```
RRRRRRRRRRR      MMM          MMM      SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSSSSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMMMMMM    MMMMMMM   SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRR          RRR MMM   MMM   MMM   MMM SSS
RRRRRRRRRRR      MMM          MMM      SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM      SSSSSSSSS
RRR    RRR       MMM          MMM               SSS
RRR    RRR       MMM          MMM               SSS
RRR    RRR       MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR       RRR    MMM          MMM               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR       RRR    MMM          MMM               SSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
RRR          RRR MMM          MMM  SSSSSSSSSSS
```

FILEID**RMSOSETDD

```
RRRRRRRR    MM       MM    SSSSSSSS    000000     SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  DDDDDDDD    DDDDDDDD
RRRRRRRR    MM       MM    SSSSSSSS    000000     SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  DDDDDDDD    DDDDDDDD
RR      RR  MMMM   MMMM  SS           00    00  SS           EE              TT      DD      DD  DD      DD
RR      RR  MMMMM MMMMM  SS           00    00  SS           EE              TT      DD      DD  DD      DD
RR      RR  MM MMM MM MM  SS           00  0000  SS           EE              TT      DD      DD  DD      DD
RR      RR  MM  M  MM MM  SS           00  0000  SS           EE              TT      DD      DD  DD      DD
RRRRRRRR    MM     MM       SSSSSS    00 00 00    SSSSSS    EEEEEEEE        TT      DD      DD  DD      DD
RRRRRRRR    MM     MM       SSSSSS    00 00 00    SSSSSS    EEEEEEEE        TT      DD      DD  DD      DD
RR  RR      MM     MM             SS  0000  00          SS  EE              TT      DD      DD  DD      DD
RR  RR      MM     MM             SS  0000  00          SS  EE              TT      DD      DD  DD      DD
RR    RR    MM     MM             SS  00    00          SS  EE              TT      DD      DD  DD      DD    ....
RR    RR    MM     MM             SS  00    00          SS  EE              TT      DD      DD  DD      DD    ....
RR      RR  MM     MM  SSSSSSSS      000000    SSSSSSSS    EEEEEEEEEE      TT      DDDDDDDD    DDDDDDDD      ....
RR      RR  MM     MM  SSSSSSSS      000000    SSSSSSSS    EEEEEEEEEE      TT      DDDDDDDD    DDDDDDDD      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1          $BEGIN  RMSOSETDD,000,RM$RMS,<SET DEFAULT DIRECTORY>
0000     2
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY       *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    18 ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
```

```
0000     27 ;++
0000     28 ;
0000     29 ; Facility: rms32
0000     30 ;
0000     31 ; Abstract:
0000     32 ;       this routine sets (and optionally returns) the default directory
0000     33 ;       string in the process i/o control page.
0000     34 ;
0000     35 ; Environment:
0000     36 ;       vax/vms
0000     37 ;
0000     38 ; Author:
0000     39 ;       tim halvorsen   SEP-1979
0000     40 ;
0000     41 ; Modified By:
0000     42 ;
0000     43 ;       V03-004 RAS0245         Ron Schaefer            24-Jan-1984
0000     44 ;               Make sure that the default directory is NOT rooted
0000     45 ;               as that leads to effectively a null directory string
0000     46 ;               which is not good.  Use extended NAM block fields.
0000     47 ;
0000     48 ;       V03-003 KBT0589         Keith B. Thompson       22-Aug-1983
0000     49 ;               Correct error path from RM$XPFN
0000     50 ;
0000     51 ;       V03-002 KBT0521         Keith B. Thompson       23-May-1983
0000     52 ;               RM$XPFN moved so change ref to JSB
0000     53 ;
0000     54 ;       V03-001 KBT0193         Keith B. Thompson       23-Aug-1982
0000     55 ;               Reorganize psects
0000     56 ;
0000     57 ;       V02-002 REFORMAT        Ron Schaefer    30-Jul-1980     09:27
0000     58 ;               Reformat the source.
0000     59 ;
0000     60 ;--
```

```
0000     62          .SBTTL  DEFINITIONS
0000     63
0000     64 ;
0000     65 ; include files
0000     66 ;
0000     67
0000     68 ;
0000     69 ; macros
0000     70 ;
0000     71
0000     72          $FABDEF                         ; fab definitions
0000     73          $NAMDEF                         ; nam definitions
0000     74          $IFBDEF                         ; ifab definitions
0000     75          $FWADEF                         ; fwa definitions
0000     76          $PSLDEF                         ; psl definitions
0000     77          $RMSDEF                         ; rms error codes
0000     78
0000     79 ;
0000     80 ; equated symbols
0000     81 ;
0000     82
0000     83 ;
0000     84 ; own storage
0000     85 ;
```

```
                                        .SBTTL  RMS$SETDDIR, SET DEFAULT DIRECTORY STRING

                                ;++
                                ;
                                ; RMS$SETDDIR:  set default directory string
                                ;
                                ;       this routine moves the given default directory string
                                ;       to the process i/o control page.  the resultant string
                                ;       is optionally returned to the caller.
