```
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
```

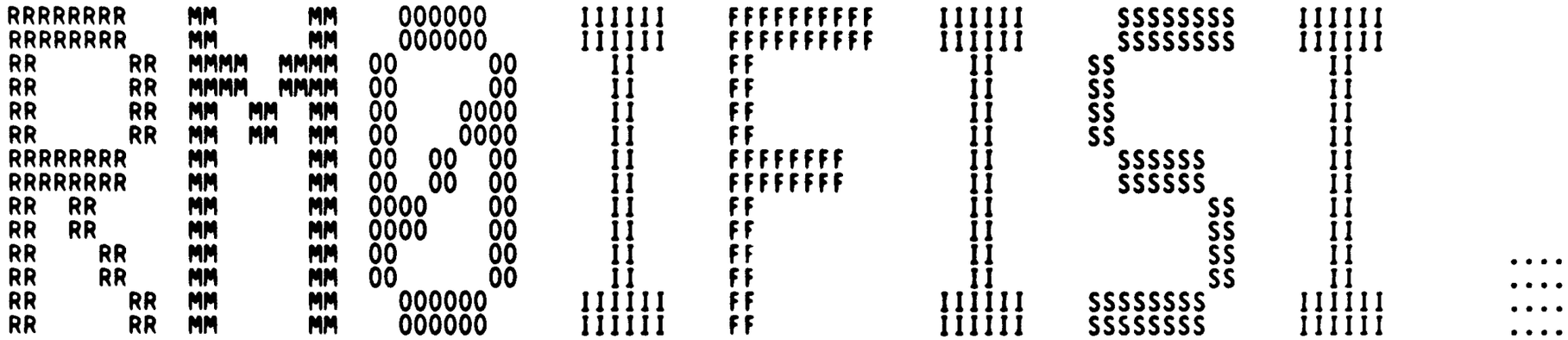
RRRRRRRR MM MM 000000 IIIIII FFFFFFFFFF IIIIII SSSSSSSS IIIIII
RRRRRRRR MM MM 000000 IIIIII FFFFFFFFFF IIIIII SSSSSSSS IIIIII
RR RR MMMM MMMM 00 00 II FF II SS II
RR RR MMMM MMMM 00 00 II FF II SS II
RR RR MM MM MM 00 0000 II FF II SS II
RR RR MM MM MM 00 0000 II FF II SS II
RRRRRRRR MM MM 00 00 00 II FFFFFFFF II SSSSSS II
RRRRRRRR MM MM 00 00 00 II FFFFFFFF II SSSSSS II
RR RR MM MM 0000 00 II FF II SS II
RR RR MM MM 0000 00 II FF II SS II
RR RR MM MM 00 00 II FF II SS II
RR RR MM MM 00 00 II FF II SS II
RR RR MM MM 000000 IIIIII FF IIIIII SSSSSSSS IIIIII
RR RR MM MM 000000 IIIIII FF IIIIII SSSSSSSS IIIIII
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          $BEGIN  RMOIFISI,000,RM$RMS0,<IFAB AND IRAB INDEX TABLE ROUTINES>,-
0000     2                                                <NOWRT,QUAD>
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; Facility: RMS32
0000    31 ;
0000    32 ; Abstract:
0000    33 ;                   This module contains the routines to allocate
0000    34 ;                   ifab and irab index table slots and to return
0000    35 ;                   the ifab or irab address from an ifi or isi
0000    36 ;                   value respectively.
0000    37 ;
0000    38 ; Environment:
0000    39 ;                   Star processor running Starlet exec.
0000    40 ;
0000    41 ; Author: L. F. Laverdure       Creation Date: 3-JAN-1977
0000    42 ;
0000    43 ; Modified By:
0000    44 ;
0000    45 ;       V03-005 LJA0050         Laurie J. Anderson      12-Jan-1983
0000    46 ;                   Fill in IFI/ISI field in IFB/IRB upon finding slot in
0000    47 ;                   table.
0000    48 ;                   Move popping of R6 (on restart) in rm$gtslt.  If error, on
0000    49 ;                   restart, R6 was not restored.
