```
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
```

```
MM      MM    000000    MM      MM  MM      MM    000000    PPPPPPPP    LL          IIIIII      000000
MM      MM    000000    MM      MM  MM      MM    000000    PPPPPPPP    LL          IIIIII      000000
MMMM  MMMM  OO      OO  MMMM  MMMM  MMMM  MMMM  OO      OO  PP      PP  LL            II      OO      OO
MMMM  MMMM  OO      OO  MMMM  MMMM  MMMM  MMMM  OO      OO  PP      PP  LL            II      OO      OO
MM  MM  MM  OO      OO  MM  MM  MM  MM  MM  MM  OO      OO  PP      PP  LL            II      OO      OO
MM  MM  MM  OO      OO  MM  MM  MM  MM  MM  MM  OO      OO  PP      PP  LL            II      OO      OO
MM      MM  OO      OO  MM      MM  MM      MM  OO      OO  PPPPPPPP    LL            II      OO      OO
MM      MM  OO      OO  MM      MM  MM      MM  OO      OO  PPPPPPPP    LL            II      OO      OO
MM      MM  OO      OO  MM      MM  MM      MM  OO      OO  PP          LL            II      OO      OO
MM      MM  OO      OO  MM      MM  MM      MM  OO      OO  PP          LL            II      OO      OO
MM      MM  OO      OO  MM      MM  MM      MM  OO      OO  PP          LL            II      OO      OO
MM      MM  OO      OO  MM      MM  MM      MM  OO      OO  PP          LL            II      OO      OO
MM      MM    000000    MM      MM  MM      MM    000000    PP          LLLLLLLLLL  IIIIII      000000
MM      MM    000000    MM      MM  MM      MM    000000    PP          LLLLLLLLLL  IIIIII      000000

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 %TITLE 'Network Service MOP line I/O modules'
;      2     0002  0 MODULE MOMMOPLIO (
;      3     0003  0                 LANGUAGE (BLISS32),
;      4     0004  0                 ADDRESSING_MODE (NONEXTERNAL=LONG_RELATIVE),
;      5     0005  0                 ADDRESSING_MODE (EXTERNAL=LONG_RELATIVE),
;      6     0006  0                 IDENT = 'V04-000'
;      7     0007  0                 ) =
;      8     0008  1 BEGIN
;      9     0009  1 !
;     10     0010  1 !*****************************************************************************
;     11     0011  1 !*                                                                           *
;     12     0012  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;     13     0013  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;     14     0014  1 !*  ALL RIGHTS RESERVED.                                                      *
;     15     0015  1 !*                                                                           *
;     16     0016  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;     17     0017  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;     18     0018  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;     19     0019  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;     20     0020  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;     21     0021  1 !*  TRANSFERRED.                                                              *
;     22     0022  1 !*                                                                           *
;     23     0023  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;     24     0024  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;     25     0025  1 !*  CORPORATION.                                                              *
;     26     0026  1 !*                                                                           *
;     27     0027  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;     28     0028  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;     29     0029  1 !*                                                                           *
;     30     0030  1 !*                                                                           *
;     31     0031  1 !*****************************************************************************
;     32     0032  1 !
;     33     0033  1
;     34     0034  1 !++
;     35     0035  1 ! FACILITY:  DECnet-VAX Network Management Maintenance Operations Module (MOM)
;     36     0036  1 !
;     37     0037  1 ! ABSTRACT:
;     38     0038  1 !       This module contains routines to handle I/O for MOP
;     39     0039  1 !       functions.
;     40     0040  1 !
;     41     0041  1 ! ENVIRONMENT:  VAX/VMS Operating System
;     42     0042  1 !
;     43     0043  1 ! AUTHOR:  Kathy Perko
;     44     0044  1 !
;     45     0045  1 ! CREATION DATE:  11-Jan-1983
;     46     0046  1 !
;     47     0047  1 ! MODIFIED BY:
;     48     0048  1 !       V03-005 MKP0005         Kathy Perko             30-May-1984
;     49     0049  1 !               Eliminate echo parameter from SETMODE QIO to NI driver.
;     50     0050  1 !               This is because, if the target device is a QNA, the echo
;     51     0051  1 !               option is not implemented.
;     52     0052  1 !
;     53     0053  1 !       V03-004 MKP0004         Kathy Perko             29-April-1984
;     54     0054  1 !               When sending and receiving MOP messages, the receive NI
;     55     0055  1 !               address is currently never alternated between the hardware
;     56     0056  1 !               and DECnet address.  Fix it so that it will be if the CIB
;     57     0057  1 !               is marked "alternate".  Also, get rid of Request ID stuff.
```

```
;    58          0058  1 !                 It'll just slow things down.
;    59          0059  1 !
;    60          0060  1 !       V03-003 MKP0003         Kathy Perko             11-Feb-1984
;    61          0061  1 !               Use NI Remote Console protocol for boot message instead of
;    62          0062  1 !               NI Load/dump protocol.
;    63          0063  1 !               Use system ID message to get target's NI address instead of
;    64          0064  1 !               alternately trying the hardware and the DECnet NI addresses.
;    65          0065  1 !
;    66          0066  1 !       V03-002 MKP0002         Kathy Perko             12-Sept-1983
;    67          0067  1 !               If area routing is turned off, use a physical address
;    68          0068  1 !               with an area number of one instead of zero.
;    69          0069  1 !
;    70          0070  1 !       V03-001 MKP0001         Kathy Perko             8-May-1983
;    71          0071  1 !               Change check on read completion status and quit
;    72          0072  1 !               trying to send to target if it's anything except SS$_TIMEOUT.
;    73          0073  1 !
;    74          0074  1 !--
;    75          0075  1
```

```
;    77          0076  1 %SBTTL 'Declarations'
;    78          0077  1
;    79          0078  1 !
;    80          0079  1 ! TABLE OF CONTENTS:
;    81          0080  1 !
;    82          0081  1
;    83          0082  1 FORWARD ROUTINE
;    84          0083  1     mom$mopopen           : NOVALUE,
;    85          0084  1     mom$mopsetsubstate    : NOVALUE,
;    86          0085  1     mom$init_CIB          : NOVALUE,
;    87          0086  1     mom$set_NI_addr       : NOVALUE,
;    88          0087  1     mom$mopsndrcv,
;    89          0088  1     mom_mop_receive,
;    90          0089  1     mom$mop_receive_qiow,
;    91          0090  1     mom$mopsend,
;    92          0091  1     mom_mapmoperrors;
;    93          0092  1
;    94          0093  1 !
;    95          0094  1 ! INCLUDE FILES:
;    96          0095  1 !
;    97          0096  1
;    98          0097  1 LIBRARY 'LIB$:MOMLIB.L32';
;    99          0098  1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
;   100          0099  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
;   101          0100  1
;   102          0101  1 !
;   103          0102  1 ! EQUATED SYMBOLS:
;   104          0103  1 !
;   105          0104  1
;   106          0105  1 LITERAL
;   107          0106  1     mom$k_moprcvefn = 1,                ! MOP receive event flag
;   108          0107  1     mom$m_moprcvefn = 1 ^ mom$k_moprcvefn,
;   109          0108  1     mom$k_mopsndefn = 2,                ! MOP send event flag
;   110          0109  1     mom$m_mopsndefn = 1 ^ mom$k_mopsndefn,
;   111          0110  1     mom$k_moptimefn = 3,                ! MOP I/O timer event flag
;   112          0111  1     mom$m_moptimefn = 1 ^ mom$k_moptimefn;
;   113          0112  1
;   114          0113  1
;   115          0114  1 !
;   116          0115  1 ! EXTERNAL REFERENCES:
;   117          0116  1 !
;   118          0117  1
;   119          0118  1 $mom_externals;
;   120          0119  1
;   121          0120  1 EXTERNAL
;   122          0121  1     mom$gq_timeout:     VECTOR [0],
;   123          0122  1     mom$gq_dle_namdsc:  REF VECTOR;
;   124          0123  1
;   125          0124  1 EXTERNAL ROUTINE
;   126          0125  1     mom$bld_reply,
;   127          0126  1     mom$debug_msg,
;   128          0127  1     mom$debug_qio,
;   129          0128  1     mom$debug_txt;
;   130          0129  1
```

```
;  132          0130  1 %SBTTL 'mom$mopopen  Open a circuit for MOP mode access'
;  133          0131  1 GLOBAL ROUTINE mom$mopopen (mop_channel) : NOVALUE =
;  134          0132  1
;  135          0133  1 !++
;  136          0134  1 ! FUNCTIONAL DESCRIPTION:
;  137          0135  1 !
;  138          0136  1 !       This routine opens a channel to a circuit for service operations
;  139          0137  1 !       via a request to NETACP.
;  140          0138  1 !
;  141          0139  1 ! INPUTS:
;  142          0140  1 !
;  143          0141  1 !       MOP_CHANNEL = Address at which to return channel over which MOP QIOs
;  144          0142  1 !             are to be done.
;  145          0143  1 !
;  146          0144  1 ! OUTPUTS:
;  147          0145  1 !
;  148          0146  1 !       Success/failure of operation.
;  149          0147  1 !
;  150          0148  1 !       MOM$GQ_TIMEOUT  Timer for MOP I/O has been set up.
;  151          0149  1 !       MOP_CHANNEL = Address for returning channel for which all MOP I/O
;  152          0150  1 !             is initialized.
;  153          0151  1 !--
;  154          0152  1
;  155          0153  2 BEGIN
;  156          0154  2
;  157          0155  2 LOCAL
;  158          0156  2     retry_count,
;  159          0157  2     iosb        : $IOSB,
;  160          0158  2     circuit_dsc : VECTOR [2],
;  161          0159  2     status;
;  162          0160  2
;  163          0161  2 !
;  164          0162  2 ! Save the service timer (specified in Msec).
;  165          0163  2 !
;  166          0164  2 mom$gq_timeout [0] = -10 * 1000 * .mom$ab_service_data [svd$gk_pcli_sti,
;  167          0165  2                                                        svd$l_param];
;  168          0166  2 !
;  169          0167  2 ! Assign a channel to NET for use in controlling and accessing the circuit.
;  170          0168  2 !
;  171        P 0169  2 status = $ASSIGN (CHAN = .mop_channel,
;  172          0170  2                     DEVNAM = mom$gq_dle_namdsc);
;  173          0171  2 mom_mapmoperrors (.status, 0, mom$signal);
;  174          0172  2
;  175          0173  2 !
;  176          0174  2 ! Request use of the circuit for MOP functions.  This request causes
;  177          0175  2 ! NETACP to allow Service to issue QIOs directly to the circuit - not a
;  178          0176  2 ! normal thing, since NETDRIVER is normally the only DECnet module
;  179          0177  2 ! which does this.
;  180          0178  2 !
;  181          0179  2 circuit_dsc [0] = .mom$ab_service_data [svd$gk_pcno_sli, svd$b_string_len];
;  182          0180  2 circuit_dsc [1] = mom$ab_service_data [svd$gk_pcno_sli, svd$t_string];
;  183          0181  2 !
;  184          0182  2 ! Get NETACP to put the service device into a state where Service can
;  185          0183  2 ! issue QIOs to it.  Issue the QIO to tell NETACP the circuit is needed
;  186          0184  2 ! for a service function.  Do 3 retries.
;  187          0185  2 !
;  188          0186  2 retry_count = 3;
```

```
;  189          0187  2 WHILE true DO
;  190          0188  3     BEGIN
;  191          0189  3     retry_count = .retry_count - 1;
;  192        P 0190  3     status = $QIOW (CHAN = ..mop_channel,
;  193        P 0191  3                     FUNC = io$_access,
;  194        P 0192  3                     IOSB = iosb,
;  195          0193  3                     P1 = circuit_dsc);
;  196          0194  3     mom$debug_qio (dbg$c_mopio, .status, iosb, 0, circuit_dsc,
;  197          0195  3                    0, 0, $ASCID ('IO$_ACCESS QIO on MOP channel'));
;  198          0196  4     IF (NOT .status) OR (NOT .iosb [ios$w_status])
;  199          0197  3     THEN
;  200          0198  4         BEGIN
;  201          0199  4         IF .retry_count LEQ 0 THEN
;  202          0200  4             mom_mapmoperrors (.status, iosb, mom$signal);
;  203          0201  4         END
;  204          0202  3     ELSE
;  205          0203  3         EXITLOOP;
;  206          0204  2     END;
;  207          0205  2 !
;  208          0206  2 ! Check to see if the circuit is an NI circuit.
;  209          0207  2 !
;  210          0208  2 IF .iosb[ios$l_info] THEN
;  211          0209  2     mom$gl_service_flags [mom$v_ni_circ] = true;
;  212          0210  2 !
;  213          0211  2 ! Set the circuit substate to -AUTOSERVICE.
;  214          0212  2 !
