NAM block is present
57    0850 C8     9E    00B1            MOVAB   FAL$L_NAM2(R8),R7       ; Get address of NAM2 block
      6002 8F     B0    00B6            MOVW    #<NAM$C_BLN@8+NAM$C_BID>,-
             67         00B7                    NAM$B_BID(R7)           ; Insert NAM block ID and length
      0A00 C8     9E    00BB            MOVAB   FAL$T_EXPAND2(R8),-     ; Store address of expanded string
        0C A7           00BF                    NAM$L_ESA(R7)           ;  buffer in NAM block
        FF 8F     90    00C1            MOVB    #FAL$K_EXPAND2,-        ; Store size of expanded string
        0A A7           00C4                    NAM$B_ESS(R7)           ;  buffer in NAM block
      0B00 C8     9E    00C6            MOVAB   FAL$T_RESULT2(R8),-     ; Store address of resultant string
        04 A7           00CA                    NAM$L_RSA(R7)           ;  buffer in NAM block
        FF 8F     90    00CC            MOVB    #FAL$K_RESULT2,-        ; Store size of resultant string
        02 A7           00CF                    NAM$B_RSS(R7)           ;  buffer in NAM block
   08 A7    01    90    00D1            MOVB    #NAM$M_PWD,NAM$B_NOP(R7); Do not mask out password in node spec
                        00D5
                        00D5    ;
                        00D5    ; Link the secondary NAM to the secondary FAB.
                        00D5    ;
                        00D5
      0850 C8     DE    00D5            MOVAL   FAL$L_NAM2(R8),-        ; Store NAM2 pointer in FAB2
        28 AA           00D9                    FAB$L_NAM(R10)          ;
5A    0200 C8     DE    00DB            MOVAL   FAL$L_FAB(R8),R10       ; Restore R10 to FAB pointer
50      01        D0    00E0            MOVL    #1,R0                   ; Return success code
                  05    00E3            RSB
                  00E4          .SBTTL  FAL$INIT_XABCHN
                  00E4
                  00E4  ;++
                  00E4  ; This routine initializes the XAB chain and related XAB fields in the FAL
                  00E4  ; work area in preparation for receiving XAB related messages from partner
                  00E4  ; after the file has been opened or created (e.g., XABs are input to the RMS
                  00E4  ; $EXTEND and $CLOSE services).
                  00E4  ;--
                  00E4
                  00E4  FAL$INIT_XABCHN::                           ; Entry point
       24 AA   D4 00E4          CLRL    FAB$L_XAB(R10)              ; Remove any XABs from chain
       24 AA   DE 00E7          MOVAL   FAB$L_XAB(R10),-            ;
       7C A8      00EA                  FAL$L_CHAIN_NXT(R8)         ; Save address of next chain pointer
       72 A8   B4 00EC          CLRW    FAL$W_RECEIVED(R8)          ; Zero mask of receiv^d XABs to chain
       74 A8   D4 00EF          CLRL    FAL$L_ALLXABINI(R8)         ; Zero list of ALLXABs initialized
       78 A8   D4 00F2          CLRL    FAL$L_KEYXABINI(R8)         ; Zero list of KEYXABs initialized
    50    01   D0 00F5          MOVL    #1,R0                       ; Return success code
               05 00F8          RSB                                 ; Exit state with status code in R0
    00F9          .SBTTL  SUPPORT ROUTINES
000000F9          .PSECT  FAL$CODE        NOSHR,EXE,RD,NOWRT,BYTE
    00F9
    00F9  ;++
    00F9  ; Functional Description:
    00F9  ;
    00F9  ;       The following routines are called by other action routines, not by the
    00F9  ;       state table manager (FAL$STATE).
    00F9  ;
    00F9  ;       These routines initialize the specified XAB. For Allocation and Key
    00F9  ;       Definition XABs, the area ID and key of reference value, respectively,
    00F9  ;       is an input parameter.
    00F9  ;
    00F9  ; Calling Sequence:
    00F9  ;
    00F9  ;       BSBW    FAL$INIT_ALLXAB
    00F9  ;       BSBW    FAL$INIT_DATXAB
    00F9  ;       BSBW    FAL$INIT_KEYXAB
    00F9  ;       BSBW    FAL$INIT_PROXAB
    00F9  ;       BSBW    FAL$INIT_RDTXAB
    00F9  ;       BSBW    FAL$INIT_SUMXAB
    00F9  ;
    00F9  ; Input Parameters:
    00F9  ;
    00F9  ;       R6      AID value for FAL$INIT_ALLXAB; REF value for FAL$INIT_KEYXAB
    00F9  ;
    00F9  ; Implicit Inputs:
    00F9  ;
    00F9  ;       None
    00F9  ;
    00F9  ; Output Parameters:
    00F9  ;
    00F9  ;       R0      Status code
    00F9  ;       R1-R5   Destroyed
    00F9  ;       R6      Unchanged
    00F9  ;       R7      Address of XAB initialized
    00F9  ;
    00F9  ; Implicit Outputs:
    00F9  ;
    00F9  ;       None
    00F9  ;
    00F9  ; Completion Codes:
    00F9  ;
    00F9  ;       R0      1 = success; 0 = failure
    00F9  ;
    00F9  ; Side Effects:
    00F9  ;
    00F9  ;       None
    00F9  ;
    00F9  ;--
    00F9
    00F9          ASSUME  XAB$B_COD+1 EQ XAB$B_BLN
    00F9          ASSUME  FAL$K_ALLXAB GE XAB$C_ALLLEN
    00F9          ASSUME  FAL$K_DATXAB GE XAB$C_DATLEN
    00F9          ASSUME  FAL$K_KEYXAB GE XAB$C_KEYLEN_V2
    00F9          ASSUME  FAL$K_PROXAB GE XAB$C_PROLEN
    00F9          ASSUME  FAL$K_RDTXAB GE XAB$C_RDTLEN
```

