```
RRRRRRRRRRR    MMM        MMM    SSSSSSSSSSSS
RRRRRRRRRRR    MMM        MMM    SSSSSSSSSSSS
RRRRRRRRRRR    MMM        MMM    SSSSSSSSSSSS
RRR       RRR  MMMMMM  MMMMMM  SSS
RRR       RRR  MMMMMM  MMMMMM  SSS
RRR       RRR  MMMMMM  MMMMMM  SSS
RRR       RRR  MMM  MMM  MMM   SSS
RRR       RRR  MMM  MMM  MMM   SSS
RRR       RRR  MMM  MMM  MMM   SSS
RRRRRRRRRRR    MMM        MMM    SSSSSSSSS
RRRRRRRRRRR    MMM        MMM    SSSSSSSSS
RRRRRRRRRRR    MMM        MMM    SSSSSSSSS
RRR   RRR      MMM        MMM          SSS
RRR   RRR      MMM        MMM          SSS
RRR   RRR      MMM        MMM          SSS
RRR     RRR    MMM        MMM          SSS
RRR     RRR    MMM        MMM          SSS
RRR     RRR    MMM        MMM          SSS
RRR       RRR  MMM        MMM    SSSSSSSSSSSS
RRR       RRR  MMM        MMM    SSSSSSSSSSSS
RRR       RRR  MMM        MMM    SSSSSSSSSSSS
```

```
RRRRRRR     MM      MM   000000   FFFFFFFFF    AAAAAA    BBBBBBBB    CCCCCCC  HH      HH  KK       KK
RRRRRRR     MM      MM   000000   FFFFFFFFF    AAAAAA    BBBBBBBB    CCCCCCC  HH      HH  KK       KK
RR    RR    MMMM  MMMM   00    00 FF          AA    AA   BB    BB   CC        HH      HH  KK      KK
RR    RR    MMMM  MMMM   00    00 FF          AA    AA   BB    BB   CC        HH      HH  KK     KK
RR    RR    MM MM MM MM  00  0000 FF          AA    AA   BB    BB   CC        HH      HH  KK    KK
RR    RR    MM MM MM MM  00  0000 FF          AA    AA   BB    BB   CC        HH      HH  KK   KK
RRRRRRR     MM      MM   00    00 FFFFFFF      AA    AA   BBBBBBB    CC        HHHHHHHHHH  KKKKK
RRRRRRR     MM      MM   00    00 FFFFFFF      AA    AA   BBBBBBB    CC        HHHHHHHHHH  KKKKK
RR  RR      MM      MM   0000  00 FF          AAAAAAAAA   BB    BB   CC        HH      HH  KK   KK
RR  RR      MM      MM   0000  00 FF          AAAAAAAAA   BB    BB   CC        HH      HH  KK   KK
RR    RR    MM      MM   00    00 FF          AA    AA   BB    BB   CC        HH      HH  KK    KK       ....
RR    RR    MM      MM   00    00 FF          AA    AA   BB    BB   CC        HH      HH  KK    KK       ....
RR    RR    MM      MM   000000  FF          AA    AA   BBBBBBB   CCCCCCC    HH      HH  KK     KK       ....
RR    RR    MM      MM   000000  FF          AA    AA   BBBBBBB   CCCCCCC    HH      HH  KK      KK      ....

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
```

```
0000     1              $BEGIN  RMOFABCHK,000,RM$RMS0,<COMMON FAB CHECKING>
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     ;++
0000    29     ; Facility: rms32
0000    30     ;
0000    31     ; Abstract:
0000    32     ;               this routine performs common fab call argument
0000    33     ;               list and fab validation.
0000    34     ;
0000    35     ; Environment:
0000    36     ;               star processor running starlet exec.
0000    37     ;
0000    38     ; Author: L F Laverdure,         creation date: 4-JAN-1977
0000    39     ;
0000    40     ; Modified By:
0000    41     ;
0000    42     ;       V03-004 DGB0043         Donald G. Blair         02-May-1984
0000    43     ;               If the PIO$V_INHAST bit is set when we start an
0000    44     ;               RMS operation, we conclude that the caller must be
0000    45     ;               at exec AST level or higher and that he would break
0000    46     ;               RMS synchronization rules if allowed to continue.
0000    47     ;               Return Error.
0000    48     ;
0000    49     ;       V03-003 RAS0171         Ron Schaefer            19-Jul-1983
0000    50     ;               Change RAS0162 to a new specific structure-less
0000    51     ;               error.
0000    52     ;
0000    53     ;       V03-002 RAS0162         Ron Schaefer            17-Jun-1983
0000    54     ;               Detect and report the AST/non-AST caller's mode
0000    55     ;               wait hang condition, by checking the low bit of the
0000    56     ;               FAB's BLN field.
0000    57     ;
```