;  215          0213  2 IF .mom$gl_service_flags [mom$v_autoservice] THEN
;  216          0214  2     mom$mopsetsubstate (nma$c_linss_ase,
;  217          0215  2                         ..mop_channel);
;  218          0216  1 END;                            ! End of MOM$MOPOPEN


                                                .TITLE  MOMMOPLIO Network Service MOP line I/O modules
                                                .IDENT  \V04-000\

                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

20 4F 49 51 20 53 53 45 43 43 41 5F 24 4F 49  00000 P.AAB:  .ASCII  \IO$_ACCESS QIO on MOP channel\          ;
   6C 65 6E 6E 61 68 63 20 50 4F 4D 20 6E 6F  0000F                                                       ;
                                              0001D        .BLKB   3
                                   0000001D   00020 P.AAA:  .LONG   29                                     ;
                                   00000000'  00024        .ADDRESS P.AAB                                 ;

                                                .EXTRN  MOM$GL_LOGMASK, MOM$GL_SVD_INDEX
                                                .EXTRN  MOM$AB_SERVICE_DATA
                                                .EXTRN  MOM$GB_FUNCTION
                                                .EXTRN  MOM$GB_OPTION_BYTE
                                                .EXTRN  MOM$GB_ENTITY_CODE
                                                .EXTRN  MOM$AB_ENTITY_BUF
                                                .EXTRN  MOM$GQ_ENTITY_BUF_DSC
                                                .EXTRN  MOM$GL_SERVICE_FLAGS
                                                .EXTRN  MOM$AB_NPARSE_BLK
                                                .EXTRN  MOM$AB_NICE_RCV_BUF
                                                .EXTRN  MOM$AB_NICE_XMIT_BUF
                                                .EXTRN  MOM$GQ_NICE_RCV_BUF_DSC
                                                .EXTRN  MOM$GL_NICE_RCV_MSG_LEN
                                                .EXTRN  MOM$GQ_NICE_XMIT_BUF_DSC
```

```
.EXTRN  MOM$AB_MSGBLOCK
.EXTRN  MOM$AB_ACPQIO_BUFFER
.EXTRN  MOM$GQ_ACPQIO_BUF_DSC
.EXTRN  MOM$AB_CIB, MOM$AB_LOOP_CIB
.EXTRN  MOM$AB_TRIGGER_CIB
.EXTRN  MOM$AB_MOP_XMIT_BUF
.EXTRN  MOM$GQ_MOP_XMIT_BUF_DSC
.EXTRN  MOM$AB_MOP_RCV_BUF
.EXTRN  MOM$GQ_MOP_RCV_BUF_DSC
.EXTRN  MOM$AB_MOP_MSG, MOM$GQ_MOP_MSG_DSC
.EXTRN  MOM$GW_EVT_CODE
.EXTRN  MOM$GB_EVT_POPR
.EXTRN  MOM$GB_EVT_PRSN
.EXTRN  MOM$GB_EVT_PSER
.EXTRN  SVD$GK_PCNO_ADD
.EXTRN  SVD$GK_PCNO_SDV
.EXTRN  SVD$GK_PCNO_CPU
.EXTRN  SVD$GK_PCNO_STY
.EXTRN  SVD$GK_PCNO_DAD
.EXTRN  SVD$GK_PCNO_DCT
.EXTRN  SVD$GK_PCNO_IHO
.EXTRN  SVD$GK_PCNO_NNA
.EXTRN  SVD$GK_PCNO_SLI
.EXTRN  SVD$GK_PCNO_SPA
.EXTRN  SVD$GK_PCNO_HWA
.EXTRN  SVD$GK_PCNO_SNV
.EXTRN  SVD$GK_PCNO_LOA
.EXTRN  SVD$GK_PCNO_SLO
.EXTRN  SVD$GK_PCNO_TLO
.EXTRN  SVD$GK_PCNO_DFL
.EXTRN  SVD$GK_PCNO_SID
.EXTRN  SVD$GK_PCNO_DUM
.EXTRN  SVD$GK_PCNO_SDU
.EXTRN  SVD$GK_PCNO_$HNA
.EXTRN  SVD$GK_PCNO_$HHW
.EXTRN  SVD$GK_PCNO_$FTY
.EXTRN  SVD$GK_PCNO_PHA
.EXTRN  SVD$GK_PCNO_$DA
.EXTRN  SVD$GK_PCNO_LPC
.EXTRN  SVD$GK_PCNO_LPL
.EXTRN  SVD$GK_PCNO_LPD
.EXTRN  SVD$GK_PCNO_LPH
.EXTRN  SVD$GK_PCNO_LPA
.EXTRN  SVD$GK_PCNO_LPN
.EXTRN  SVD$GK_PCNO_$LNA
.EXTRN  SVD$GK_PCNO_$LNH
.EXTRN  SVD$GK_PCNO_LAN
.EXTRN  SVD$GK_PCNO_$LNN
.EXTRN  SVD$GK_PCNO_$LAH
.EXTRN  SVD$GK_PCLI_STI
.EXTRN  SVD$C_ENTRY_COUNT
.EXTRN  MOM$GQ_TIMEOUT, MOM$GQ_DLE_NAMDSC
.EXTRN  MOM$BLD_REPLY, MOM$DEBUG_MSG
.EXTRN  MOM$DEBUG_QIO, MOM$DEBUG_TXT
.EXTRN  SYS$ASSIGN, SYS$QIOW

.PSECT  $CODE$,NOWRT,2
```

```
                                                003C 00000          .ENTRY  MOM$MOPOPEN, Save R2,R3,R4,R5                   ; 0131
                                    55 00000000G    EF   9E 00002   MOVAB   MOM$GL_SERVICE_FLAGS, R5                      ;
                                    54 00000000V    EF   9E 00009   MOVAB   MOM_MAPMOPERRORS, R4                         ;
                                    5E              10   C2 00010   SUBL2   #16, SP                                     ;
                00000000G   EF 00000000*    EF FFFFD8F0 8F   C5 00013   MULL3   #-10000, <<MOM$AB_SERVICE_DATA+-           ; 0164
                                                                            <SVD$GK_PCLI_STI*137>>+9>, MOM$GQ_TIMEOUT   ;
                                                        7E   7C 00023   CLRQ    -(SP)                                   ; 0170
                                                  04    AC   DD 00025   PUSHL   MOP_CHANNEL                             ;
                                        00000000G   EF   9F 00028   PUSHAB  MOM$GQ_DLE_NAMDSC                           ;
                                    00000000G   00      04   FB 0002E   CALLS   #4, SYS$ASSIGN                          ;
                                                52      50   D0 00035   MOVL    R0, STATUS                              ;
                                                        01   DD 00038   PUSHL   #1                                      ; 0171
                                                        7E   D4 0003A   CLRL    -(SP)                                   ;
                                                        52   DD 0003C   PUSHL   STATUS                                  ;
                                                64      03   FB 0003E   CALLS   #3, MOM_MAPMOPERRORS                    ;
                                                6E 00000000*    EF   9A 00041   MOVZBL  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-; 0179
                                                                            >+8>, CIRCUIT_DSC                           ;
                                            04  AE 00000000*    EF   9E 00048   MOVAB   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-; 0180
                                                                            >+9>, CIRCUIT_DSC+4                         ;
                                                53              03   D0 00050   MOVL    #3, RETRY_COUNT                 ; 0186
                                                                53   D7 00053 1$:   DECL    RETRY_COUNT                     ; 0189
                                                                7E   7C 00055   CLRQ    -(SP)                           ; 0193
                                                                7E   7C 00057   CLRQ    -(SP)                           ;
                                                                7E   D4 00059   CLRL    -(SP)                           ;
                                                          14    AE   9F 0005B   PUSHAB  CIRCUIT_DSC                     ;
                                                                7E   7C 0005E   CLRQ    -(SP)                           ;
                                                          28    AE   9F 00060   PUSHAB  IOSB                            ;
                                                                32   DD 00063   PUSHL   #50                             ;
                                                          04    BC   DD 00065   PUSHL   @MOP_CHANNEL                    ;
                                                                7E   D4 00068   CLRL    -(SP)                           ;
                                        00000000G   00          0C   FB 0006A   CALLS   #12, SYS$QIOW                   ;
                                                    52          50   D0 00071   MOVL    R0, STATUS                      ;
                                                        00000000'   EF   9F 00074   PUSHAB  P.AAA                       ; 0195
                                                                7E   7C 0007A   CLRQ    -(SP)                           ; 0194
                                                          0C    AE   9F 0007C   PUSHAB  CIRCUIT_DSC                     ;
                                                                7E   D4 0007F   CLRL    -(SP)                           ;
                                                          1C    AE   9F 00081   PUSHAB  IOSB                            ;
                                                                52   DD 00084   PUSHL   STATUS                          ;
                                                                05   DD 00086   PUSHL   #5                              ;
                                        00000000G   EF          08   FB 00088   CALLS   #8, MOM$DEBUG_QIO               ;
                                                    04          52   E9 0008F   BLBC    STATUS, 2$                      ; 0196
                                                    10    08    AE   E8 00092   BLBS    IOSB, 3$                        ;
                                                                53   D5 00096 2$:   TSTL    RETRY_COUNT                     ; 0199
                                                                B9   14 00098   BGTR    1$                              ;
                                                                01   DD 0009A   PUSHL   #1                              ; 0200
                                                          0C    AE   9F 0009C   PUSHAB  IOSB                            ;
                                                                52   DD 0009F   PUSHL   STATUS                          ;
                                                64              03   FB 000A1   CALLS   #3, MOM_MAPMOPERRORS            ;
                                                                AD   11 000A4   BRB     1$                              ; 0196
                                                03    0C    AE   E9 000A6 3$:   BLBC    IOSB+4, 4$                      ; 0208
                                                65          02   88 000AA   BISB2   #2, MOM$GL_SERVICE_FLAGS          ; 0209
                                                0C          65   E9 000AD 4$:   BLBC    MOM$GL_SERVICE_FLAGS, 5$        ; 0213
                                                          04    BC   DD 000B0   PUSHL   @MOP_CHANNEL                    ; 0215
                                                                06   DD 000B3   PUSHL   #6                              ; 0214
                                        00000000V   EF          02   FB 000B5   CALLS   #2, MOM$MOPSETSUBSTATE          ;
                                                                    04 000BC 5$:   RET                                  ; 0216
```

; Routine Size: 189 bytes,   Routine Base: $CODE$ + 0000

```
;  220          0217  1 %SBTTL 'mom$mopsetsubstate  Set the circuit substate'
;  221          0218  1 GLOBAL ROUTINE mom$mopsetsubstate (substate, mop_chan) : NOVALUE =
;  222          0219  1
;  223          0220  1 !++
;  224          0221  1 ! FUNCTIONAL DESCRIPTION:
;  225          0222  1 !
;  226          0223  1 !       This routine modifies the substate of a circuit that has been opened
;  227          0224  1 !       for service operations.
;  228          0225  1 !
;  229          0226  1 ! INPUTS:
;  230          0227  1 !
;  231          0228  1 !       SUBSTATE        New circuit substate code.
;  232          0229  1 !
;  233          0230  1 !       MOP_CHAN        Channel for MOP circuit control has been initialized.
;  234          0231  1 !
;  235          0232  1 ! OUTPUTS:
;  236          0233  1 !
;  237          0234  1 !       Success/failure of operation.