```
                            00F9   369          ASSUME  FAL$K_SUMXAB GE XAB$C_SUMLEN
```

```
                     00F9          .SBTTL  FAL$INIT_DATXAB, FAL$INIT_PROXAB
                     00F9          .SBTTL  FAL$INIT_SUMXAB, FAL$INIT_RDTXAB
                     00F9          .SBTTL  FAL$INIT_ALLXAB, FAL$INIT_KEYXAB
                     00F9
                     00F9   ;++
                     00F9   ; This routine initializes the Date and Time XAB.
                     00F9   ;--
                     00F9
                     00F9   FAL$INIT_DATXAB::                            ; Entry point
 57   0320 C8   DE   00F9          MOVAL   FAL$L_DATXAB(R8),R7          ; Get address of DATXAB
                     00FE          $ZERO_FILL-                          ; Zero DATXAB
                     00FE                  DST=(R7)-                    ;
                     00FE                  SIZE=#FAL$K_DATXAB           ;
      2C12 8F   B0   0104          MOVW    #<XAB$C_DATLEN@8+XAB$C_DAT>,-
           67        0108                  XAB$B_COD(R7)                ; Insert DATXAB ID and length
      0091      31   0109          BRW     INIT_SUC                     ; All done
                     010C
                     010C   ;++
                     010C   ; This routine initializes the Protection XAB.
                     010C   ;--
                     010C
                     010C   FAL$INIT_PROXAB::                            ; Entry point
 57   034C C8   DE   010C          MOVAL   FAL$L_PROXAB(R8),R7          ; Get address of PROXAB
                     0111          $ZERO_FILL-                          ; Zero PROXAB
                     0111                  DST=(R7)-                    ;
                     0111                  SIZE=#FAL$K_PROXAB           ;
      5813 8F   B0   0119          MOVW    #<XAB$C_PROLEN@8+XAB$C_PRO>,-
           67        011D                  XAB$B_COD(R7)                ; Insert PROXAB ID and length
           7D   11   011E          BRB     INIT_SUC                     ; All done
                     0120
                     0120   ;++
                     0120   ; This routine initializes the Summary XAB.
                     0120   ;--
                     0120
                     0120   FAL$INIT_SUMXAB::                            ; Entry point
 57   03A4 C8   DE   0120          MOVAL   FAL$L_SUMXAB(R8),R7          ; Get address of SUMXAB
                     0125          $ZERO_FILL-                          ; Zero SUMXAB
                     0125                  DST=(R7)-                    ;
                     0125                  SIZE=#FAL$K_SUMXAB           ;
      0C16 8F   B0   012B          MOVW    #<XAB$C_SUMLEN@8+XAB$C_SUM>,-
           67        012F                  XAB$B_COD(R7)                ; Insert SUMXAB ID and length
           6B   11   0130          BRB     INIT_SUC                     ; All done
                     0132
                     0132   ;++
                     0132   ; This routine initializes the Revision Date and Time XAB.
                     0132   ;--
                     0132
                     0132   FAL$INIT_RDTXAB::                            ; Entry point
 57   03B0 C8   DE   0132          MOVAL   FAL$L_RDTXAB(R8),R7          ; Get address of RDTXAB
                     0137          $ZERO_FILL-                          ; Zero RDTXAB
                     0137                  DST=(R7)-                    ;
                     0137                  SIZE=#FAL$K_RDTXAB           ;
      141E 8F   B0   013D          MOVW    #<XAB$C_RDTLEN@8+XAB$C_RDT>,-
           67        0141                  XAB$B_COD(R7)                ; Insert RDTXAB ID and length
           59   11   0142          BRB     INIT_SUC                     ; All done
                     0144
                     0144   ;++
                                   0144  ; This routine initializes the Allocation XAB (by area ID).
                                   0144  ;--
                                   0144
                                   0144  FAL$INIT_ALLXAB::                              ; Entry point
