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
```

```
                                                  H 16
**FILE**ID**RM2CONN
```

```
RRRRRRRR    MM      MM    222222      CCCCCCCC    000000    NN      NN  NN      NN
RRRRRRRR    MM      MM    222222      CCCCCCCC    000000    NN      NN  NN      NN
RR      RR  MMMM  MMMM  22      22  CC          00      00  NN      NN  NN      NN
RR      RR  MMMM  MMMM  22      22  CC          00      00  NN      NN  NN      NN
RR      RR  MM  MM  MM          22  CC          00      00  NNNN    NN  NNNN    NN
RR      RR  MM  MM  MM          22  CC          00      00  NNNN    NN  NNNN    NN
RRRRRRRR    MM      MM        22    CC          00      00  NN  NN  NN  NN  NN  NN
RRRRRRRR    MM      MM        22    CC          00      00  NN  NN  NN  NN  NN  NN
RR  RR      MM      MM      22      CC          00      00  NN    NNNN  NN    NNNN
RR  RR      MM      MM      22      CC          00      00  NN    NNNN  NN    NNNN
RR    RR    MM      MM    22        CC          00      00  NN      NN  NN      NN       ....
RR    RR    MM      MM    22        CC          00      00  NN      NN  NN      NN       ....
RR      RR  MM      MM  2222222222    CCCCCCCC    000000    NN      NN  NN      NN       ....
RR      RR  MM      MM  2222222222    CCCCCCCC    000000    NN      NN  NN      NN       ....

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          $BEGIN  RM2CONN,000,RM$RMS2,<RELATIVE-SPECIFIC CONNECT>
0000
0000    ;
```

```
0000    ;
0000    ;++
0000    ; Facility: rms32
0000    ;
0000    ; Abstract:
0000    ;       routine to perform relative file organization specific
0000    ;       connect stream processing.
0000    ;
0000    ; Environment:
0000    ;               star processor running starlet exec.
0000    ;
0000    ; Author: L F Laverdure,        creation date: 19-OCT-1977
0000    ;
0000    ; Modified By:
0000    ;
0000    ;       V03-007 RAS0125         Ron Schaefer            28-Feb-1983
0000    ;               Fix bad psect introduced by RAS0119.
0000    ;
0000    ;       V03-006 KPL0002         Peter Lieberwirth       21-Jan-1983
0000    ;               Fix many broken branches.
0000    ;
0000    ;       V03-005 RAS0119         Ron Schaefer            19-Jan-1983
0000    ;               Correct RAS0092 to compute the correct record number for
0000    ;               the NRP; otherwise you get holes in the record numbers.
0000    ;
0000    ;       V03-004 KBT0131         Keith B. Thompson       20-Aug-1982
0000    ;               Reorganize psects
0000    ;
0000    ;       V03-003 KBT0122         Keith B. Thompson       6-Aug-1982
0000    ;               Remove ref. to set_sidb_adr
0000    ;
0000    ;       V03-002 RAS0092         Ron Schaefer            27-Jul-1982
```

```
0000    ;               Correct connect to EOF logic to deal correctly with
0000    ;               sparse files.
0000    ;
0000    ;       V03-001 JWH0003         Jeffrey W. Horn         16-Mar-1982
0000    ;               Prevent connect to eof and block io to be used together
0000    ;               with relative files.
0000    ;
0000    ;       V02-016 JWH0002         Jeffrey W. Horn         19-Feb-1982
0000    ;               Fix problem with connect to eof option for shared sequential
0000    ;               files.
0000    ;
0000    ;       V02-015 JWH0001         Jeffrey W. Horn         26-Oct-1981
0000    ;               Added connect to eof option for relative files.  Search
0000    ;               backwards from EBK untill a record is found, set NRP to
0000    ;               that record plus 1.
0000    ;
0000    ;       V02-014 KPL0001         Peter Lieberwirth       24-Jul-1981
0000    ;               Fix broken branch.
0000    ;
0000    ;       V02-013 REFORMAT        C D Saether     30-Jul-1980     23:07
0000    ;
0000    ;       V012    CDS0044         C D Saether     22-Oct-1979     15:37
0000    ;               allow for connect to eof option for shared seq file
0000    ;               use seq mbf when shared seq file, blk io bdb handled
0000    ;               by rm$bdballoc now
0000    ;
0000    ;       V011    JAK0020         J A Krycka      11-Sep-1979     10:00
0000    ;               remove network code.
0000    ;
0000    ;       V010    CDS0012         C Saether       24-Jul-1979     14:35
0000    ;               remove references to ifb$w_bks_bytes and ifb$w_bks_recs.
0000    ;
0000    ;       V009    WSK0001         W Koenig        5-Feb-1979      12:50
0000    ;       on errors, branch to rm$ex_nirab_shr instead of rm$ex_nostr
0000    ;--
0000    ;
0000
```