;  238          0235  1 !--
;  239          0236  1
;  240          0237  2 BEGIN
;  241          0238  2
;  242          0239  2 LOCAL
;  243          0240  2     iosb    : $iosb,
;  244          0241  2     status;
;  245          0242  2
;  246          0243  2
;  247        P 0244  2 status = $QIOW (CHAN = .mop_chan,         ! Channel
;  248        P 0245  2                 FUNC = io$_setmode,      ! Modify substate function
;  249        P 0246  2                 IOSB = iosb,             ! Address of I/O status block
;  250          0247  2                 P4   = substate);        ! New substate value
;  251          0248  2 mom$debug_qio (dbg$c_mopio, .status, iosb, 0, 0, substate, 0,
;  252          0249  2                $ASCID ('Set circuit substate'));
;  253          0250  2
;  254          0251  2 mom_mapmoperrors (.status, iosb, mom$signal)
;  255          0252  2
;  256          0253  1 END;                                ! End of MOM$MOPSETSUBSTATE


                                                            .PSECT  $PLIT$,NOWRT,NOEXE,2

62 75 73 20 74 69 75 63 72 69 63 20 74 65 53  00028 P.AAD:  .ASCII  \Set circuit substate\                    ;
                                  65 74 61 74 73  00037                                                     ;
                                     00000014     0003C P.AAC:  .LONG   20                                  ;
                                     00000000'    00040         .ADDRESS P.AAD                              ;


                                                            .PSECT  $CODE$,NOWRT,2

                                        0004 00000          .ENTRY  MOM$MOPSETSUBSTATE, Save R2              ; 0218
                                  5E       08   C2 00002     SUBL2   #8, SP                                  ;
                                           7E   7C 00005     CLRQ    -(SP)                                   ; 0247
                                     04    AC   9F 00007     PUSHAB  SUBSTATE                                ;
                                           7E   7C 0000A     CLRQ    -(SP)                                   ;
                                           7E   7C 0000C     CLRQ    -(SP)                                   ;
```

```
                                 7E  D4 0000E       CLRL    -(SP)                                          ;
                              20 AE  9F 00010       PUSHAB  IOSB                                           ;
                                 23  DD 00013       PUSHL   #35                                            ;
                              08 AC  DD 00015       PUSHL   MOP_CHAN                                       ;
                                 7E  D4 00018       CLRL    -(SP)                                          ;
                00000000G     00   0C  FB 0001A    CALLS   #12, SYS$QIOW                                  ;
                              52     50  D0 00021    MOVL    R0, STATUS                                     ;
                        00000000'    EF  9F 00024    PUSHAB  P.AAC                                          ; 0249
                                 7E  D4 0002A       CLRL    -(SP)                                          ; 0248
                              04 AC  9F 0002C       PUSHAB  SUBSTATE                                       ;
                                 7E  7C 0002F       CLRQ    -(SP)                                          ;
                              14 AE  9F 00031       PUSHAB  IOSB                                           ;
                                 52  DD 00034       PUSHL   STATUS                                         ;
                                 05  DD 00036       PUSHL   #5                                             ;
                00000000G     EF   08  FB 00038    CALLS   #8, MOM$DEBUG_QIO                              ;
                                 01  DD 0003F       PUSHL   #1                                             ; 0251
                              04 AE  9F 00041       PUSHAB  IOSB                                           ;
                                 52  DD 00044       PUSHL   STATUS                                         ;
                00000000V     EF   03  FB 00046    CALLS   #3, MOM_MAPMOPERRORS                           ;
                                     04  0004D      RET                                                    ; 0253

; Routine Size:  78 bytes,    Routine Base:  $CODE$ + 00BD
```

```
;  258          0254  1 %SBTTL 'mom$init_CIB     Initialize Channel Information Block'
;  259          0255  1 GLOBAL ROUTINE mom$init_CIB (CIB,
;  260          0256  1                                function,
;  261          0257  1                                phys_addr_svd,
;  262          0258  1                                node_addr_svd,
;  263          0259  1                                hardware_addr_svd) : NOVALUE =
;  264          0260  1
;  265          0261  1 !++
;  266          0262  1 ! FUNCTIONAL DESCRIPTION:
;  267          0263  1 !       This routine is called when setting up to perform a maintenance
;  268          0264  1 !       operation.  It initializes the Channel Information Block (CIB),
;  269          0265  1 !       which is used to control establishing the connection with the
;  270          0266  1 !       target.
;  271          0267  1 !
;  272          0268  1 ! INPUTS:
;  273          0269  1 !       CIB     = Address of CIB to initialize.
;  274          0270  1 !       FUNCTION = load, dump, trigger, or test (NICE function code)
;  275          0271  1 !       PHYS_ADDR_SVD = Index of SVD entry containing the NI physical
;  276          0272  1 !               address to  connect to (not always used).
;  277          0273  1 !       NODE_ADDR_SVD = Index of SVD entry containing the node address to
;  278          0274  1 !               connect to (not always used).
;  279          0275  1 !       HARDWARE_ADDR_SVD = Index of SVD entry containing the NI hardware
;  280          0276  1 !               address to connect to (not always used).
;  281          0277  1 !
;  282          0278  1 ! IMPLICIT INPUTS:
;  283          0279  1 !
;  284          0280  1 ! OUTPUTS:
;  285          0281  1 !       Channel Information Block is initialized.
;  286          0282  1 !--
;  287          0283  1
;  288          0284  2 BEGIN
;  289          0285  2
;  290          0286  2 MAP
;  291          0287  2     CIB: REF BBLOCK;
;  292          0288  2
;  293          0289  2 !
;  294          0290  2 ! P2 parameter buffer for SETMODE to NI driver (XXDRIVER).
;  295          0291  2 !
;  296          0292  2 OWN
;  297          0293  2     p2_buffer: BBLOCK [cib$s_setmode_p2_buf] ALIGN (0)
;  298          0294  2                INITIAL (WORD (nma$c_pcli_bus), LONG (0),
;  299          0295  2                         WORD (nma$c_pcli_pad), LONG (0),
;  300          0296  2                         WORD (nma$c_pcli_dch), LONG (nma$c_state_off),
;  301          0297  2                         WORD (nma$c_pcli_crc), LONG (nma$c_state_on),
;  302          0298  2                         WORD (nma$c_pcli_pty), LONG (0),
;  303          0299  2                         WORD (nma$c_pcli_con), LONG (nma$c_lincn_nor),
;  304          0300  2                         WORD (nma$c_pcli_acc), LONG (nma$c_acc_lim),
;  305          0301  2                         WORD (nma$c_pcli_des), WORD (8),
;  306          0302  2                         WORD (nma$c_linmc_set),REP 3 OF WORD (0)),
;  307          0303  2     in_case_eko: BBLOCK [8];
;  308          0304  2
;  309          0305  2 CIB [cib$w_flags] = 0;
;  310          0306  2 CIB [cib$l_retry_cnt] = 5;
;  311          0307  2 !
;  312          0308  2 ! Most of the CIB fields are necessary for NI circuits only.  Point to point
;  313          0309  2 ! circuits (like DMCs) don't need as much.
;  314          0310  2 !
```

```
;   315          0311  2 IF NOT .mom$gl_service_flags [mom$v_ni_circ] THEN
;   316          0312  2     CIB [cib$v_target_addr_fixed] = true
;   317          0313  2 ELSE
;   318          0314  3     BEGIN
;   319          0315  3     !
;   320          0316  3     ! Build P2 buffer containing the parameters to set up the NI for the
;   321          0317  3     ! maintenance operation.
;   322          0318  3     !
;   323          0319  3     CH$MOVE (cib$s_setmode_p2_buf,
;   324          0320  3              P2_buffer,
;   325          0321  3              CIB [cib$t_setmode_p2_buf]);
;   326          0322  3     !
;   327          0323  3     ! Set up the NI protocol type and padding to give the NI driver.  The NI has
;   328          0324  3     ! the capability to run several protocols at once.  MOM will set it up to
;   329          0325  3     ! run the Load/Dump, the Remote Console, or the Loopback protocol.  Padding
;   330          0326  3     ! on means the NI driver will add a word of count to a buffer when it
;   331          0327  3     ! transmits it.  Loop messages use a skip field instead.
;   332          0328  3     !
;   333          0329  3     IF .function EQL nma$c_fnc_tes THEN
;   334          0330  4         BEGIN
;   335          0331  4         CIB [cib$l_p2_protocol] = mom$k_loop_ni_prot;
;   336          0332  4         CIB [cib$l_p2_padding] = nma$c_state_off;
;   337          0333  4         CIB [cib$l_p2_buf_siz] = 1500;
;   338          0334  4         END
;   339          0335  3     ELSE
;   340          0336  4         BEGIN
;   341          0337  4         IF .function EQL nma$c_fnc_tri THEN
;   342          0338  4             CIB [cib$l_p2_protocol] = mom$k_console_ni_prot
;   343          0339  4         ELSE
;   344          0340  4             CIB [cib$l_p2_protocol] = mom$k_loadump_ni_prot;
;   345          0341  4         CIB [cib$l_p2_padding] = nma$c_state_on;
;   346          0342  4         CIB [cib$l_p2_buf_siz] = 1498;          ! Leave room for padding.
;   347          0343  3         END;
;   348          0344  3     !
;   349          0345  3     ! Set up the NI address for the target.
;   350          0346  3     !
;   351          0347  3     IF .mom$ab_service_data [.phys_addr_svd, svd$v_msg_param] THEN
;   352          0348  3         !
;   353          0349  3         ! If there was a Physical Address specified in the NICE command
;   354          0350  3         ! (operservice) or it's autoservice (and the target's physical address
;   355          0351  3         ! was in the NI header), do the operation to that address on the NI.
;   356          0352  3         !
;   357          0353  4         BEGIN
;   358          0354  4         CH$MOVE (mom$k_ni_addr_length,
;   359          0355  4                  mom$ab_service_data [.phys_addr_svd, svd$t_string],
;   360          0356  4                  CIB [cib$t_ni_phys_addr]);
;   361          0357  4         CIB [cib$v_target_addr_fixed] = true;
;   362          0358  4         END
;   363          0359  3     ELSE
;   364          0360  3         !
;   365          0361  3         ! This operation does not have a Physical Address supplied.  In
;   366          0362  3         ! this case, if the hardware address is available, set up to alternate
;   367          0363  3         ! trying the target's hardware NI address and it's HIORD NI address
;   368          0364  3         ! (the DEC NI prefix concatenated with the node's address).
;   369          0365  3         !
;   370          0366  4         BEGIN
;   371          0367  4         CIB [cib$l_ni_hiord_pref] = mom$k_ni_prefix;
```

```
;  372     0368  4          CIB [cib$w_ni_hiord_node] = .mom$ab_service_data [.node_addr_svd,
;  373     0369  4                                                           svd$[_param];
;  374     0370  4
;  375     0371  4          !
;  376     0372  4          ! Initialize the SVD physical address to something.
;  377     0373  4          !
;  378     0374  4          CH$MOVE (mom$k_ni_addr_length,
;  379     0375  4                   CIB [cib$t_ni_hiord_addr],
;  380     0376  4                   mom$ab_service_data [.phys_addr_svd, svd$t_string]);
;  381     0377  4          mom$ab_service_data [.phys_addr_svd, svd$b_string_len] =
;  382     0378  4                                                  mom$k_ni_addr_length;
;  383     0379  4          IF .mom$ab_service_data [.hardware_addr_svd, svd$b_string_len]
;  384     0380  5                                                               NEQ 0 THEN
;  385     0381  5              BEGIN
;  386     0382  5              CH$MOVE (mom$k_ni_addr_length,
;  387     0383  5                       mom$ab_service_data [.hardware_addr_svd, svd$t_string],
;  388     0384  5                       CIB [cib$t_ni_hardwr_addr]);
;  389     0385  5              CIB [cib$v_target_addr_fixed] = false;
;  390     0386  5              !
;  391     0387  5              ! When attempting to establish communication with the target, try
;  392     0388  5              ! the HIORD address first.
;  393     0389  5              !
;  394     0390  5              CH$MOVE (mom$k_ni_addr_length,
;  395     0391  5                       CIB [cib$t_ni_hardwr_addr],
;  396     0392  5                       CIB [cib$t_ni_phys_addr]);
;  397     0393  4              END
;  398     0394  4          ELSE
;  399     0395  4          !
;  400     0396  4          ! If there is no hardware address in the volatile database, simply try
;  401     0397  4          ! communicating with the target using the HIORD NI address.
;  402     0398  5          !
;  403     0399  5              BEGIN
;  404     0400  5              CH$MOVE (mom$k_ni_addr_length,
;  405     0401  5                       CIB [cib$t_ni_hiord_addr],
;  406     0402  5                       CIB [cib$t_ni_phys_addr]);
;  407     0403  5              CIB [cib$v_target_addr_fixed] = true;
;  408     0404  3              END;
;  409     0405  3          !
;  410     0406  3          ! Issue the SETMODE to tell the NI driver what protocol and NI address
;  411     0407  3          ! I want to use.  The NI address set up will be the one to try first.
;  412     0408  3          !