                  1F   56   D1     0144          CMPL    R6,#FAL$K_MAX_AID              ; Return error if area ID value
                       58   1A     0147          BGTRU   INIT_ERR                       ;  is too large
            50    20   56   C5     0149          MULL3   R6,#FAL$K_ALLXAB,R0            ; Using AID as an index, compute
        57   0C00 C840      9E     014D          MOVAB   FAL$L_ALLXAB(R8)[R0],R7 ;  address of Allocation XAB to use
                                   0153          $ZERO_FILL-                            ; Zero ALLXAB
                                   0153                  DST=(R7)-                      ;
                                   0153                  SIZE=#FAL$K_ALLXAB             ;
                2014 8F     B0     0159          MOVW    #<XAB$C_ALLLEN@8+XAB$C_ALL>,-
                       67          015D                  XAB$B_COD(R7)                  ; Insert ALLXAB ID and length
             17 A7     56   90     015E          MOVB    R6,XAB$B_AID(R7)               ; Store area ID value
                                   0162          $SETBIT R6,FAL$L_ALLXABINI(R8)         ; Denote which ALLXAB was initialized
                       34   11     0167          BRB     INIT_SUC                       ; All done
                                   0169
                                   0169  ;++
                                   0169  ; This routine initializes the Key Definition XAB (by key of reference).
                                   0169  ;--
                                   0169
                                   0169  FAL$INIT_KEYXAB::                              ; Entry point
                  1F   56   D1     0169          CMPL    R6,#FAL$K_MAX_REF              ; Return error if key of reference value
                       33   1A     016C          BGTRU   INIT_ERR                       ;  is too large
  50    0000004C 8F    56   C5     016E          MULL3   R6,#FAL$K_KEYXAB,R0            ; Using REF as an index, compute
        57   1000 C840      9E     0176          MOVAB   FAL$L_KEYXAB(R8)[R0],R7 ;  address of KEYXAB to use
                                   017C          $ZERO_FILL-                            ; Zero KEYXAB
                                   017C                  DST=(R7)-                      ;
                                   017C                  SIZE=#FAL$K_KEYXAB             ;
            50    20   56   C5     0184          MULL3   R6,#FAL$K_KEYNAM,R0            ; Using REF as an index, compute
     38 A7   1C00 C840      9E     0188          MOVAB   FAL$L_KEYNAM(R8)[R0],-  ;  address of key name buffer to use
                                   018F                  XAB$L_KNM(R7)          ;  and store address in XAB
                4015 8F     B0     018F          MOVW    #<XAB$C_KEYLEN_V2@8+XAB$C_KEY>,-
                       67          0193                  XAB$B_COD(R7)                  ; Insert KEYXAB ID and length
             17 A7     56   90     0194          MOVB    R6,XAB$B_REF(R7)               ; Store key of reference value
                                   0198          $SETBIT R6,FAL$L_KEYXABINI(R8)         ; Denote which KEYXAB was initialized
                                   019D
                                   019D  ;++
                                   019D  ; Common exit paths.
                                   019D  ;--
                                   019D
                                   019D  INIT_SUC:                                      ;
                  50   01   D0     019D          MOVL    #1,R0                          ; Return success code
                            05     01A0          RSB                                    ; Exit
                                   01A1  INIT_ERR:                                      ;
                       50   D4     01A1          CLRL    R0                             ; Return failure code
                            05     01A3          RSB                                    ; Exit
                                   01A4
                                   01A4          .END                                   ; End of module
```