0000    50 ;
0000    51 ;       V03-004 LJA0024         Laurie Anderson         05-Oct-1982
0000    52 ;                   Fix some problems with restarting when the IFI/ISI is large
0000    53 ;                   enough to cause the allocation of a chain of index tables.
0000    54 ;
0000    55 ;       V03-003 LJA0018         Laurie Anderson         03-Sep-1982
0000    56 ;                   To enable the restart to re-open/re-connect to same IFI/ISI,
0000    57 ;                   modify RM$GTSLT.  If provided with an IFI/ISI on input, then
```

```
0000     58 ;                  use that one rather than the next available empty slot.
0000     59 ;                  If that slot can not be provided, then return an error.
0000     60 ;
0000     61 ;       V03-002 KBT0305         Keith B. Thompson       28-Aug-1982
0000     62 ;                  Reorganize psects
0000     63 ;
0000     64 ;       V03-001 KBT0102         Keith B. Thompson       13-Jul-1982
0000     65 ;                  Clean up psects
0000     66 ;
0000     67 ;       V02-013 CDS0001         C Saether               10-Dec-1981
0000     68 ;                  Rename psect.
0000     69 ;
0000     70 ;       V02-012 REFORMAT        K. E. Kinnear           31-Jul-1980     8:51
0000     71 ;
0000     72 ;--
0000     73
```

```
                                B 15
0000     75          .SBTTL  DECLARATIONS
0000     76
0000     77 ;
0000     78 ; Include Files:
0000     79 ;
0000     80
0000     81 ;
0000     82 ; Macros:
0000     83 ;
0000     84
0000     85          $FABDEF
0000     86          $RABDEF
0000     87          $IFBDEF
0000     88          $IRBDEF
0000     89          $IMPDEF
0000     90          $PSLDEF
0000     91          $RMSDEF
0000     92
0000     93 ;
0000     94 ; Equated Symbols:
0000     95 ;
0000     96
0000     97 ;
0000     98 ; Own Storage:
0000     99 ;
0000    100
```

```
                     0000    102          .SBTTL  RM$GTIADR - INDEX TO TABLE ADDRESS CONVERSION ROUTINE
                     0000    103
                     0000    104 ;++
                     0000    105 ; RM$GTIADR -- Index to Table Address Conversion Routine.
                     0000    106 ;
                     0000    107 ;  This subroutine converts an ifi or isi table index value
                     0000    108 ;  into the address of the related ifab or irab respectively.
                     0000    109 ;
                     0000    110 ; Calling sequence:
                     0000    111 ;
                     0000    112 ;       BSBW    RM$GTIADR
                     0000    113 ;
                     0000    114 ; Input Parameters:
                     0000    115 ;
                     0000    116 ;       R11     image impure area address
                     0000    117 ;       R9      index value (ifi or isi)
                     0000    118 ;       R7      mode of caller (only if ifi or isi indicates direct
                     0000    119 ;                               access to a process permanent file)
                     0000    120 ;       R0      IFAB or IRAB index table offset / 4
                     0000    121 ;
                     0000    122 ; Implicit Inputs:
                     0000    123 ;
                     0000    124 ;       none
                     0000    125 ;
                     0000    126 ; Output Parameters:
                     0000    127 ;
                     0000    128 ;       R9      IFAB or IRAB address
                     0000    129 ;       R0      destroyed
                     0000    130 ;
                     0000    131 ;    if ifi or isi < 0 and not user mode:
                     0000    132 ;
                     0000    133 ;       R11     process i/o impure area addr
                     0000    134 ;
                     0000    135 ; Implicit Outputs:
                     0000    136 ;
                     0000    137 ;       none
                     0000    138 ;
                     0000    139 ; Completion Codes:
                     0000    140 ;
                     0000    141 ;       Z-bit set if invalid ifi or isi, clear otherwise.