```
                   0000     96         .SBTTL  DECLARATIONS
                   0000     97
                   0000     98 ;
                   0000     99 ; Include Files:
                   0000    100 ;
                   0000    101
                   0000    102 ;
                   0000    103 ; Macros:
                   0000    104 ;
                   0000    105
                   0000    106         $BKTDEF
                   0000    107         $DLCDEF
                   0000    108         $FABDEF
                   0000    109         $IFBDEF
                   0000    110         $IMPDEF
                   0000    111         $IRBDEF
                   0000    112         $RABDEF
                   0000    113         $RMSDEF
                   0000    114
                   0000    115 ;
                   0000    116 ; Equated Symbols:
                   0000    117 ;
                   0000    118
         00000020  0000    119         ROP=RAB$L_ROP*8                 ; bit offset to rop
                   0000    120
                   0000    121 ;
                   0000    122 ; Own Storage:
                   0000    123 ;
                   0000    124
```

```
0000          .SBTTL  RM$CONNECT2 - RELATIVE-SPECIFIC CONNECT ROUTINE
0000
0000    ;++
0000    ; RM$CONNECT2 - connect for relative file organization
0000    ; RM$CONNECT_BIO - just  ›t bdb for block i/o connect
0000    ;
0000    ; this module performs the following functions required for
0000    ; connecting to relative files.
0000    ;
0000    ;       1.  performs various validity checks
0000    ;       2.  if connect for block i/o allocate a bdb  and exit
0000    ;       3.  allocate bdb's and buffers:
0000    ;               1 minimum, otherwise system default
0000    ;       4.  perform various irab initializations
0000    ;
0000    ; Calling sequence:
0000    ;
0000    ;       entered via case branch from rm$connect
0000    ;
0000    ; Input Parameters:
0000    ;
0000    ;       r11     impure area address
0000    ;       r10     ifab address
0000    ;       r9      irab address
0000    ;       r8      rab address
0000    ;
0000    ; Implicit Inputs:
0000    ;
0000    ;       the contents of the rab and irab
0000    ;
0000    ; Output Parameters:
0000    ;
0000    ;       r0      status
0000    ;
0000    ; Implicit Outputs:
0000    ;
0000    ;       sets various fields in the irab and ifab.
0000    ;
0000    ; Completion Codes:
0000    ;
0000    ;       the standard rms status code is set into r0 and return
0000    ;       is made to user (not caller) via rm$exsuc (or rm$ex_nostr
0000    ;       after calling rm$ccln1 if an error is detected, thus
0000    ;       deallocating all irab - related internal structures).
0000    ;
0000    ; Side Effects:
0000    ;
0000    ;       none
0000    ;
0000    ;--
0000
```

