```
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
```

```
RRRRRRRR    MM      MM      11      PPPPPPPP    UU      UU  TTTTTTTTTT
RRRRRRRR    MM      MM      11      PPPPPPPP    UU      UU  TTTTTTTTTT
RR      RR  MMMM  MMMM    1111      PP      PP  UU      UU      TT
RR      RR  MMMM  MMMM    1111      PP      PP  UU      UU      TT
RR      RR  MM  MM  MM      11      PP      PP  UU      UU      TT
RR      RR  MM  MM  MM      11      PP      PP  UU      UU      TT
RRRRRRRR    MM      MM      11      PPPPPPPP    UU      UU      TT
RRRRRRRR    MM      MM      11      PPPPPPPP    UU      UU      TT
RR  RR      MM      MM      11      PP          UU      UU      TT
RR  RR      MM      MM      11      PP          UU      UU      TT
RR    RR    MM      MM      11      PP          UU      UU      TT          ....
RR    RR    MM      MM      11      PP          UU      UU      TT          ....
RR      RR  MM      MM    111111    PP          UUUUUUUUUU      TT          ....
RR      RR  MM      MM    111111    PP          UUUUUUUUUU      TT          ....

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          $BEGIN  RM1PUT,000,RM$RMS1,<SEQUENTIAL SPECIFIC PUT>
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    26 ;
0000    27 ;++
0000    28 ; Facility: RMS32
0000    29 ;
0000    30 ; Abstract:
0000    31 ;                This module provides sequential file organization
0000    32 ;                specific processing for the $put function.
0000    33 ;
0000    34 ;
0000    35 ; Environment:
0000    36 ;                Star processor running Starlet exec.
0000    37 ;
0000    38 ; Author: L. F. Laverdure        Creation Date: 17-FEB-1977
0000    39 ;
0000    40 ; Modified By:
0000    41 ;
0000    42 ;       V03-002 RAS0280         Ron Schaefer            27-Mar-1984
0000    43 ;               Eliminate call to RM$CHKEOF1 by doing the check inline.
0000    44 ;               Eliminate some spurious branches as well.
0000    45 ;
0000    46 ;       V03-001 KBT0143         Keith B. Thompson       20-Aug-1982
0000    47 ;               Reorganize psects
0000    48 ;
0000    49 ;       V02-016 CDS0001         C Saether                9-Nov-1981
0000    50 ;               Change brw to jmp to fix broken branch.
0000    51 ;
0000    52 ;       V02-015 REFORMAT        K. E. Kinnear           31-Jul-1980     9:01
0000    53 ;
0000    54 ;       V01-014 PSK0012         P. S. Knibbe            14-Feb-1980     11:30
0000    55 ;               Put to a block foreign device sets eof bit. Put
0000    56 ;               to a device with tpt and truncate access sets eof bit.
0000    57 ;
```

```
                                ;  V01-013 JAK0001          J. A. Krycka            27-Aug-1978     13:42
                                ;        Miscellaneous clean-up prior to DECNET V1.0 code freeze.
                                ;        Add code to support network access by key.
                                ;
                                ;--
                                ;
                                
```

```
                     0000     66          .SBTTL  DECLARATIONS
                     0000     67
                     0000     68 ;
                     0000     69 ; Include Files:
                     0000     70 ;
                     0000     71
                     0000     72 ;
                     0000     73 ; Macros:
                     0000     74 ;
                     0000     75
                     0000     76          $IFBDEF
                     0000     77          $DEVDEF
                     0000     78          $FABDEF
                     0000     79          $RABDEF
                     0000     80          $IRBDEF
                     0000     81          $RMSDEF
                     0000     82
                     0000     83 ;
                     0000     84 ; Equated Symbols:
                     0000     85 ;
                     0000     86
00000020             0000     87          ROP=RAB$L_ROP*8                 ; bit offset to rop field
                     0000     88
                     0000     89 ;
                     0000     90 ; Own Storage:
                     0000     91 ;
                     0000     92
```

```
