C 13

```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
```

```
RRRRRRRR    MM      MM    SSSSSSSS    000000    DDDDDDDD      IIIIII      SSSSSSSS  PPPPPPPP    LL
RRRRRRRR    MM      MM    SSSSSSSS    000000    DDDDDDDD      IIIIII      SSSSSSSS  PPPPPPPP    LL
RR      RR  MMMM  MMMM  SS          00      00  DD      DD      II      SS          PP      PP  LL
RR      RR  MMMM  MMMM  SS          00      00  DD      DD      II      SS          PP      PP  LL
RR      RR  MM  MM  MM  SS          00    0000  DD      DD      II      SS          PP      PP  LL
RR      RR  MM  MM  MM  SS          00    0000  DD      DD      II      SS          PP      PP  LL
RRRRRRRR    MM      MM    SSSSSS    00  00  00  DD      DD      II        SSSSSS    PPPPPPPP    LL
RRRRRRRR    MM      MM    SSSSSS    00  00  00  DD      DD      II        SSSSSS    PPPPPPPP    LL
RR  RR      MM      MM          SS  0000    00  DD      DD      II              SS  PP          LL
RR  RR      MM      MM          SS  0000    00  DD      DD      II              SS  PP          LL
RR    RR    MM      MM          SS  00      00  DD      DD      II              SS  PP          LL              ....
RR    RR    MM      MM          SS  00      00  DD      DD      II              SS  PP          LL              ....
RR      RR  MM      MM  SSSSSSSS      000000    DDDDDDDD      IIIIII    SSSSSSSS    PP          LLLLLLLLLL      ....
RR      RR  MM      MM  SSSSSSSS      000000    DDDDDDDD      IIIIII    SSSSSSSS    PP          LLLLLLLLLL      ....


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
```

