```
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**RM1PUTREC

RM1
Sym
$$.
$$R
$$R
$$R
$$R
BDB
BDB
BDB
BDB
BDB
CCT
CR
DEV
DEV
DEV
DEV
ERR
EXI
EXI
FAB
FAB
FAB
FAB
FAB
FAB
FAB
FF
FTN
IFB
IFB
IFB
IFB
IFB
IMP
IMP
IRB
IRB
IRB
IRB
IRB
IRB
LF
LF
MAP
MOV
MOV
MOV
NOT
NTM
PIC
PRM
RAB
RAB
RAB
RAB
RAB

Table of contents

(2) 90 DECLARATIONS
(3) 135 RM$MAPFTN - ROUTINE TO CONVERT FROM FTN TO PRN FORMAT
(4) 180 RM$PUT_UNIT_REC

```
0000     1          $BEGIN  RM1PUTREC,000,RM$RMS1,<INTERNAL PUT SEQ FOR UNIT RECORD DEVICE>
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 performs a $put for the sequential file
0000    32 ;                organization on a unit record device.
0000    33 ;
0000    34 ; Environment:
0000    35 ;                star processor running starlet exec.
0000    36 ;
0000    37 ; Author: L F Laverdure,          creation date: 17-FEB-19-77
0000    38 ;
0000    39 ; Modified By:
0000    40 ;
0000    41 ;       V03-004 JEJ0001         J E Johnson             27-Feb-1984
0000    42 ;                Correct the mishandling of the fixed control field in
0000    43 ;                VFC process permanent files when accessed across the network.
0000    44 ;
0000    45 ;       V03-003 JAK0131         J A Krycka              06-Feb-1984
0000    46 ;                Fix bug preventing return of RFA for network access.
0000    47 ;
0000    48 ;       V03-002 KBT0145         Keith B. Thompson       20-Aug-1982
0000    49 ;                Reorganize psects
0000    50 ;
0000    51 ;       V03-001 KBT0083         Keith B. Thompson       13-Jul-1982
0000    52 ;                Clean up psects
0000    53 ;
0000    54 ;       V02-022 KPL0001         Peter Lieberwirth       31-Dec-1981
0000    55 ;                Fix more broken branches.
0000    56 ;
0000    57 ;       V02-021 TMK0043         Todd M. Katz            26-Dec-1981
```

```
0000     58 ;                  Fix a broken branch by changing a BRW MOVDAT1 to a JMP.
0000     59 ;
0000     60 ;        V02-020 RAS0052         Ron Schaefer            15-Dec-1981
0000     61 ;                Fix carriage control for stm files, and non-CCL devices.
0000     62 ;
0000     63 ;        V02-019 REFORMAT        Maria del C. Nasr       24-Jul-1980
0000     64 ;
0000     65 ;        V018    JAK0033         J A Krycka      4-JAN-1980      11:00
0000     66 ;                Perform appropriate carriage control handling for network
0000     67 ;                process permanent files.
0000     68 ;
0000     69 ;        v017    CDS0062         C D Saether     7-DEC-1979      13:50
0000     70 ;                return second longword iosb in stv of rab
0000     71 ;
0000     72 ;        v016    PSK0001         P S Knibbe      23-NOV-1979     12:30
0000     73 ;                set irab end of file bit on foreign magtape put
0000     74 ;
0000     75 ;        v015    CDS0030         C D Saether     11-SEP-1979     16:15
0000     76 ;                stop breaking up records to terminals also
0000     77 ;
0000     78 ;        v014    CDS0027         C D Saether     18-AUG-1979     00:55
0000     79 ;                put to unit record device goes straight from user buffer
0000     80 ;                terminals are only device where records get broken up
0000     81 ;
0000     82 ;        V012    JAK0015         J A Krycka      15-DEC-1978     11:00
0000     83 ;                Fix network bug that drops first character of record for
0000     84 ;                files with fortran carriage control.
