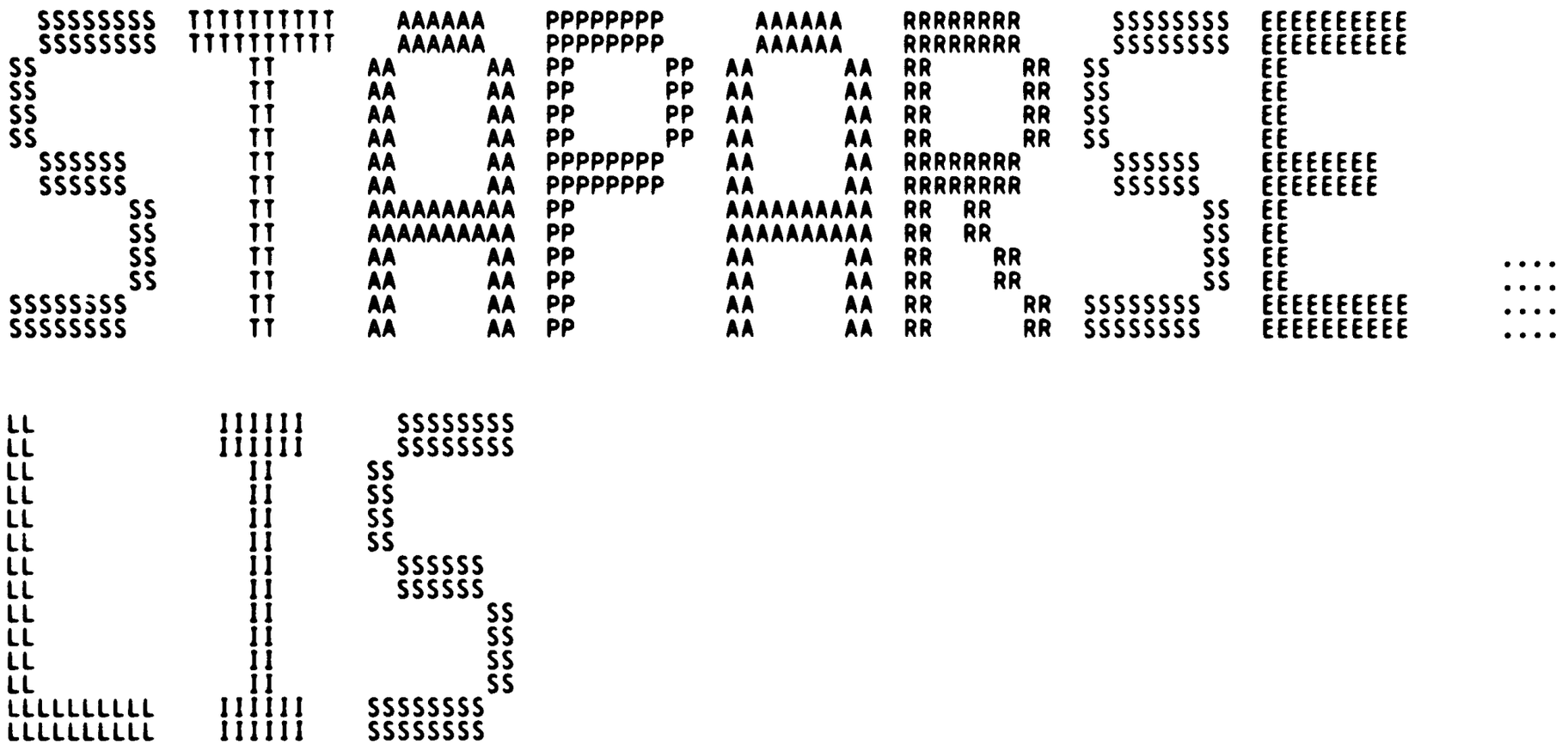
```
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK          KKK  UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK          KKK  UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK          KKK  UUU          UUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK          KKK  UUUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK          KKK  UUUUUUUUUUUUUUUU  PPP
```

FILEID**STAPARSE



```
0000     1          .TITLE  STAPARSE       Standalone $PARSE subroutines
0000     2          .IDENT  'V04-000'
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  ; FACILITY:
0000    30  ;       Backup/Restore
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;       This module contains the external subroutines and data required for
0000    34  ;       RMS PARSE and XPFN to operate.