;  413     0409  3          mom$set_NI_addr (.CIB);
;  414     0410  2          END;
;  415     0411  1 END;                                ! End of mom$init_CIB


                                                     .PSECT  $OWN$,NOEXE,2

                                        0AF1  00000 P2_BUFFER:
                                                     .WORD   2801
                                    00000000  00002          .LONG   0
                                        0B1A  00006          .WORD   2842
                                    00000000  00008          .LONG   0
                                        0B1B  0000C          .WORD   2843
                                    00000001  0000E          .LONG   1
                                        0B1C  00012          .WORD   2844
```

```
                               00000000  00014             .LONG    0                                                   ;
                                   0B0E  00018             .WORD    2830                                                ;
                               00000000  0001A             .LONG    0                                                   ;
                                   0456  0001E             .WORD    1110                                                ;
                               00000000  00020             .LONG    0                                                   ;
                                   0B1E  00024             .WORD    2846                                                ;
                               00000002  00026             .LONG    2                                                   ;
                                   0B21  0002A             .WORD    2849                                                ;
                                   0008  0002C             .WORD    8                                                   ;
                                   0001  0002E             .WORD    1                                                   ;
                                  0000#  00030             .WORD    0[3]                                                ;
                                         00036             .BLKB    2
                                         00038 IN_CASE_EKO:
                                                           .BLKB    8


                                                           .PSECT   $CODE$,NOWRT,2

                                   03FC  00000             .ENTRY   MOM$INIT_CIB, Save R2,R3,R4,R5,R6,R7,R8,R9     ; 0255
             59 00000000G     EF     9E  00002             MOVAB    MOM$AB_SERVICE_DATA+9, R9                      ;
             57          04   AC     D0  00009             MOVL     CIB, R7                                         ; 0305
             58          10   A7     9E  0000D             MOVAB    16(R7), R8                                      ;
                              68     B4  00011             CLRW     (R8)                                            ;
         12  A7               05     D0  00013             MOVL     #5, 18(R7)                                      ; 0306
   04 00000000G               EF 01  E0  00017             BBS      #1, MOM$GL_SERVICE_FLAGS, 1$                    ; 0311
             68               01     88  0001F             BISB2    #1, (R8)                                        ; 0312
                                     04  00022             RET                                                      ;
16  A7 00000000'              EF 36  28  00023 1$:         MOVC3    #54, P2_BUFFER, 22(R7)                          ; 0321
             12          08   AC     D1  0002C             CMPL     FUNCTION, #18                                   ; 0329
                              11     12  00030             BNEQ     2$                                              ;
         30  A7          90   8F     9A  00032             MOVZBL   #144, 48(R7)                                    ; 0331
         1E  A7               01     D0  00037             MOVL     #1, 30(R7)                                      ; 0332
         18  A7        05DC   8F     3C  0003B             MOVZWL   #1500, 24(R7)                                   ; 0333
                              1D     11  00041             BRB      5$                                              ; 0329
             11          08   AC     D1  00043 2$:         CMPL     FUNCTION, #17                                   ; 0337
                              08     12  00047             BNEQ     3$                                              ;
         30  A7        0260   8F     3C  00049             MOVZWL   #608, 48(R7)                                    ; 0338
                              06     11  0004F             BRB      4$                                              ;
         30  A7        0160   8F     3C  00051 3$:         MOVZWL   #352, 48(R7)                                    ; 0340
                         1E   A7     D4  00057 4$:         CLRL     30(R7)                                          ; 0341
         18  A7        05DA   8F     3C  0005A             MOVZWL   #1498, 24(R7)                                   ; 0342
     56  0C  AC    00000089   8F     C5  00060 5$:         MULL3    #137, PHYS_ADDR_SVD, R6                         ; 0347
     08  FE A946              00     E1  00069             BBC      #0, MOM$AB_SERVICE_DATA+7[R6], 6$               ;
46   A7     6946              06     28  0006F             MOVC3    #6, MOM$AB_SERVICE_DATA+9[R6], 70(R7)           ; 0356
                              49     11  00075             BRB      8$                                              ; 0357
         04  A7    000400AA   8F     D0  00077 6$:         MOVL     #262314, 4(R7)                                  ; 0367
     50  10  AC    00000089   8F     C5  0007F             MULL3    #137, NODE_ADDR_SVD, R0                         ; 0369
                            6940     9F  00088             PUSHAB   MOM$AB_SERVICE_DATA+9[R0]                       ; 0368
         08  A7               9E     B0  0008B             MOVW     @(SP)+, 8(R7)                                   ;
   6946  04  A7               06     28  0008F             MOVC3    #6, 4(R7), MOM$AB_SERVICE_DATA+9[R6]            ; 0375
         FF A946              06     90  00095             MOVB     #6, MOM$AB_SERVICE_DATA+8[R6]                   ; 0376
     50  14  AC    00000089   8F     C5  0009A             MULL3    #137, HARDWARE_ADDR_SVD, R0                     ; 0378
                         FF A940     95  000A3             TSTB     MOM$AB_SERVICE_DATA+8[R0]                       ; 0379
                              11     13  000A7             BEQL     7$                                              ;
0A   A7     6940              06     28  000A9             MOVC3    #6, MOM$AB_SERVICE_DATA+9[R0], 10(R7)           ; 0383
            68                01     8A  000AF             BICB2    #1, (R8)                                        ; 0384
```

```
                46   A7        0A   A7          06  28 000B2          MOVC3   #6, 10(R7), 70(R7)          ; 0391
                                                09  11 000B8          BRB     9$                          ; 0378
                46   A7        04   A7          06  28 000BA 7$:      MOVC3   #6, 4(R7), 70(R7)           ; 0401
                                    68          01  88 000C0 8$:      BISB2   #1, (R8)                    ; 0402
                                                57  DD 000C3 9$:      PUSHL   R7                          ; 0409
                      00000000V   EF            01  FB 000C5          CALLS   #1, MOM$SET_NI_ADDR         ;
                                                    04 000CC          RET                                 ; 0411

; Routine Size:  205 bytes,    Routine Base:  $CODE$ + 010B
```

```
;   417          0412  1 %SBTTL 'mom$set_NI_addr     Tell NI driver targets NI address'
;   418          0413  1 GLOBAL ROUTINE mom$set_NI_addr (CIB) : NOVALUE =
;   419          0414  1
;   420          0415  1 !++
;   421          0416  1 ! FUNCTIONAL DESCRIPTION:
;   422          0417  1 !       This routine is called when setting up to perform a maintenance
;   423          0418  1 !       operation over an NI circuit.  It tells the NI driver what
;   424          0419  1 !       protocol and destination NI address (among other things) to use
;   425          0420  1 !       when transmitting and receiving MOP messages.
;   426          0421  1 !
;   427          0422  1 ! INPUTS:
;   428          0423  1 !       CIB     = Address of Channel Information Block.  This contains the
;   429          0424  1 !                 NI addresses to try, and a prebuilt P2 buffer to give
;   430          0425  1 !                 the NI driver.
;   431          0426  1 !
;   432          0427  1 ! IMPLICIT INPUTS:
;   433          0428  1 !
;   434          0429  1 ! OUTPUTS:
;   435          0430  1 !--
;   436          0431  1
;   437          0432  2 BEGIN
;   438          0433  2
;   439          0434  2 MAP
;   440          0435  2     CIB: REF BBLOCK;
;   441          0436  2
;   442          0437  2 LOCAL
;   443          0438  2     p2_dsc: VECTOR [2],
;   444          0439  2     iosb: $iosb,
;   445          0440  2     status;
;   446          0441  2
;   447          0442  2 p2_dsc [0] = cib$s_setmode_p2_buf;
;   448          0443  2 p2_dsc [1] = CIB [cib$t_setmode_p2_buf];
;   449          0444  2
;   450          0445  2 IF (NOT .CIB [cib$v_target_addr_fixed]) THEN
;   451          0446  2     !
;   452          0447  2     ! If the operation wasn't requested with a specific NI physical address,
;   453          0448  2     ! then alternately retry the hardware address from the volatile
;   454          0449  2     ! database and the NI HIORD address (the DEC NI prefix concatenated
;   455          0450  2     ! with the node number.)  This is done because the target's UNA responds
;   456          0451  2     ! to it's hardware address after it's been powered up, but when DECnet
;   457          0452  2     ! is started up, it will change the NI address that the UNA responds to
;   458          0453  2     ! to the HIORD address.  After a crash, the target's UNA will still be
;   459          0454  2     ! running the DECnet NI address.  Since there is no way to tell which
;   460          0455  2     ! one the target is currently answering to, alternately try both.
;   461          0456  2     !
;   462          0457  3     BEGIN
;   463          0458  3     IF CH$EQL (mom$k_ni_addr_length,
;   464          0459  3                CIB [cib$t_ni_hiord_addr],
;   465          0460  3                mom$k_ni_addr_length,
;   466          0461  3                CIB [cib$t_ni_phys_addr], 0) THEN
;   467          0462  3         CH$MOVE (mom$k_ni_addr_length,
;   468          0463  3                  CIB [cib$t_ni_hardwr_addr],
;   469          0464  3                  CIB [cib$t_ni_phys_addr])
;   470          0465  3     ELSE
;   471          0466  3         CH$MOVE (mom$k_ni_addr_length,
;   472          0467  3                  CIB [cib$t_ni_hiord_addr],
;   473          0468  3                  CIB [cib$t_ni_phys_addr]);
```

```
;  474          0469  2     END;
;  475          0470  2
;  476        P 0471  2 status = $QIOW (CHAN = .CIB [cib$l_chan],
;  477        P 0472  2                 FUNC = io$_setmode OR io$m_ctrl,
;  478        P 0473  2                 IOSB = iosb,
;  479        P 0474  2                 P2 = p2_dsc,
;  480          0475  2                 P3 = CIB [cib$t_ni_phys_addr]);
;  481          0476  2 mom$debug_qio (dbg$c_mopio,
;  482          0477  2                .status,
;  483          0478  2                iosb, 0, p2_dsc, CIB [cib$t_ni_phys_addr], 0,
;  484          0479  2                $ASCID ('IO$_SETMODE QIO on MOP channel'));
;  485          0480  2 !
;  486          0481  2 ! If the SETMODE didn't work, get out.
;  487          0482  2 !
;  488          0483  2 mom_mapmoperrors (.status, iosb, mom$signal);
;  489          0484  2
;  490          0485  1 END;                                ! End of mom$set_NI_addr


                                                            .PSECT  $PLIT$,NOWRT,NOEXE,2

4F  49  51  20  45  44  4F  4D  54  45  53  5F  24  4F  49  00044 P.AAF:  .ASCII  \IO$_SETMODE QIO on MOP channel\          ;
6C  65  6E  6E  61  68  63  20  50  4F  4D  20  6E  6F  20  00053                                                          ;
                                                            00062       .BLKB   2
                                                0000001E    00064 P.AAE:  .LONG   30                                      ;
                                                00000000'   00068       .ADDRESS P.AAF                                    ;


                                                            .PSECT  $CODE$,NOWRT,2

                                                        007C 00000      .ENTRY  MOM$SET_NI_ADDR, Save R2,R3,R4,R5,R6      ; 0413
                                                    5E  10   C2 00002   SUBL2   #16, SP                                   ;
                                                08  AE  36   D0 00005   MOVL    #54, P2_DSC                               ; 0442
                                                    56  04   AC D0 00009   MOVL    CIB, R6                               ; 0443
                                                0C  AE  16   A6 9E 0000D   MOVAB   22(R6), P2_DSC+4                      ;
                                                    16  10   A6 E8 00012   BLBS    16(R6), 2$                            ; 0445
46  A6          04  A6          06   29 00016   CMPC3   #6, 4(R6), 70(R6)                                               ; 0461
                                    08   12 0001C   BNEQ    1$                                                          ;
46  A6          0A  A6          06   28 0001E   MOVC3   #6, 10(R6), 70(R6)                                             ; 0464
                                    06   11 00024   BRB     2$                                                          ;
46  A6          04  A6          06   28 00026 1$: MOVC3   #6, 4(R6), 70(R6)                                            ; 0468
                                    7E   7C 0002C 2$: CLRQ    -(SP)                                                     ; 0475
                                    7E   D4 0002E   CLRL    -(SP)                                                       ;
                                46  A6   9F 00030   PUSHAB  70(R6)                                                      ;
                                18  AE   9F 00033   PUSHAB  P2_DSC                                                      ;
                                    7E   7C 00036   CLRQ    -(SP)                                                       ;
                                    7E   D4 00038   CLRL    -(SP)                                                       ;
                                20  AE   9F 0003A   PUSHAB  IOSB                                                       ;
                            7E      0223 8F   3C 0003D   MOVZWL  #547, -(SP)                                            ;
                                    66   DD 00042   PUSHL   (R6)                                                        ;
                                    7E   D4 00044   CLRL    -(SP)                                                       ;
                    00000000G   00      0C   FB 00046   CALLS   #12, SYS$QIOW                                        ;
                            52          50   D0 0004D   MOVL    R0, STATUS                                            ;
                        00000000'       EF   9F 00050   PUSHAB  P.AAE                                                 ; 0479
                                    7E   D4 00056   CLRL    -(SP)                                                       ; 0478
```

```
                                 46   A6  9F 00058        PUSHAB  70(R6)                                        ;
                                 14   AE  9F 0005B        PUSHAB  P2_DSC                                      ; 0476
                                      7E  D4 0005E        CLRL    -(SP)                                       ; 0478
                                 14   AE  9F 00060        PUSHAB  IOSB                                        ; 0476
                                      52  DD 00063        PUSHL   STATUS                                      ; 0478
                                      05  DD 00065        PUSHL   #5                                          ;
                    00000000G   EF        08  FB 00067    CALLS   #8, MOM$DEBUG_QIO                           ;
                                      01  DD 0006E        PUSHL   #1                                          ; 0483
                                 04   AE  9F 00070        PUSHAB  IOSB                                        ;
                                      52  DD 00073        PUSHL   STATUS                                      ;
                    00000000V   EF        03  FB 00075    CALLS   #3, MOM_MAPMOPERRORS                        ;
                                          04  0007C        RET                                                 ; 0485

; Routine Size:  125 bytes,    Routine Base:  $CODE$ + 01D8
```

```
;  492          0486  1 %SBTTL 'mom$mopsndrcv     Send and receive a MOP mode message'
;  493          0487  1 GLOBAL ROUTINE mom$mopsndrcv (send_CIB, xmit_msg_dsc,
;  494          0488  1                             rcv_CIB, recv_buf_dsc, recv_msg_len,
;  495          0489  1                             skip_msg_dsc) =
;  496          0490  1
;  497          0491  1 !++
;  498          0492  1 ! FUNCTIONAL DESCRIPTION:
;  499          0493  1 !