0000     85 ;
0000     86 ;--
0000     87 ;
0000     88
```

```
                     0000     90          .SBTTL  DECLARATIONS
                     0000     91
                     0000     92 ;
                     0000     93 ; Include Files:
                     0000     94 ;
                     0000     95
                     0000     96 ;
                     0000     97 ; Macros:
                     0000     98 ;
                     0000     99
                     0000    100          $IRBDEF
                     0000    101          $RABDEF
                     0000    102          $IFBDEF
                     0000    103          $DEVDEF
                     0000    104          $FABDEF
                     0000    105          $BDBDEF
                     0000    106          $IMPDEF
                     0000    107          $RMSDEF
                     0000    108
                     0000    109 ;
                     0000    110 ; Equated Symbols:
                     0000    111 ;
                     0000    112
00000020             0000    113          SPACE=32
0000000A             0000    114          LF=10
0000000B             0000    115          VT=11
0000000C             0000    116          FF=12
0000000D             0000    117          CR=13
                     0000    118
                     0000    119 ;
                     0000    120 ; Own Storage:
                     0000    121 ;
                     0000    122 ;
                     0000    123 ;  fortran to 'pre/post' standard carriage control mapping table
                     0000    124 ;
                     0000    125
                     0000    126 CCTL_TABLE:
                     0000    127                                                  ; entries are fortran byte, pre, post
8D 01 20             0000    128          .BYTE   ^A/ /,1,128+CR                  ; " " - single space
8D 02 30             0003    129          .BYTE   ^A/0/,2,128+CR                  ; "0" - double space
8D 8C 31             0006    130          .BYTE   ^A/1/,128+FF,128+CR             ; "1" - form feed
8D 00 2B             0009    131          .BYTE   ^A/+/,0,128+CR                  ; "+" - overprint
00 01 24             000C    132          .BYTE   ^A/$/,1,0                       ; "$" - prompt
00 00 00             000F    133          .BYTE   0,0,0                           ; null
```

```
                     0012   135          .SBTTL  RM$MAPFTN - ROUTINE TO CONVERT FROM FTN TO PRN FORMAT
                     0012   136
                     0012   137 ;++
                     0012   138 ; RM$MAPFTN - This routine converts the fortran carriage control
                     0012   139 ;            character in R0 into the equivalent pre/post carriage
                     0012   140 ;            control word.
                     0012   141 ;
                     0012   142 ; Calling sequence:
                     0012   143 ;
                     0012   144 ;       bsbw    rm$mapftn
                     0012   145 ;
                     0012   146 ; inputs:
                     0012   147 ;
                     0012   148 ;       r0      fortran carriage control character
                     0012   149 ;
                     0012   150 ; outputs:
                     0012   151 ;
                     0012   152 ;       r2      pre/post carriage control word
                     0012   153 ;
                     0012   154 ; note:  this routine always succeeds. no other registers destroyed.
                     0012   155 ;
                     0012   156 ;--
                     0012   157
                     0012   158 RM$MAPFTN::
52   EB AF   9E      0012   159          MOVAB   B^CCTL_TABLE,R2         ; addr of mapping table
   82   50   91      0016   160 10$:     CMPB    R0,(R2)+                ; match on char?
        0A   13      0019   161          BEQL    MAPCTL                  ; branch if yes
        82   B5      001B   162          TSTW    (R2)+                   ; bump to next entry
        F7   12      001D   163          BNEQ    10$                     ; continue if more
                     001F   164
                     001F   165 ;
                     001F   166 ;  no match  -  give default of line feed before, cr after
                     001F   167 ;
                     001F   168
                     001F   169          ASSUME  IRB$B_POST_CCTL EQ IRB$B_PRE_CCTL+1
52   8D01 8F    B0   001F   170          MOVW    #1+<<T28+CR>@8>,R2      ; lf=rec-cr
                05   0024   171          RSB
                     0025   172
                     0025   173 ;
                     0025   174 ;  pick up pre and post cctl from table
                     0025   175 ;
                     0025   176
     52   62    B0   0025   177 MAPCTL: MOVW    (R2),R2                 ; get pre and post
                05   0028   178          RSB
```

```
0029   180          .SBTTL  RM$PUT_UNIT_REC
0029   181
0029   182 ;++
0029   183 ; RM$PUT_UNIT_REC - This routine performs a $PUT to a unit
0029   184 ;       record device, setting carriage control as required
0029   185 ;       and handling crossing of block boundaries.