0000    35  ;
0000    36  ; ENVIRONMENT:
0000    37  ;       VAX/VMS user mode.
0000    38  ;
0000    39  ;
0000    40  ;--
0000    41  ;
0000    42  ; AUTHOR: M. Jack, CREATION DATE: 27-Dec-1980
0000    43  ;
0000    44  ;
0000    45  ; MODIFIED BY:
0000    46  ;
0000    47  ;       V03-013 RAS0293         Ron Schaefer            15-Apr-1984
0000    48  ;               Add RM$RETPAG entry point and dummy entries for
0000    49  ;               RM$SLIST_ERR and RM$SLIST_ERR_CNT.
0000    50  ;
0000    51  ;       V03-012 DAS0001         David Solomon           29-Feb-1984
0000    52  ;               Add dummy entry point RM$FSET_ALT to track RMSOPARSE change.
0000    53  ;
0000    54  ;       V03-011 RAS0221         Ron Schaefer             9-Dec-1983
0000    55  ;               Add dummy entry points:  RM$DEACCESS, RM$GET1PAG, RM$RET1PAG,
0000    56  ;               RM$GETBLK1 in order to track RMS.
0000    57  ;
```

```
0000     58 ;       V03-010 RAS0212         Ron Schaefer            16-Nov-1983
0000     59 ;               Fix version/implementation skew in RM$ASSIGN by moving it
0000     60 ;               to a conditionalized version of the RMS module RMOPRFLNM
0000     61 ;               for both RMS and stand-alone BACKUP.
0000     62 ;
0000     63 ;       V03-009 JWT0136         Jim Teague              19-Sep-1983
0000     64 ;               Be sure to clean out IFI between calls to FABCHK,
0000     65 ;               also can remove RM$FSETI entry point.
0000     66 ;
0000     67 ;       V03-008 KBT0529         Keith B. Thompson       31-May-1983
0000     68 ;               Change PSECT RM$RMS to PIC and add RM$GETBLK and RM$RETBLK
0000     69 ;
0000     70 ;       V03-007 ACG0332         Andrew C. Goldstein,    5-May-1983  15:30
0000     71 ;               Track changes in RMS testpoints
0000     72 ;
0000     73 ;       V03-006 RAS0127         Ron Schaefer     4-Mar-1983
0000     74 ;               Add global symbol RM$RETSPC.
0000     75 ;
0000     76 ;       V03-005 RAS0125         Ron Schaefer    21-Feb-1983
0000     77 ;               Add global symbol RM$GETSPC.
0000     78 ;
0000     79 ;       V03-004 RAS0111         Ron Schaefer    22-Dec-1982
0000     80 ;               Change symbols:
0000     81 ;                       FWA$T_SHRFILDEV  -> FWA$T_SHRFILBUF
0000     82 ;                       IFB$W_DEVBUFSIZ  -> IFB$L_DEVBUFSIZ
0000     83 ;
0000     84 ;       V03-003 RAS0098         Ron Schaefer    16-Sep-1982
0000     85 ;               Change PSECT attributes of RM$RMS to match RMS.
0000     86 ;
0000     87 ;       V03-002 RAS0097         Ron Schaefer    14-Sep-1982
0000     88 ;               Track RMS changes:
0000     89 ;               1.  Change spelling of RMS$$PARSE to RMS$PARSE.
0000     90 ;               2.  Add RM$FSETI_ALT entry point.
0000     91 ;               3.  Change code PSECT to RM$RMS to prevent truncation errors.