;  500          0494  1 !       This routine performs a transmit operation followed by a receive
;  501          0495  1 !       operation.  Actually, the receive is issued before the transmit
;  502          0496  1 !       to insure that a buffer is available for the receive.
;  503          0497  1 !
;  504          0498  1 ! INPUTS:
;  505          0499  1 !
;  506          0500  1 !       SEND_CIB = Information Block for channel to transmit MOP message on.
;  507          0501  1 !       XMIT_MSG_DSC = Addres of descriptor of data to be transmitted.
;  508          0502  1 !       RCV_CIB = Information Block for channel to receive response MOP
;  509          0503  1 !                 message on.
;  510          0504  1 !       RECV_BUF_DSC    Address of descriptor of receive buffer.
;  511          0505  1 !       RECV_MSG_LEN    Address of word to contain length of received data.
;  512          0506  1 !       SKIP_MSG_DSC  = Address of descriptor of received message to skip.
;  513          0507  1 !                       On the NI, a node requesting a Program Load sends it
;  514          0508  1 !                       repeatedly until a response is received from a host.
;  515          0509  1 !                       This means that there could be more than one Progam
;  516          0510  1 !                       Load request backed up by the time the MOM sends it's
;  517          0511  1 !                       response.  So, skip over these until the real response
;  518          0512  1 !                       is received.
;  519          0513  1 !                       Set to -1 if this routine should skip over all received
;  520          0514  1 !                       messages which were sent to the multicast NI address.
;  521          0515  1 !
;  522          0516  1 ! IMPLICIT INPUTS:
;  523          0517  1 !       The channel to the target has been opened, and, if it's a NI circuit,
;  524          0518  1 !       initted for the first attempted destination address.
;  525          0519  1 !
;  526          0520  1 ! OUTPUTS:
;  527          0521  1 !
;  528          0522  1 !       Success/failure of operation.
;  529          0523  1 !--
;  530          0524  1
;  531          0525  2 BEGIN
;  532          0526  2
;  533          0527  2 MAP
;  534          0528  2     send_CIB      : REF BBLOCK,
;  535          0529  2     rcv_CIB       : REF BBLOCK,
;  536          0530  2     xmit_msg_dsc  : REF VECTOR,
;  537          0531  2     recv_buf_dsc  : REF VECTOR,
;  538          0532  2     skip_msg_dsc  : REF VECTOR;
;  539          0533  2
;  540          0534  2 OWN
;  541          0535  2     iosb    : $IOSB,
;  542          0536  2     ni_header : BBLOCK [nih$k_ni_header_len];
;  543          0537  2
;  544          0538  2 LOCAL
;  545          0539  2     retry,
;  546          0540  2     status,
;  547          0541  2     rstatus,
;  548          0542  2     skip_msg_max,
```

```
;  549          0543  2     eflags;
;  550          0544  2
;  551          0545  2 !
;  552          0546  2 ! Post a receive for the response MOP message before sending the MOP message
;  553          0547  2 ! which the response will be for.
;  554          0548  2 !
;  555          0549  2 rstatus = mom_mop_receive (.rcv_CIB [cib$l_chan], .recv_buf_dsc, iosb, ni_header);
;  556          0550  2 IF NOT .rstatus THEN
;  557          0551  2     RETURN mom_mapmoperrors (.rstatus, 0, mom$nosignal);
;  558          0552  2 skip_msg_max = 20;
;  559          0553  2 retry = .send_CIB [cib$l_retry_cnt];
;  560          0554  2
;  561          0555  2 WHILE .retry GTR 0 DO
;  562          0556  3     BEGIN
;  563          0557  3     !
;  564          0558  3     ! Transmit the MOP message to the target load, loop, or dump node.
;  565          0559  3     !
;  566          0560  3     status = mom$mopsend (.send_CIB,
;  567          0561  3                           .xmit_msg_dsc [1],
;  568          0562  3                           .xmit_msg_dsc [0]);
;  569          0563  3     IF NOT .status THEN
;  570          0564  3         EXITLOOP;
;  571          0565  3     !
;  572          0566  3     ! Issue a read for a response from the target, and then wait for the
;  573          0567  3     ! or a timeout, whichever comes first.
;  574          0568  3     !
;  575          0569  3     WHILE true DO
;  576          0570  4         BEGIN
;  577          0571  4         !
;  578          0572  4         ! Start the timer for the response MOP message just sent.
;  579          0573  4         !
;  580        P 0574  4         $SETIMR (EFN    = mom$k_moptimefn,
;  581          0575  4                  DAYTIM = mom$gq_timeout);
;  582        P 0576  4         $WFLOR (EFN  = 0,
;  583          0577  4                 MASK = mom$m_moptimefn OR mom$m_moprcvefn);
;  584          0578  4         !
;  585          0579  4         ! If the timer expired then return an error.  If the receive
;  586          0580  4         ! completed then check it.
;  587          0581  4         !
;  588        P 0582  4         $READEF (EFN   = 0,
;  589          0583  4                  STATE = eflags);
;  590          0584  4         IF .eflags <mom$k_moptimefn,1> THEN
;  591          0585  5             BEGIN
;  592          0586  5             mom$debug_txt (dbg$c_mopio,
;  593          0587  5                            $ASCID ('QIO to MOP channel has timed out'));
;  594          0588  5             status = ss$_timeout;
;  595          0589  5             EXITLOOP;
;  596          0590  5             END
;  597          0591  4         ELSE
;  598          0592  4             !
;  599          0593  4             ! The receive did not time out.  Check to see if it completed
;  600          0594  4             ! with an error.
;  601          0595  4             !
;  602          0596  5             BEGIN
;  603          0597  5             $CANTIM ();
;  604          0598  5             mom$debug_qio (dbg$c_mopio, .rstatus, iosb, 0, 0, 0, 0,
;  605          0599  5                            $ASCID ('MOP receive completion status'));
```

```
;  606          0600  5              status = .iosb [ios$w_status];
;  607          0601  5              (.recv_msg_len)<0,16> = .iosb [ios$w_count];
;  608          0602  5              !
;  609          0603  5              ! If the receive completion status was bad or the received
;  610          0604  5              ! message was 0 bytes in length, retransmit the send MOP message.
;  611          0605  5              ! The latter check is for point-to-point lines.
;  612          0606  5              !
;  613          0607  5              IF (NOT .status) OR
;  614          0608  5                 (.recv_msg_len)<0,16> EQL 0 THEN
;  615          0609  5                  EXITLOOP;
;  616          0610  5              IF .status THEN
;  617          0611  6                  BEGIN
;  618          0612  6                  !
;  619          0613  6                  ! A response was received from the target, so the correct NI
;  620          0614  6                  ! address is in use.
;  621          0615  6                  !
;  622          0616  6                  send_CIB [cib$v_target_addr_fixed] = true;
;  623          0617  6                  rcv_CIB [cib$v_target_addr_fixed] = true;
;  624          0618  6                  mom$debug_msg ( dbg$c_mopio,
;  625          0619  6                                  .recv_buf_dsc [1],
;  626          0620  6                                  .(.recv_msg_len)<0,16>,
;  627          0621  6                                  $ASCID ('MOP message received'));
;  628          0622  6                  !
;  629          0623  6                  ! On NI links, at the beginning of each program load
;  630          0624  6                  ! (secondary, tertiary, or operating system), there is a chance
;  631          0625  6                  ! that the load or dump request has been retransmitted.  If
;  632          0626  6                  ! so, throw it away, and issue another read.  The caller can
;  633          0627  6                  ! specify either a specific message to skip or that multicasts
;  634          0628  6                  ! be skipped.
;  635          0629  6                  !
;  636          0630  6                  IF .skip_msg_dsc EQL 0 THEN
;  637          0631  6                      EXITLOOP
;  638          0632  6                  ELSE
;  639          0633  7                      BEGIN
;  640          0634  7                      IF .skip_msg_dsc GTR 0 THEN
;  641          0635  8                          BEGIN
;  642          0636  8                          IF CH$NEQ (.skip_msg_dsc [0], .skip_msg_dsc [1],
;  643          0637  8                                     .skip_msg_dsc [0], .recv_buf_dsc [1], 0) THEN
;  644          0638  8                              EXITLOOP;
;  645          0639  8                          END
;  646          0640  7                      ELSE
;  647          0641  7                          !
;  648          0642  7                          ! Skip over multicasts.
;  649          0643  7                          !
;  650          0644  7                          IF NOT .ni_header [nih$b_multicast] THEN
;  651          0645  7                              EXITLOOP;
;  652          0646  6                      END;
;  653          0647  5                  END;
;  654          0648  5              skip_msg_max = .skip_msg_max - 1;
;  655          0649  5              !
;  656          0650  5              ! Just in case I'm skipping a message that the target is
;  657          0651  5              ! retransmitting over and over, limit the number of skipped
;  658          0652  5              ! messages.
;  659          0653  5              !
;  660          0654  5              IF .skip_msg_max EQL 0 THEN
;  661          0655  6                  BEGIN
;  662          0656  6                  status = ss$_timeout;
```

```
;  663          0657  6                 EXITLOOP;
;  664          0658  5                 END;
;  665          0659  5             rstatus = mom_mop_receive (.rcv_CIB [cib$l_chan],
;  666          0660  5                                        .recv_buf_dsc,
;  667          0661  5                                        iosb,
;  668          0662  5                                        ni_header);
;  669          0663  5             IF NOT .rstatus THEN
;  670          0664  5                 RETURN mom_mapmoperrors (.rstatus, 0, mom$nosignal);
;  671          0665  4             END;
;  672          0666  3         END;
;  673          0667  3         IF .status NEQ ss$_timeout THEN
;  674          0668  3             EXITLOOP;
;  675          0669  3         !
;  676          0670  3         ! Decrement the retry count and retry the I/O.
;  677          0671  3         !
;  678          0672  3         retry = .retry - 1;
;  679          0673  3         !
;  680          0674  3         ! For circuit loop tests, if the NI address of the target is not
;  681          0675  3         ! already known, set up to try an alternate one.  The HIORD and
;  682          0676  3         ! hardware addresses for the target are alternately retried.
;  683          0677  3         !
;  684          0678  3         IF .mom$gl_service_flags [mom$v_ni_circ] AND
;  685          0679  3            .retry GTR 0 THEN
;  686          0680  4             BEGIN
;  687          0681  4             IF NOT .send_CIB [cib$v_target_addr_fixed] THEN
;  688          0682  4                 mom$set_NI_addr (.send_CIB);
;  689          0683  4             IF .send_CIB NEQ .rcv_CIB AND
;  690          0684  4                NOT .rcv_CIB [cib$v_target_addr_fixed] THEN
;  691          0685  4                 mom$set_NI_addr (.rcv_CIB);
;  692          0686  3             END;
;  693          0687  2         END;
;  694          0688  2 !
;  695          0689  2 ! Check the NI header to see if the MOP message was sent to the
;  696          0690  2 ! NI multicast address (meaning that any NI node enabled for multicast
;  697          0691  2 ! can respond to the message.)
;  698          0692  2 !
;  699          0693  2 IF .mom$gl_service_flags [mom$v_ni_circ] AND
;  700          0694  2    .status THEN
;  701          0695  3     BEGIN
;  702          0696  3     IF .ni_header [nih$b_multicast] THEN
;  703          0697  3         mom$gl_service_flags [mom$v_ni_multicast] = true
;  704          0698  3     ELSE
;  705          0699  3         mom$gl_service_flags [mom$v_ni_multicast] = false;
;  706          0700  2     END;
;  707          0701  2 !
;  708          0702  2 ! If no response was received from the target node, cancel the I/Os
;  709          0703  2 ! on the MOP channels.  The only time the send channel isn't the same
;  710          0704  2 ! as the receive channel is for loop with assist on the Ethernet.
;  711          0705  2 !