0029   186 ;
0029   187 ; Calling sequence:
0029   188 ;
0029   189 ;       bsbw    rm$put_unit_rec
0029   190 ;
0029   191 ; Input Parameters:
0029   192 ;
0029   193 ;       r11     impure area pointer
0029   194 ;       r10     ifab addr
0029   195 ;       r9      irab addr
0029   196 ;       r8      rab addr
0029   197 ;       r6      user record size
0029   198 ;       r5      user record addr (first block probed)
0029   199 ;
0029   200 ; Implicit Inputs:
0029   201 ;
0029   202 ;       the contents of the rab and related irab and ifab.
0029   203 ;
0029   204 ; Output Parameters:
0029   205 ;
0029   206 ;       r0      status code
0029   207 ;       r1-r7   destroyed
0029   208 ;
0029   209 ; Implicit Outputs:
0029   210 ;
0029   211 ;       The RAB and the internal structures are updated to reflect
0029   212 ;       the results of the put. (see functional spec).
0029   213 ;
0029   214 ; Completion Codes:
0029   215 ;
0029   216 ;       standard rms.
0029   217 ;
0029   218 ; Side Effects:
0029   219 ;
0029   220 ;       none
0029   221 ;
0029   222 ;--
0029   223
```

```
                     0029   225 RMS$PUT_UNIT_REC::
                     0029   226         $TSTPT  PUTREC1
54   20 A9   D0      002F   227         MOVL    IRB$L_CURBDB(R9),R4     ; get current bdb addr
        04   12      0033   228         BNEQ    10$
54   3C A9   D0      0035   229         MOVL    IRB$L_NXTBDB(R9),R4     ; so use nxtbdb instead then
                     0039   230
                     0039   231 ;++
                     0039   232 ;
                     0039   233 ; determine required carriage control
                     0039   234 ;
                     0039   235 ; 4 types of carriage control may be specified:
                     0039   236 ;       none            = record
                     0039   237 ;       fab$v_cr        = lf-record-cr
                     0039   238 ;       fab$v_ftn       = 1st char of record determines, as follows:
                     0039   239 ;                         space = lf - record - cr
                     0039   240 ;                         0     = lf,lf - record - cr
                     0039   241 ;                         1     = ff - record - cr
                     0039   242 ;                         $     = lf - record
                     0039   243 ;                         +     = record - cr
                     0039   244 ;                         null  = record
                     0039   245 ;                         other = lf - record - cr
                     0039   246 ;       fab$v_prn       = print file carriage control specified in vfc header
                     0039   247 ;                         as a pre and post field indicating carriage control
                     0039   248 ;                         to be performed before and after printing the
                     0039   249 ;                         record.  the pre and post carriage control bytes
                     0039   250 ;                         have the following format:
                     0039   251 ;
                     0039   252 ;                           bit 7 = 0
                     0039   253 ;                               bits 6-0 give # of new lines
                     0039   254 ;                           bit 7 = 1
                     0039   255 ;                             bit 6 = 0
                     0039   256 ;                               bit 5 = 0
                     0039   257 ;                                 bits 4-0 give the ascii control character
                     0039   258 ;                                 to print (c0 set)
                     0039   259 ;                               bit 5 = 1
                     0039   260 ;                                 bits 4-0 give the ascii control character
                     0039   261 ;                                 to print (c1 set)
                     0039   262 ;                             bit 6 = 1
                     0039   263 ;                               bit 5 = 0
                     0039   264 ;                                 bits 4-0 have device-specific interpretation
                     0039   265 ;                                 (reserved)
                     0039   266 ;                               bit 5 - 1
                     0039   267 ;                                 (reserved)
                     0039   268 ;
                     0039   269 ;--
                     0039   270
                     0039   271         ASSUME  IRB$B_POST_CCTL EQ IRB$B_PRE_CCTL+1
                     0039   272         ASSUME  IMP$W_RMSSTATUS EQ 0
                     0039   273         ASSUME  IMP$V_IIOS EQ 0
   0D   E0           0039   274 10$:    BBS     #DEV$V_NET,-            ; If we are a net device,
04 6A                003B   275                 IFB$L_PRIM_DEV(R10),15$ ; then force some ccl device processing
   01   E1           003D   276         BBC     #DEV$V_CCL,-            ; no cctl if not ccl device
7B 6A                003F   277                 IFB$L_PRIM_DEV(R10),MOVDAT1
64 A9   B4           0041   278 15$:    CLRW    IRB$B_PRE_CCTL(R9)      ; initialize (null carriage ctl)
52 51 AA   90        0044   279         MOVB    IFB$B_RAT(R10),R2       ; get rat for file
                     0048   280
                     0048   281 ;
```

```
                                0048   282 ; for a network $put function, determine pre and post carriage control only if
                                0048   283 ; it is a process permanent file in print file format.
