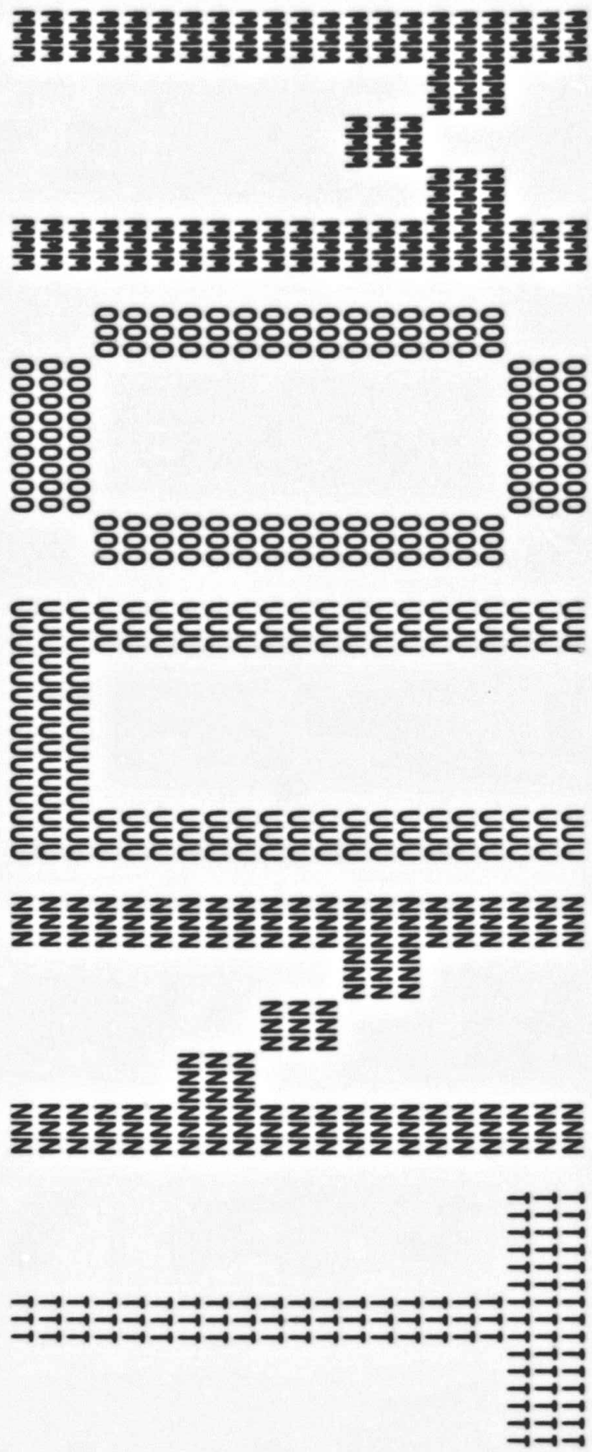

```
MM      MM  RRRRRRRR    DDDDDDDD    BBBBBBBB    LL          KK      KK
MM      MM  RRRRRRRR    DDDDDDDD    BBBBBBBB    LL          KK      KK
MMMM  MMMM  RR      RR  DD      DD  BB      BB  LL          KK      KK
MMMM  MMMM  RR      RR  DD      DD  BB      BB  LL          KK      KK
MM  MM  MM  RR      RR  DD      DD  BB      BB  LL          KK    KK
MM  MM  MM  RR      RR  DD      DD  BB      BB  LL          KK    KK
MM      MM  RRRRRRRR    DD      DD  BBBBBBBB    LL          KKKKKK
MM      MM  RRRRRRRR    DD      DD  BBBBBBBB    LL          KKKKKK
MM      MM  RR  RR      DD      DD  BB      BB  LL          KK    KK
MM      MM  RR  RR      DD      DD  BB      BB  LL          KK    KK
MM      MM  RR    RR    DD      DD  BB      BB  LL          KK      KK      ....