;  712          0706  2 IF .retry LEQ 0 THEN
;  713          0707  3     BEGIN
;  714          0708  3     $CANCEL (CHAN = .send_CIB [cib$l_chan]);
;  715          0709  3     IF .send_CIB [cib$l_chan] NEQ .rcv_CIB [cib$l_chan] THEN
;  716          0710  3         $CANCEL (CHAN = .rcv_CIB [cib$l_chan]);
;  717          0711  2     END;
;  718          0712  2 RETURN mom_mapmoperrors (.status, 0, mom$nosignal)
;  719          0713  2
```

```
;  720          0714  1 END;                                ! End of mom$mopsndrcv


                                                           .PSECT  $PLIT$,NOWRT,NOEXE,2

6E 61 68 63 20 50 4F 4D 20 6F 74 20 4F 49 51  0006C P.AAH:  .ASCII  \QIO to MOP channel has timed out\                ;
6F 20 64 65 6D 69 74 20 73 61 68 20 6C 65 6E  0007B                                                                  ;
                                              0008A                                                                  ;
                                   00000020   0008C P.AAG:  .LONG   32                                              ;
                                   00000000'  00090         .ADDRESS P.AAH                                         ;
6D 6F 63 20 65 76 69 65 63 65 72 20 50 4F 4D  00094 P.AAJ:  .ASCII  \MOP receive completion status\                 ;
   73 75 74 61 74 73 20 6E 6F 69 74 65 6C 70  000A3                                                                  ;
                                              000B1         .BLKB   3                                              ;
                                   0000001D   000B4 P.AAI:  .LONG   29                                             ;
                                   00000000'  000B8         .ADDRESS P.AAJ                                         ;
63 65 72 20 65 67 61 73 73 65 6D 20 50 4F 4D  000BC P.AAL:  .ASCII  \MOP message received\                           ;
                                        64 65 76 69 65 72  000CB                                                     ;
                                   00000014   000D0 P.AAK:  .LONG   20                                             ;
                                   00000000'  000D4         .ADDRESS P.AAL                                         ;

                                                           .PSECT  $OWN$,NOEXE,2

                                              00040 IOSB:  .BLKB   8
                                              00048 NI_HEADER:
                                                           .BLKB   14

                                                           .EXTRN  SYS$SETIMR, SYS$WFLOR
                                                           .EXTRN  SYS$READEF, SYS$CANTIM
                                                           .EXTRN  SYS$CANCEL

                                                           .PSECT  $CODE$,NOWRT,2

                                     0FFC 00000            .ENTRY  MOM$MOPSNDRCV, Save R2,R3,R4,R5,R6,R7,R8,-      ; 0487
                                                                   R9,R10,R11                                       ;
                                5E      04   C2 00002              SUBL2   #4, SP                                  ;
                     00000000'     EF   9F 00005              PUSHAB  NI_HEADER                                   ; 0549
                     00000000'     EF   9F 0000B              PUSHAB  IOSB                                        ;
                                55      0C   AC 7D 00011              MOVQ    RCV_CIB, R5                         ;
                                             56 DD 00015              PUSHL   R6                                  ;
                                             65 DD 00017              PUSHL   (R5)                                ;
00000000V                       EF           04 FB 00019              CALLS   #4, MOM_MOP_RECEIVE                 ;
                                59           50 D0 00020              MOVL    R0, RSTATUS                         ;
                                03           59 E8 00023              BLBS    RSTATUS, 1$                         ; 0550
                                           0104 31 00026              BRW     11$                                 ;
                                5B           14 D0 00029 1$:          MOVL    #20, SKIP_MSG_MAX                   ; 0552
                                54      04   AC D0 0002C              MOVL    SEND_CIB, R4                        ; 0553
                                5A      12   A4 D0 00030              MOVL    18(R4), RETRY                       ;
                                58      08   AC D0 00034              MOVL    XMIT_MSG_DSC, R8                    ; 0561
                                             5A D5 00038 2$:          TSTL    RETRY                               ; 0555
                                             03 14 0003A              BGTR    4$                                  ;
                                           0127 31 0003C 3$:          BRW     15$                                 ;
                                      08     BC DD 0003F 4$:          PUSHL   @XMIT_MSG_DSC                       ; 0562
                                      04     A8 DD 00042              PUSHL   4(R8)                               ; 0561
                                             54 DD 00045              PUSHL   R4                                  ; 0560
00000000V                       EF           03 FB 00047              CALLS   #3, MOM$MOPSEND                     ;
                                57           50 D0 0004E              MOVL    R0, STATUS                          ;
```

```
                                E8                  57  E9 00051                BLBC    STATUS, 3$                          ; 0563
                                                    7E  7C 00054 5$:         CLRQ    -(SP)                               ; 0575
                                   00000000G        EF  9F 00056             PUSHAB  MOM$GQ_TIMEOUT                      ;
                                                    03  DD 0005C             PUSHL   #3                                  ;
                          00000000G  00             04  FB 0005E             CALLS   #4, SYS$SETIMR                      ;
                                                    0A  DD 00065             PUSHL   #10                                 ; 0577
                                                    7E  D4 00067             CLRL    -(SP)                               ;
                          00000000G  00             02  FB 00069             CALLS   #2, SYS$WFLOR                       ;
                                                    5E  DD 00070             PUSHL   SP                                  ; 0583
                                                    7E  D4 00072             CLRL    -(SP)                               ;
                          00000000G  00             02  FB 00074             CALLS   #2, SYS$READEF                      ;
                    11                6E            03  E1 0007B             BBC     #3, EFLAGS, 6$                      ; 0584
                                   00000000'        EF  9F 0007F             PUSHAB  P.AAG                               ; 0587
                                                    05  DD 00085             PUSHL   #5                                  ; 0586
                          00000000G  EF             02  FB 00087             CALLS   #2, MOM$DEBUG_TXT                   ;
                                                    76  11 0008E             BRB     9$                                  ; 0588
                                                    7E  7C 00090 6$:         CLRQ    -(SP)                               ; 0597
                          00000000G  00             02  FB 00092             CALLS   #2, SYS$CANTIM                      ;
                                   00000000'        EF  9F 00099             PUSHAB  P.AAI                               ; 0599
                                                    7E  7C 0009F             CLRQ    -(SP)                               ; 0598
                                                    7E  7C 000A1             CLRQ    -(SP)                               ;
                                   00000000'        EF  9F 000A3             PUSHAB  IOSB                                ;
                                                    59  DD 000A9             PUSHL   RSTATUS                             ;
                                                    05  DD 000AB             PUSHL   #5                                  ;
                          00000000G  EF             08  FB 000AD             CALLS   #8, MOM$DEBUG_QIO                   ;
                                57  00000000'        EF  3C 000B4             MOVZWL  IOSB, STATUS                        ; 0600
                             14   BC  00000000'        EF  B0 000BB             MOVW    IOSB+2, @RECV_MSG_LEN               ; 0601
                                6D                  57  E9 000C3             BLBC    STATUS, 12$                         ; 0607
                                                    6B  13 000C6             BEQL    12$                                 ; 0608
                                37                  57  E9 000C8             BLBC    STATUS, 8$                          ; 0610
                          10    A4                  01  88 000CB             BISB2   #1, 16(R4)                          ; 0616
                          10    A5                  01  88 000CF             BISB2   #1, 16(R5)                          ; 0617
                                   00000000'        EF  9F 000D3             PUSHAB  P.AAK                               ; 0621
                                7E                 14   BC  3C 000D9             MOVZWL  @RECV_MSG_LEN, -(SP)                ; 0620
                                                   04   A6  DD 000DD             PUSHL   4(R6)                               ; 0619
                                                    05  DD 000E0             PUSHL   #5                                  ; 0618
                          00000000G  EF             04  FB 000E2             CALLS   #4, MOM$DEBUG_MSG                   ;
                                50                 18   AC  D0 000E9             MOVL    SKIP_MSG_DSC, R0                    ; 0630
                                                    44  13 000ED             BEQL    12$                                 ;
                                                    0A  15 000EF             BLEQ    7$                                  ; 0634
                    04    B6          04    B0            60  29 000F1             CMPC3   (R0), @4(R0), @4(R6)                ; 0636
                                                    09  13 000F7             BEQL    8$                                  ;
                                                    38  11 000F9             BRB     12$                                 ; 0638
                                31 00000000'        EF  E9 000FB 7$:         BLBC    NI_HEADER, 12$                      ; 0644
                                                    5B  D7 00102 8$:         DECL    SKIP_MSG_MAX                        ; 0648
                                                    07  12 00104             BNEQ    10$                                 ; 0654
                                57      022C        8F  3C 00106 9$:         MOVZWL  #556, STATUS                        ; 0656
                                                    26  11 0010B             BRB     12$                                 ; 0655
                                   00000000'        EF  9F 0010D 10$:        PUSHAB  NI_HEADER                           ; 0659
                                   00000000'        EF  9F 00113             PUSHAB  IOSB                                ;
                                                    56  DD 00119             PUSHL   R6                                  ; 0660
                                                    65  DD 0011B             PUSHL   (R5)                                ; 0659
                          00000000V  EF             04  FB 0011D             CALLS   #4, MOM_MOP_RECEIVE                 ;
                                59                  50  D0 00124             MOVL    R0, RSTATUS                         ;
                                03                  59  E9 00127             BLBC    RSTATUS, 11$                        ; 0663
                                                FF27    31 0012A             BRW     5$                                  ;
                                                    7E  7C 0012D 11$:        CLRQ    -(SP)                               ; 0664
```

```
                                 59  DD 0012F          PUSHL   RSTATUS                                          ;
                                 74  11 00131          BRB     19$                                             ;
                0000022C   8F    57  D1 00133 12$:     CMPL    STATUS, #556                                    ; 0667
                                 2A  12 0013A          BNEQ    15$                                             ;
                                 5A  D7 0013C          DECL    RETRY                                           ; 0672
             1D 00000000G   EF    01  E1 0013E          BBC     #1, MOM$GL_SERVICE_FLAGS, 14$                   ; 0678
                                 1B  15 00146          BLEQ    14$                                             ; 0679
                         07  10   A4  E8 00148          BLBS    16(R4), 13$                                     ; 0681
                                 54  DD 0014C          PUSHL   R4                                              ; 0682
                FE30     CF      01  FB 0014E          CALLS   #1, MOM$SET_NI_ADDR                             ;
                         55      54  D1 00153 13$:     CMPL    R4, R5                                          ; 0683
                                 0B  13 00156          BEQL    14$                                             ;
                         07  10   A5  E8 00158          BLBS    16(R5), 14$                                     ; 0684
                                 55  DD 0015C          PUSHL   R5                                              ; 0685
                FE20     CF      01  FB 0015E          CALLS   #1, MOM$SET_NI_ADDR                             ;
                             FED2  31  00163 14$:      BRW     2$                                              ; 0555
             1A 00000000G   EF    01  E1 00166 15$:     BBC     #1, MOM$GL_SERVICE_FLAGS, 17$                   ; 0693
                         17      57  E9 0016E          BLBC    STATUS, 17$                                     ; 0694
                         09 00000000'  EF  E9 00171     BLBC    NI_HEADER, 16$                                  ; 0696
                00000000G   EF   20  88 00178          BISB2   #32, MOM$GL_SERVICE_FLAGS                       ; 0697
                                 07  11 0017F          BRB     17$                                             ;
                00000000G   EF   20  8A 00181 16$:     BICB2   #32, MOM$GL_SERVICE_FLAGS                       ; 0699
                                 5A  D5 00188 17$:     TSTL    RETRY                                           ; 0706
                                 17  14 0018A          BGTR    18$                                             ;
                                 64  DD 0018C          PUSHL   (R4)                                            ; 0708
                00000000G   00   01  FB 0018E          CALLS   #1, SYS$CANCEL                                  ;
                         65      64  D1 00195          CMPL    (R4), (R5)                                      ; 0709
                                 09  13 00198          BEQL    18$                                             ;
                                 65  DD 0019A          PUSHL   (R5)                                            ; 0710
                00000000G   00   01  FB 0019C          CALLS   #1, SYS$CANCEL                                  ;
                                 7E  7C 001A3 18$:     CLRQ    -(SP)                                           ; 0712
                                 57  DD 001A5          PUSHL   STATUS                                          ;
                00000000V   EF   03  FB 001A7 19$:     CALLS   #3, MOM_MAPMOPERRORS                           ;
                                     04  001AE          RET                                                     ; 0714

; Routine Size:  431 bytes,    Routine Base:  $CODE$ + 0255
```

```
;   722          0715  1 %SBTTL 'mom_mop_receive    Receive a MOP mode message'
;   723          0716  1 ROUTINE mom_mop_receive (mop_rcv_chan,
;   724          0717  1                                recv_buf_dsc,
;   725          0718  1                                iosb,
;   726          0719  1                                ni_header_addr) =
;   727          0720  1 !++
;   728          0721  1 ! FUNCTIONAL DESCRIPTION:
;   729          0722  1 !       This routine is called to issue a QIO receive on the MOP channel
;   730          0723  1 !
;   731          0724  1 ! INPUTS:
;   732          0725  1 !       MOP_RCV_CHAN = QIO channel to issue receive on.
;   733          0726  1 !       RECV_BUF_DSC    = Address of descriptor of receive message buffer.