                                ;
                                ; inputs:
                                ;
                                ;       4(ap) = address of descriptor of new default directory string
                                ;               (optional - if not supplied, no change is made)
                                ;       8(ap) = address of word to receive length of resultant string
                                ;               (optional - if not supplied, no length is returned)
                                ;       12(ap)= address of descriptor of buffer to receive resultant string
                                ;               (optional - if not supplied, string is not returned)
                                ;
                                ; outputs:
                                ;
                                ;       r0 = status code (standard rms codes)
                                ;
                                ;       pio$gt_ddstring may be set to the resultant directory string
                                ;--
                                ;

                                        .ENABL  LSB
                                        $ENTRY  RMS$SETDDIR
                                
                                        $TSTPT  SETDDIR
                                
 55  00000000'9F    9E          MOVAB   @#PIO$GT_DDSTRING,R5    ; address of counted string
     56    85    9A              MOVZBL  (R5)+,R6                ; length of string
                                
                                ;
                                ; if caller requested existing string, return it also
                                ;
                                
       51    0C AC    D0          MOVL    12(AP),R1               ; address of buffer descriptor
             19    13              BEQL    5$                      ; branch if none
                                   IFNORD  #8,(R1),ERRIAL          ; error if not readable
          61    56    B1          CMPW    R6,(R1)                 ; buffer long enough?
             25    1A              BGTRU   ERRIAL                  ; error if not
                                   IFNOWRT (R1),@4(R1),ERRIAL      ; error if not writable
04 B1 61  00  65    56    2C      MOVC5   R6,(R5),#0,(R1),@4(R1)  ; return existing string to caller
                                
                                ;
                                ; if caller requested existing string length, return it
                                ;
                                
       51    08 AC    D0      5$: MOVL    8(AP),R1                ; address to receive length
             09    13              BEQL    10$                     ; branch if not supplied
                                   IFNOWRT #2,(R1),ERRIAL          ; error if not writable
          61    56    B0          MOVW    R6,(R1)                 ; return length to caller
                                
                                ;
```

```
                                003E   144 ; if a new string is not supplied, return the existing string
                                003E   145 ;
                                003E   146
                     04 AC      D5  003E   147 10$:    TSTL    4(AP)                   ; any input string given?
                        0B      12  0041   148        BNEQ    20$                     ; branch if yes
                      0089      31  0043   149        BRW     EXITOK                  ; if not, exit with success
                                0046   150
                                0046   151 ;
                                0046   152 ; process error with caller's arguments
                                0046   153 ;
                                0046   154
                                0046   155 ERRIAL: RMSERR  IAL
                      0084      31  004B   156        BRW     EXIT
                                004E   157
                                004E   158 ;
                                004E   159 ; allocate a fab on the stack to be used as input to xpfn
                                004E   160 ;
                                004E   161
           5E    00000050 8F    C2  004E   162 20$:    SUBL2   #FAB$C_BLN,SP           ; allocate storage for fab
                    58    5E    D0  0055   163        MOVL    SP,R8                   ; save address of fab
68    0050 8F    00    6E    00  2C  0058   164        MOVC5   #0,(SP),#0,#FAB$C_BLN,(R8); zero fab
                                0060   165        ASSUME  FAB$B_BLN EQ FAB$B_BID+1
                 68    5003 8F    B0  0060   166        MOVW    #FAB$C_BID+<FAB$C_BLN@8>,FAB$B_BID(R8); set id and length
                                0065   167
                                0065   168 ;
                                0065   169 ; allocate a nam block on the stack to be used as input to xpfn
                                0065   170 ;
                                0065   171
           5E    00000060 8F    C2  0065   172        SUBL2   #NAM$C_BLN,SP           ; allocate storage for nam block
                    57    5E    D0  006C   173        MOVL    SP,R7                   ; save address of nam block
                 28 A8    5E    D0  006F   174        MOVL    SP,FAB$L_NAM(R8)        ; save address in fab
67    0060 8F    00    6E    00  2C  0073   175        MOVC5   #0,(SP),#0,#NAM$C_BLN,(R7); zero nam block
                                007B   176        ASSUME  NAM$B_BLN EQ NAM$B_BID+1
                 67    6002 8F    B0  007B   177        MOVW    #NAM$C_BID+<NAM$C_BLN@8>,NAM$B_BID(R7); set id and length
                                0080   178        SSB     #NAM$V_SYNCHK,NAM$B_NOP(R7);fast parse
                                0085   179
                                0085   180 ;
                                0085   181   allocate an expanded string buffer for resultant string
                                0085   182 ;
                                0085   183
           5E    000000FF 8F    C2  0085   184        SUBL2   #NAM$C_MAXRSS,SP        ; allocate storage for expanded name
                    0C A7    5E    D0  008C   185        MOVL    SP,NAM$L_ESA(R7)        ; save address in nam block
                 0A A7    FF 8F    90  0090   186        MOVB    #NAM$C_MAXRSS,NAM$B_ESS(R7); set buffer length in nam block
                                0095   187
                                0095   188 ;
                                0095   189 ; parse the input string.  a call frame is created here so that if
                                0095   190 ; fseti detects any errors and ret's, we never lose control.