0000     92 ;
0000     93 ;       V03-001 MLJ0082         Martin L. Jack, 15-Mar-1982  15:48
0000     94 ;               Set DEV$V_FOR and DEV$V_MNT to correct problem with mounting of
0000     95 ;               continuation tapes.  Initialize FWA$T_SHRFILDEV in RM$ASSIGN to
0000     96 ;               track RMS change.
0000     97 ;
0000     98 ;       V02-003 MLJ0079         Martin L. Jack, 17-Feb-1982  13:51
0000     99 ;               Set up R11 in RM$FABCHK to point to a dummy image I/O impure
0000    100 ;               page.  This is now used by $PARSE.
0000    101 ;
0000    102 ;       V02-002 MLJ0063         Martin L. Jack, 23-Dec-1981  1:58
0000    103 ;               Correct RM$ALDBUF to accept size in R5.
0000    104 ;
0000    105 ;       V02-001 MLJ0054         Martin L. Jack, 22-Nov-1981  22:51
0000    106 ;               Integrate GET_VM and FREE_VM jacket routines.
0000    107 ;
0000    108 ;**
```

```
                        00000000   110          .PSECT  RM$RMS,PIC,EXE,NOWRT,GBL
                              0000 111
                              0000 112          $BDBDEF
                              0000 113          $CCBDEF
                              0000 114          $DCDEF
                              0000 115          $DEVDEF
                              0000 116          $DIBDEF
                              0000 117          $FABDEF
                              0000 118          $FWADEF
                              0000 119          $IFBDEF
                              0C00 120          $IODEF
                              0000 121          $PSLDEF
                              0000 122          $UCBDEF
                              0000 123
                      0FFC    0000 124          .ENTRY  SYS$PARSE,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
          00000002'GF   17    0002 125          JMP     G^RMS$PARSE+2
                              0008 126
                              0008 127 NT$NWA_INIT::
     50   00000000'8F   D0    0008 128          MOVL    #RMS$_SYN,R0            ; set error code
                        05    000F 129          RSB                             ; return
                              0010 130
                              0010 131 RMS$SLIST_ERRS::                         ; no search list errors
                      0000    0010 132          .WORD   0                       ;  are appropriate
                  00000000    0012 133 RMS$SLIST_ERR_CNT == 0
                              0012 134
                              0012 135 RMS$CLEANUP::
                        05    0012 136          RSB
                              0013 137
                              0013 138 RMS$CLSCU::
                        50 DD 0013 139          PUSHL   R0                      ; save error code
                        FB 10 0015 140          BSBB    RMS$CLEANUP             ; cleanup ifab and stuff
                        50 8ED0 0017 141        POPL    R0
                        03 11 001A 142          BRB     RMS$EX_NOSTR            ; and do structureless exit
                              001C 143
                              001C 144 RMS$EXSUC::
                     50 01 D0 001C 145          MOVL    #1,R0                   ; set success
                              001F 146 RMS$EX_NOSTR::
                  00 50 10 E2 001F 147          BBSS    #16,R0,10$              ; add rms facility code
                  08 A8 50 D0 0023 148 10$:     MOVL    R0,FAB$L_STS(R8)        ; and store in fab/rab
                        04    0027 149          RET                             ; exit rms
                              0028 150
                              0028 151 RMS$FABCHK::
                     57 03 D0 0028 152          MOVL    #PSL$C_USER,R7          ; set previous mode user
     5B   00000000'EF 9E 002B 153          MOVAB   PIO$GW_IIOIMPA,R11      ; point to fake image i/o impure area
                  58 04 AC D0 0032 154          MOVL    4(AP),R8                ; get fab address
                     08 A8 7C 0036 155          CLRQ    FAB$L_STS(R8)           ; clear sts and stv
                     02 A8 B4 0039 156          CLRW    FAB$W_IFI(R8)           ; clear ifi
                  59 02 A8 3C 003C 157          MOVZWL  FAB$W_IFI(R8),R9        ; set r9 = ifi value
                        05    0040 158          RSB                             ; return