MM      MM  RR    RR    DD      DD  BB      BB  LL          KK      KK      ....
MM      MM  RR      RR  DDDDDDDD    BBBBBBBB    LLLLLLLLLL  KK      KK      ....
MM      MM  RR      RR  DDDDDDDD    BBBBBBBB    LLLLLLLLLL  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
```

```
;     1          0001  0 MODULE MRDBLK (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !**********************************************************************
;     9          0009  1 !*                                                                    *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                             *
;    13          0013  1 !*                                                                    *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    19          0019  1 !*  TRANSFERRED.                                                       *
;    20          0020  1 !*                                                                    *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    23          0023  1 !*  CORPORATION.                                                       *
;    24          0024  1 !*                                                                    *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    27          0027  1 !*                                                                    *
;    28          0028  1 !*                                                                    *
;    29          0029  1 !**********************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  MOUNT Utility Structure Levels 1 & 2
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This routine reads a block from the disk being mounted.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:
;    40          0040  1 !
;    41          0041  1 !       STARLET operating system, including privileged system services
;    42          0042  1 !       and internal exec routines.
;    43          0043  1 !
;    44          0044  1 !--
;    45          0045  1 !
;    46          0046  1 !
;    47          0047  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  18-Oct-1977  11:35
;    48          0048  1 !
;    49          0049  1 ! MODIFIED BY:
;    50          0050  1 !
;    51          0051  1 !       V03-002 HH0041          Hai Huang                   24-Jul-1984
;    52          0052  1 !               Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.B32'.
;    53          0053  1 !
;    54          0054  1 !       V03-001 STJ0246         Steven T. Jeffreys,     04-Apr-1982
;    55          0055  1 !               Create a common I/O routine that will be used for
;    56          0056  1 !               all sycnronous I/O done within $MOUNT.
;    57          0057  1 !
```

```
;    58          0058  1 !       V02-001 ACG0167         Andrew C. Goldstein,    18-Apr-1980  13:38
;    59          0059  1 !               Previous revision history moved to MOUNT.REV
;    60          0060  1 !**
;    61          0061  1
;    62          0062  1
;    63          0063  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    64          0064  1 REQUIRE 'SRC$:MOUDEF.B32';
;    65          0596  1
;    66          0597  1
;    67          0598  1 FORWARD ROUTINE
;    68          0599  1         READ_BLOCK,                     ! read a disk block
;    69          0600  1         WRITE_BLOCK;                    ! write a disk block
```

```
;    71          0601  1 GLOBAL ROUTINE READ_BLOCK (LBN, BUFFER) =
;    72          0602  1
;    73          0603  1 !++
;    74          0604  1 !
;    75          0605  1 ! FUNCTIONAL DESCRIPTION:
;    76          0606  1 !
;    77          0607  1 !       This routine reads the indicated block from the device that
;    78          0608  1 !       CHANNEL is assigned to.
;    79          0609  1 !
;    80          0610  1 ! CALLING SEQUENCE:
;    81          0611  1 !       READ_BLOCK (ARG1, ARG2)
;    82          0612  1 !
;    83          0613  1 ! INPUT PARAMETERS:
;    84          0614  1 !       ARG1: LBN of block
;    85          0615  1 !       ARG2: address of buffer to read into
;    86          0616  1 !
;    87          0617  1 ! IMPLICIT INPUTS:
;    88          0618  1 !       CHANNEL: channel number to use
;    89          0619  1 !
;    90          0620  1 ! OUTPUT PARAMETERS:
;    91          0621  1 !       NONE
;    92          0622  1 !
;    93          0623  1 ! IMPLICIT OUTPUTS:
;    94          0624  1 !       NONE
;    95          0625  1 !
;    96          0626  1 ! ROUTINE VALUE:
;    97          0627  1 !       I/O or system service status of request
;    98          0628  1 !
;    99          0629  1 ! SIDE EFFECTS:
;   100          0630  1 !       block read
;   101          0631  1 !