RM
VO

Table of contents

```
    (2)      180       DECLARATIONS
    (3)      250       RMS$DISPLAY - $DISPLAY ROUTINE
    (6)      397       DSPXAB - Handle general, non-ISAM  XAB attributes
    (7)      489       DSPFAB - Handle FAB attributes
    (8)      539       DSPRAB - Handle RAB attributes
    (9)      562       ISAM_XABS - Handle Indexed file XAB attributes
    (10)     579       READ_ATTR - SUBROUTINE TO READ FILE ATTRIBUTES
```

```
0000            $BEGIN  RMSODISPL,000,RM$RMS,<DISPATCH FOR DISPLAY OPERATION>
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    ; FACILITY: RMS32
0000    ;
0000    ; ABSTRACT:
0000    ;               This module is the highest level control routine
0000    ;               to perform the $DISPLAY function.
0000    ;
0000    ; ENVIRONMENT:
0000    ;               STAR processor running STARLET EXEC.
0000    ;
0000    ; AUTHOR: L F Laverdure,        CREATION DATE: 19-Jan-1978
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-021 JWT0175         Jim Teague              12-Apr-1984
0000    ;               Complete the implementation of access mode ATRs.
0000    ;
0000    ;       V03-020 JWT0173         Jim Teague               1-Apr-1984
0000    ;               Disable access mode ATRs for now.
0000    ;
0000    ;       V03-019 JWT0172         Jim Teague              28-Mar-1984
0000    ;               Put byte specifying EXEC mode access in last longword
0000    ;               of ATR work area.
0000    ;
0000    ;       V03-018 DAS0001         David Solomon           25-Mar-1984
0000    ;               Add $ATRDEF.
0000    ;
0000    ;       V03-017 DGB0028         Donald G. Blair         22-Mar-1984
0000    ;               Have RM$XAB_SCAN process protection xab AFTER call to
0000    ;               ACP as part of implementation of ACL's.
0000    ;
```

```
0000    ;       V03-016 JWT0166         Jim Teague              20-Mar-1984
0000    ;               Use access-mode attributes in those places where
0000    ;               RMS requests the ACP to probe user structures.
0000    ;               Also use a dynamically-allocated scratch page for
0000    ;               accumulating ATRs for QIOs.
0000    ;
0000    ;       V03-015 DGB0009         Donald G. Blair         01-Mar-1984
0000    ;               Make changes related to ACP calls as part of the
0000    ;               restructuring necessary to implement access mode
0000    ;               protected files.  Also use RM$XABOPN_ARGS rather than
0000    ;               XABOPN_ARGS.
0000    ;
0000    ;       V03-014 RAS0214         Ron Schaefer            22-Nov-1983
0000    ;               Fix RAS0210 to set R10 more carefully.
0000    ;               Fix network $DISPLAY for task-to-task.
0000    ;
0000    ;       V03-013 RAS0210         Ron Schaefer             4-Nov-1983
0000    ;               Revise this in several ways:
0000    ;                   Use the real FWA for storage;
0000    ;                   Use RM$RET_DEV_CHAR to get device characteristics;
0000    ;                   Properly store the RFM and ORG fields.
0000    ;
0000    ;       V03-012 RAS0193         Ron Schaefer            20-Sep-1983
0000    ;               Increase size of attribute buffer to 16.
0000    ;               This is a temp fix; real solution is to use existing FWA.
0000    ;
0000    ;       V03-011 RAS0163         Ron Schaefer            27-Jun-1983
0000    ;               Eliminate reference to RM$BASIC_ERR.
0000    ;
0000    ;       V03-010 TSK001          Tamar Krichevsky        12-Jun-1983
0000    ;               Fix broken branches to journaling routines.
0000    ;
0000    ;       V03-009 LJA0071         Laurie J. Anderson      29-Apr-1983
0000    ;               Fix problem with displaying terminals
0000    ;               When displaying FAB, fill in ALQ field, too.
0000    ;               Note: Still need to handle shared file case in filling in ALQ
0000    ;
0000    ;       V03-008 RAS0148         Ron Schaefer            26-Apr-1983
0000    ;               Add initial support for extended XABPRO.
0000    ;
0000    ;       V03-007 LJA0060         Laurie J. Anderson      21-Feb-1983
0000    ;               Add the display of RAB and FAB fields which are available
0000    ;               in the IRAB and IFAB and so could be displayed.
0000    ;               Add the display of the NAM block, if linked to FAB, from
0000    ;               the permanent FWA space.  Add some more comments.
0000    ;
0000    ;       V03-006 LJA0051         Laurie J. Anderson      13-Jan-1983
0000    ;               Add ability to display RAB, and add "wild ISI" support
0000    ;               for Context Extraction.  "Wild ISI" support involves the
0000    ;               return of the "next ISI" in the linked list of IRAB's.
0000    ;               This value is returned in the STV.
0000    ;
0000    ;       V03-005 JWH0115         Jeffrey W. Horn         22-Oct-1982
0000    ;               Fix bug in V03-004 to reference correct register
0000    ;               for FIB in READ_ATTR.
0000    ;
0000    ;       V03-004 JWH0108         Jeffrey W. Horn         28-Sep-1982
```

```
0000  ;             Add processing for $XABJNL.
0000  ;
0000  ;     V03-003 KBT03xx         Keith B. Thompson       10-Aug-1982
0000  ;             Remove $FRBDEF
0000  ;
0000  ;     V03-002 KBT0178         Keith B. Thompson       23-Aug-1982
0000  ;             Reorganize psects and rename entry point to single '$'
0000  ;
0000  ;     V03-001 CDS0003         C Saether       30-Mar-1982
0000  ;             Correct display of isam xab's so that BIO connect
0000  ;             after BRO open works.
0000  ;
0000  ;     V02-015 CDS0002         C Saether       22-Jan-1982
0000  ;             Use RM$ALLOC_BUF routine to allocate/deallocate
0000  ;             buffers so that global buffers are handled correctly.
0000  ;
0000  ;     V02-014 CDS0001         C Saether       29-Aug-1981
0000  ;             Remove references to BCB's.  Use BLB's instead.
0000  ;
0000  ;     V02-013 JAK0063         J A Krycka      28-AUG-1981
0000  ;             Add support for network $DISPLAY.
0000  ;
0000  ;     V02-012 MCN0007         Maria del C. Nasr       12-May-1981
0000  ;             Define new symbol for old length of backup date and time XAB.
0000  ;
0000  ;     V02-011 REFORMAT        Frederick E. Deen, Jr.      28-Jul-1980
0000  ;             This code was reformatted to adhere to RMS standards
0000  ;
0000  ;     V010    CDS0060         C Saether       6-Dec-1979
0000  ;             Fixup relative to V009.  Use RM$BDBALLOC_ALT to allocate
0000  ;             buffer, etc., for ISAM.