                                0095   191 ;
                                0095   192
                    51    04 AC    D0  0095   193        MOVL    4(AP),R1                ; get address of input descriptor
                                0099   194        IFNORD  #8,(R1),ERRIAL          ; error if not readable
                       50    61    7D  009F   195        MOVQ    (R1),R0                 ; get descriptor
                                00A2   196        IFNORD  R0,(R1),ERRIAL          ; error if string not readable
                    34 A8    50    90  00A8   197        MOVB    R0,FAB$B_FNS(R8)        ; set length into fab
                    2C A8    61    DE  00AC   198        MOVAL   (R1),FAB$L_FNA(R8)      ; and address also
                    DE'AF    00    FB  00B0   199        CALLS   #0,B^PARSE              ; parse the string - get any rms errors
                          1B 50    E9  00B4   200        BLBC    R0,EXIT                 ; branch if error
```

```
                     00B7   201
                     00B7   202 ;
                     00B7   203 ; store the resultant directory spec in the control region
                     00B7   204 ;
                     00B7   205
        54    3A A7   9A   00B7   206          MOVZBL  NAM$B_DIR(R7),R4        ; get resultant string length
              1A      13   00BB   207          BEQL    ERRDIR                  ; bad if null
        55    48 A7   D0   00BD   208          MOVL    NAM$L_DIR(R7),R5        ; get resultant string address
53   00000000'9F      9E   00C1   209          MOVAB   @#PIO$GT_DDSTRING,R3    ; address to store string
           83    54   90   00C8   210          MOVB    R4,(R3)+                ; store length of string
     63    65    54   28   00CB   211          MOVC3   R4,(R5),(R3)            ; store string behind it
           50    01   D0   00CF   212 EXITOK:  MOVL    #1,R0
                      00D2   213 EXIT:   SSB     #16,R0                  ; mark as rms error
                   04 00D6   214          RET                             ; return with status
                      00D7   215
                      00D7   216 ERRDIR: RMSERR  DIR
                   F4 11 00DC   217          BRB     EXIT
                      00DE   218
                      00DE   219          .DSABL  LSB
```

```
                                00DE   221 ;++
                                00DE   222 ;
                                00DE   223 ; PARSE:        subroutine to parse the input string.  this is a separate
                                00DE   224 ;               procedure so that if any internal rms routine detects an
                                00DE   225 ;               error, then it can perform a ret without our losing control.
                                00DE   226 ;
                                00DE   227 ; inputs:
                                00DE   228 ;
                                00DE   229 ;       4(ap),r8 = fab address
                                00DE   230 ;
                                00DE   231 ; outputs:
                                00DE   232 ;
                                00DE   233 ;       r0 = status
                                00DE   234 ;
                                00DE   235 ;--
                                00DE   236
                          0080  00DE   237 PARSE:  .WORD   ^M<R7>                  ; save nam address over call
                                00E0   238
                                00E0   239 ;
                                00E0   240 ; allocate internal ifab to be used only during this routine
                                00E0   241 ;
                                00E0   242
      5B   00000000'9F    9E  00E0   243         MOVAB   @#PIO$GW_PIOIMPA,R11    ; set impure area address
                  57    01    DO  00E7   244         MOVL    #PSL$C_EXEC,R7          ; caller's access mode
                     FF13'  30  00EA   245         BSBW    RM$FSETI_ALT            ; allocate internal ifab
                                00ED   246
                                00ED   247 ;
                                00ED   248 ; set the previous mode to exec so that nam probes will work
                                00ED   249 ;
                                00ED   250
                        7E    DC  00ED   251         MOVPSL  -(SP)                   ; use current as next
   6E    02    16    01    F0  00EF   252         INSV    #PSL$C_EXEC,#PSL$V_PRVMOD,#PSL$S_PRVMOD,(SP) ; make prvmod=exec
                     F8'AF    9F  00F4   253         PUSHAB  B^10$                   ; just beyond rei
                              02  00F7   254         REI
                                00F8   255
                                00F8   256 ;
                                00F8   257 ; parse the input string