                              0041 159
                              0041 160 RMS$FSETI_ALT::
                  52 AC 8F 9A 0041 161          MOVZBL  #IFB$C_BLN_SEQ,R2       ; block size for ifab
                        0046 30 0045 162          BSBW    RMS$GETSPC              ; get ifab space
                     59 51 D0 0048 163          MOVL    R1,R9                   ; save ifab address
                  08 A9 0B 90 004B 164          MOVB    #IFB$C_BID,IFB$B_BID(R9)        ; block type for ifab
               09 A9 AC 8F 90 004F 165          MOVB    #IFB$C_BLN_SEQ,IFB$B_BLN(R9)    ; block size for ifab
                  50 40 A9 9E 0054 166          MOVAB   IFB$L_BDB_FLNK(R9),R0   ; set up bdb link pointers
```

```
                60    50    D0   0058   167          MOVL    R0,(R0)                         ;
              04 A0   50    D0   005B   168          MOVL    R0,4(R0)                        ;
00000208'     EF      59    D0   005F   169          MOVL    R9,IFAB_ADDRESS                 ; save ifab address
              02 A8   01    B0   0066   170          MOVW    #1,FAB$W_IFI(R8)                ; store ifi value
                                 006A   171 RM$FSET_ALT::
              0A A9   57    90   006A   172          MOVB    R7,IFB$B_MODE(R9)               ; save caller's mode
              18 A9   5C    D0   006E   173          MOVL    AP,IFB$L_ARGLST(R9)             ; save pointer to arglist
                 5A   59    D0   0072   174          MOVL    R9,R10                          ; copy ifab addr
              24 A9   58    D0   0075   175          MOVL    R8,IFB$L_LAST_FAB(R9)           ; save addr this fab
                      05    0079   176          RSB
                                 007A   177
                                 007A   178 RM$GETBLK::
                                 007A   179 RM$GETBLK1::
                      52    DD   007A   180          PUSHL   R2                              ; save # of longwords
                52    52    02   9C   007C   181          ROTL    #2,R2,R2                        ; convert to bytes
                         0C   10   0080   182          BSBB    RM$GETSPC                       ; get the space
                      05 50    E9   0082   183          BLBC    R0,10$                          ; branch on error
                09 A1    8E    F6   0085   184          CVTLB   (SP)+,9(R1)                     ; store length
                               05   0089   185          RSB                                     ; return
                         51 8ED0   008A   186 10$:     POPL    R1                              ; clean stack
                               05   008D   187          RSB
                                 008E   188
                                 008E   189 RM$GETSPC1::
                                 008E   190 RM$GETSPC::
                      52    DD   008E   191          PUSHL   R2                              ; length desired
00000000'     GF      01    FB   0090   192          CALLS   #1,G^GET_ZERO_VM                ; allocate and clear memory
                 51   50    D0   0097   193          MOVL    R0,R1                           ; get address of memory
                 50   01    D0   009A   194          MOVL    #1,R0                           ; set success return
                               05   009D   195          RSB
                                 009E   196
                                 009E   197 RM$GET1PAG::
           52    0200 8F    3C   009E   198          MOVZWL  #512,R2                         ; get page
                                 00A3   199 RM$GETPAG::
                      52    DD   00A3   200          PUSHL   R2                              ; save input length
                         E7   10   00A5   201          BSBB    RM$GETSPC                       ; get the space
                      53    51   D0   00A7   202          MOVL    R1,R3                           ; return page address in r3
                      52    8E   D0   00AA   203          MOVL    (SP)+,R2                        ; return length obtained
                               05   00AD   204          RSB
                                 00AE   205
                                 00AE   206 RM$ALDBUF::
                      55    DD   00AE   207          PUSHL   R5                              ; save length to allocate