0000  ;
0000  ;     V009    CDS0051         C Saether       2-Nov-1979
0000  ;             Use RM$ALLOC_BUF to handle allocate buffer for
0000  ;             ISAM org only - others don't need it
0000  ;
0000  ;     V008    DMB0002          D M Bousquet   13-Feb-1979
0000  ;             Modified to always allocate a BDB, BUFFER and BCB (if SHARED)
0000  ;             and release everything we allocated at finish
0000  ;
0000  ;     V007    DMB0001          D M Bousquet    10-Jan-1979
0000  ;             Changed RM$RETBDB to a CACHE and RELEASE CALL
0000  ;
0000  ;     V006    CDS0001         C D Saether     2-Jan-1979
0000  ;             Resolve out of range BRANCH
0000  ;
0000  ;     V005    RAN0002         R A Newell      6-Sep-1978
0000  ;             RMS-32 ISAM modifications.  Processing of summary, key
0000  ;             and area xabs
0000  ;
0000  ;     V004    JAK0001         J A Krycka      27-Aug-1978
0000  ;             Revise NETWORK ERROR processing
0000  ;
0000  ; REVISION HISTORY:
0000  ;
0000  ;     L F Laverdure,      10-Oct-1978
0000  ;     X0003 - Deletion of call to CHK_IDLE
```

```
                      0000   172 ;
                      0000   173 ;       L F Laverdure,      11-Aug-1978
                      0000   174 ;       X0002 -  Return real BLK RAT bit for MT
                      0000   175 ;
                      0000   176 ;--
                      0000   177 ;
                      0000   178
```

```
                 0000         .SBTTL  DECLARATIONS
                 0000
                 0000   ;
                 0000   ; INCLUDE FILES:
                 0000   ;
                 0000
                 0000   ;
                 0000   ; MACROS:
                 0000   ;
                 0000
                 0000           $IFBDEF
                 0000           $IRBDEF
                 0000           $FABDEF
                 0000           $RABDEF
                 0000           $RMSDEF
                 0000           $ATRDEF
                 0000           $PSLDEF
                 0000           $IODEF
                 0000           $DEVDEF
                 0000           $FIBDEF
                 0000           $XABDEF
                 0000           $XABALLDEF
                 0000           $XABDATDEF
                 0000           $XABFHCDEF
                 0000           $XABPRODEF
                 0000           $XABRDTDEF
                 0000           $XABJNLDEF
                 0000           $CSHDEF
                 0000           $RLSDEF
                 0000           $FWADEF
                 0000
                 0000   ;
                 0000   ; EQUATED SYMBOLS:
                 0000   ;
                 0000
      00000010   0000           C_MAXATTR=16                    ; max. # attribute list entries / QIO
                 0000
                 0000   ;
                 0000   ; OWN STORAGE:
                 0000   ;
                 0000   ;
                 0000   ;  Argument list for XAB chain processing (allocation XABS only)
                 0000   ;
                 0000
                 0000   ALL_XAB_ARGS:
      00'20 14   0000           .BYTE   XAB$C_ALL,XAB$C_ALLLEN,XBC$C_DSPALL
      00'24 12   0003           .BYTE   XAB$C_DAT,XAB$C_DATLEN_V2,XBC$C_OPNDAT  ; (temporarily here)
            00   0006           .BYTE   0
                 0007
                 0007   ;
                 0007   ;  XAB scan args for other XABS requiring a READ attributes
                 0007   ;
                 0007
                 0007   DSP_XAB_ARGS:
      00'2C 1D   0007           .BYTE   XAB$C_FHC,XAB$C_FHCLEN,XBC$C_DSPFHC
      00'10 13   000A           .BYTE   XAB$C_PRO,XAB$C_PROLEN_V3,XBC$C_OPNPRO
      00'14 1E   000D           .BYTE   XAB$C_RDT,XAB$C_RDTLEN,XBC$C_OPNRDT
```

```
           00  0010   237          .BYTE   0
               0011   238
               0011   239 ;
               0011   240 ;  XAB scan args for XABS requiring modification after a READ attributes
               0011   241 ;
               0011   242
               0011   243 DSP_XAB_ARGS1:
    00'2C  1D  0011   244          .BYTE   XAB$C_FHC,XAB$C_FHCLEN,XBC$C_DSPFHC1
    00'20  14  0014   245          .BYTE   XAB$C_ALL,XAB$C_ALLLEN,XBC$C_DSPALL1
    00'3C  22  0017   246          .BYTE   XAB$C_JNL,XAB$C_JNLLEN,XBC$C_OPNJNL
    00'10  13  001A   247          .BYTE   XAB$C_PRO,XAB$C_PROLEN_V3,XBC$C_OPNPRO1
           00  001D   248          .BYTE   0
```

```
001E           .SBTTL  RMS$DISPLAY - $DISPLAY ROUTINE
001E
001E    ;++
001E    ; RMS$DISPLAY - Highest level $DISPLAY processing
001E    ;
001E    ;  This routine performs the highest level $DISPLAY processing.
001E    ;  its functions include:
001E    ;
001E    ;       1. Determine whether FAB or RAB display, and do common setup
001E    ;       2. Check for all streams idle, exiting with error if not
001E    ;       3. If this is a RAB display, just fill in RAB information available
001E    ;          from the IRAB and exit.
001E    ;       4. Allocate an attribute list work area and process the XAB chain
001E    ;       5. Do a READ of the file attributes
001E    ;       6. Reprocess the XAB chain for any XABS requiring modification
001E    ;          after the READ attributes.
001E    ;       7. Deallocate the attribute list work area
001E    ;       8. Fill in FAB fields available from the IFAB.
001E    ;       9. Check for a NAM block.  If present, fill in with information from FWA
001E    ;       10. Exit to the user, generating an AST if requested
001E    ;
001E    ;
001E    ; CALLING SEQUENCE:
001E    ;
001E    ;       Entered from EXEC as a result of user's calling SYS$DISPLAY
001E    ;       (e.g., by using the $DISPLAY macro).
001E    ;
001E    ; INPUT PARAMETERS:
001E    ;
001E    ;       AP      User's argument list addr
001E    ;
001E    ; IMPLICIT INPUTS:
001E    ;
001E    ;       The contents of the FAB or RAB and RELATED XABS.
001E    ;
001E    ; OUTPUT PARAMETERS:
001E    ;
001E    ;       R0      STATUS code
001E    ;       R1      destroyed
001E    ;
001E    ; IMPLICIT OUTPUTS:
001E    ;
001E    ;       The various fields of the RELATED XABS are filled in.
001E    ;
001E    ;       A completion AST is queued if so specified by the user.
001E    ;
001E    ; COMPLETION CODES:
001E    ;
001E    ;       Standard RMS (see functional spec for list).
001E    ;
001E    ; SIDE EFFECTS:
001E    ;
001E    ;       None.
001E    ;
001E    ;--
001E
```

```
                        001E
                        001E   ;++
                        001E   ;
                        001E   ;  Entry point for $DISPLAY service