```
                            0000  RM$CONNECT2::
                            0000          $TSTPT  CONNECT2
                            0006
                            0006  ;
                            0006  ;  if open or create was done with bro specified (mixed block & record i/o),
                            0006  ;  use the bio rop bit to determine whether to connect for block or record
                            0006  ;  operations.  note: any subsequent connects must be of the same type.
                            0006  ;
                            0006
   09 22 AA    06     E5    0006          BBCC    #FAB$V_BRO,IFB$B_FAC(R10),5$ ; branch if bro not specified
      0A 68    2B     E1    000B          BBC     #RAB$V_BIO+ROP,(R8),CHKRFM ; branch if bio not wanted
   44 22 AA    05     E3    000F          BBCS    #FAB$V_BIO,IFB$B_FAC(R10),RM$CONNECT_BIO ; set block i/o, branch
   3F 22 AA    05     E0    0014  5$:     BBS     #FAB$V_BIO,IFB$B_FAC(R10),RM$CONNECT_BIO ; branch if block io
                            0019
                            0019  ;
                            0019  ; record i/o.
                            0019  ;
                            0019
                            0019          ASSUME  FAB$C_UDF EQ 0
         50 AA    95        0019  CHKRFM: TSTB    IFB$B_RFMORG(R10)        ; r'm = undefined?
            03    12        001C          BNEQ    50$                      ; continue if no
          013C    31        001E          BRW     ERRRFM                   ; branch if yes
            54    D4        0021  50$:    CLRL    R4                       ; set default for rm$bdballoc
                            0023                                           ; to seq disk file
                            0023
                            0023  ;
                            0023  ; calculate cell size for records
                            0023  ;
                            0023
     03    50 AA    91      0023          CMPB    IFB$B_RFMORG(R10),#FAB$C_VFC ; vfc rec format?
              05    12      0027          BNEQ    60$                      ; branch if not
   62 A9   5F AA    90      0029          MOVB    IFB$B_FSZ(R10),IRB$W_CSIZ(R9) ; yes-initialize record size
   06 6A      38    E0      002E  60$:    BBS     #IFB$V_SEQFIL,(R10),65$ ; br if shared seq file
        62 A9       B6      0032          INCW    IRB$W_CSIZ(R9)           ; add in control byte
      54    03      D0      0035          MOVL    #3,R4                    ; index for relative defaults
     01    50 AA    91      0038  65$:    CMPB    IFB$B_RFMORG(R10),#FAB$C_FIX ; fixed rec. len?
              04    13      003C          BEQL    70$                      ; branch if yes
     62 A9    02    A0      003E          ADDW2   #2,IRB$W_CSIZ(R9)        ; add in 2-byte size field
                            0042  70$:
   62 A9   60 AA    A0      0042          ADDW2   IFB$W_MRS(R10),IRB$W_CSIZ(R9) ; & max record size
                            0047                                           ; giving total cell size
      55   5E AA    9A      0047          MOVZBL  IFB$B_BKS(R10),R5        ; get bkt size in blocks
    55   55   09    78      004B          ASHL    #9,R5,R5                 ; r5 gets bkt size in bytes
     55   62 A9     B1      004F          CMPW    IRB$W_CSIZ(R9),R5        ; is cell size <= bkt size?
              03    1B      0053          BLEQU   RM$CONNECT_BIO           ; LEQU means record fits
           00F6     31      0055          BRW     ERRIFA                   ; Otherwise means record don't fit
                            0058                                           ; so presume file header's been
                            0058                                           ; messed with
                            0058  RM$CONNECT_BIO::
      00000000'EF   16      0058          JSB     RM$BDBALLOC              ; allocate appropriate number
                            005E                                           ; of buffers using r4 to index
                            005E                                           ; to defaults, also allocates
                            005E                                           ; a lock bdb if write accessed
                            005E                                           ; allocates bdb without buffer
                            005E                                           ; for block i/o connect
                            005E
           03 50    E8      005E          BLBS    R0,SETNRP                ; continue if success
           00E4  31  0061          BRW     EXIT                       ; exit on error
     06 68   28  E0  0064  SETNRP: BBS     #RAB$V_EOF+ROP,(R8),CEOF ; branch if eof bit set in rop
         40 A9   D6  0068          INCL    IRB$L_NRP_VBN(R9)          ; set nrp to vbn1 (rp = 0)
          FF92'  31  006B          BRW     RM$EXSUC                   ; exit with success
                     006E
                     006E
                     006E
                     006E  ;++
                     006E  ;
                     006E  ; Connect to EOF Processing
                     006E  ;
                     006E  ;--
                     006E
     0D 6A   38  E1  006E  CEOF:   BBC     #IFB$V_SEQFIL,(R10),RELF ; br if rel file
                     0072
                     0072
                     0072  ;
                     0072  ;  SET NRP for Shared Sequential Files
                     0072  ;
                     0072
   40 A9   74 AA  D0  0072          MOVL    IFB$L_EBK(R10),IRB$L_NRP(R9)    ; set nrp from end block
             03  12  0077          BNEQ    1$                              ; if non-zero leave it.
         40 A9   D6  0079          INCL    IRB$L_NRP(R9)                   ; else set nrp to record 1
          FF81'  31  007C  1$:     BRW     RM$EXSUC                        ; exit with success
                     007F
                     007F
                     007F
                     007F
                     007F  ;
                     007F  ;  Search for last record if CONNECT to EOF for relative files
                     007F  ;
                     007F
     03 68   2B  E1  007F  RELF:   BBC     #RAB$V_BIO+ROP,(R8),5$          ; Branch if not block io,
           009E  31  0083          BRW     ERRROP                          ;  otherwise error.
      00000000'EF  16  0086  5$:     JSB     RM$LOCK_PROLOG                  ; Lock Prolog
          03 50  E8  008C          BLBS    R0,10$                          ; Continue on success
           00B6  31  008F          BRW     EXIT
        00D0 8F  BB  0092  10$:    PUSHR   #^M<R4,R6,R7>                   ; Save plg bdb adr & R6,R7
      7E   5E AA  9A  0096          MOVZBL  IFB$B_BKS(R10),-(SP)            ; Get Bucket Size
 56    74 AA   00B0 CA  C3  009A          SUBL3   IFB$L_DVBN(R10),IFB$L_EBK(R10),R6 ; Calculate VBN of block past
          56   6E  C6  00A1          DIVL    (SP),R6                         ;        last bucket
 56   00B0 CA  56   6E  7A  00A4          EMUL    (SP),R6,IFB$L_DVBN(R10),R6
      7E  6E   09  78  00AB          ASHL    #9,(SP),-(SP)                   ; Calculate records per block
     57   62 A9  3C  00AF          MOVZWL  IRB$W_CSIZ(R9),R7               ; Get recordsize longword
          6E   57  C6  00B3          DIVL2   R7,(SP)
                     00B6
                     00B6
                     00B6  ;
                     00B6  ;  Begining of Bucket Read/Check Loop
                     00B6  ;             (SP) :   Records per block
                     00B6  ;             4(SP) :  Bucket Size as longword
                     00B6  ;
                     00B6
     56   04 AE  C2  00B6  20$:    SUBL    4(SP),R6                 ; Look at preivous bucket
   00B0 CA   56  D1  00BA          CMPL    R6,IFB$L_DVBN(R10)       ; Branch if current VBN less then first
             5E  19  00BF          BLSS    60$                      ;        data block.
        51   56  D0  00C1          MOVL    R6,R1                    ; Read Bucket with : VBN
```