                     0000    142 ;
                     0000    143 ; Side Effects:
                     0000    144 ;
                     0000    145 ;       none
                     0000    146 ;
                     0000    147 ;--
                     0000    148
                     0000    149 RM$GTIADR::
           59    B5  0000    150          TSTW    R9                              ; non-standard ifi/isi?
           19    15  0002    151          BLEQ    50$                             ; branch if ppf format or zero
 50    6B40    D0  0004    152 5$:      MOVL    (R11)[R0],R0                    ; get table addr
20 AB    59    B1  0008    153 10$:     CMPW    R9,IMP$W_ENTPERSEG(R11)         ; in this table segment?
           05    1A  000C    154          BGTRU   30$                             ; branch if not
 59    6049    D0  000E    155          MOVL    (R0)[R9],R9                     ; set ifab/irab addr
                     0012    156                                                  ;  note: may be zero.
                05  0012    157          RSB
                     0013    158
```

```
                                0013   159 ;
                                0013   160 ;  The input ifi/isi value does not map to this segment
                                0013   161 ;  of the index table  -  try the next segment.
                                0013   162 ;
                                0013   163
          59   20 AB    A2   0013   164 30$:    SUBW2   IMP$W_ENTPERSEG(R11),R9 ; reduce the index value
               50   60    D0   0017   165         MOVL    (R0),R0                ; get link to next table seg
                    EC    12   001A   166         BNEQ    10$                    ; and keep trying, if any more
                          05   001C   167 40$:    RSB                            ; return with z-bit set
                               001D   168
                               001D   169 ;
                               001D   170 ;  A negative isi or ifi has been input.
                               001D   171 ;
                               001D   172 ;  2 valid cases exist, depending upon the state of bit 14:
                               001D   173 ;       - (bit 14=0) - a non-user mode caller desires to operate
                               001D   174 ;                      on the ppf directly
                               001D   175 ;       - (bit 14=1) - a caller desires to operate on the ppf indirectly
                               001D   176 ;
                               001D   177 ;  If access allowed switch to the process permanent files ifab or
                               001D   178 ;  irab table in the process i/o segment.
                               001D   179 ;
                               001D   180
                    FD    13   001D   181 50$:    BEQL    40$                    ; branch if zero ifi/isi (error)
               05 59    0E   E0   001F   182         BBS     #FAB$V_PPF_IND,R9,55$  ; branch if indirect access
                  03    57   91   0023   183         CMPB    R7,#PSL$C_USER         ; direct access attempt from user mode?
                        0E   13   0026   184         BEQL    60$                    ; error if user
     5B   00000000'9F    DE   0028   185 55$:    MOVAL   @#PIO$GW_PIOIMPA,R11   ; switch impure pointers
          59   FFC0 8F    AA   002F   186         BICW2   #<<1@RAB$V_PPF_RAT>-1> \ <<1@16>-1>,R9; clear all but table index
                        CE   12   0034   187         BNEQ    5$                     ; and continue if non-zero
                               0036   188
                               0036   189 ;
                               0036   190 ;  User attempting direct access to a process permanent file
                               0036   191 ;  or has a bad isi value.
                               0036   192 ;
                               0036   193
     59                   D4   0036   194 60$:    CLRL    R9                     ; it's a no-no
                          05   0038   195         RSB
```

```
                     0039   197          .SBTTL  RM$GTSLT - INDEX TABLE SLOT ALLOCATION ROUTINE
                     0039   198
                     0039   199 ;++
                     0039   200 ; RM$GTSLT -- Index Table Slot Allocation Routine.
                     0039   201 ;
                     0039   202 ;  This routine scans the ifab or irab index table for
                     0039   203 ;  an empty slot and if found sets the slot to the associated
                     0039   204 ;  ifab or irab address, returning the index value of the slot.
                     0039   205 ;
                     0039   206 ;  If an IFI/ISI is provided on input, this routine will attempt to
                     0039   207 ;  allocate that slot in the IFAB/IRAB index table.  If that slot is
                     0039   208 ;  not available (due to DME or slot already in use), then an error is
                     0039   209 ;  returned.
                     0039   210 ;
                     0039   211 ;  Calling Sequence:
                     0039   212 ;
                     0039   213 ;       BSBW    RM$GTSLT
                     0039   214 ;
                     0039   215 ;  Input Parameters:
                     0039   216 ;
                     0039   217 ;       R11     process or image impure area address
                     0039   218 ;       R9      IFAB or IRAB address
                     0039   219 ;       R5      IFAB or IRAB index table start address
                     0039   220 ;
                     0039   221 ;  If this is a restart operation, the following additional inputs:
                     0039   222 ;       R6      IFI/ISI (the desired particular slot)
                     0039   223 ;
                     0039   224 ;  Implicit Inputs:
                     0039   225 ;
                     0039   226 ;       none
                     0039   227 ;
                     0039   228 ;  Output Parameters:
                     0039   229 ;
                     0039   230 ;       R6      table index value (i.e., ifi or isi)
                     0039   231 ;       R0      status code
                     0039   232 ;       R1,R2,R3,R4,R5 destroyed
                     0039   233 ;
                     0039   234 ;  Implicit Outputs:
                     0039   235 ;
                     0039   236 ;       The allocated table slot is set to the contents of r9.
                     0039   237 ;
                     0039   238 ;  Completion Codes:
                     0039   239 ;
                     0039   240 ;       Standard RMS, in particular, success or dme.
                     0039   241 ;
                     0039   242 ;  Side Effects:
                     0039   243 ;
                     0039   244 ;       none
                     0039   245 ;
                     0039   246 ;--
                     0039   247
                     0039   248 RM$GTSLT::
                     0039   249          ASSUME  IFB$V_RESTART EQ IRB$V_RESTART
  1C 69   3B   E0    0039   250          BBS     #IFB$V_RESTART,(R9),23$ ; Branch if restart operation
                 56  D4  003D   251          CLRL    R6                      ; build index value here
51   55   04   C1    003F   252 10$:      ADDL3   #4,R5,R1                ; R1 points to first table entry,
                     0043   253                                          ;  while saving R5 - pointing to link
```

```
                50    20 AB    3C   0043    254          MOVZWL  IMP$W_ENTPERSEG(R11),R0 ; # entries per table segment
                   56    D6   0047    255 20$:     INCL    R6                      ; bump index
                   81    D5   0049    256          TSTL    (R1)+                   ; zero slot?
                   55    13   004B    257          BEQL    50$                     ; branch if yes - go use it
                F7 50    F5   004D    258          SOBGTR  R0,20$                  ; keep scanning segment
                            0050    259
                            0050    260 ;
                            0050    261 ;  No free slots this table segment  -  try next if any.
                            0050    262 ;
                            0050    263
                   65    D5   0050    264          TSTL    (R5)                    ; another segment?
                   28    13   0052    265          BEQL    30$                     ; branch if none
                55 65    D0   0054    266          MOVL    (R5),R5                 ; update link
                   E6    11   0057    267          BRB     10$
                            0059    268 ;
                            0059    269 ;  If this is a restart operation, attempt to allocate the same slot in the
                            0059    270 ;  index table.
                            0059    271 ;
                            0059    272
                   56    DD   0059    273 23$:     PUSHL   R6                      ; Save the desired IFI/ISI
             51    55 04    C1   005B    274          ADDL3   #4,R5,R1                ; R1 points to first table entry,
                            005F    275                                             ;  while saving R5 - pointing to link
                20 AB 56    B1   005F    276 25$:     CMPW    R6,IMP$W_ENTPERSEG(R11) ; Is this IFI/ISI in this table segment?
                      17    1A   0063    277          BGTRU   30$                     ; Branch if not in this segment
                   6546    D5   0065    278          TSTL    (R5)[R6]                ; Is this slot already being used?
                      5D   12   0068    279          BNEQ    ERRRST                  ; Used - return error
                6546  59    D0   006A    280          MOVL    R9,(R5)[R6]             ; Put IFAB/IRAB address in table slot
                   22 AB    B6   006E    281          INCW    IMP$W_NUM_IFABS(R11)    ; count # ifabs in use
                      3C    11   0071    282          BRB     60$                     ; Return success
                            0073    283 ;
                            0073    284 ; Had to allocate more table segment space.  See if index maps into this segment
                            0073    285 ;
                            0073    286
             56    20 AB    A2   0073    287 27$:     SUBW2   IMP$W_ENTPERSEG(R11),R6 ; reduce the index value
                55    65    D0   0077    288          MOVL    (R5),R5                 ; Update the index table links
                      E3    11   007A    289          BRB     25$                     ; try next table segment
                            007C    290
```

```
                                007C   232
                                007C   233 ;
                                007C   234 ;  No free slots in existing segments.
                                007C   235 ;  If process i/o segment, return an error, otherwise
                                007C   236 ;  add a segment to the table.
                                007C   237 ;
                                007C   238
                                007C   239          ASSUME  IMP$W_RMSSTATUS EQ 0
                                007C   240          ASSUME  IMP$V_IIOS EQ 0
                                007C   241
                   41 6B   E9   007C   242 30$:     BLBC    (R11),ERRDME            ; branch if pio segment
               51  00000000'9F DE   007F   243          MOVAL   @#PIO$GL_IIOFSPLH,R1    ; impure area addr
                52    20 AB   3C   0086   244          MOVZWL  IMP$W_ENTPERSEG(R11),R2 ; # entries
                         52   D6   008A   245          INCL    R2                      ; +1 = # longwords
                52   52   02   78   008C   246          ASHL    #2,R2,R2                ; = # bytes required
                        FFF0' 30   0090   247          BSBW    RM$GETSPC_ALT           ; get this much space
                      1C 50   E9   0093   248          BLBC    R0,RETURN               ; branch if no space
                                0096   249
                                0096   250 ;
                                0096   251 ;  Link in the new segment (address of space returned in R1 by get_spc)
                                0096   252 ;  and go back and use it.
                                0096   253 ;
                                0096   254
                   65   51   D0   0096   255          MOVL    R1,(R5)                 ; update the old link
                   51   04   C0   0099   256          ADDL2   #4,R1                   ; and point R1 to the table
                D3 69   3B   E0   009C   257          BBS     #IFB$V_RESTART,(R9),27$ ; Branch if restart
                         A5   11   00A0   258          BRB     20$                     ; and go use it
                                00A2   259
                                00A2   260 ;
                                00A2   261 ;  Found a free slot.
                                00A2   262 ;  Store the IFAB or IRAB address in the slot and return.
                                00A2   263 ;  Store the IFI/ISI value into the IFAB/IRAB
                                00A2   264 ;
                                00A2   265
             71    59   D0   00A2   266 50$:     MOVL    R9,-(R1)                ; store structure address
               22 AB   B6   00A5   267          INCW    IMP$W_NUM_IFABS(R11)    ; count # ifabs in use
                                00A8   268
                                00A8   269 ;
                                00A8   270 ;  Flag the ifi or isi value if this is the process i/o segment.
                                00A8   271 ;
                                00A8   272
                                00A8   273          ASSUME  IMP$W_RMSSTATUS EQ 0
                                00A8   274          ASSUME  IMP$V_IIOS EQ 0
                                00A8   275
                04 6B   E8   00A8   276          BLBS    (R11),60$               ; branch if image i/o segment
                         56   D5   00AB   277          SSB     #15,R6                  ; set pio segment flag
                                00AF   278 60$:
                                00AF   279          ASSUME  IFB$W_IFI EQ IRB$W_ISI
                                00AF   280          RMSSUC
                                00B2   281 RETURN:
          03 69   3B   E1   00B2   282          BBC     #IFB$V_RESTART,(R9),80$ ; If not restarting, do not pop
                   56 8ED0   00B6   283          POPL    R6                      ; Restore the IFI/ISI
          28 A9    56   90   00B9   284 80$:     MOVB    R6,IFB$W_IFI(R9)        ; Fill in IFI/ISI value into IFAB/IRAB
                   50   D5   00BD   285          TSTL    R0                      ; Set condition code for return
                         05   00BF   286          RSB
                                00C0   287
                                00C0   288 ;
```

```
                     00C0   349 ;  Couldn't allocate another page for new IRAB or IFAB
                     00C0   350 ;  table segment because in process i/o segment.