                                00F8   258 ;
                                00F8   259
            00000000'EF    16  00F8   260 10$:     JSB     RM$XPFN                 ; parse/expand file name string
                  14    50    E9  00FE   261         BLBC    R0,30$                  ; branch if error
            0B    6A    1C    E0  0101   262         BBS     #FWA$V_WILD_DIR,(R10),20$       ; error if any wild cards
            07    6A    3C    E0  0105   263         BBS     #FWA$V_EXP_ROOT,(R10),20$       ; error if rooted dir
               3800    8F    B3  0109   264         BITW    #<FWA$M_TYPE!FWA$M_VERSION!FWA$M_NAME>,-
                     6A          010D   265                 FWA$Q_FLAGS(R10)        ; name, type and version should not
                                 010E   266                                         ; be specified on set default
                        05    13  010E   267         BEQLU   30$                     ; they weren't specified
                              0110   268 20$:    RMSERR  DIR
                        50    DD  0115   269 30$:    PUSHL   R0                      ; save return status
                     FEE6'    30  0117   270         BSBW    RM$CLEANUP              ; cleanup ifab, fwa
                        50  8ED0  011A   271         POPL    R0                      ; restore status
                              04  011D   272         RET
                                011E   273
                                011E   274         .END
```

```
$$.PSECT_EP                     = 00000000
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
ERRDIR                            000000D7 R      01
ERRIAL                            00000046 R      01
EXIT                              000000D2 R      01
EXITOK                            000000CF R      01
FAB$B_BID                       = 00000000
FAB$B_BLN                       = 00000001
FAB$B_FNS                       = 00000034
FAB$C_BID                       = 00000003
FAB$C_BLN                       = 00000050
FAB$L_FNA                       = 0000002C
FAB$L_NAM                       = 00000028
FWA$M_NAME                      = 00002000
FWA$M_TYPE                      = 00001000
FWA$M_VERSION                   = 00000800
FWA$Q_FLAGS                     = 00000000
FWA$V_EXP_ROOT                  = 0000003C
FWA$V_WILD_DIR                  = 0000001C
NAM$B_BID                       = 00000000
NAM$B_BLN                       = 00000001
NAM$B_DIR                       = 0000003A
NAM$B_ESS                       = 0000000A
NAM$B_NOP                       = 00000008
NAM$C_BID                       = 00000002
NAM$C_BLN                       = 00000060
NAM$C_MAXRSS                    = 000000FF
NAM$L_DIR                       = 00000048
NAM$L_ESA                       = 0000000C
NAM$V_SYNCHK                    = 00000003
PARSE                             000000DE R      01
PIO$A_TRACE                       ********   X    01
PIO$GT_DDSTRING                   ********   X    01
PIO$GW_PIOIMPA                    ********   X    01
PSL$C_EXEC                      = 00000001
PSL$S_PRVMOD                    = 00000002
PSL$V_PRVMOD                    = 00000016
RM$CLEANUP                        ********   X    01
RM$FSETI_ALT                      ********   X    01
RM$XPFN                           ********   X    01
RMS$SETDDIR                     = FFFFFFFE RG     01
RMS$_DIR                        = 000184CC
RMS$_IAL                        = 0001854C
TPT$C_SETDDIR                     ********   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	0000011E (286.)	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	00:00:00.07	00:00:00.95
Command processing	108	00:00:00.82	00:00:07.53
Pass 1	314	00:00:10.16	00:00:22.31
Symbol table sort	0	00:00:01.40	00:00:01.90
Pass 2	62	00:00:01.81	00:00:03.93
Symbol table output	7	00:00:00.09	00:00:00.23
Psect synopsis output	2	00:00:00.04	00:00:00.11
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	524	00:00:14.39	00:00:37.01

The working set limit was 1500 pages.
55631 bytes (109 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1104 non-local and 8 local symbols.
274 source lines were read in Pass 1, producing 13 object records in Pass 2.
23 pages of virtual memory were used to define 22 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	11
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	18

1233 GETS were required to define 18 macros.

There were no errors, warnings or information messages.

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

RMS0TRUNC
LIS

STAPRFLNM
LIS

RPGCVTPTO
LIS

RPGHANDLE
LIS

RPGMOVE1
LIS

RPGLIB
REQ

RMSGBL
LIS

RPGRTL

RPGDSPLY
LIS

RPGRTL
MAP

RPGPROLOG
REQ

RPGEXTIND
LIS

RPGLIB
LIS

RMS0SRCH
LIS

RMS0WAIT
LIS

RPGBTZ
LIS

RPGMOVE2
LIS

RPGDIVIDE
LIS

RPGIOEXCE
LIS

RPGDEF
REQ

RMS0UPDAT
LIS

RPGERROR
LIS