;   102          0632  1 !--
;   103          0633  1
;   104          0634  2 BEGIN
;   105          0635  2
;   106          0636  2 LOCAL
;   107          0637  2         STATUS,                         ! system status code
;   108          0638  2         IO_STATUS       : VECTOR [2];   ! I/O status block
;   109          0639  2
;   110          0640  2 EXTERNAL
;   111          0641  2         CHANNEL;                        ! channel number for I/O
;   112          0642  2
;   113        P 0643  2 STATUS = DO_IO (CHAN = .CHANNEL,
;   114        P 0644  2                 FUNC = IO$_READLBLK,
;   115        P 0645  2                 IOSB = IO_STATUS[0],
;   116        P 0646  2                 EFN  = MOUNT_EFN,
;   117        P 0647  2                 P1   = .BUFFER,
;   118        P 0648  2                 P2   = 512,
;   119          0649  2                 P3   = .LBN);
;   120          0650  2
;   121          0651  2 IF .STATUS THEN STATUS = .(IO_STATUS[0])<0,16>;
;   122          0652  2 RETURN .STATUS;
;   123          0653  2
;   124          0654  1 END;                                    ! end of routine READ_BLOCK

                                                                .TITLE  MRDBLK
```

```
                                                        .IDENT  \V04-000\

                                                        .EXTRN  CHANNEL, COMMON_IO

                                                        .PSECT  $CODE$,NOWRT,2

                                        0000 00000      .ENTRY  READ_BLOCK, Save nothing                        ; 0601
                                5E              08 C2 00002     SUBL2   #8, SP                                  ;
                                              7E 7C 00005       CLRQ    -(SP)                                   ; 0649
                                              7E D4 00007       CLRL    -(SP)                                   ;
                                          04  AC DD 00009       PUSHL   LBN                                     ;
                                7E      0200  8F 3C 0000C       MOVZWL  #512, -(SP)                             ;
                                          08  AC DD 00011       PUSHL   BUFFER                                  ;
                                              7E 7C 00014       CLRQ    -(SP)                                   ;
                                          20  AE 9F 00016       PUSHAB  IO_STATUS                               ;
                                              21 DD 00019       PUSHL   #33                                     ;
                                       0000G  CF DD 0001B       PUSHL   CHANNEL                                 ;
                                              1A DD 0001F       PUSHL   #26                                     ;
                        00000000G     00      0C FB 00021       CALLS   #12, COMMON_IO                          ;
                                              03 50 E9 00028    BLBC    STATUS, 1$                              ; 0651
                                              50 6E 3C 0002B    MOVZWL  IO_STATUS, STATUS                       ;
                                                 04 0002E 1$:   RET                                             ; 0654

; Routine Size:  47 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  126     0655  1 GLOBAL ROUTINE WRITE_BLOCK (LBN, BUFFER) =
;  127     0656  1
;  128     0657  1 !++
;  129     0658  1 !
;  130     0659  1 ! FUNCTIONAL DESCRIPTION:
;  131     0660  1 !
;  132     0661  1 !       This routine writes the indicated block to the device that
;  133     0662  1 !       CHANNEL is assigned to.
;  134     0663  1 !
;  135     0664  1 ! CALLING SEQUENCE:
;  136     0665  1 !       WRITE_BLOCK (ARG1, ARG2)
;  137     0666  1 !
;  138     0667  1 ! INPUT PARAMETERS:
;  139     0668  1 !       ARG1: LBN of block
;  140     0669  1 !       ARG2: address of buffer to write from
;  141     0670  1 !
;  142     0671  1 ! IMPLICIT INPUTS:
;  143     0672  1 !       CHANNEL: channel number to use
;  144     0673  1 !
;  145     0674  1 ! OUTPUT PARAMETERS:
;  146     0675  1 !       NONE
;  147     0676  1 !
;  148     0677  1 ! IMPLICIT OUTPUTS:
;  149     0678  1 !       NONE
;  150     0679  1 !