;   734          0727  1 !       IOSB            = Address of IOSB.
;   735          0728  1 !       NI_HEADER_ADDR  = Address of buffer in which to return NI header.
;   736          0729  1 !
;   737          0730  1 ! OUTPUTS:
;   738          0731  1 !       Success/failure of operation.
;   739          0732  1 !--
;   740          0733  2 BEGIN
;   741          0734  2
;   742          0735  2 MAP
;   743          0736  2     recv_buf_dsc: REF VECTOR;
;   744          0737  2
;   745          0738  2 LOCAL
;   746          0739  2     retry,
;   747          0740  2     rstatus;
;   748          0741  2
;   749          0742  2 retry = 3;
;   750          0743  2 WHILE .retry GTR 0 DO
;   751          0744  3     BEGIN
;   752        P 0745  3     rstatus = $QIO (EFN  = mom$k_moprcvefn,
;   753        P 0746  3                     CHAN = .mop_rcv_chan,
;   754        P 0747  3                     FUNC = io$_readvblk,
;   755        P 0748  3                     IOSB = .iosb,
;   756        P 0749  3                     P1   = .recv_buf_dsc [1],
;   757        P 0750  3                     P2   = .recv_buf_dsc [0],
;   758          0751  3                     P5   = .ni_header_addr);
;   759          0752  3     IF NOT .rstatus THEN
;   760          0753  3         mom$debug_qio (dbg$c_mopio, .rstatus, 0, 0, 0, 0, 0,
;   761          0754  3                        $ASCID ('MOP receive completion status'))
;   762          0755  3     ELSE
;   763          0756  3         EXITLOOP;
;   764          0757  3     retry = .retry - 1;
;   765          0758  2     END;
;   766          0759  2 RETURN .rstatus;
;   767          0760  1 END;                       ! of  mom_mop_receive


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

6D  6F  63  20  65  76  69  65  63  65  72  20  50  4F  4D  000D8 P.AAN:  .ASCII  \MOP receive completion status\                ;
    73  75  74  61  74  73  20  6E  6F  69  74  65  6C  70  000E7                                                              ;
                                                            000F5       .BLKB   3
                                                  0000001D  000F8 P.AAM:  .LONG   29                                        ;
                                                  00000000' 000FC         .ADDRESS P.AAN                                    ;
```

```
                                                        .EXTRN  SYS$QIO

                                                        .PSECT  $CODE$,NOWRT,2

                                        001C 00000 MOM_MOP_RECEIVE:
                                                        .WORD   Save R2,R3,R4                   ; 0716
                                    54          03  D0 00002           MOVL    #3, RETRY       ; 0742
                                    52      08  AC  D0 00005           MOVL    RECV_BUF_DSC, R2 ; 0751
                                            54  D5 00009 1$:         TSTL    RETRY           ; 0743
                                            41  15 0000B             BLEQ    2$              ;
                                            7E  D4 0000D             CLRL    -(SP)           ; 0751
                                        10  AC  DD 0000F             PUSHL   NI_HEADER_ADDR  ;
                                            7E  7C 00012             CLRQ    -(SP)           ;
                                        08  BC  DD 00014             PUSHL   @RECV_BUF_DSC   ;
                                        04  A2  DD 00017             PUSHL   4(R2)           ;
                                            7E  7C 0001A             CLRQ    -(SP)           ;
                                        0C  AC  DD 0001C             PUSHL   IOSB            ;
                                        31  DD 0001F                 PUSHL   #49             ;
                                        04  AC  DD 00021             PUSHL   MOP_RCV_CHAN    ;
                                        01  DD 00024                 PUSHL   #1              ;
                00000000G   00          0C  FB 00026             CALLS   #12, SYS$QIO    ;
                            53          50  D0 0002D             MOVL    R0, RSTATUS     ;
                            1B          53  E8 00030             BLBS    RSTATUS, 2$     ; 0752
                                00000000' EF  9F 00033             PUSHAB  P.AAM           ; 0754
                                            7E  7C 00039             CLRQ    -(SP)           ; 0753
                                            7E  7C 0003B             CLRQ    -(SP)           ;
                                            7E  D4 0003D             CLRL    -(SP)           ;
                                            53  DD 0003F             PUSHL   RSTATUS         ;
                                            05  DD 00041             PUSHL   #5              ;
                00000000G   EF              08  FB 00043             CALLS   #8, MOM$DEBUG_QIO ;
                                            54  D7 0004A             DECL    RETRY           ; 0757
                                            BB  11 0004C             BRB     1$              ; 0743
                                50          53  D0 0004E 2$:         MOVL    RSTATUS, R0     ; 0759
                                            04  00051                 RET                     ; 0760

; Routine Size:  82 bytes,    Routine Base:  $CODE$ + 0404
```

```
;  769          0761  1 %SBTTL 'mom$mop_receive_qiow    Receive and wait for a MOP mode message'
;  770          0762  1 GLOBAL ROUTINE mom$mop_receive_qiow (mop_rcv_chan,
;  771          0763  1                                     mop_msg_dsc,
;  772          0764  1                                     ni_header_addr) =
;  773          0765  1
;  774          0766  1 !++
;  775          0767  1 ! FUNCTIONAL DESCRIPTION:
;  776          0768  1 !       This routine performs a receive operation on the specified MOP
;  777          0769  1 !       channel. This is currently used only to get the MOP message which
;  778          0770  1 !       causes SERVICE to be started for an autoservice function.
;  779          0771  1 !
;  780          0772  1 ! INPUTS:
;  781          0773  1 !       MOP_RCV_CHAN = Channel to receive response MOP message on.
;  782          0774  1 !       MOP_MSG_DSC     Address of descriptor of receive buffer.
;  783          0775  1 !       NI_HEADER_ADDR - Address at which to put NI header which was
;  784          0776  1 !               received with the MOP message.  Used to determine if
;  785          0777  1 !               a MOP message was multicast.
;  786          0778  1 !
;  787          0779  1 ! OUTPUTS:
;  788          0780  1 !       If the QIO does not complete successfully, the first longword of
;  789          0781  1 !               the MOP message descriptor is set to zero.
;  790          0782  1 !--
;  791          0783  1
;  792          0784  2 BEGIN
;  793          0785  2
;  794          0786  2 MAP
;  795          0787  2     mop_msg_dsc: REF VECTOR;
;  796          0788  2
;  797          0789  2 OWN
;  798          0790  2     iosb    : $iosb;
;  799          0791  2
;  800          0792  2 LOCAL
;  801          0793  2     rstatus;
;  802          0794  2
;  803        P 0795  2 rstatus = $QIOW (EFN  = mom$k_moprcvefn,
;  804        P 0796  2                  CHAN = .mop_rcv_chan,
;  805        P 0797  2                  FUNC = io$_readvblk OR io$m_now,
;  806        P 0798  2                  IOSB = iosb,
;  807        P 0799  2                  P1   = .mop_msg_dsc [1],
;  808        P 0800  2                  P2   = .mop_msg_dsc [0],
;  809          0801  2                  P5   = .ni_header_addr);
;  810          0802  2 IF NOT .rstatus THEN
;  811          0803  2     mom$debug_qio (dbg$c_mopio, .rstatus, iosb, 0, 0, 0, 0,
;  812          0804  2                    $ASCID ('MOP receive completion status'));
;  813          0805  2
;  814          0806  2 !
;  815          0807  2 ! Check to see ef the receive completed with an error.
;  816          0808  2 !
;  817          0809  2 IF .rstatus THEN
;  818          0810  3     BEGIN
;  819          0811  3     mom$debug_qio (dbg$c_mopio, .rstatus, iosb, 0, 0, 0, 0,
;  820          0812  3                    $ASCID ('MOP receive completion status'));
;  821          0813  3     rstatus = .iosb [ios$w_status];
;  822          0814  3     IF .rstatus THEN
;  823          0815  4         BEGIN
;  824          0816  4         mop_msg_dsc [0] = .iosb [ios$w_count];
;  825          0817  4         mom$debug_msg ( dbg$c_mopio,
```

```
;  826          0818  4                          .mop_msg_dsc [1],
;  827          0819  4                          .mop_msg_dsc [0],
;  828          0820  4                          $ASCID ('MOP message received'));
;  829          0821  3              END;
;  830          0822  2          END;
;  831          0823  2      IF NOT .rstatus THEN
;  832          0824  2          mop_msg_dsc [0] = 0;
;  833          0825  2
;  834          0826  2      RETURN .rstatus;
;  835          0827  1      END;                                ! End of mom$mop_receive_qiow


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

6D 6F 63 20 65 76 69 65 63 65 72 20 50 4F 4D  00100 P.AAP:   .ASCII  \MOP receive completion status\          ;
   73 75 74 61 74 73 20 6E 6F 69 74 65 6C 70  0010F                                                        ;
                                              0011D          .BLKB   3                                    ;
                                   0000001D   00120 P.AAO:   .LONG   29                                   ;
                                   00000000'  00124          .ADDRESS P.AAP                               ;
6D 6F 63 20 65 76 69 65 63 65 72 20 50 4F 4D  00128 P.AAR:   .ASCII  \MOP receive completion status\      ;
   73 75 74 61 74 73 20 6E 6F 69 74 65 6C 70  00137                                                        ;
                                              00145          .BLKB   3                                    ;
                                   0000001D   00148 P.AAQ:   .LONG   29                                   ;
                                   00000000'  0014C          .ADDRESS P.AAR                               ;
63 65 72 20 65 67 61 73 73 65 6D 20 50 4F 4D  00150 P.AAT:   .ASCII  \MOP message received\               ;
                                        64 65 76 69 65 72  0015F                                          ;
                                   00000014   00164 P.AAS:   .LONG   20                                   ;
                                   00000000'  00168          .ADDRESS P.AAT                               ;

                                                                .PSECT  $OWN$,NOEXE,2

                                              00056          .BLKB   2
                                              00058 IOSB:    .BLKB   8


                                                                .PSECT  $CODE$,NOWRT,2

                                         007C 00000          .ENTRY  MOM$MOP_RECEIVE_QIOW, Save R2,R3,R4,R5,R6    ; 0762
                 56 00000000G   EF    9E 00002          MOVAB   MOM$DEBUG_QIO, R6                            ;
                 55 00000000'   EF    9E 00009          MOVAB   P.AAO, R5                                    ;
                 54 00000000'   EF    9E 00010          MOVAB   IOSB, R4                                     ;
                                7E    D4 00017          CLRL    -(SP)                                        ; 0801
                          0C    AC    DD 00019          PUSHL   NI_HEADER_ADDR                               ;
                                7E    7C 0001C          CLRQ    -(SP)                                        ;
                 52       08    AC    D0 0001E          MOVL    MOP_MSG_DSC, R2                              ;
                                62    DD 00022          PUSHL   (R2)                                         ;
                          04    A2    DD 00024          PUSHL   4(R2)                                        ;
                                7E    7C 00027          CLRQ    -(SP)                                        ;
                                54    DD 00029          PUSHL   R4                                           ;
                 7E       71    8F    9A 0002B          MOVZBL  #113, -(SP)                                  ;
                          04    AC    DD 0002F          PUSHL   MOP_RCV_CHAN                                 ;
                                01    DD 00032          PUSHL   #1                                           ;
00000000G        00             0C    FB 00034          CALLS   #12, SYS$QIOW                                ;
                 53             50    D0 0003B          MOVL    R0, RSTATUS                                  ;
                 10             53    E8 0003E          BLBS    RSTATUS, 1$                                  ; 0802
```

```
                                55  DD 00041          PUSHL   R5                                          ; 0804
                                7E  7C 00043          CLRQ    -(SP)                                       ; 0803
                                7E  7C 00045          CLRQ    -(SP)                                       ;
                                18  BB 00047          PUSHR   #^M<R3,R4>                                  ;
                                05  DD 00049          PUSHL   #5                                          ;
                            66  08  FB 0004B          CALLS   #8, MOM$DEBUG_QIO                           ;
                            2C  53  E9 0004E          BLBC    RSTATUS, 2$                                 ; 0809
                        28      A5  9F 00051  1$:     PUSHAB  P.AAQ                                       ; 0812
                                7E  7C 00054          CLRQ    -(SP)                                       ; 0811
                                7E  7C 00056          CLRQ    -(SP)                                       ;
                                18  BB 00058          PUSHR   #^M<R3,R4>                                  ;
                                05  DD 0005A          PUSHL   #5                                          ;
                            66  08  FB 0005C          CALLS   #8, MOM$DEBUG_QIO                           ;
                            53  64  3C 0005F          MOVZWL  IOSB, RSTATUS                               ; 0813
                            18  53  E9 00062          BLBC    RSTATUS, 2$                                 ; 0814
                            62  02  A4 3C 00065       MOVZWL  IOSB+2, (R2)                                ; 0816
                                44  A5 9F 00069      PUSHAB  P.AAS                                       ; 0820
                                62  DD 0006C          PUSHL   (R2)                                        ; 0819
                                04  A2 DD 0006E      PUSHL   4(R2)                                       ; 0818
                                05  DD 00071          PUSHL   #5                                          ; 0817
                00000000G   EF      04  FB 00073      CALLS   #4, MOM$DEBUG_MSG                           ;
                            02  53  E8 0007A          BLBS    RSTATUS, 3$                                 ; 0823
                                62  D4 0007D  2$:     CLRL    (R2)                                        ; 0824
                            50  53  D0 0007F  3$:     MOVL    RSTATUS, R0                                 ; 0826
                                04  00082            RET                                                 ; 0827

; Routine Size:  131 bytes,    Routine Base:  $CODE$ + 0456
```

```
;   837          0828  1 %SBTTL 'mom$mopsend  Send a MOP mode message'
;   838          0829  1 GLOBAL ROUTINE mom$mopsend (send_CIB, adr, len) =
;   839          0830  1
;   840          0831  1 !++
;   841          0832  1 ! FUNCTIONAL DESCRIPTION:
;   842          0833  1 !