                      52    55   D0   00B0   208          MOVL    R5,R2                           ; copy to correct register
                         D9   10   00B3   209          BSBB    RM$GETSPC                       ; get the space
                      53    51   D0   00B5   210          MOVL    R1,R3                           ; return page buffer in r3
          52    00000050 8F    D0   00B8   211          MOVL    #BDB$C_BLN,R2                   ; length of BDB
                         CD   10   00BF   212          BSBB    RM$GETSPC                       ; get the space
                      54    51   D0   00C1   213          MOVL    R1,R4                           ; return bdb buffer in r4
                44 BA    61   0E   00C4   214          INSQUE  (R1),@IFB$L_BDB_BLNK(R10)       ; insert into bdb queue at tail
                18 A4    53   D0   00C8   215          MOVL    R3,BDB$L_ADDR(R4)               ; set address of buffer
                      52    8E   D0   00CC   216          MOVL    (SP)+,R2                        ; restore length to allocate
                               05   00CF   217          RSB
                                 00D0   218
                                 00D0   219 RM$RET1PAG::
           52    0200 8F    3C   00D0   220          MOVZWL  #512,R2
                         08   11   00D5   221          BRB     RM$RETSPC
                                 00D7   222
                                 00D7   223 RM$RETBLK::
```

```
                                00D7   224 RM$RETBLK1::
                   52   09 A4   9A  00D7   225          MOVZBL  9(R4),R2                ; get length from block
                52 52   52 02   9C  00DB   226          ROTL    #2,R2,R2                ; convert to bytes
                                    00DF   227
                                    00DF   228 RM$RETPAG::
                                    00DF   229 RM$RETSPC1::
                                    00DF   230 RM$RETSPC::
                         54   DD  00DF   231          PUSHL   R4                      ; address to free
                         52   DD  00E1   232          PUSHL   R2                      ; length to free
          00000000'GF    02   FB  00E3   233          CALLS   #2,G^FREE_VM            ; free memory
                      50   01   D0  00EA   234          MOVL    #1,R0                   ; set success return
                                05  00ED   235          RSB                             ; return
                                    00EE   236
                                    00EE   237 RM$GTIADR::
          59    00000208'EF     D0  00EE   238          MOVL    IFAB_ADDRESS,R9         ; get ifab address
                                05  00F5   239          RSB
                                    00F6   240
                                    00F6   241 RM$DEACCESS::
                                    00F6   242 RM$INIT_SWB::
                                    00F6   243 RM$NEXTDIR::
                      50   01   D0  00F6   244          MOVL    #1,R0
                                05  00F9   245          RSB
                                    00FA   246
                                    00FA   247 PIO$GT_DDSTRING::
5D 45 58 45 53 59 53 5B 00'         00FA   248          .ASCIC  "[S/SEXE]"              ; default directory
                                08  00FA
                                    0103   249
                                00000000   250          .PSECT  DATA,WRT,NOEXE
                                    0000   251 PIO$GW_IIOIMPA:
                00000200            0000   252          .BLKB   512                     ; dummy image I/O impure area
                                    0200   253 PIO$A_TRACE::
                00000204            0200   254          .BLKL                           ; dummy trace area
                                    0204   255 PIO$GW_STATUS::
                00000208            0204   256          .BLKL                           ; global status word
                                    0208   257 IFAB_ADDRESS:
                0000020C            0208   258          .BLKL                           ; saves address of IFAB
                                    020C   259
                00000000            020C   260 TPT$L_PARSE==          0               ; force all $TSTPT's to
                00000000            020C   261 TPT$L_NTXLATLOG==      0               ; use the single longword