                                0048   284 ; note: this avoids fortran carriage control processing for network
                                0048   285 ;       non-process permanent files at the local node which is correct
                                0048   286 ;       because the carriage control character must be passed to the
                                0048   287 ;       remote fal where it will be handled.
                                0048   288 ;
                                C048   289
                09 6A    0D     E1     0048   290         BBC     #DEV$V_NET,(R10),20$     ; branch if not network access
                     69  6B     E8     004C   291         BLBS    (R11),MOVDAT             ; branch if not ppf
                65 52    02     E1     004F   292         BBC     #FAB$V_PRN,R2,MOVDAT     ; branch if not a 'print' file
                         07     11     0053   293         BRB     30$                      ; process pre/post carriage control
                0D 52    02     E1     0055   294 20$:     BBC     #FAB$V_PRN,R2,NOTPRN     ; branch if not a 'print' file
                     41  6B     E8     0059   295         BLBS    (R11),PRNFMT             ; branch if not ppf
                                005C   296
                                005C   297 ;
                                005C   298 ;  this is a process-permanent print file.
                                005C   299 ;  extract rat value from isi.
                                005C   300 ;
                                005C   301
        52   02 A8    08     06     EF     005C   302 30$:     EXTZV   #RAB$V_PPF_RAT,#RAB$S_PPF_RAT,RAB$W_ISI(R8),R2
                37 52    02     E0     0062   303         BBS     #FAB$V_PRN,R2,PRNFMT     ; branch if print format
                                0066   304
                                0066   305 ;
                                0066   306 ;  check for cr, ftn, or no carriage control
                                0066   307 ;
                                0066   308
                45 52    01     E0     0066   309 NOTPRN: BBS     #FAB$V_CR,R2,LF_REC_CR   ; branch if 'cr'
                                006A   310         ASSUME  FAB$V_FTN EQ 0
                     22 52      E8     006A   311         BLBS    R2,FTN_REC               ; branch if FTN cctl
                                006D   312
                                006D   313 ;
                                006D   314 ;  null carriage control.  do nothing unless it's a stream file.
                                006D   315 ;
                                006D   316
                                006D   317         ASSUME  FAB$C_STM+1 EQ FAB$C_STMLF
                                006D   318         ASSUME  FAB$C_STMLF+1 EQ FAB$C_STMCR
                                006D   319         ASSUME  FAB$C_STMCR EQ FAB$C_MAXRFM
           50 AA    91     006D   320         CMPB    IFB$B_RFMORG(R10),-      ; stm file?
                04             0070   321                 #FAB$C_STM
                45       1F    0071   322         BLSSU   MOVDAT                   ; nope, proceed
                56       B5    0073   323         TSTW    R6                       ; zero len record
                38       13    0075   324         BEQL    LF_REC_CR                ; needs a terminator
                32       BB    0077   325         PUSHR   #^M<R1,R4,R5>            ; save size, record and buff addr
      51   FF A546     9E     0079   326         MOVAB   -1(R5)[R6],R1            ; setup for term check
           50   01     D0     007E   327         MOVL    #1,R0                    ; check only last char
        54   50 AA     9A     0081   328         MOVZBL  IFB$B_RFMORG(R10),R4      ; get format type
              FF78'     30     0085   329         BSBW    RM$STM_TERM              ; check for terminator
                32       BA    0088   330         POPR    #^M<R1,R4,R5>            ; restore regs
           2B 50       E8     008A   331         BLBS    R0,MOVDAT                ; already have a terminator
                20       11    008D   332         BRB     LF_REC_CR                ; add one
                                008F   333
                                008F   334 ;
                                008F   335 ;  fortran carriage control.  pick up control byte and interpret.