;  151     0680  1 ! ROUTINE VALUE:
;  152     0681  1 !       I/O or system service status of request
;  153     0682  1 !
;  154     0683  1 ! SIDE EFFECTS:
;  155     0684  1 !       block written
;  156     0685  1 !
;  157     0686  1 !--
;  158     0687  1
;  159     0688  2 BEGIN
;  160     0689  2
;  161     0690  2 LOCAL
;  162     0691  2         STATUS,                         ! system status code
;  163     0692  2         IO_STATUS       : VECTOR [2];   ! I/O status block
;  164     0693  2
;  165     0694  2 EXTERNAL
;  166     0695  2         CHANNEL;                        ! channel number for I/O
;  167     0696  2
;  168   P 0697  2 STATUS = DO_IO (CHAN = .CHANNEL,
;  169   P 0698  2                  FUNC = IO$_WRITELBLK,
;  170   P 0699  2                  IOSB = IO_STATUS[0],
;  171   P 0700  2                  EFN  = MOUNT_EFN,
;  172   P 0701  2                  P1   = .BUFFER,
;  173   P 0702  2                  P2   = 512,
;  174     0703  2                  P3   = .LBN);
;  175     0704  2
;  176     0705  2 IF .STATUS THEN STATUS = .(IO_STATUS[0])<0,16>;
;  177     0706  2 RETURN .STATUS;
;  178     0707  2
;  179     0708  1 END;                                    ! end of routine WRITE_BLOCK
```

```
                                 0000 00000          .ENTRY  WRITE_BLOCK, Save nothing                ; 0655
                          5E       08   C2 00002     SUBL2   #8, SP                                  ;
                                   7E   7C 00005     CLRQ    -(SP)                                   ; 0703
                                   7E   D4 00007     CLRL    -(SP)                                   ;
                              04   AC   DD 00009     PUSHL   LBN                                     ;
                          7E  0200   8F   3C 0000C     MOVZWL  #512, -(SP)                             ;
                              08   AC   DD 00011     PUSHL   BUFFER                                  ;
                                   7E   7C 00014     CLRQ    -(SP)                                   ;
                              20   AE   9F 00016     PUSHAB  IO_STATUS                               ;
                                   20   DD 00019     PUSHL   #32                                     ;
                            0000G  CF   DD 0001B     PUSHL   CHANNEL                                 ;
                                   1A   DD 0001F     PUSHL   #26                                     ;
                 00000000G  00         0C   FB 00021     CALLS   #12, COMMON_IO                      ;
                            03         50   E9 00028     BLBC    STATUS, 1$                          ; 0705
                            50         6E   3C 0002B     MOVZWL  IO_STATUS, STATUS                   ;
                                       04   0002E 1$:    RET                                         ; 0708

; Routine Size:  47 bytes,    Routine Base:  $CODE$ + 002F
```

```
;  181     0709  1 GLOBAL ROUTINE COMMON_IO (EFN,CHAN,FUNC,IOSTS,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)=
;  182     0710  1
;  183     0711  1 !++
;  184     0712  1 !
;  185     0713  1 ! FUNCTIONAL DESCRIPTION:
;  186     0714  1 !
;  187     0715  1 !       This routine is the common I/O routine for all synchronous I/O
;  188     0716  1 !       done by the $MOUNT system service.  Event flags are used very
;  189     0717  1 !       cautiously to prevent confusion arising from an outside agent
;  190     0718  1 !       setting and clearing event flags indiscrimenently.
;  191     0719  1 !
;  192     0720  1 ! CALLING SEQUENCE:
;  193     0721  1 !       Identical to $QIO.  A macro, DO_IO, has been defined in MOUDEF
;  194     0722  1 !       so that existing code can use this routine by replacing the
;  195     0723  1 !       call to $QIOW with an invokation of DO_IO without having to
;  196     0724  1 !       change the order or names of routine arguements.
;  197     0725  1 !