```
DAP$B_BITCNT                        00000035          FAL$B_RBK_CACHE                 00000012
DAP$B_BLKCNT                        00000056          FAL$B_RCVBUFIDX                 00000011
DAP$B_CTLFUNC                       00000040          FAL$B_VALUE                     00000010
DAP$B_DCODE_FID                     00000019          FAL$C_WRKBLN                    00002000
DAP$B_DCODE_MAC                     0000001B          FAL$INIT                        00000000 RG     02
DAP$B_DCODE_MSG                     0000001A          FAL$INIT_ALLXAB                 00000144 RG     02
DAP$B_FLAGS                         00000031          FAL$INIT_DATXAB                 000000F9 RG     02
DAP$B_KRF                           00000047          FAL$INIT_KEYXAB                 00000169 RG     02
DAP$B_LEN256                        00000034          FAL$INIT_PROXAB                 0000010C RG     02
DAP$B_LENGTH                        00000033          FAL$INIT_RDTXAB                 00000132 RG     02
DAP$B_RAC                           00000046          FAL$INIT_RENAME                 00000094 RG     02
DAP$B_STREAMID                      00000032          FAL$INIT_SUMXAB                 00000120 RG     02
DAP$B_TYPE                          00000030          FAL$INIT_XABCHN                 000000E4 RG     02
DAP$B_X_FIELD                       00000024          FAL$K_ALLXAB                  = 00000020
DAP$C_BLN                           000000C0          FAL$K_DATXAB                  = 0000002C
DAP$K_BLN                           000000C0          FAL$K_EXPAND                  = 000000FF
DAP$K_RAC_D                       = 00000000          FAL$K_EXPAND2                 = 000000FF
DAP$K_SEQ_ACC                     = 00000000          FAL$K_FAB                     = 00000050
DAP$K_VALID_R2F                   = 0000EDBE          FAL$K_FAB2                    = 00000050
DAP$L_CMWA                          00000030          FAL$K_FHCXAB                  = 0000002C
DAP$L_CRC_RSLT                      00000020          FAL$K_KEYNAM                  = 00000020
DAP$L_DCODE_STS                     00000018          FAL$K_KEYXAB                  = 0000004C
DAP$L_MSG_MASK                      0000001C          FAL$K_MAX_AID                 = 0000001F
DAP$L_ROP                           00000050          FAL$K_MAX_REF                 = 0000001F
DAP$L_SSPWA                         00000080          FAL$K_NAM                     = 00000060
DAP$L_TEMP                          00000090          FAL$K_NAM2                    = 00000060
DAP$M_BITCNT                      = 00000008          FAL$K_PROXAB                  = 00000058
DAP$M_BLKCNT                      = 00000040          FAL$K_RAB                     = 00000044
DAP$M_SEGMENT                     = 00000040          FAL$K_RDTXAB                  = 00000014
DAP$M_TMP1$                       = 00000008          FAL$K_RESULT                  = 000000FF
DAP$M_TMP2$                       = FFF80000          FAL$K_RESULT2                 = 000000FF
DAP$Q_DCODE_FLG                     00000000          FAL$K_SUMXAB                  = 0000000C
DAP$Q_KEY                           00000048          FAL$K_WRKBLN                    00002000
DAP$Q_MSG_BUF1                      00000008          FAL$L_ALLXAB                    00000C00
DAP$Q_MSG_BUF2                      00000010          FAL$L_ALLXABINI                 00000074
DAP$Q_SYSPEC                        00000038          FAL$L_CHAIN_NXT                 0000007C
DAP$W_CTLMENU                       00000044          FAL$L_DATXAB                    00000320
DAP$W_DISPLAY2                      00000054          FAL$L_FAB                       00000200
DAP$W_PARTNER                       00000006          FAL$L_FAB2                      00000800
DAP$W_VERSION                       00000004          FAL$L_FHCXAB                    000002F4
FAB$B_BID                         = 00000000          FAL$L_FOP                       000001F8
FAB$B_BLN                         = 00000001          FAL$L_KEYNAM                    00001C00
FAB$C_BID                         = 00000003          FAL$L_KEYXAB                    00001000
FAB$C_BLN                         = 00000050          FAL$L_KEYXABINI                 00000078
FAB$L_FNA                         = 0000002C          FAL$L_NAM                       00000294
FAB$L_FOP                         = 00000004          FAL$L_NAM2                      00000850
FAB$L_NAM                         = 00000028          FAL$L_NUMBER                    000001FC
FAB$L_XAB                         = 00000024          FAL$L_PROXAB                    0000034C
FAB$V_NAM                         = 00000018          FAL$L_RAB                       00000250
FAL$B_ACCFUNC                       000001F6          FAL$L_RCVBUF                    0000005C
FAL$B_ACCOPT                        000001F5          FAL$L_RDTXAB                    000003B0
FAL$B_DATATYPE                      000001F4          FAL$L_RMS_PTR                   0000006C
FAL$B_DISABLE                       00000006          FAL$L_STB                       000000C0
FAL$B_ENABLE                        00000005          FAL$L_SUMXAB                    000003A4
FAL$B_LOGGING                       00000004          FAL$L_TEMP                      000003F4
FAL$B_MISCOPT                       00000007          FAL$L_USE_SC1                   000000A8
FAL$B_RAC                           000001F7          FAL$L_USE_SC2                   000000AC
```