                                008F   336 ;
                                008F   337
                56       D5    008F   338 FTN_REC:TSTL    R6                       ; zero length record?
```

```
                 1C   13  0091   339          BEQL    LF_REC_CR                ; branch if yes (same as blank)
                 56   D7  0093   340          DECL    R6                       ; decrement size
              50 85   90  0095   341          MOVB    (R5)+,R0                 ; get fortran cctl byte
                 FF77 30  0098   342          BSBW    RM$MAPFTN                ; and map to pre/post format
                 17   11  009B   343          BRB     MOVPREPOST
                         009D   344
                         009D   345 ;
                         009D   346 ;  'prn' carriage control.
                         009D   347 ;  record header buffer contains explicit 'standard' carriage control.
                         009D   348 ;
                         009D   349
           50    2C A8   D0  009D   350 PRNFMT: MOVL    RAB$L_RHB(R8),R0         ; get record header buffer addr
                 15   13  00A1   351          BEQL    MOVDAT                   ; branch if none (=null cctl)
                         00A3   352          IFNORD  #2,(R0),ERRRHB           ; branch if rhb not readable by caller
           64 A9 60   B0  00A9   353          MOVW    (R0),IRB$B_PRE_CCTL(R9)  ; set carriage control
                 09   11  00AD   354          BRB     MOVDAT
                         00AF   355
                         00AF   356 ;
                         00AF   357 ;  line feed before, carriage return after record
                         00AF   358 ;
                         00AF   359
                         00AF   360 LF_REC_CR:
        52    8D01 8F   B0  00AF   361          MOVW    #1+<<128+CR>@8>,R2       ; convert to pre/post format
                         00B4   362
                         00B4   363 MOVPREPOST:
           64 A9 52   B0  00B4   364          MOVW    R2,IRB$B_PRE_CCTL(R9)    ; save in irab for print
```

```
                          00B8    366
                          00B8    367 ;
                          00B8    368 ; move data record into buffer
                          00B8    369 ;
                          00B8    370
                          00B8    371 MOVDAT:
      35 6A    3E    E0   00B8    372             BBS     #IFB$V_DAP,(R10),NTMOVE ; branch if network file access
                          00BC    373 MOVDAT1:                                    ; return from ntmove
      4C A4    55    D0   00BC    374             MOVL    R5,BDB$L_CURBUFADR(R4)  ; addr of buffer to put from
      14 A4    56    B0   00C0    375             MOVW    R6,BDB$W_NUMB(R4)       ; size to put
            64 A9    B0   00C4    376             MOVW    IRB$B_PRE_CCTL(R9),-
            4A A4         00C7    377                      BDB$B_PRE_CCTL(R4)     ; reset carriage control
             FF34'   30   00C9    378             BSBW    RM$SEQPUTR              ; put the record
            15 50    E9   00CC    379             BLBC    R0,EXIT
               05    E1   00CF    380             BBC     #DEV$V_SQD,-
            08 6A         00D1    381                      IFB$L_PRIM_DEV(R10),10$; branch if not magtape
               18    E1   00D3    382             BBC     #DEV$V_FOR,-
            04 6A         00D5    383                      IFB$L_PRIM_DEV(R10),10$; branch if not foreign
                          00D7    384             SSB     #IRB$V_EOF,(R9)         ; set end of file bit
   0C A8    10 A9    D0   00DB    385 10$:        MOVL    IRB$L_IOS4(R9),RAB$L_STV(R8)
                          00E0    386                                             ; copy 2nd longword iosb to stv
      03 6A    3E    E0   00E0    387             BBS     #IFB$V_DAP,(R10),EXIT1  ; branch if network access
            10 A8    7C   00E4    388 EXIT:       CLRQ    RAB$L_RFA0(R8)          ; zero rfa
             FF16'   31   00E7    389 EXIT1:      BRW     RM$EXRMS
                          00EA    390
                          00EA    391 ;++
                          00EA    392 ;
                          00EA    393 ;  handle bad record neader buffer error
                          00EA    394 ;
                          00EA    395 ;--
                          00EA    396
                          00EA    397 ERRRHB:
                          00EA    398             RMSERR  RHB
         F3    11         00EF    399             BRB     EXIT
                          00F1    400
```

```
                     00F1   402
                     00F1   403 ;++
                     00F1   404 ;
                     00F1   405 ; network specific code to move record header (if vfc format) and data
                     00F1   406 ; record into one bdb buffer, note: size of header + record can not
                     00F1   407 ; exceed device buffer size ( = bdb buffer size) for this release!!!