                00000000            020C   262 TPT$L_PARSES==         0               ; at PIO$A_TRACE
                00000000            020C   263 TPT$L_XLATLOG==        0
                00000000            020C   264 TPT$L_XPFN==           0
                                    020C   265
                                    020C   266          .END
```

```
BDB$C_BLN                       = 00000050
BDB$L_ADDR                      = 00000018
FAB$L_STS                       = 00000008
FAB$W_IFI                       = 00000002
FREE_VM                           ********    X   01
GET_ZERO_VM                       ********    X   01
IFAB_ADDRESS                      00000208 R      03
IFB$B_BID                       = 00000008
IFB$B_BLN                       = 00000009
IFB$B_MODE                      = 0000000A
IFB$C_BID                       = 0000000B
IFB$C_BLN_SEQ                   = 000000AC
IFB$L_ARGLST                    = 00000018
IFB$L_BDB_BLNK                  = 00000044
IFB$L_BDB_FLNK                  = 00000040
IFB$L_LAST_FAB                  = 00000024
NT$NWA_INIT                       00000008 RG     01
PIO$A_TRACE                       00000200 RG     03
PIO$GT_DDSTRING                   000000FA RG     01
PIO$GW_IIOIMPA                    00000000 R      03
PIO$GW_STATUS                     00000204 RG     03
PSL$C_USER                      = 00000003
RMSALDBUF                         000000AE RG     01
RMSCLEANUP                        00000012 RG     01
RMSCLSCU                          00000013 RG     01
RMSDEACCESS                       000000F6 RG     01
RMSEXSUC                          0000001C RG     01
RMSEX_NOSTR                       0000001F RG     01
RMSFABCHK                         00000028 RG     01
RMSFSETI_ALT                      00000041 RG     01
RMSFSET_ALT                       0000006A RG     01
RMSGET1PAG                        0000009E RG     01
RMSGETBLK                         0000007A RG     01
RMSGETBLK1                        0000007A RG     01
RMSGETPAG                         000000A3 RG     01
RMSGETSPC                         0000008E RG     01
RMSGETSPC1                        0000008E RG     01
RMSGTIADR                         000000EE RG     01
RMSINIT_SWB                       000000F6 RG     01
RMSNEXTDIR                        000000F6 RG     01
RMSRET1PAG                        000000D0 RG     01
RMSRETBLK                         000000D7 RG     01
RMSRETBLK1                        000000D7 RG     01
RMSRETPAG                         000000DF RG     01
RMSRETSPC                         000000DF RG     01
RMSRETSPC1                        000000DF RG     01
RMSSLIST_ERRS                     00000010 RG     01
RMSSLIST_ERR_CNT                = 00000000  G
RMS$PARSE                         ********      X 01
RMS$_SYN                          ********      X 01
SYS$PARSE                         00000000 RG     01
TPT$L_NTXLATLOG                 = 00000000  G
TPT$L_PARSE                     = 00000000  G
TPT$L_PARSES                    = 00000000  G
TPT$L_XLATLOG                   = 00000000  G
TPT$L_XPFN                      = 00000000  G
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RM$RMS	00000103 (259.)	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
DATA	0000020C (524.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	12	00:00:00.09	00:00:00.59
Command processing	109	00:00:00.98	00:00:06.63
Pass 1	411	00:00:14.86	00:00:43.38
Symbol table sort	0	00:00:02.66	00:00:06.34
Pass 2	66	00:00:02.47	00:00:08.63
Symbol table output	8	00:00:00.10	00:00:00.22
Psect synopsis output	2	00:00:00.03	00:00:00.06
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	611	00:00:21.20	00:01:05.86

The working set limit was 1500 pages.
86466 bytes (169 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1820 non-local and 2 local symbols.
266 source lines were read in Pass 1, producing 19 object records in Pass 2.
18 pages of virtual memory were used to define 17 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SHRLIB]RMS.MLB;1	4
_255DUA28:[SYSLIB]STARLET.MLB;2	8
TOTALS (all libraries)	14

1878 GETS were required to define 14 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:STAPARSE/OBJ=OBJ$:STAPARSE MSRC$:STAPARSE/UPDATE=(ENH$:STAPARSE)+SHRLIB$:RMS.MLB/LIB+EXECML$/LIB
```

TAPEUTIL
LIS

STAINVOL
LIS

UTILITY
LIS

STATOSYS
LIS

STAPARSE
LIS