                     00C0   351 ;
                     00C0   352
                     00C0   353 ERRDME:
                     00C0   354          RMSERR  DME
           EB   11   00C5   355          BRB     RETURN                  ; return - restore R7 first
                     00C7   356
                     00C7   357 ;
                     00C7   358 ; Indicate that the index table slot has already been used.
                     00C7   359 ;
                     00C7   360 ERRRST:
                     00C7   361          ASSUME  <IFB$C_BID&1> EQ 1              ;  in case it's really irab
                     00C7   362          ASSUME  <IRB$C_BID&1> EQ 0
                     00C7   363          ASSUME  IFB$B_BID EQ IRB$B_BID
07 08 A9   E8        00C7   364          BLBS    IFB$B_BID(R9),100$              ; Do we have a ifab or irab
                     00CB   365          RMSERR  ISI                             ; IRAB - Indicate ISI error
           E0   11   00D0   366          BRB     RETURN                          ; return - restore R7 first
                     00D2   367 100$:    RMSERR  IFI                             ; IFAB - Indicate IFI error
           D9   11   00D7   368          BRB     RETURN                          ; return - restore R7 first
                     00D9   369
                     00D9   370          .END
```

```
$$.PSECT_EP                     = 00000000
$$RMSTEST                       = 0000001A
$$RMS_PBUGCHK                   = 00000010
$$RMS_TBUGCHK                   = 00000008
$$RMS_UMODE                     = 00000004
ERRDME                            000000C0 R      01
ERRRST                            000000C7 R      01
FAB$V_PPF_IND                   = 0000000E
IFB$B_BID                       = 00000008
IFB$C_BID                       = 0000000B
IFB$V_RESTART                   = 0000003B
IFB$W_IFI                       = 00000028
IMP$V_IIOS                      = 00000000
IMP$W_ENTPERSEG                 = 00000020
IMP$W_NUM_IFABS                 = 00000022
IMP$W_RMSSTATUS                 = 00000000
IRB$B_BID                       = 00000008
IRB$C_BID                       = 0000000A
IRB$V_RESTART                   = 0000003B
IRB$W_ISI                       = 00000028
PIO$GL_IIOFSPLH                   ********    X   01
PIO$GW_PIOIMPA                    ********    X   01
PSL$C_USER                      = 00000003
RAB$V_PPF_RAT                   = 00000006
RETURN                            000000B2 R      01
RMS$GETSPC_ALT                    ********      X 01
RMS$GTIADR                        00000000 RG     01
RMS$GTSLT                         00000039 RG     01
RMS$_DME                        = 000184D4
RMS$_IFI                        = 00018564
RMS$_ISI                        = 00018584
```

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$RMSO	000000D9 (217.)	01 (1.)	PIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC QUAD
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	35	00:00:00.97	00:00:01.12
Command processing	125	00:00:00.96	00:00:03.89
Pass 1	307	00:00:08.76	00:00:24.91
Symbol table sort	0	00:00:01.17	00:00:01.91
Pass 2	78	00:00:01.75	00:00:04.32
Symbol table output	4	00:00:00.06	00:00:00.06
Psect synopsis output	2	00:00:00.01	00:00:00.17
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	553	00:00:12.49	00:00:36.39

The working set limit was 1500 pages.
50127 bytes (98 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 958 non-local and 18 local symbols.
370 source lines were read in Pass 1, producing 13 object records in Pass 2.
21 pages of virtual memory were used to define 20 macros.

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

Macro library name	Macros defined
_255DUA28:[RMS.OBJ]RMS.MLB;1	10
_255DUA28:[SYS.OBJ]LIB.MLB;1	1
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	16

1077 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

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

RM0EXTRMS
LIS

RM0DIRSCN
LIS

RM0CRECOM
LIS

RM0FABCHK
LIS

RM0CHKSUM
LIS

RM0FWASET
LIS

RM0EXTEND
LIS

RM0IFISI
LIS

RM0JOURNL
LIS

RM0FSETI
LIS

RM0FSET
LIS

RM0COMCLN
LIS

RM0DUMMY
LIS

RM0FILFNC
LIS