                        001E   ;
                        001E   ;--
                        001E
                        001E          $ENTRY  RMS$DISPLAY
                        001E          $TSTPT  DISPLAY
                        0024
                        0024   ;
                        0024   ;  Decide if this is a FAB or RAB display.
                        0024   ;
                        0024
      58   04 AC    D0  0024          MOVL    4(AP),R8                        ; Get FAB or RAB address
                        0028          ASSUME  FAB$B_BID EQ RAB$B_BID
                        0028          IFNORD  #<FAB$B_BID+1>,(R8),1$           ; Structure not accessible.
         68   03    91  002E          CMPB    #FAB$C_BID,FAB$B_BID(R8)        ; Is this a display for RAB?
              15    12  0031          BNEQ    5$                              ; Yes, RAB display
                        0033   ;++
                        0033   ;
                        0033   ;  This is a FAB display.  Do common setup for FAB.  Check for Network
                        0033   ;  operation and go do just that.
                        0033   ;
                        0033   ;--
           FFCA'    30  0033          BSBW    RM$FSET                         ; do common setup
                        0036                                                  ;  NOTE: does not return on
                        0036                                                  ;  error
      12 69   3E    E1  0036          BBC     #IFB$V_DAP,(R9),10$             ; Check for network operation
           FFC3'    30  003A          BSBW    NT$DISPLAY                      ; Get file attributes from
           FFC0'    31  003D          BRW     RM$EXRMS                        ;  remote system and exit RMS
                        0040
                        0040   ;
                        0040   ;  Error out if cannot read first word of the inputted structure (FAB or RAB)
                        0040   ;
   50   000187BC 8F  D0  0040   1$:    MOVL    #RMS$_STR,R0                    ; Return structure error to user
                    04  0047          RET                                     ; and exit RMS
                        0048   ;++
                        0048   ;
                        0048   ;  This is a RAB display.  Do common setup for RAB.
                        0048   ;
                        0048   ;--
                        0048   5$:    $RABSET                                 ; common setup
                        004C                                                  ;  NOTE: does not return on error
                        004C   ;
                        004C   ;  Supply "Wild ISI" information.  Put it into STV of user input structure.
                        004C   ;
                        004C   ;  Pick up, in the case of a FAB input, the ISI of the first stream which
                        004C   ;  is connected to the file.  In the case of a RAB input, the ISI of the next
                        004C   ;  stream which is connected to the file.  If such an ISI exists, in both cases.
                        004C   ;  Because the top portion of both the IFAB and IRAB are similar, the same code
                        004C   ;  can be used.
                        004C   ;
                        004C   ;  One is to assume that the user has called $DISPLAY with an IFI and is now
                        004C   ;  interested in the streams connected to the file.  Returning the first ISI
                        004C   ;  with the FAB $DISPLAY, the user will call $DISPLAY with each of the values
                        004C   ;  returned in the STV until zero.
```

```
                   004C      ;
                   004C                ASSUME  IFB$L_IRAB_LNK  EQ IRB$L_IRAB_LNK
                   004C                ASSUME  IFB$W_IFI       EQ IRB$W_ISI
                   004C                ASSUME  FAB$L_STV EQ RAB$L_STV
                   004C      10$:
50   1C A9   D0    004C                MOVL    IFB$L_IRAB_LNK(R9),R0           ; Pick up first IRAB address
        04   13    0050                BEQL    20$                             ; No streams connected
50   28 A0   3C    0052                MOVZWL  IRB$W_ISI(R0),R0                ; save this IRAB's ISI
                   0056      20$:
0C A8   50   D0    0056                MOVL    R0,FAB$L_STV(R8)                ; return this ISI (or 0) in STV
                   005A                RMSSUC                                  ; anticipate SUCCESS
                   005D      ;
                   005D      ;  If this is a RAB input, then go to fill in the RAB information and
                   005D      ;  handle any RAB XAB's attached.
                   005D      ;
                   005D                ASSUME  <IFB$C_BID&1> EQ 1              ; Is this a FAB or RAB display
                   005D                ASSUME  <IRB$C_BID&1> EQ 0
                   005D                ASSUME  IFB$B_BID EQ IRB$B_BID
                   005D
  03 08 A9   E8    005D                BLBS    IFB$B_BID(R9),25$
      012A   31    0061                BRW     DSPRAB                          ; display RAB stuff
                   0064
        03   E0    0064      25$:      BBS     #DEV$V_DIR,-                    ; err not file structured
     03 69         0066                        IFB$L_PRIM_DEV(R9),30$
      00BA   31    0068                BRW     DSPFAB                          ; Display FAB and exit.
23 A9   02   91    006B      30$:      CMPB    #IFB$C_IDX,IFB$B_ORGCASE(R9)    ; is this ISAM file?
        09   12    006F                BNEQ    DSPXAB                          ; no, skip ISAM XABS
                   0071
      014F   30    0071                BSBW    ISAM_XABS                       ; process ISAM XABS
     03 50   E8    0074                BLBS    R0,DSPXAB
      0111   31    0077      D_XIT:    BRW     DSPXIT                          ; exit on error
                   007A
```

```
                        007A          .SBTTL  DSPXAB - Handle general, non-ISAM  XAB attributes
                        007A
                        007A     ;
                        007A     ;  Allocate FIB work area
                        007A     ;
                        007A
     52   40 8F    9A   007A     DSPXAB: MOVZBL  #FIB$C_LENGTH,R2                 ; size for work area
            FF7F'  30   007E             BSBW    RM$GETSPC1                       ; allocate work area
           F3 50   E9   0081             BLBC    R0,D_XIT                         ; get out on errors