;  198     0726  1 ! INPUT PARAMETERS:
;  199     0727  1 !       See $QIOW for the details.
;  200     0728  1 !
;  201     0729  1 ! IMPLICIT INPUTS:
;  202     0730  1 !       NONE
;  203     0731  1 !
;  204     0732  1 ! OUTPUT PARAMETERS:
;  205     0733  1 !       NONE
;  206     0734  1 !
;  207     0735  1 ! IMPLICIT OUTPUTS:
;  208     0736  1 !       NONE
;  209     0737  1 !
;  210     0738  1 ! ROUTINE VALUE:
;  211     0739  1 !       I/O or system service status of request
;  212     0740  1 !
;  213     0741  1 ! SIDE EFFECTS:
;  214     0742  1 !       Some I/O has been attempted, and at exit, the event flag
;  215     0743  1 !       MOUNT_EFN is always set.
;  216     0744  1 !
;  217     0745  1 !--
;  218     0746  1
;  219     0747  2 BEGIN                                   ! Start of routine COMMON_IO
;  220     0748  2
;  221     0749  2 LOCAL
;  222     0750  2         IOSB            : REF BBLOCK VOLATILE,
;  223     0751  2         IO_STATUS       : BBLOCK [8],
;  224     0752  2         STATUS;
;  225     0753  2
;  226     0754  2 ! If an I/O status block was not specified, use a local one.
;  227     0755  2 !
;  228     0756  2
;  229     0757  2 IF (IOSB = .IOSTS) EQL 0
;  230     0758  2 THEN
;  231     0759  3     BEGIN
;  232     0760  3     CH$FILL (0, 8, IO_STATUS);
;  233     0761  3     IOSB = IO_STATUS;
;  234     0762  2     END;
;  235     0763  2
;  236     0764  2 ! Issue the I/O request.
;  237     0765  2 !
```

```
;   238          0766  2
;   239        P 0767  2 STATUS = $QIOW  (EFN    = .EFN,
;   240        P 0768  2                  CHAN   = .CHAN,
;   241        P 0769  2                  FUNC   = .FUNC,
;   242        P 0770  2                  IOSB   = .IOSB,
;   243        P 0771  2                  ASTADR = .ASTADR,
;   244        P 0772  2                  ASTPRM = .ASTPRM,
;   245        P 0773  2                  P1     = .P1,
;   246        P 0774  2                  P2     = .P2,
;   247        P 0775  2                  P3     = .P3,
;   248        P 0776  2                  P4     = .P4,
;   249        P 0777  2                  P5     = .P5,
;   250        P 0778  2                  P6     = .P6
;   251          0779  2                 );
;   252          0780  2
;   253          0781  2 ! If the return status from $QIOW indicates success, make sure that
;   254          0782  2 ! the I/O actually completed.  If it did not, then wait for the event
;   255          0783  2 ! flag again, after resetting it to a known state.
;   256          0784  2 !
;   257          0785  2
;   258          0786  2 IF .STATUS
;   259          0787  2 THEN
;   260          0788  3     BEGIN
;   261          0789  3     WHILE (.IOSB [0,0,16,0] EQL 0) DO
;   262          0790  4         BEGIN
;   263          0791  4         $CLREF (EFN = .EFN);
;   264          0792  5         IF (.IOSB [0,0,16,0] EQL 0)
;   265          0793  4         THEN
;   266          0794  4             $WAITFR (EFN = .EFN);
;   267          0795  3         END;
;   268          0796  3     STATUS = .IOSB [0,0,16,0];
;   269          0797  2     END;
;   270          0798  2
;   271          0799  2
;   272          0800  2 ! Set the specified event flag and return.
;   273          0801  2 !