                     00F1   408 ;
                     00F1   409 ;--
                     00F1   410
                     00F1   411 NTMOVE:
                     00F1   412
                     00F1   413 ;
                     00F1   414 ; check to see if record plus header fit
                     00F1   415 ; ifb$b_fsz is zero for non-vfc record types
                     00F1   416 ;
                     00F1   417
50   5F AA   9A      00F1   418          MOVZBL  IFB$B_FSZ(R10),R0          ; fixed size into r0
   50   56   A0      00F5   419          ADDW2   R6,R0                      ; total size in r0
        4D   1D      00F8   420          BVS     35$                        ; error if overflow on add
16 A4   50   B1      00FA   421          CMPW    R0,BDB$W_SIZE(R4)          ; fit in buffer?
        47   1A      00FE   422          BGTRU   35$                        ; error if record larger
        54   DD      0100   423          PUSHL   R4                         ; save bdb addr
53   18 A4   D0      0102   424          MOVL    BDB$L_ADDR(R4),R3          ; addr of buffer to move to
   50   AA   91      0106   425          CMPB    IFB$B_RFMORG(R10),-
        03           0109   426                   #FAB$C_VFC                ; is this vfc rec?
        29   12      010A   427          BNEQ    20$                        ; go move it
                     010C   428
                     010C   429 ;
                     010C   430 ; move header and record into one buffer and put as one record
                     010C   431 ;
                     010C   432
   57   53   D0      010C   433          MOVL    R3,R7                      ; save bdb buffer address
   7E   55   7D      010F   434          MOVQ    R5,-(SP)                   ; push rec addr and size onto stack
56   5F AA   9A      0112   435          MOVZBL  IFB$B_FSZ(R10),R6          ; size of record header
55   2C A8   D0      0116   436          MOVL    RAB$L_RHB(R8),R5           ; address of header buffer
        3B   13      011A   437          BEQL    50$                        ; branch if none spec'd
      FEE1'  30      011C   438          BSBW    RM$PROBEREAD               ; check out header buffer
     2C 50   E9      011F   439          BLBC    R0,40$                     ; branch if problems
63   65   56   28    0122   440          MOVC3   R6,(R5),(R3)               ; move header part
   55   8E   7D      0126   441 10$:     MOVQ    (SP)+,R5                   ; rec addr to r5, rec len to r6
     09 6B   E8      0129   442          BLBS    (R11),20$                  ; branch if not ppf
04 51 AA   02   E1   012C   443          BBC     #FAB$V_PRN,IFB$B_RAT(R10),20$
                     0131   444                                             ; branch if not a 'print' file
67   64 A9   B0      0131   445          MOVW    IRB$B_PRE_CCTL(R9),(R7)    ; jam pre/post carriage control into
                     0135   446                                             ;  record header if ppf and prn set
63   65   56   28    0135   447 20$:     MOVC3   R6,(R5),(R3)               ; move record beyond header
   54   8E   D0      0139   448          MOVL    (SP)+,R4                   ; get back bdb addr
55   18 A4   D0      013C   449          MOVL    BDB$L_ADDR(R4),R5          ; addr of buffer to r5
56   53   55   C3    0140   450          SUBL3   R5,R3,R6                   ; total length in r6
      FF75   31      0144   451          BRW     MOVDAT1                    ; rejoin mainline
                     0147   452
                     0147   453 ;
                     0147   454 ; error handling
                     0147   455 ;
                     0147   456
                     0147   457 35$:     RMSERR  RSZ                        ; record too big
        96   11      014C   458          BRB     EXIT                       ; exit
```

```
                                014E    459
                        07  BA  014E    460 40$:    POPR    #^M<R0,R1,R2>           ; clean off stack - not same regs
                                0150    461         RMSERR  RHB                     ; can not access rhb
                        8D  11  0155    462         BRB     EXIT                    ; exit
                                0157    463