```
          4C A9   D4  00C4   292          CLRL    IRB$L_RP_OFF(R9)        ;                          RP_OFF
                      00C7   293          $CSHFLAGS       <>              ;                          Cache Flags
    00000000'EF   16  00C9   294          JSB     RM$READBKT2             ; Go read it
          59 50   E9  00CF   295          BLBC    R0,ERRORE               ; Branch if error
                      00D2   296          ASSUME  IMP$W_RMSSTATUS EQ 0
36 00000000'9F    04  E0  00D2   297      BBS     #IMP$" IORUNDOWN,@#PIO$GW_PIOIMPA,50$ ; Exit if rundown in prog
             57   D4  00DA   298          CLRL    R7                      ; Count of found records in R7
                      00DC   299
                      00DC   300
                      00DC   301 ;
                      00DC   302 ;  Loop to Check Buc'`t for exsisting records
                      00DC   303 ;
                      00DC   304
          53   01  D0  00DC   305          MOVL    #1,R3                   ; Count the records/bucket
       50   62 A9  3C  00DF   306          MOVZWL  IRB$W_CSIZ(R9),R0       ; Size of each record
       03 65   03  E1  00E3   307 30$:     BBC     #DLC$V_REC,(R5),35$     ; Branch if record does not exist
          57   53  D0  00E7   308          MOVL    R3,R7                   ; Save record number if found
          55   50  C0  00EA   309 35$:     ADDL2   R0,R5                   ; Point at next record
       F2 53   6E  F3  00ED   310          AOBLEQ  (SP),R3,30$             ; Loop if not last record
                      00F1   311
    000000Ov'EF   16  00F1   312 40$:     JSB     RM$RLNERR               ; Release bucket
          31 50   E9  00F7   313          BLBC    R0,ERRORE               ; Branch if error on release
             57   D5  00FA   314          TSTL    R7                      ; Did we find a record?
             B8   13  00FC   315          BEQL    20$                     ; No, go look at prev bucket
                      00FE   316
                      00FE   317 ;
                      00FE   318 ;  Found a record, compute it's Relitive Record Number
                      00FE   319 ;
                      00FE   320
     56   00B0 CA  C2  00FE   321          SUBL    IFB$L_DVBN(R10),R6      ; Number of data blocks before VBN fnd
      56   04 AE   C6  0103   322          DIVL    4(SP),R6                ; and compute # buckets before VBN fnd
          56   6E  C4  0107   323          MULL    (SP),R6                 ; Compute # records before VBN found
   40 A9   01 A647  9E  010A   324          MOVAB   1(R6)[R7],IRB$L_NRP_VBN(R9) ; Compute next record number
                      0110   325
             8E   7C  0110   326 50$:     CLRQ    (SP)+                   ; Pop temp space
        00D0 8F   BA  0112   327          POPR    #^M<R4,R6,R7>           ; Restore plg bdb adr & R6,R7
    00000000'EF   16  0116   328          JSB     RM$RLSPLG               ; Release prolog
          FEE1'   31  011C   329          BRW     RM$EXSUC
                      011F   330
                      011F   331
                      011F   332 ;
                      011F   333 ;  At Begining of file
                      011F   334 ;
                      011F   335
          40 A9   D6  011F   336 60$:     INCL    IRB$L_NRP_VBN(R9)       ; Set nrp to record #1
             EC   11  0122   337          BRB     50$
                      0124   338
```