                        .SBTTL  RM$PUT1 - HIGH LEVEL SEQUENTIAL $PUT

                ;++
                ; RM$PUT1 -- High Level Sequential $PUT.
                ;
                ; This module performs the following functions:
                ;
                ;       1.  Calls rm$putsetup1 to perform various setups.
                ;
                ;       2.  Initializes the current record size to zero.
                ;
                ;       3.  Verifies that rac = sequential or key (if rfm=fix).
                ;
                ;       4.  If device is unit record calls RM$PUT_UNIT_REC.
                ;           Otherwise, verifies positioning at eof
                ;           and calls RM$PUT_BLK_DEV unless rac=key in which case calls
                ;           RM$UPDATE_ALT.
                ;
                ;
                ; Calling Sequence:
                ;
                ;        Entered via case branch from RM$PUT at RM$PUT1.
                ;
                ; Input Parameters:
                ;
                ;       R11     impure area address
                ;       R10     IFAB addr
                ;       R9      IRAB addr
                ;       R8      rab addr
                ;
                ; Implicit Inputs:
                ;
                ;       The contents of the rab and related IRAB and IFAB.
                ;
                ; Output Parameters:
                ;
                ;       R7 thru R1      destroyed
                ;       R0              status
                ;
                ; Implicit Outputs:
                ;
                ;       Various fields of the rab are filled in to reflect
                ;       the status of the operation (see functional spec
                ;       for details).
                ;
                ;       The IRAB is similarly updated.
                ;
                ; Completion Codes:
                ;
                ;       Standard rms (see functional spec).
                ;
                ; Side Effects:
                ;
                ;       none
                ;
                ;--
                
```

```
                     0000   152 RM$PUT1::
                     0000   153         $TSTPT  PUT1
              FFF7'  30   0006   154         BSBW    RM$PUTSETUP1            ; perform various put setups
            62 A9    B4   0009   155         CLRW    IRB$W_CSIZ(R9)          ; indicate no current record
            6D 50    E9   000C   156         BLBC    R0,ERROR
                     000F   157
                     000F   158 ;
                     000F   159 ; Verify that record access mode is sequential.
                     000F   160 ;
                     000F   161
                     000F   162         ASSUME  RAB$C_SEQ EQ 0
                     000F   163
            1E A8    95   000F   164         TSTB    RAB$B_RAC(R8)
               21    12   0012   165         BNEQ    CHKRRN
                     0014   166
                     0014   167         ASSUME  DEV$V_REC EQ 0
                     0014   168
            1B 6A    E8   0014   169         BLBS    IFB$L_PRIM_DEV(R10),DAPDEV      ; branch if unit record device
                     0017   170
                     0017   171 ;
                     0017   172 ;  Sequential $PUT to a block device: must be positioned at eof.
                     0017   173 ;
                     0017   174
                     0017   175 BLKDEV:
         14 69       21   E0  0017   176         BBS     #IRB$V_EOF,(R9),PUTBLK1 ; branch if already eof
      74 AA    40 A9    D1   001B   177         CMPL    IRB$L_NRP_VBN(R9),IFB$L_EBK(R10) ; at eof?
               5D       1F   0020   178         BLSSU   ERRNEF                  ; branch if not yet
               07       1A   0022   179         BGTRU   PUTBLK                  ; branch if definitely
                     0024   180                                                 ; in current eof block
      5C AA    44 A9    B1   0024   181         CMPW    IRB$W_NRP_OFF(R9),IFB$W_FFB(R10) ; how about byte position?
               54       1F   0029   182         BLSSU   ERRNEF                  ; nope - not there yet
                     002B   183 PUTBLK: SSB     #IRB$V_EOF,(R9)                 ; eof - set the flag