    63  56  00  63     00  2C  0157    464 50$:    MOVC5   #0,(R3),#0,R6,(R3)      ; supply zero record header
                        C7  11  015D    465         BRB     10$                     ;
                                015F    466
                                015F    467         .END
```

```
$$.PSECT_EP                     = 00000000
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
BDB$B_PRE_CCTL                  = 0000004A
BDB$L_ADDR                      = 00000018
BDB$L_CURBUFADR                 = 0000004C
BDB$W_NUMB                      = 00000014
BDB$W_SIZE                      = 00000016
CCTL_TABLE                        00000000 R      01
CR                              = 0000000D
DEV$V_CCL                       = 00000001
DEV$V_FOR                       = 00000018
DEV$V_NET                       = 0000000D
DEV$V_SQD                       = 00000005
ERRRHB                            000000EA R      01
EXIT                              000000E4 R      01
EXIT1                             000000E7 R      01
FAB$C_MAXRFM                    = 00000006
FAB$C_STM                       = 00000004
FAB$C_STMCR                     = 00000006
FAB$C_STMLF                     = 00000005
FAB$C_VFC                       = 00000003
FAB$V_CR                        = 00000001
FAB$V_FTN                       = 00000000
FAB$V_PRN                       = 00000002
FF                              = 0000000C
FTN_REC                           000C008F R      01
IFB$B_FSZ                       = 0000005F
IFB$B_RAT                       = 00000051
IFB$B_RFMORG                    = 00000050
IFB$L_PRIM_DEV                  = 00000000
IFB$V_DAP                       = 0000003E
IMP$V_IIOS                      = 00000000
IMP$W_RMSSTATUS                 = 00000000
IRB$B_POST_CCTL                 = 00000065
IRB$B_PRE_CCTL                  = 00000064
IRB$L_CURBDB                    = 00000020
IRB$L_IOS4                      = 00000010
IRB$L_NXTBDB                    = 0000003C
IRB$V_EOF                       = 00000021
LF                              = 0000000A
LF_REC_CR                         000000AF R      01
MAPCTL                            00000025 R      01
MOVDAT                            000000B8 R      01
MOVDAT1                           000000BC R      01
MOVPREPOST                        000000B4 R      01
NOTPRN                            00000066 R      01
NTMOVE                            000000F1 R      01
PIO$A_TRACE                       ******** X      01
PRNFMT                            0000009D R      01
RAB$L_RFA0                      = 00000010
RAB$L_RHB                       = 0000002C
RAB$L_STV                       = 0000000C
RAB$S_PPF_RAT                   = 00000008
RAB$V_PPF_RAT                   = 00000006
RAB$W_ISI                       = 00000002
RM$EXRMS                          ********   X    01
RM$MAPFTN                         00000012 RG     01
RM$PROBEREAD                      ********   X    01
RM$PUT_UNIT_REC                   00000029 RG     01
RM$SEQ@TUR                        ********   X    01
RM$STM_TERM                       ********   X    01
RMS$_RAB                        = 0001866C
RMS$_RSZ                        = 000186A4
SPACE                           = 00000020
TPT$L_PUTREC1                     ********   X    01
VT                              = 0000000B
```

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	0000015F (351.)	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	00:00:00.85
Command processing	109	00:00:00.78	00:00:04.18
Pass 1	307	00:00:09.93	00:00:30.44
Symbol table sort	0	00:00:01.31	00:00:02.31
Pass 2	89	00:00:02.03	00:00:05.09
Symbol table output	8	00:00:00.09	00:00:00.64
Psect synopsis output	2	00:00:00.02	00:00:00.08
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	547	00:00:14.24	00:00:43.59

The working set limit was 1500 pages.
56424 bytes (111 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1066 non-local and 12 local symbols.
467 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
_255DUA28:[RMS.OBJ]RMS.MLB;1	11
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	18

1176 GETS were required to define 18 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:RM1PUTREC/OBJ=OBJ$:RM1PUTREC MSRC$:RM1PUTREC/UPDATE=(ENH$:RM1PUTREC)+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