         56   51   D0   0084             MOVL    R1,R6                            ; set FIB addr
     5A   38 A9    D0   0087             MOVL    IFB$L_FWA_PTR(R9),R10            ; Set up FWA pointer
              0E   BB   008B             PUSHR   #^M<R1,R2,R3>             ; Save regs
            FF70'  30   008D             BSBW    RM$GET1PAG                ; Get scratch page
     58 AA    53   D0   0090             MOVL    R3,FWA$L_ATR_WORK(R10)    ;  for ATR work area
         55   53   D0   0094             MOVL    R3,R5                     ; Put address in R5
   01FC C3    01   D0   0097             MOVL    #PSL$C_EXEC,508(R3)       ; Keep exec mode byte in last lword
              0E   BA   009C             POPR    #^M<R1,R2,R3>             ; Restore regs
                        009E
                        009E     ;
                        009E     ;  Process ALLOCATION XAB, if any.
                        009E     ;
                        009E
     03 69    38   E1   009E             BBC     #IFB$V_SEQFIL,(R9),7$            ; really SEQUENTIAL FILE
                        00A2                                                      ;  masquerading as RELATIVE?
                        00A2             ASSUME  <IFB$C_SEQ + 1> EQ IFB$C_REL
                        00A2
         23 A9    97    00A2             DECB    IFB$B_ORGCASE(R9)                ; don't confuse XAB scan
   5C    FF57 CF  9E    00A5     7$:     MOVAB   ALL_XAB_ARGS,AP                  ; XAB scan arg. list addr
                        00AA     ;
                        00AA     ; set access mode to user before calling XAB_SCAN
                        00AA     ;
         85   01   B0   00AA             MOVW    #1,(R5)+                         ; 1 byte length
         85   2D   B0   00AD             MOVW    #ATR$C_ACCESS_MODE,(R5)+         ; access mode
     85   0A A9    9E   00B0             MOVAB   IFB$B_MODE(R9),(R5)+             ; user mode
                        00B4
            FF49'  30   00B4             BSBW    RM$XAB_SCAN                      ; process any ALLOCATION XAB
           47 50   E9   00B7             BLBC    R0,DSPCLN                        ; get out on error
                        00BA
                        00BA     ;
                        00BA     ;  Do a read of the file attributes.  RM$XAB_SCAN has set up the attribute
                        00BA     ;  list, if a Allocation XAB was found.
                        00BA     ;
              54   D5   00BA             TSTL    R4                               ; any XAB processed?
              0B   13   00BC             BEQL    9$                               ; if not, R5 still points
                        00BE                                                      ;  to the right place - go on
            0116   30   00BE             BSBW    READ_ATTR                        ; go read attributes
           3D 50   E9   00C1             BLBC    R0,DSPCLN                        ; get out on error
                        00C4
                        00C4     ; Reset R5 pointer.  Since there is already a user-mode ATR there, use it
                        00C4     ;
   55    08   58 AA    C1   00C4         ADDL3   FWA$L_ATR_WORK(R10),#8,R5        ; set addr of work area
                        00C9
                        00C9     ;
                        00C9     ;  Process other XABS, if any
                        00C9     ;
                        00C9     9$:
   5C    FF3A CF  9E    00C9             MOVAB   DSP_XAB_ARGS,AP                  ; XAB scan arg. list addr
```

```
                 FF2F'  30  00CE          BSBW    RMS$XAB_SCAN                      ; process the XABS
                 2D 50  E9  00D1          BLBC    R0,DSPCLN                         ; get out on error
          57   00A0 C9  9A  00D4          MOVZBL  IFB$B_JNLFLG(R9),R7               ; save current journal flags
                            00D9   ;
                            00D9   ; put an exec-mode ATR here for picking up jounaling stuff
                            00D9   ;
               85   01  B0  00D9          MOVW    #1,(R5)+                          ; 1 byte length
               85   2D  B0  00DC          MOVW    #ATR$C_ACCESS_MODE,(R5)+          ; access mode
 85   58 AA   000001FC 8F  C1  00DF          ADDL3   #508,FWA$L_ATR_WORK(R10),(R5)+    ; Byte that specifies exec mode
                            00E8
          00000000'EF   16  00E8          JSB     RMS$RTVJNL                        ; set up attributes for journaling
                 00E6   30  00EE          BSBW    READ_ATTR                         ; read file attributes
                 0D 50  E9  00F1          BLBC    R0,DSPCLN                         ; get out on error
          5C   FF19 CF  9E  00F4          MOVAB   DSP_XAB_ARGS1,AP                  ; FHC & ALLOCATION XAB scan args
                 FF04'  30  00F9          BSBW    RMS$XAB_SCAN                      ; process 2nd half of all or
                            00FC                                                    ;  FHC XAB
          00A0 C9   57  90  00FC          MOVB    R7,IFB$B_JNLFLG(R9)               ; restore journal flags
                            0101
                            0101   ;
                            0101   ;  Deallocate FIB and ATR work area
                            0101   ;
                            0101
                    3F  BB  0101   DSPCLN: PUSHR   #^M<R0,R1,R2,R3,R4,R5>           ; Save regs
            54   58 AA  D0  0103          MOVL    FWA$L_ATR_WORK(R10),R4            ; Point to work page
                 FEF6'  30  0107          BSBW    RMS$RET1PAG                       ;  and deallocate it
                 58 AA  D4  010A          CLRL    FWA$L_ATR_WORK(R10)               ; Indicate no work area now
                    3F  BA  010D          POPR    #^M<R0,R1,R2,R3,R4,R5>            ; Restore regs
            52   40 8F  9A  010F          MOVZBL  #FIB$C_LENGTH,R2                  ; size of FIB
               54   56  D0  0113          MOVL    R6,R4                             ; right register to return
                    50  DD  0116          PUSHL   R0                                ; save status
                 FEE5'  30  0118          BSBW    RMS$RETSPC1                       ; deallocate work space
                  50 8ED0  011B          POPL    R0                                ; restore status
            03 69   38  E1  011E          BBC     #IFB$V_SEQFIL,(R9),DSPFAB         ; skip if not SEQ file SHARED