;   843          0834  1 !       This routine transmits a single MOP message.
;   844          0835  1 !
;   845          0836  1 ! INPUTS:
;   846          0837  1 !
;   847          0838  1 !       SEND_CIB = Informaton Block of channel over which to transmit MOP message
;   848          0839  1 !       ADR = Address of data to be transmitted.
;   849          0840  1 !       LEN = Length in bytes of data to be transmitted.
;   850          0841  1 !
;   851          0842  1 ! OUTPUTS:
;   852          0843  1 !
;   853          0844  1 !       Success/failure of operation.
;   854          0845  1 !--
;   855          0846  1
;   856          0847  2 BEGIN
;   857          0848  2
;   858          0849  2 MAP
;   859          0850  2     send_CIB : REF BBLOCK;
;   860          0851  2
;   861          0852  2 LOCAL
;   862          0853  2     iosb    : $iosb,
;   863          0854  2     wstatus;
;   864          0855  2
;   865          0856  2 mom$debug_msg ( dbg$c_mopio,
;   866          0857  2                 .adr,
;   867          0858  2                 .len,
;   868          0859  2                 $ASCID ('Transmitting MOP message'));
;   869        P 0860  2 wstatus = $QIOW (EFN  = mom$k_mopsndefn,
;   870        P 0861  2                  CHAN = .send_CIB [cib$l_chan],
;   871        P 0862  2                  FUNC = io$_writevblk,
;   872        P 0863  2                  IOSB = iosb,
;   873        P 0864  2                  P1   = .adr,
;   874          0865  2                  P2   = .len);
;   875          0866  2
;   876          0867  2 !
;   877          0868  2 ! Dump the transmit's completion status to the debug log.
;   878          0869  2 !
;   879          0870  2 mom$debug_qio (dbg$c_mopio, .wstatus, iosb, 0, 0, 0, 0,
;   880          0871  2                 $ASCID ('MOP transmit completion status'));
;   881          0872  2 RETURN mom_mapmoperrors (.wstatus, iosb, mom$nosignal)
;   882          0873  2
;   883          0874  1 END;                                ! End of MOM$MOPSEND


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

4F  4D  20  67  6E  69  74  74  69  6D  73  6E  61  72  54  0016C P.AAV:   .ASCII  \Transmitting MOP message\                ;
                        65  67  61  73  73  65  6D  20  50  0017B                                                          ;
                                                00000018  00184 P.AAU:   .LONG   24                                       ;
                                                00000000' 00188         .ADDRESS P.AAV                                    ;
6F  63  20  74  69  6D  73  6E  61  72  74  20  50  4F  4D  0018C P.AAX:   .ASCII  \MOP transmit completion status\       ;
```

```
73 75 74 61 74 73 20 6E 6F 69 74 65 6C 70 6D  0019B
                                              001AA                 .BLKB   2                                                       ;
                                    0000001E  001AC P.AAW:          .LONG   30                                                      ;
                                    00000000' 001B0                .ADDRESS P.AAX                                                  ;


                                                                   .PSECT  $CODE$,NOWRT,2

                                        0004 00000                 .ENTRY  MOM$MOPSEND, Save R2                                    ; 0829
                                  5E      08  C2 00002               SUBL2   #8, SP                                                  ;
                                     00000000'  EF  9F 00005         PUSHAB  P.AAU                                                   ; 0859
                                  7E        08  AC  7D 0000B         MOVQ    ADR, -(SP)                                              ; 0857
                                              05  DD 0000F         PUSHL   #5                                                      ; 0856
                        00000000G  EF             04  FB 00011      CALLS   #4, MOM$DEBUG_MSG                                       ;
                                              7E  7C 00018         CLRQ    -(SP)                                                   ; 0865
                                              7E  7C 0001A         CLRQ    -(SP)                                                   ;
                                  7E        08  AC  7D 0001C         MOVQ    ADR, -(SP)                                              ;
                                              7E  7C 00020         CLRQ    -(SP)                                                   ;
                                            20  AE  9F 00022         PUSHAB  IOSB                                                    ;
                                              30  DD 00025         PUSHL   #48                                                     ;
                                            04  BC  DD 00027         PUSHL   @SEND_CIB                                               ;
                                              02  DD 0002A         PUSHL   #2                                                      ;
                        00000000G  00             0C  FB 0002C      CALLS   #12, SYS$QIOW                                           ;
                                  52              50  D0 00033      MOVL    R0, WSTATUS                                             ;
                                     00000000'  EF  9F 00036         PUSHAB  P.AAW                                                   ; 0871
                                              7E  7C 0003C         CLRQ    -(SP)                                                   ; 0870
                                              7E  7C 0003E         CLRQ    -(SP)                                                   ;
                                            14  AE  9F 00040         PUSHAB  IOSB                                                    ;
                                              52  DD 00043         PUSHL   WSTATUS                                                 ;
                                              05  DD 00045         PUSHL   #5                                                      ;
                        00000000G  EF             08  FB 00047      CALLS   #8, MOM$DEBUG_QIO                                       ;
                                              7E  D4 0004E         CLRL    -(SP)                                                   ; 0872
                                            04  AE  9F 00050         PUSHAB  IOSB                                                    ;
                                              52  DD 00053         PUSHL   WSTATUS                                                 ;
                        00000000V  EF             03  FB 00055      CALLS   #3, MOM_MAPMOPERRORS                                    ;
                                                  04 0005C         RET                                                             ; 0874

; Routine Size:  93 bytes,    Routine Base:  $CODE$ + 04D9
```

```
;  885          0875  1 %SBTTL 'mom_mapmoperrors      Map MOP error codes'
;  886          0876  1 ROUTINE mom_mapmoperrors (code, iosb, signal_flag) =
;  887          0877  1
;  888          0878  1 !++
;  889          0879  1 ! FUNCTIONAL DESCRIPTION:
;  890          0880  1 !
;  891          0881  1 !       This routine sets up the message block information for MOP errors.
;  892          0882  1 !
;  893          0883  1 ! INPUTS:
;  894          0884  1 !
;  895          0885  1 !       CODE            QIO status code.
;  896          0886  1 !       IOSB            Address of I/O status block.
;  897          0887  1 !       SIGNAL_FLAG     Indicates whether or not to signal if there is
;  898          0888  1 !                       an error.  0 = don't signal, 1 = signal.
;  899          0889  1 !
;  900          0890  1 ! OUTPUTS:
;  901          0891  1 !
;  902          0892  1 !       If an error is indicated, the appropriate NICE message information
;  903          0893  1 !       is stored in the message block (MOM$AB_MSGBLOCK).
;  904          0894  1 !
;  905          0895  1 !       The actual status of the operation is returned.  The status
;  906          0896  1 !       comes from either the value in R0 or the value in the I/O status
;  907          0897  1 !       block.
;  908          0898  1 !--
;  909          0899  1
;  910          0900  2 BEGIN
;  911          0901  2
;  912          0902  2 MAP
;  913          0903  2     iosb : REF $IOSB;
;  914          0904  2
;  915          0905  2 LOCAL
;  916          0906  2     status,
;  917          0907  2     msgsize;
;  918          0908  2
;  919          0909  2 status = .code;
;  920          0910  2
;  921          0911  2 IF .code THEN
;  922          0912  2     IF .iosb NEQA 0 THEN
;  923          0913  2         status = .iosb [ios$w_status];
;  924          0914  2
;  925          0915  2 IF NOT .status THEN
;  926          0916  2     SELECTONEU .status OF
;  927          0917  2         SET
;  928          0918  2         [OTHERWISE]:
;  929          0919  3             BEGIN
;  930          0920  3
;  931          0921  3             mom$ab_msgblock [msb$l_flags] = msb$m_msg_fld;
;  932          0922  3             mom$ab_msgblock [msb$b_code] = nma$c_sts_[co;
;  933          0923  3             mom$ab_msgblock [msb$l_text] = .status;
;  934          0924  3             IF .signal_flag THEN
;  935          0925  4                 BEGIN
;  936          0926  4                 mom$bld_reply (mom$ab_msgblock, msgsize);
;  937          0927  4                 $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
;  938          0928  3                 END;
;  939          0929  2             END;
;  940          0930  2         TES;
;  941          0931  2
```

```
;  942          0932  2 RETURN .status
;  943          0933  2
;  944          0934  1 END;                                   ! End of mom_mapmoperrors


                                                000C 00000 MOM_MAPMOPERRORS:
                                                           .WORD   Save R2,R3                                      ; 0876
                                    53 00000000G   EF  9E 00002            MOVAB   MOM$AB_MSGBLOCK, R3          ;
                                    5E             04  C2 00009            SUBL2   #4, SP                       ;
                                    50       04    AC  D0 0000C            MOVL    CODE, STATUS                 ; 0909
                                    09       04    AC  E9 00010            BLBC    CODE, 1$                     ; 0911
                                             08    AC  D5 00014            TSTL    IOSB                         ; 0912
                                             04    13  00017               BEQL    1$                           ;
                                    50       08    BC  3C 00019            MOVZWL  @IOSB, STATUS                ; 0913
                                    52             50  D0 0001D 1$:        MOVL    STATUS, R2                   ; 0915
                                    2F             52  E8 00020            BLBS    R2, 2$                       ;
                                    63             04  D0 00023            MOVL    #4, MOM$AB_MSGBLOCK          ; 0921
                              04    A3             0A  8E 00026            MNEGB   #10, MOM$AB_MSGBLOCK+4       ; 0922
                              0C    A3             52  D0 0002A            MOVL    R2, MOM$AB_MSGBLOCK+12       ; 0923
                                    20       0C    AC  E9 0002E            BLBC    SIGNAL_FLAG, 2$              ; 0924
                                          4008    8F  BB 00032            PUSHR   #^M<R3,SP>                   ; 0926
                         00000000G  EF              02  FB 00036            CALLS   #2, MOM$BLD_REPLY            ;
                                                    6E  DD 0003D            PUSHL   MSGSIZE                      ; 0927
                                    00000000G      EF  9F 0003F            PUSHAB  MOM$AB_NICE_XMIT_BUF         ;
                                    02070000       8F  DD 00045            PUSHL   #34013184                    ;
                         00000000G  00             03  FB 0004B            CALLS   #3, LIB$SIGNAL               ;
                                    50             52  D0 00052 2$:        MOVL    R2, R0                       ; 0932
                                                   04  00055              RET                                  ; 0934

; Routine Size:  86 bytes,    Routine Base:  $CODE$ + 0536
```

```
;  946          0935  1 END                                   ! End of module
;  947          0936  1
;  948          0937  0 ELUDOM


                                                                .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $PLIT$                        436  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       1420  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                          96  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics
;
;                                    --------- Symbols ---------      Pages       Processing
;       File                         Total    Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[MOM.OBJ]MOMLIB.L32;1     194        51        26          21        00:00.1
;  _$255$DUA28:[SHRLIB]NMALIBRY.L32;1    887        17         1          47        00:00.2
;  _$255$DUA28:[SYSLIB]STARLET.L32;1    9776        17         0         581        00:02.1


;                                       COMMAND QUALIFIERS

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

;  949          0938  0
; Size:         1420 code + 532 data bytes
; Run Time:       00:29.8
; Elapsed Time:   01:05.2
; Lines/CPU Min:   1891
; Lexemes/CPU-Min: 19316
; Memory Used:  199 pages
; Compilation Complete
```

MOMMSG
LIS

MOMMAIN
LIS

MOMPARSE
LIS

MOMSERSTA
LIS

MOMMOPSTA
LIS

MOMSERVIC
LIS

MOMMOPLIO
LIS

MOMTESSTA
LIS

MOMRSXDEF
LIS

MOMSUBS
LIS