                     002F   184
                     002F   185 PUTBLK1:
              FFCE'  31   002F   186         BRW     RM$PUT_BLK_DEV          ; do the put
                     0032   187
                     0032   188 ;
                     0032   189 ; $PUT to a unit record device.
                     0032   190 ;
                     0032   191
              FFCB'  31   0032   192 DAPDEV: BRW     RM$PUT_UNIT_REC
                     0035   193
                     0035   194 ;
                     0035   195 ;  Code to perform random mode $PUT by relative record #.
                     0035   196 ;
                     0035   197
                     0035   198 CHKRRN:
      F9 6A       3E   E0  0035   199         BBS     #IFB$V_DAP,(R10),DAPDEV ; branch if network operation
      01    1E A8    91   0039   200         CMPB    RAB$B_RAC(R8),#RAB$C_KEY; keyed access?
               57       12   003D   201         BNEQ    ERRRAC                  ; no - it's a problem
      53 6A    1C   E1  003F   202         BBC     #DEV$V_RND,IFB$L_PRIM_DEV(R10),ERRRAC; branch if not disk
      30 6A    2D   E0  0043   203         BBS     #IFB$V_SQO,(R10),ERRSQO ; branch if sqo set
            FFB6'  30   0047   204         BSBW    RM$SEQKEY               ; convert rrn to rfa
         2F 50    E9   004A   205         BLBC    R0,ERROR                ; get out on errors
      14 68       24   E0  004D   206         BBS     #RAB$V_UIF+ROP,(R8),10$ ; branch if put anywhere is ok
      10 6A       18   E0  0051   207         BBS     #DEV$V_FOR,IFB$L_PRIM_DEV(R10),10$; branch if foreign device
   74 AA    10 A8    D1   0055   208         CMPL    RAB$W_RFA(R8),IFB$L_EBK(R10); check for attempt to put
```

```
                         005A  209                                        ;  beyond eof
              33   1F    005A  210          BLSSU   ERRNEF1                ; branch if before eof (error)
              07   1A    005C  211          BGTRU   10$                    ; branch if beyond
5C AA   14    A8   B1    005E  212          CMPW    RAB$W_RFA+4(R8),IFB$W_FFB(R10); in eof blk - check offset
              2A   1F    0063  213          BLSSU   ERRNEF1                ; branch if before eof
   62 A9      56   B0    0065  214 10$:      MOVW    R6,IRB$W_CSIZ(R9)      ; set current record size
      51      56   D0    0069  215          MOVL    R6,R1                  ; restore record size
48 A9    10    A8   7D    006C  216          MOVQ    RAB$W_RFA(R8),IRB$L_RP_VBN(R9); set rp from rfa
         4E    A9   B4    0071  217          CLRW    IRB$L_RP_OFF+2(R9)     ; insure valid long word offset
           FF89'   31    0074  218          BRW     RM$UPDATE_ALT          ; go do the random put
```

```
                     0077   220
                     0077   221 ;++
                     0077   222 ;
                     0077   223 ; Error Processors:
                     0077   224 ;
                     0077   225 ;--
                     0077   226
                     0077   227 ERRSQO:
                     0077   228         RMSERR  SQO                     ; seq output only and user
                     007C   229
           FF81'  31 007C   230 ERROR:  BRW     RM$EXRMS                ; get out
                     007F   231
                     007F   232 ;
                     007F   233 ;    If device is foreign or truncate is allowed
                     007F   234 ;        then set eof and continue
                     007F   235 ;        else error
                     007F   236 ;
                     007F   237
             18    E0 007F   238 ERRNEF: BBS     #DEV$V_FOR,-                ; continue
           A8 6A        0081   239                 IFB$L_PRIM_DEV(R10),PUTBLK  ; if foreign
                     0083   240
                     0083   241 ;
                     0083   242 ; Not foreign - is truncate permitted ?
                     0083   243 ;
                     0083   244
        08 68    21   E1 0083   245         BBC     #RAB$V_TPT+ROP,(R8),ERRNEF1 ; no - real error
     9F 22 AA    04   E0 0087   246         BBS     #FAB$V_TRN,IFB$B_FAC(R10),PUTBLK; yes - if truncate