                 23 A9  96  0122          INCB    IFB$B_ORGCASE(R9)                 ; back to RELATIVE disguise
```

```
                            0125              .SBTTL  DSPFAB - Handle FAB attributes
                            0125
                            0125      ;
                            0125      ;  Display (fill in) FAB associated fields which are available in the IFAB
                            0125      ;
                            0125
             63 50    E9    0125      DSPFAB: BLBC    R0,DSPXIT                       ; get out on error
                            0128
       5A    38 A9    D0    0128              MOVL    IFB$L_FWA_PTR(R9),R10           ; Set up FWA pointer
                            012C      ;
                            012C      ;        Note still need to handle shared file case in copying the all. quant.
                            012C      ;
    10 A8    70 A9    D0    012C              MOVL    IFB$L_HBK(R9),FAB$L_ALQ(R8)     ; Copy in allocation quantity
                            0131
             FECC'    30    0131              BSBW    RM$RET_DEV_CHAR                 ; Copy in Device Characteristics
                            0134
                            0134      ;
                            0134      ;       Fill in all the misc FAB fields in alphabetical order
                            0134      ;
                            0134
    3E A8    5E A9    90    0134              MOVB    IFB$B_BKS(R9),FAB$B_BKS(R8)     ; Copy in Bucket Size
 3C A8    0094 C9     B0    0139              MOVW    IFB$L_ASDEVBSIZ(R9),FAB$W_BLS(R8)       ; Copy in Block-size
    14 A8    4C A9    B0    013F              MOVW    IFB$W_RTDEQ(R9),FAB$W_DEQ(R8)   ; Copy in Default Extend Quant.
    16 A8    22 A9    90    0144              MOVB    IFB$B_FAC(R9),FAB$B_FAC(R8)     ; Copy in File access
    3F A8    5F A9    90    0149              MOVB    IFB$B_FSZ(R9),FAB$B_FSZ(R8)     ; Copy in Record header sz for VFC
    48 A8    64 A9    B0    014E              MOVW    IFB$W_GBC(R9),FAB$W_GBC(R8)     ; Copy in Global Buffer Count
 38 A8    00AC C9     D0    0153              MOVL    IFB$L_MRN(R9),FAB$L_MRN(R8)     ; Copy in Max record Number
    36 A8    60 A9    B0    0159              MOVW    IFB$W_MRS(R9),FAB$W_MRS(R8)     ; Copy in Maximum Record Size
       50    23 A9    9A    015E              MOVZBL  IFB$B_ORGCASE(R9),R0            ; Pick up file organization
    02 69       38    E1    0162              BBC     #IFB$V_SEQFIL,(R9),10$          ; skip if not SEQ file SHARED
                50    D7    0166              DECL    R0                              ; make it really SEQ
     04     04  50    F0    0168      10$:    INSV    R0,#FAB$V_ORG,#FAB$S_ORG,-
             1D A8          016C                      FAB$B_ORG(R8)                   ;  and shift over for FAB
    1E A8    51 A9    90    016E              MOVB    IFB$B_RAT(R9),FAB$B_RAT(R8)     ; Copy in Record attributes
    1F A8    50 A9    90    0173              MOVB    IFB$B_RFMORG(R9),FAB$B_RFM(R8)  ; Copy in Record Format
       50    14 BA    9E    0178              MOVAB   @FWA$Q_FIB+4(R10),R0            ; Get address of FIB from FWA
                05    13    017C              BEQL    20$                             ; No FIB => no windows
    1C A8    03 A0    90    017E              MOVB    FIB$B_WSIZE(R0),FAB$B_RTV(R8)   ; Copy in Retrieval Window
    17 A8    4E A9    90    0183      20$:    MOVB    IFB$B_SHR(R9),FAB$B_SHR(R8)     ; Copy in Sharing bits
                            0188
                            0188      ;
                            0188      ;  Check for a NAM block and if present, fill in any information from the FWA
                            0188      ;  Fill in the NAM block and resultant name string and return any errors found.
                            0188      ;
                            0188
             FE75'    30    0188              BSBW    RM$FILLNAM                      ; Fill in NAM block
                            018B
             FE72'    31    018B      DSPXIT: BRW     RM$EXRMS                        ; exit RMS
                            018E
```

```
                              018E   539           .SBTTL  DSPRAB - Handle RAB attributes
                              018E   540 ;
                              018E   541 ;  DSPRAB - Display the RAB information into the inputted RAB from the IRAB
                              018E   542 ;
                              018E   543 DSPRAB:
        37 A8     55 A9   90  018E   544           MOVB    IRB$B_MBC(R9),RAB$B_MBC(R8)     ; Copy in Multi-block count
        36 A8     5C A9   90  0193   545           MOVB    IRB$B_MBF(R9),RAB$B_MBF(R8)     ; Copy in Multi-buffer count
                              0198   546           CASE    TYPE=B,SRC=IFB$B_ORGCASE(R10),-
                              0198   547                   DISPLIST=<SEQ,REL,ISAM>         ; Case on File organization
                              01A3   548 ;
                              01A3   549 SEQ:
                              01A3   550 REL:
        10 A8     40 A9   D0  01A3   551           MOVL    IRB$L_NRP_VBN(R9),RAB$L_RFA0(R8)        ; Copy RFA
        14 A8     44 A9   B0  01A8   552           MOVW    IRB$W_NRP_OFF(R9),RAB$W_RFA4(R8)        ;
                     DC   11  01AD   553           BRB     DSPXIT                                  ; Exit from RAB display
                              01AF   554 ;
                              01AF   555 ISAM:
       10 A8    00B0 C9   D0  01AF   556           MOVL    IRB$L_UDR_VBN(R9),RAB$L_RFA0(R8)        ; Copy RFA
       14 A8    00BC C9   B0  01B5   557           MOVW    IRB$W_UDR_ID(R9),RAB$W_RFA4(R8)         ;
       35 A8    00C3 C9   90  01BB   558           MOVB    IRB$B_CUR_KREF(R9),RAB$B_KRF(R8)        ; Copy Key of Reference
                              01C1   559
                     C8   11  01C1   560           BRB     DSPXIT                                  ; exit RMS
```

```
                                     01C3   562          .SBTTL  ISAM_XABS - Handle Indexed file XAB attributes
                                     01C3   563 ;
                                     01C3   564 ;  This does SCAN for INDEXED file org SUMMARY, KEY, and AREA XAB'S
                                     01C3   565 ;  RM$ALLOC_BUF needs the IFB pointer in R10.
                                     01C3   566 ;
                                     01C3   567
                                     01C3   568 ISAM_XABS:
                    55   01    7D    01C3   569          MOVQ    #1, R5                              ; 1 block buffer, no lock blb.