```
                              0124
                              0124   ;++
                              0124   ;
                              0124   ; handle errors
                              0124   ;
                              0124   ;--
                              0124
                              0124   ;
                              0124   ;  EOF and BIO combination illegal for relative files.
                              0124   ;
                              0124
                              0124   ERRROP:
                              0124           RMSERR          ROP
                    1A   11   0129           BRB     CLNUP
                              012B
                              012B   ;
                              012B   ;  read error on bucket, assume EBK messed up.
                              012B   ;
                              012B
                    8E   7C   012B   ERRORE: CLRQ    (SP)+                   ; Pop temp space
               00D0 8F   BA   012D           POPR    #^M<R4,R6,R7>           ; Restore plg bdb adr & R6,R7
            00000000'EF  16   0131           JSB     RM$RLSPLG               ; release prolog
      0000827A 8F   50   D1   0137           CMPL    R0,#<RMS$_EOF & ^XFFFF> ; end of file error?
                    05   12   013E           BNEQ    CLNUP                   ; branch if not
                              0140           RMSERR  IFA                     ; assume file header messed up
                              0145   ;
                              0145   ; fall thru to cleanup
                              0145   ;
                              0145
                 FEB8'   30   0145   CLNUP:  BSBW    RM$CCLN1                ; deallocate irab
            00000000'EF  17   0148   EXIT:   JMP     RM$EX_NIRAB_SHR         ; and exit
                              014E
                              014E   ;
                              014E   ;  calculated that bucket holds 0 records.  must be bad data in file header.
                              014E   ;
                              014E
                              014E   ERRIFA:
   0C A8    000185D4 8F  D0   014E           MOVL    #RMS$_MRS,RAB$L_STV(R8) ; stv secondary error code
                              0156           RMSERR  IFA                     ; illegal file attributes
                    E8   11   015B           BRB     CLNUP                   ; and cleanup
                              015D
                              015D   ;
                              015D   ;  undefined record format for record i/o
                              015D   ;
                              015D
            00000000'EF  17   015D   ERRRFM: JMP     RM$CONN_ERRRFM          ; go report error
                              0163
                              0163           .END
```