RMOFABCHK
VO4-000

COMMON FAB CHECKING

G 10

16-SEP-1984 00:21:02   VAX/VMS Macro VO4-00
5-SEP-1984 16:21:44   [RMS.SRC]RMOFABCHK.MAR;1

Page 2
(1)

RM
VO

```
0000    58 ;          V03-001 KBT0206        Keith B. Thompson      23-Aug-1982
0000    59 ;                  Reorganize psects
0000    60 ;
0000    61 ;          V007    REFORMAT        Ken Henderson   29-JUL-1980      15:42
0000    62 ;                  code was reformatted
0000    63 ;
0000    64 ;--
0000    65
```

H 10

RMOFABCHK   COMMON FAB CHECKING   16-SEP-1984 00:21:02 VAX/VMS Macro V04-00   Page 3   RM
V04-000   DECLARATIONS   5-SEP-1984 16:21:44   [RMS.SRC]RMOFABCHK.MAR;1   (2)   V0

```
0000    67              .SBTTL   DECLARATIONS
0000    68
0000    69  ;
0000    70  ; Include Files:
0000    71  ;
0000    72
0000    73  ;
0000    74  ; Macros:
0000    75  ;
0000    76
0000    77              $FABDEF
0000    78              $PIODEF
0000    79              $PSLDEF
0000    80              $RMSDEF
0000    81
0000    82  ;
0000    83  ; Equated Symbols:
0000    84  ;
0000    85
0000    86  ;
0000    87  ; Own Storage:
0000    88  ;
0000    89
```

I 10

RMOFABCHK   COMMON FAB CHECKING   16-SEP-1984 00:21:02 VAX/VMS Macro V04-00   Page 4   RM
V04-000   RM$FABCHK - COMMON ARGLIST AND FAB VALID 5-SEP-1984 16:21:44   [RMS.SRC]RMOFABCHK.MAR;1   (3)   V0

```
0000    91              .SBTTL   RM$FABCHK - COMMON ARGLIST AND FAB VALIDATION ROUTINE
0000    92
0000    93      ;++
0000    94      ; RM$FABCHK
0000    95      ;
0000    96      ;   this routine performs the following functions:
0000    97      ;
0000    98      ;           1. setup r11 to point to the image i/o impure area
0000    99      ;              (it may be changed in fset to the process i/o impure area)
0000    100     ;           2. check argument list for accessibility and validity
0000    101     ;           3. check fab for accessibility and validity
0000    102     ;           4. set r8 to address of fab
0000    103     ;           5. clear sts and stv
0000    104     ;           6. inhibit rms internal asts
0000    105     ;           7. set r9 to the value of ifi
0000    106     ;
0000    107     ;
0000    108     ;
0000    109     ; Calling sequence:
0000    110     ;
0000    111     ;           bsbw    rm$fabchk
0000    112     ;
0000    113     ; Input Parameters:
0000    114     ;
0000    115     ;           ap      arglist addr
0000    116     ;
0000    117     ; Implicit Inputs:
0000    118     ;
0000    119     ;           the contents of the arglist and the bid, bln, and ifi fields
0000    120     ;           of the fab.
0000    121     ;
0000    122     ; Output Parameters:
0000    123     ;
0000    124     ;           r11     impure area address
0000    125     ;           r9      ifi value
0000    126     ;           r8      fab address
0000    127     ;           r7      mode of caller
0000    128     ;
0000    129     ; Implicit Outputs:
0000    130     ;
0000    131     ;           the sts and stv fields of the fab are zeroed.
0000    132     ;
0000    133     ; Completion Codes:
0000    134     ;
0000    135     ;           z-bit set if ifi = zero, else clear.
0000    136     ;           if any errors, the rms error code is set into r0
0000    137     ;           and return is made to the user (not caller).
0000    138     ;
0000    139     ; Side Effects:
0000    140     ;
0000    141     ;           rms internal asts are inhibited.
0000    142     ;
0000    143     ;--
0000    144
```

J 10

RMOFABCHK   COMMON FAB CHECKING   16-SEP-1984 00:21:02 VAX/VMS Macro V04-00   Page 5   RM
V04-000   RMSFABCHK - COMMON ARGLIST AND FAB VALID 5-SEP-1984 16:21:44 [RMS.SRC]RMOFABCHK.MAR;1   (5)   V0

```
                                 0000    146
                                 0000    147  ;
                                 0000    148  ; set up pointer to impure area based on the mode of the caller
                                 0000    149  ;
                                 0000    150
                                 0000    151  RMSFABCHK::
                    5B    DC     0000    152          MOVPSL  R11                         ; get psl
        57  5B  02  16    EF     0002    153          EXTZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R11,R7
                                 0007    154                                              ; extract the previous mode
        5B      00000000'9F  DE  0007    155          MOVAL   a#PIO$GW_IIOIMPA,R11        ; image io impure area addr
                                 000E    156
                                 000E    157  ;
                                 000E    158  ; perform accessibility checks
                                 000E    159  ;
                                 000E    160
            58    04 AC    D0    000E    161          MOVL    4(AP),R8                    ; get fab address
                                 0012    162          IFNOWRT #FAB$C_BLN,(R8),ERRFAB      ; fab writeable?
                                 001A    163          ASSUME  FAB$B_BID EQ 0
                03    68    91   001A    164          CMPB    (R8),#FAB$C_BID             ; is it a fab?
                      29    12   001D    165          BNEQ    ERRFAB
            50 8F  01 A8    91   001F    166          CMPB    FAB$B_BLN(R8),#FAB$C_BLN;   is it long enough?
                      1B    1F   0024    167          BLSSU   ERRBLN
            10 01 A8    E8       0026    168          BLBS    FAB$B_BLN(R8),ERRACT        ; is this FAB busy?
                                 002A    169                                              ; continue if not
                                 002A    170
                                 002A    171  ;
                                 002A    172  ; zero the sts and stv fab fields
                                 002A    173  ;
                                 002A    174
                                 002A    175          ASSUME  FAB$L_STS+4 EQ FAB$L_STV
            08 A8    7C          002A    176          CLRQ    FAB$L_STS(R8)
                                 002D    177
                                 002D    178  ;
                                 002D    179  ; Disable AST's.  If the PIO$V_INHAST bit is already set, we
                                 002D    180  ; conclude that the caller must be at exec ast level or higher
                                 002D    181  ; (otherwise, he could not have kicked off an RMS operation
                                 002D    182  ; while RMS was already in progress) and would break RMS
                                 002D    183  ; synchronization rules if allowed to continue.  Return RMS$_BUSY
                                 002D    184  ; status when this happens.
                                 002D    185  ;
                                 002D    186
                    00    E2     002D    187          BBSS    #PIO$V_INHAST,-             ; set inhast bit.  err if already set.
        05  00000000'9F          002F    188                  a#PIO$GW_STATUS,ERRACT
            59    02 A8    3C    0035    189          MOVZWL  FAB$W_IFI(R8),R9            ; set r9 = ifi value
                      05         0039    190          RSB
                                 003A    191
                                 003A    192
                                 003A    193  ;
                                 003A    194  ; an error has occurred in validating the argument list or fab
                                 003A    195  ;
                                 003A    196  ; since an error code cannot be safely stored in the fab,
                                 003A    197  ; no attempt to generate an err= ast will be made.
                                 003A    198  ; r0 will be set to the appropriate error code and an
                                 003A    199  ; exception, if enabled, will be generated upon ret.
                                 003A    200  ;
                                 003A    201  ;
                                 003A    202
```

RMOFABCHK
V04-000

K 10

COMMON FAB CHECKING   16-SEP-1984 00:21:02 VAX/VMS Macro V04-00   Page 6
RMSFABCHK - COMMON ARGLIST AND FAB VALID 5-SEP-1984 16:21:44 [RMS.SRC]RMOFABCHK.MAR;1   (5)

```
           003A     203 ERRACT:
           003A     204         RMSERR   BUSY                           ; Syncrhonization problem
 0C   11   003F     205         BRB      BASIC_ERR
           0041     206
           0041     207 ERRBLN:
           0041     208         RMSERR   BLN                            ; invalid block length
 05   11   0046     209         BRB      BASIC_ERR
           0048     210
           0048     211 ERRFAB:
           0048     212         RMSERR   FAB                            ; invalid fab
           004D     213
           004D     214 BASIC_ERR:
           004D     215         SSB      #16,R0                         ; prefix the facility code
           0051     216                                                 ; ... to the error code
 04   0051     217         RET                                          ; and return to caller
           0052     218
           0052     219         .END
```

L 10

RMOFABCHK   COMMON FAB CHECKING   16-SEP-1984 00:21:02   VAX/VMS Macro V04-00   Page   7
Symbol table   5-SEP-1984 16:21:44   [RMS.SRC]RMOFABCHK.MAR;1   (5)

```
$$.PSECT_EP             = 00000000
$$RMSTEST              = 0000001A
$$RMS_PBUGCHK          = 00000010
$$RMS_TBUGCHK          = 00000008
$$RMS_UMODE            = 00000004
BASIC_ERR                0000004D R      01
ERRACT                   0000003A R      01
ERRBLN                   00000041 R      01
ERRFAB                   00000048 R      01
FAB$B_BID              = 00000000
FAB$B_BLN              = 00000001
FAB$C_BID              = 00000003
FAB$C_BLN              = 00000050
FAB$L_STS              = 00000008
FAB$L_STV              = 0000000C
FAB$W_IFI              = 00000002
PIO$GQ_IIOIMPA           ******** X      01
PIO$GW_STATUS            ******** X      01
PIO$V_INHAST           = 00000000
PSL$S_PRVMOD           = 00000002
PSL$V_PRVMOD           = 00000016
RM$FABCHK                00000000 RG     01
RMS$_BLN              = 0001842C
RMS$_BUSY             = 0001848C
RMS$_FAB              = 0001850C
```

+------------------+
! 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$RMS0	00000052 (	82.)	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	00:00:01.21
Command processing	115	00:00:00.74	00:00:06.48
Pass 1	206	00:00:04.66	00:00:15.03
Symbol table sort	0	00:00:00.39	00:00:00.64
Pass 2	52	00:00:01.01	00:00:02.53
Symbol table output	5	00:00:00.05	00:00:00.37
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	410	00:00:06.97	00:00:26.28

The working set limit was 1350 pages.
24174 bytes (48 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 453 non-local and 1 local symbols.
219 source lines were read in Pass 1, producing 13 object records in Pass 2.
18 pages of virtual memory were used to define 17 macros.

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

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                  7
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                  1
_$255$DUA28:[SYSLIB]STARLET.MLB;2               5
TOTALS (all libraries)                         13
```

558 GETS were required to define 13 macros.

There were no errors, warnings or information messages.

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