                       FE37'   30    01C6   570          BSBW    RM$ALLOC_BUF                        ; Allocate buffer, desc.
                    0A 50      E9    01C9   571          BLBC    R0,50$                              ; out on allocation failure
                                     01CC   572                                                      ;  nothing wil, have been alloc
         5C   00000000'EF      9E    01CC   573 20$:     MOVAB   RM$XABOPN_ARGS,AP                   ; move addr. of XAB table in AP
                       FE2A'   30    01D3   574          BSBW    RM$XAB_SCAN                         ; scan XAB list
                               05    01D6   575 50$:     RSB                                         ; return - this deallocates the
                                     01D7   576                                                      ; buffer and desc allocated on
                                     01D7   577                                                      ; the call to RM$ALLOC_BUF.
```

```
                   01D7          .SBTTL  READ_ATTR - SUBROUTINE TO READ FILE ATTRIBUTES
                   01D7
                   01D7     ;++
                   01D7     ;  READ_ATTR - Read file attributes
                   01D7     ;
                   01D7     ;        This routine performs an IO$_ACCESS QIO to READ the file attributes
                   01D7     ;
                   01D7     ;  INPUTS:
                   01D7     ;
                   01D7     ;        R11     IMPURE AREA addr
                   01D7     ;        R10     FWA address
                   01D7     ;        R9      IFAB addr
                   01D7     ;        R8      FAB addr
                   01D7     ;        R6      FIB addr
                   01D7     ;        R5      ATTRIBUTE LIST END addr (a zero longword will be store here)
                   01D7     ;
                   01D7     ;  OUTPUTS:
                   01D7     ;
                   01D7     ;        R0              STATUS
                   01D7     ;        R1-R5,AP        Destroyed
                   01D7     ;
                   01D7     ;--
                   01D7
                   01D7     READ_ATTR:
         65    D4  01D7          CLRL    (R5)                             ; flag end of attr. list
         56    DD  01D9          PUSHL   R6                               ; build FIB descriptor (addr)
   7E    40 8F 9A  01DB          MOVZBL  #FIB$C_LENGTH,-(SP)              ;        ''            (len)
                   01DF          SSB     #FIB$V_PRSRV_ATR,-               ; specify real attributes (blk)
                   01DF                  FIB$L_ACCTL(R6)
                   01E3
                   01E3     ;
                   01E3     ;  Push P6 and P5 QIO parameters on the STACK and do the ACCESS QIO function
                   01E3     ;
                   01E3
         00    DD  01E3          PUSHL   #0                               ; P6 = 0
      58 AA    DD  01E5          PUSHL   FWA$L_ATR_WORK(R10)              ; P5 = attr. list addr
    50   32    9A  01E8          MOVZBL  #IO$_ACCESS,R0                   ; I/O function code
      FE12'    30  01EB          BSBW    RMS$FCPFNC_P4                    ; read attributes
      08 50    E8  01EE          BLBS    R0,10$                           ; Did everything go ok?
                   01F1          RMSERR  ACC,R1                           ; No, file system
      FE07'    30  01F6          BSBW    RMS$MAPERR                       ;  found something wrong
    5E   08    C0  01F9     10$: ADDL2   #8,SP                            ; dump FIB size and address
         05        01FC          RSB
                   01FD
                   01FD          .END
```

```
$$.PSECT_EP                 = 00000000                 IFB$L_FWA_PTR               = 00000038
$$RMSTEST                   = 0000001A                 IFB$L_HBK                   = 00000070
$$RMS_PBUGCHK               = 00000010                 IFB$L_IRAB_LNK              = 0000001C
$$RMS_TBUGCHK               = 00000008                 IFB$L_MRN                   = 000000AC
$$RMS_UMODE                 = 00000004                 IFB$L_PRIM_DEV              = 00000000
ALL_XAB_ARGS                  00000000 R      01       IFB$V_DAP                   = 0000003E
ATR$C_ACCESS_MODE           = 0000002D                 IFB$V_SEQFIL                = 00000038
C_MAXATTR                   = 00000010                 IFB$W_GBC                   = 00000064
DEV$V_DIR                   = 00000003                 IFB$W_IFI                   = 00000028
DSPCLR                        00000101 R      01       IFB$W_MRS                   = 00000060
DSPFAB                        00000125 R      01       IFB$W_RTDEQ                 = 0000004C
DSPRAB                        0000018E R      01       IO$_ACCESS                  = 00000032
DSPXAB                        0000007A R      01       IRB$B_BID                   = 00000008
DSPXIT                        0000018B R      01       IRB$B_CUR_KREF              = 000000C3
DSP_XAB_ARGS                  00000007 R      01       IRB$B_MBC                   = 00000055
DSP_XAB_ARGS1                 00000011 R      01       IRB$B_MBF                   = 0000005C
D_XIT                         00000077 R      01       IRB$C_BID                   = 0000000A
FAB$B_BID                   = 00000000                 IRB$L_IRAB_LNK              = 0000001C
FAB$B_BKS                   = 0000003E                 IRB$L_NRP_VBN               = 00000040
FAB$B_FAC                   = 00000016                 IRB$L_UDR_VBN               = 000000B0
FAB$B_FSZ                   = 0000003F                 IRB$W_ISI                   = 00000028
FAB$B_ORG                   = 0000001D                 IRB$W_NRP_OFF               = 00000044
FAB$B_RAT                   = 0000001E                 IRB$W_UDR_ID                = 000000BC
FAB$B_RFM                   = 0000001F                 ISAM                          000001AF R      01
FAB$B_RTV                   = 0000001C                 ISAM_XABS                     000001C3 R      01
FAB$B_SHR                   = 00000017                 NT$DISPLAY                    ********    X   01
FAB$C_BID                   = 00000003                 PIO$A_TRACE                   ********    X   01
FAB$L_ALQ                   = 00000010                 PSL$C_EXEC                  = 00000001
FAB$L_MRN                   = 00000038                 RAB$B_BID                   = 00000000
FAB$L_STV                   = 0000000C                 RAB$B_KRF                   = 00000035