```
FAL$L_USE_VER                    000000A4                 RAB$B_BLN                 = 00000001
FAL$M_ATT_MSG                  = 00000002                 RAB$C_BID                 = 00000001
FAL$M_BLK_IO                   = 00000200                 RAB$C_BLN                 = 00000044
FAL$M_FTM                      = 00000100                 RAB$L_FAB                 = 0000003C
FAL$M_WILD                     = 00000400                 XAB$B_AID                 = 00000017
FAL$Q_BLD                        00000050                 XAB$B_BLN                 = 00000001
FAL$Q_DIRNAME                    00000088                 XAB$B_COD                 = 00000000
FAL$Q_FALLOG                     00000090                 XAB$B_REF                 = 00000017
FAL$Q_FLG                        00000000                 XAB$C_ALL                 = 00000014
FAL$Q_MBX                        00000038                 XAB$C_ALLLEN              = 00000020
FAL$Q_MBXIOSB                    00000030                 XAB$C_DAT                 = 00000012
FAL$Q_RCV                        00000040                 XAB$C_DATLEN              = 0000002C
FAL$Q_RCVIOSB                    00000020                 XAB$C_FHC                 = 0000001D
FAL$Q_RMS                        00000064                 XAB$C_FHCLEN              = 0000002C
FAL$Q_STATE_CTX                  00000008                 XAB$C_KEY                 = 00000015
FAL$Q_SYSNET                     00000098                 XAB$C_KEYLEN_V2           = 00000040
FAL$Q_TEMP                       000003F8                 XAB$C_PRO                 = 00000013
FAL$Q_VOLNAME                    00000080                 XAB$C_PROLEN              = 00000058
FAL$Q_XMT                        00000048                 XAB$C_RDT                 = 0000001E
FAL$Q_XMTIOSB                    00000028                 XAB$C_RDTLEN              = 00000014
FAL$T_DAP                        00000100                 XAB$C_SUM                 = 00000016
FAL$T_DIRNAME                    00001F00                 XAB$C_SUMLEN              = 0000000C
FAL$T_EXPAND                     00000500                 XAB$L_KNM                 = 00000038
FAL$T_EXPAND2                    00000A00                 XAB$L_NXT                 = 00000004
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$T_VOLNAME                    00001E00
FAL$V_CNF_MSG                  = 00000000
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
INIT_ERR                         000001A1 R      02
INIT_SUC                         0000019D R      02
NAM$B_BID                      = 00000000
NAM$B_BLN                      = 00000001
NAM$B_ESS                      = 0000000A
NAM$B_NOP                      = 00000008
NAM$B_RSS                      = 00000002
NAM$C_BID                      = 00000002
NAM$C_BLN                      = 00000060
NAM$L_ESA                      = 0000000C
NAM$L_RSA                      = 00000004
NAM$M_PWD                      = 00000001
RAB$B_BID                      = 00000000
```

```
                                        +----------------+
                                        ! 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                        000001A4  (   420.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD  NOWRT NOVEC BYTE

                                      +------------------------+
                                      ! Performance indicators .
                                      +------------------------+

Phase                       Page faults    CPU Time       Elapsed Time
-----                       -----------    --------       ------------
Initialization                      29     00:00:00.07    00:00:01.13
Command processing                 146     00:00:00.41    00:00:02.11
Pass 1                             322     00:00:08.11    00:00:28.29
Symbol table sort                    5     00:00:01.04    00:00:03.28
Pass 2                              94     00:00:01.56    00:00:10.03
Symbol table output                 25     00:00:00.13    00:00:00.81
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               625     00:00:11.35    00:00:45.68

The working set limit was 1650 pages.
64571 bytes (127 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1130 non-local and 4 local symbols.
476 source lines were read in Pass 1, producing 13 object records in Pass 2.
25 pages of virtual memory were used to define 24 macros.

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

Macro library name                    Macros defined
------------------                    --------------
_$255$DUA28:[FAL.OBJ]FAL.MLB;1                     6
_$255$DUA28:[SYSLIB]STARLET.MLB;2                 15
TOTALS (all libraries)                            21

1320 GETS were required to define 21 macros.

There were no errors, warnings or information messages.

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

0174 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

FALACTION
LIS

FALDAPCRS
LIS

FALMACROS
MAR

FALBLDXAB
LIS

FALBLDATT
LIS

FALBLDSTS
LIS

FALDEF
MDL

FALDAPIO
LIS

FALACTINI
LIS

FALACTMSG
LIS