;   274          0802  2
;   275          0803  2 $SETEF (EFN = .EFN);
;   276          0804  3 RETURN (.STATUS)
;   277          0805  1 END;                                    ! End of routine COMMON_IO


                                                .EXTRN  SYS$QIOW, SYS$CLREF
                                                .EXTRN  SYS$WAITFR, SYS$SETEF

                                   003C 00000   .ENTRY  COMMON_IO, Save R2,R3,R4,R5                ; 0709
                          5E            0C  C2 00002   SUBL2  #12, SP                             ;
                    08    AE      10    AC  D0 00005   MOVL   IOSTS, IOSB                         ; 0757
                                        0A  12 0000A   BNEQ   1$                                  ;
08                  00          6E            00  2C 0000C   MOVC5  #0, (SP), #0, #8, IO_STATUS     ; 0760
                                              6E  9E 00011                                        ;
                    08    AE                  6E  9E 00012   MOVAB  IO_STATUS, IOSB               ; 0761
                          7E            2C    AC  7D 00016 1$:  MOVQ   P5, -(SP)                  ; 0779
                          7E            24    AC  7D 0001A      MOVQ   P3, -(SP)                  ;
                          7E            1C    AC  7D 0001E      MOVQ   P1, -(SP)                  ;
                          7E            14    AC  7D 00022      MOVQ   ASTADR, -(SP)              ;
```

```
                               28  AE  DD 00026          PUSHL   IOSB                       :
                     7E        08  AC  7D 00029          MOVQ    CHAN, -(SP)                :
                               04  AC  DD 0002D          PUSHL   EFN                        :
       00000000G  00               0C  FB 00030          CALLS   #12, SYS$QIOW              :
                  52               50  D0 00037          MOVL    R0, STATUS                 :
                  24               52  E9 0003A          BLBC    STATUS, 4$                 : 0786
                     08        BE  B5 0003D 2$:          TSTW    @IOSB                      : 0789
                               1B  12 00040              BNEQ    3$                         :
                     04        AC  DD 00042              PUSHL   EFN                        : 0791
       00000000G  00               01  FB 00045          CALLS   #1, SYS$CLREF              :
                     08        BE  B5 0004C              TSTW    @IOSB                      : 0792
                               EC  12 0004F              BNEQ    2$                         :
                     04        AC  DD 00051              PUSHL   EFN                        : 0794
       00000000G  00               01  FB 00054          CALLS   #1, SYS$WAITFR             :
                               E0  11 0005B              BRB     2$                         : 0789
                  52           08  BE  3C 0005D 3$:      MOVZWL  @IOSB, STATUS              : 0796
                               04  AC  DD 00061 4$:      PUSHL   EFN                        : 0803
       00000000G  00               01  FB 00064          CALLS   #1, SYS$SETEF              :
                  50               52  D0 0006B          MOVL    STATUS, R0                 : 0804
                                       04 0006E          RET                                : 0805

; Routine Size:  111 bytes,    Routine Base:  $CODE$ + 005E


;   278          0806   1
;   279          0807   1 END
;   280          0808   0 ELUDOM
```

```
;                               PSECT SUMMARY
;
;     Name                       Bytes                  Attributes
;
;   $CODE$                         205  NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)


;                          Library Statistics
;
;                                        -------- Symbols --------       Pages       Processing
;     File                               Total    Loaded   Percent       Mapped      Time
;
;   _$255$DUA28:[SYSLIB]LIB.L32;1        18619        16         0       1000         00:02.0


;                               COMMAND QUALIFIERS
```

```
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:MRDBLK/OBJ=OBJ$:MRDBLK MSRC$:MRDBLK/UPDATE=(ENH$:MRDBLK)

; Size:         205 code + 0 data bytes
; Run Time:         00:12.8
; Elapsed Time:     00:24.7
; Lines/CPU Min:     3787
; Lexemes/CPU-Min: 34096
; Memory Used:  99 pages
; Compilation Complete
```

0246 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RDHOME
LIS

MWTUR2
LIS

RUJMAN
LIS

MWTUR1
LIS

SRCVOL
LIS

STACP
LIS

MRDBLK
LIS

REBUILD
LIS