FAB$S_ORG                   = 00000004                 RAB$B_MBC                   = 00000037
FAB$V_ORG                   = 00000004                 RAB$B_MBF                   = 00000036
FAB$W_BLS                   = 0000003C                 RAB$L_RFA0                  = 00000010
FAB$W_DEQ                   = 00000014                 RAB$L_STV                   = 0000000C
FAB$W_GBC                   = 00000048                 RAB$W_RFA4                  = 00000014
FAB$W_MRS                   = 00000036                 READ_ATTR                     000001D7 R      01
FIB$B_WSIZE                 = 00000003                 REL                           000001A3 R      01
FIB$C_LENGTH                = 00000040                 RM$ALLOC_BUF                  ********    X   01
FIB$L_ACCTL                 = 00000000                 RM$EXRMS                      ********    X   01
FIB$V_PRSRV_ATR             = 00000011                 RM$FCPFNC_P4                  ********    X   01
FWA$L_ATR_WORK              = 00000058                 RM$FILLNAM                    ********    X   01
FWA$Q_FIB                   = 00000010                 RM$FSET                       ********    X   01
IFB$B_BID                   = 00000008                 RM$GET1PAG                    ********    X   01
IFB$B_BKS                   = 0000005E                 RM$GETSPC1                    ********    X   01
IFB$B_FAC                   = 00000022                 RM$MAPERR                     ********    X   01
IFB$B_FSZ                   = 0000005F                 RM$RET1PAG                    ********    X   01
IFB$B_JNLFLG                = 000000A0                 RM$RETSPC1                    ********    X   01
IFB$B_MODE                  = 0000000A                 RM$RET_DEV_CHAR               ********    X   01
IFB$B_ORGCASE               = 00000023                 RM$RSET                       ********    X   01
IFB$B_RAT                   = 00000051                 RM$RTVJNL                     ********    X   01
IFB$B_RFMORG                = 00000050                 RM$XABOPN_ARGS                ********    X   01
IFB$B_SHR                   = 0000004E                 RM$XAB_SCAN                   ********    X   01
IFB$C_BID                   = 0000000B                 RMS$DISPLAY                 = 0000001C RG     01
IFB$C_IDX                   = 00000002                 RMS$_ACC                    = 0001C002
IFB$C_REL                   = 00000001                 RMS$_STR                    = 000187BC
IFB$C_SEQ                   = 00000000                 SEQ                           000001A3 R      01
IFB$L_ASDEVBSIZ             = 00000094                 TPT$L_DISPLAY                 ********    X   01
```

```
                                                                   N 11
RMSODISPL                      DISPATCH FOR DISPLAY OPERATION              16-SEP-1984 01:15:10  VAX/VMS Macro V04-00         Page  17
Symbol table                                                                5-SEP-1984 16:24:48  [RMS.SRC]RMSODISPL.MAR;1          (10)

XAB$C_ALL                      = 00000014
XAB$C_ALLLEN                   = 00000020
XAB$C_DAT                      = 00000012
XAB$C_DATLEN_V2                = 00000024
XAB$C_FHC                      = 0000001D
XAB$C_FHCLEN                   = 0000002C
XAB$C_JNL                      = 00000022
XAB$C_JNLLEN                   = 0000003C
XAB$C_PRO                      = 00000013
XAB$C_PROLEN_V3                = 00000010
XAB$C_RDT                      = 0000001E
XAB$C_RDTLEN                   = 00000014
XBC$C_DSPALL                     ********    X    01
XBC$C_DSPALL1                    ********    X    01
XBC$C_DSPFHC                     ********    X    01
XBC$C_DSPFHC1                    ********    X    01
XBC$C_OPNDAT                     ********    X    01
XBC$C_OPNJNL                     ********    X    01
XBC$C_OPNPRO                     ********    X    01
XBC$C_OPNPRO1                    ********    X    01
XBC$C_OPNRDT                     ********    X    01

                                                 +----------------+
                                                 ! Psect synopsis !
                                                 +----------------+

PSECT name                     Allocation          PSECT No.  Attributes
----------                     ----------          ---------  ----------
.  ABS  .                      00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
RM$RMS                         000001FD  (  509.)  01 (  1.)    PIC   USR   CON   REL   GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                          00000000  (    0.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE

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

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   29     00:00:00.08    00:00:00.58
Command processing              112     00:00:00.76    00:00:04.84
Pass 1                          504     00:00:19.86    00:00:42.40
Symbol table sort                 0     00:00:03.13    00:00:03.59
Pass 2                          123     00:00:03.71    00:00:07.43
Symbol table output              17     00:00:00.18    00:00:00.35
Psect synopsis output             1     00:00:00.01    00:00:00.15
Cross-reference output            0     00:00:00.00    00:00:00.00
Assembler run totals            788     00:00:27.73    00:00:59.34

The working set limit was 1650 pages.
112391 bytes (220 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2200 non-local and 16 local symbols.
623 source lines were read in Pass 1, producing 15 object records in Pass 2.
39 pages of virtual memory were used to define 38 macros.
```

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

Macro library name                             Macros defined
------------------                             --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                       23
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        2
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     9
TOTALS (all libraries)                               34

2370 GETS were required to define 34 macros.

There were no errors, warnings or information messages.

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

0329 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RMS0FSCN
LIS

RM3XALLO
LIS

RMS0CONN
LIS

RMS0ENTER
LIS

RMS0ERASE
LIS

RMS0CREAT
LIS

RMS0DELET
LIS

RMS0FIND
LIS

RM3XSUMO
LIS

RMS0DISPL
LIS

RM3XKEYO
LIS

RMS0GET
LIS

RMS0CLOSE
LIS

RMS0DISC
LIS

RMS0EXTEN
LIS

RMS0BLKIO
LIS