           FF71'  31   008C   247         BRW     RM$ERRFAC                   ; else error in access
                     008F   248
                     008F   249 ERRNEF1:
                     008F   250         RMSERR  NEF                     ; put not at end of file
           E6    11  0094   251         BRB     ERROR
                     0096   252
                     0096   253 ERRRAC:
                     0096   254         RMSERR  RAC                     ; bad record access value
           DF    11  009B   255         BRB     ERROR
                     009D   256
                     009D   257         .END
```

```
$$.PSECT_EP                     =  00000000
$$RMSTEST                       =  0000001A
$$RMS_PBUGCHK                   =  00000010
$$RMS_TBUGCHK                   =  00000008
$$RMS_UMODE                     =  00000004
BLKDEV                             00000017 R      01
CHKRRN                             00000035 R      01
DAPDEV                             00000032 R      01
DEV$V_FOR                       =  00000018
DEV$V_REC                       =  00000000
DEV$V_RND                       =  0000001C
ERRNEF                             0000007F R      01
ERRNEF1                            0000008F R      01
ERROR                              0000007C R      01
ERRRAC                             00000096 R      01
ERRSQO                             00000077 R      01
FAB$V_TRN                       =  00000004
IFB$B_FAC                       =  00000022
IFB$L_EBK                       =  00000074
IFB$L_PRIM_DEV                  =  00000000
IFB$V_DAP                       =  0000003E
IFB$V_SQO                       =  0000002D
IFB$W_FFB                       =  0000005C
IRB$L_NRP_VBN                   =  00000040
IRB$L_RP_OFF                    =  0000004C
IRB$L_RP_VBN                    =  00000048
IRB$V_EOF                       =  00000021
IRB$W_CSIZ                      =  00000062
IRB$W_NRP_OFF                   =  00000044
PIO$A_TRACE                        ********    X   01
PUTBLK                             0000002B R      01
PUTBLK1                            0000002F R      01
RAB$B_RAC                       =  0000001E
RAB$C_KEY                       =  00002001
RAB$C_SEQ                       =  00000000
RAB$L_ROP                       =  00000004
RAB$V_TPT                       =  00000001
RAB$V_UIF                       =  00000004
RAB$W_RFA                       =  00000010
RMSERRFAC                          ********    X   01
RMSEXRMS                           ********    X   01
RMSPUT1                            00000000 RG     01
RMSPUTSETUP1                       ********    X   01
RMSPUT_BLK_DEV                     ********    X   01
RMSPUT_UNIT_REC                    ********    X   01
RMSSEQKEY                          ********    X   01
RMSUPDATE_ALT                      ********    X   01
RMS$_NEF                        =  000185E4
RMS$_RAC                        =  00018644
RMS$_SQO                        =  000186C4
ROP                             =  00000020
TPT$L_PUT1                         ********    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$RMS1	0000009D (157.)	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	00:00:00.10	00:00:00.42
Command processing	107	00:00:00.74	00:00:04.75
Pass 1	292	00:00:08.64	00:00:25.34
Symbol table sort	0	00:00:01.18	00:00:02.08
Pass 2	59	00:00:01.64	00:00:04.04
Symbol table output	8	00:00:00.07	00:00:00.12
Psect synopsis output	2	00:00:00.03	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	499	00:00:12.40	00:00:36.79

The working set limit was 1350 pages.
48436 bytes (95 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 979 non-local and 2 local symbols.
257 source lines were read in Pass 1, producing 13 object records in Pass 2.
20 pages of virtual memory were used to define 19 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	10
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	15

1082 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:RM1PUT/OBJ=OBJ$:RM1PUT MSRC$:RM1PUT/UPDATE=(ENH$:RM1PUT)+EXECML$/LIB+LIB$:RMS/LIB
```

RM1PUTREC
LIS

RM1PUTSET
LIS

RM1UPDATE
LIS

RM1NXTBLK
LIS

RM1PUTBLD
LIS

RM1RELBLK
LIS

RM1SEQXFR
LIS

RM1PUT
LIS

RM2CONN
LIS

RM1OPEN
LIS

RM1WTLST
LIS

RM1STMFMT
LIS