```
$$.PSECT_EP                    = 00000000                     SETNRP                      00000064 R     01
$$.TMP                         = 00000000                     TPT$L_CONNECT2              ********   X   01
$$RMSTEST                      = 0000001A
$$RMS_PBUGCHK                  = 00000010
$$RMS_TBUGCHK                  = 00000008
$$RMS_UMODE                    = 00000004
CEOF                             0000006E R      01
CHKRFM                           00000019 R      01
CLNUP                            00000145 R      01
DLC$V_REC                      = 00000003
ERRIFA                           0000014E R      01
ERRORE                           0000012B R      01
ERRRFM                           0000015D R      01
ERRROP                           00000124 R      01
EXIT                             00000148 R      01
FAB$C_FIX                      = 00000001
FAB$C_UDF                      = 00000000
FAB$C_VFC                      = 00000003
FAB$V_BIO                      = 00000005
FAB$V_BRO                      = 00000006
IFB$B_BKS                      = 0000005E
IFB$B_FAC                      = 00000022
IFB$B_FSZ                      = 0000005F
IFB$B_RFMORG                   = 00000050
IFB$L_DVBN                     = 000000B0
IFB$L_EBK                      = 00000074
IFB$V_SEQFIL                   = 00000038
IFB$W_MRS                      = 00000060
IMP$V_IORUNDOWN                = 00000004
IMP$W_RMSSTATUS                = 00000000
IRB$L_NRP                      = 00000040
IRB$L_NRP_VBN                  = 00000040
IRB$L_RP_OFF                   = 0000004C
IRB$W_CSIZ                     = 00000062
PIO$A_TRACE                      ********   X    01
PIO$GW_PIOIMPA                   ********   X    01
RAB$L_ROP                      = 00000004
RAB$L_STV                      = 0000000C
RAB$V_BIO                      = 0000000B
RAB$V_EOF                      = 00000008
RELF                             0000007F R      01
RM$BDBALLOC                      ********   X    01
RM$CCLN1                         ********   X    01
RM$CONNECT2                      00000000 RG     01
RM$CONNECT_BIO                   00000058 RG     01
RM$CONN_ERRRFM                   ********   X    01
RM$EXSUC                         ********   X    01
RM$EX_NIRAB_SHR                  ********   X    01
RM$LOCK_PROLOG                   ********   X    01
RM$READBKT2                      ********   X    01
RM$RLNERR                        ********   X    01
RM$RLSPLG                        ********   X    01
RMS$_EOF                       = 0001827A
RMS$_IFA                       = 0001C124
RMS$_MRS                       = 000185D4
RMS$_ROP                       = 0001867C
ROP                            = 00000020
```

```
                                     +----------------+
                                     ! 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$RMS2                         00000163  (   355.)  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                        30    00:00:00.08    00:00:00.60
Command processing                   114    00:00:00.68    00:00:05.02
Pass 1                               297    00:00:09.32    00:00:27.49
Symbol table sort                      0    00:00:01.25    00:00:01.79
Pass 2                                77    00:00:01.90    00:00:03.71
Symbol table output                    8    00:00:00.10    00:00:00.27
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                 530    00:00:13.35    00:00:38.90

The working set limit was 1500 pages.
51761 bytes (102 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 983 non-local and 14 local symbols.
387 source lines were read in Pass 1, producing 14 object records in Pass 2.
23 pages of virtual memory were used to define 22 macros.

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

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                         13
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                          1
_$255$DUA28:[SYSLIB]STARLET.MLB;2                       4
TOTALS (all libraries)                                 18

1102 GETS were required to define 18 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RM2CONN/OBJ=OBJ$:RM2CONN MSRC$:RM2CONN/UPDATE=(ENH$:RM2CONN)+EXECML$/LIB+LIB$:RMS/LIB
```

0322 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RM1PUTREC
LIS

RM1PUTSET
LIS

RM1UPDATE
LIS

RM1NXTBLK
LIS

RM1PUTBLD
LIS

RM1RELBLK
LIS

RM1SEQXFR
LIS

RM1PUT
LIS

RM2CONN
LIS

RM1OPEN
LIS

RM1WTLST
LIS

RM1STMFMT
LIS

0323 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY