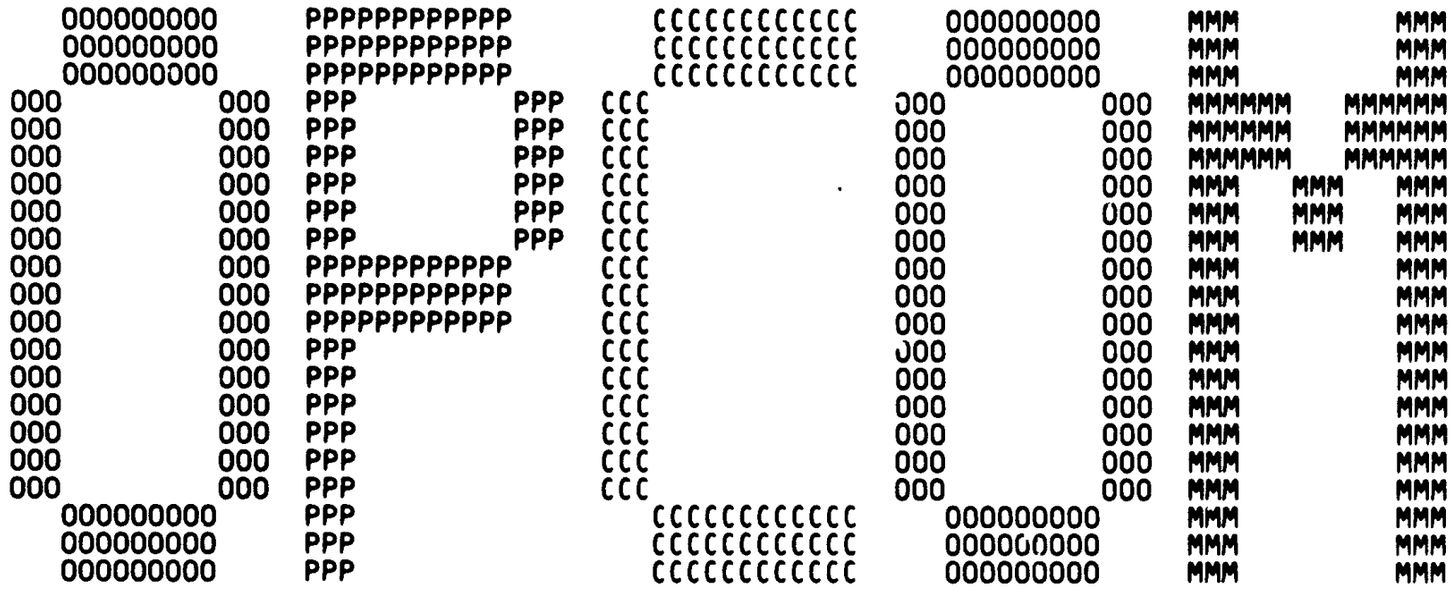


_$2

Sym

ALL

ASC

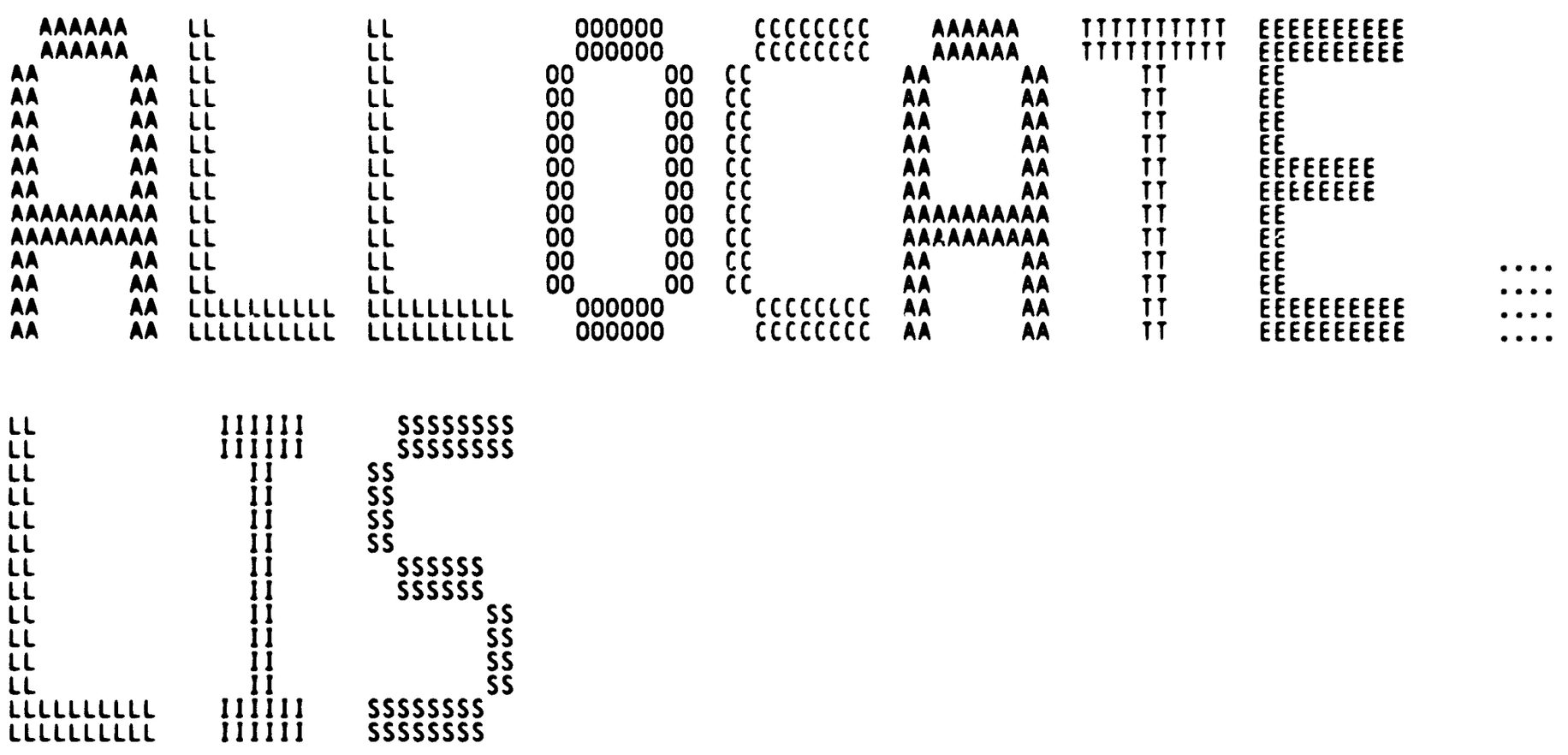
BOD
BOD
BOD
BOD
BOD
BOD
BOD
BOD
BUG
BYP
CAN
CAN
CAN
CHE
CHE

CLU
CLU
CLU
CLU
CLU
CLU
CLU

CLU
CLU
CLU
CLU
CLU
CLU
CLU
CLU

CLU
CLU

FILEID**ALLOCATE



```
;  1     0001  0 MODULE  OPC$ALLOCATE    (
;  2     0002  0                          LANGUAGE (BLISS32),
;  3     0003  0                          IDENT = 'V04-000'
;  4     0004  0                         ) =
;  5     0005  0
;  6     0006  0 !*****************************************************************************
;  7     0007  0 !*                                                                           *
;  8     0008  0 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;  9     0009  0 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 10     0010  0 !*  ALL RIGHTS RESERVED.                                                     *
; 11     0011  0 !*                                                                           *
; 12     0012  0 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 13     0013  0 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 14     0014  0 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 15     0015  0 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
; 16     0016  0 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 17     0017  0 !*  TRANSFERRED.                                                             *
; 18     0018  0 !*                                                                           *
; 19     0019  0 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 20     0020  0 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 21     0021  0 !*  CORPORATION.                                                             *
; 22     0022  0 !*                                                                           *
; 23     0023  0 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 24     0024  0 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 25     0025  0 !*                                                                           *
; 26     0026  0 !*                                                                           *
; 27     0027  0 !*****************************************************************************
; 28     0028  0
; 29     0029  0 !++
; 30     0030  0 ! FACILITY:
; 31     0031  0 !
; 32     0032  0 !       OPCOM
; 33     0033  0 !
; 34     0034  0 ! ABSTRACT:
; 35     0035  0 !
; 36     0036  0 !       This module contains the logic to allocate and deallocate
; 37     0037  0 !       the various data structures used by OPCOM.
; 38     0038  0 !
; 39     0039  0 ! Environment:
; 40     0040  0 !
; 41     0041  0 !       VAX/VMS operating system.
; 42     0042  0 !
; 43     0043  0 ! Author:
; 44     0044  0 !
; 45     0045  0 !       Steven T. Jeffreys
; 46     0046  0 !
; 47     0047  0 ! Creation date:
; 48     0048  0 !
; 49     0049  0 !       March 10, 1981
; 50     0050  0 !
; 51     0051  0 ! Revision history:
; 52     0052  0 !
; 53     0053  0 !       V03-002 CWH3002         CW Hobbs                        16-Sep-1983
; 54     0054  0 !               Put VM calls in jacket routines to aid debugging.
; 55     0055  0 !
; 56     0056  0 !       V03-001 CWH3001         CW Hobbs                        30-Jul-1983
; 57     0057  0 !               Various and sundry things to make OPCOM distributed
```

```
;    58          0058  0 !                 across the cluster.
;    59          0059  0 !
;    60          0060  0 !  V02-002        STJ0161         Steven T. Jeffreys,     08-Feb-1982
;    61          0061  0 !                 Make references to library routines use general addressing mode.
;    62          0062  0 !
;    63          0063  0 !--
;    64          0064  0
;    65          0065  1 BEGIN                                           ! Start of ALLOCATE
;    66          0066  1
;    67          0067  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    68          0068  1 LIBRARY 'LIB$:OPCOMLIB';
;    69          0069  1
;    70          0070  1 FORWARD ROUTINE
;    71          0071  1         ALLOCATE_DS,                            ! Allocate data structure
;    72          0072  1         CREATE_OCD      : NOVALUE,              ! Create a new OCD and insert it into an OCD list
;    73          0073  1         DEALLOCATE_DS,                          ! Deallocate data structure
;    74          0074  1         DEALLOCATE_MCB  : NOVALUE,              ! Deallocate an MCB.
;    75          0075  1         DEALLOCATE_OCD  : NOVALUE,              ! Deallocate an OCD
;    76          0076  1         DEALLOCATE_RQCB : NOVALUE;              ! Deallocate an RQCB.
;    77          0077  1
;    78          0078  1 BUILTIN
;    79          0079  1         INSQUE,                                 ! Insert entry onto a queue
;    80          0080  1         REMQUE;                                 ! Remove entry from a queue
```

```
;    82          0081  1 GLOBAL ROUTINE ALLOCATE_DS (DS_TYPE, DS_ADDR) =
;    83          0082  1
;    84          0083  1 !++
;    85          0084  1 ! Functional description:
;    86          0085  1 !
;    87          0086  1 !       This routine will allocate a block of process address space
;    88          0087  1 !       to be used as a data structure of the specified type.  This
;    89          0088  1 !       routine will also write some structure dependent information
;    90          0089  1 !       into the data structure.
;    91          0090  1 !
;    92          0091  1 ! Input:
;    93          0092  1 !
;    94          0093  1 !       DS_TYPE = The data structure type.  This is used as
;    95          0094  1 !                 an index into the SCB table to get some
;    96          0095  1 !                 structure dependent information.
;    97          0096  1 !
;    98          0097  1 ! Implicit Input:
;    99          0098  1 !
;   100          0099  1 !       The SCB data base.
;   101          0100  1 !
;   102          0101  1 ! Output:
;   103          0102  1 !
;   104          0103  1 !       DS_ADDR = An address of a longword to receive the data
;   105          0104  1 !                 structure address.
;   106          0105  1 !
;   107          0106  1 ! Implict output:
;   108          0107  1 !
;   109          0108  1 !       The SCB entry for this structure type is updated.
;   110          0109  1 !
;   111          0110  1 ! Side effects:
;   112          0111  1 !
;   113          0112  1 !       None.
;   114          0113  1 !
;   115          0114  1 ! Routine value:
;   116          0115  1 !
;   117          0116  1 !       TRUE             = The allocate succeeded.
;   118          0117  1 !       <anything else> = The allocate failed.
;   119          0118  1 !--
;   120          0119  1
;   121          0120  2 BEGIN                                           ! Start of ALLOCATE_DS
;   122          0121  2
;   123          0122  2 EXTERNAL ROUTINE
;   124          0123  2         CLUSUTIL_NEXT_SEQUENCE;                 ! Get a cluster-unique sequence number
;   125          0124  2
;   126          0125  2 EXTERNAL
;   127          0126  2         GLOBAL_STATUS   : BITVECTOR [32],
;   128          0127  2         LCL_NOD         : $ref_bblock,          ! Local NOD block pointer
;   129          0128  2         LCL_CSID        : LONG,                 ! CSID of local node
;   130          0129  2         SCB_TABLE       : VECTOR;               ! SCB table of pointers
;   131          0130  2
;   132          0131  2 EXTERNAL LITERAL
;   133          0132  2         SS$_BADPARAM,                           ! System status value
;   134          0133  2         MIN_DS_TYPE,                            ! Min STRUCTURE_TYPE value
;   135          0134  2         MAX_DS_TYPE;                            ! Max STRUCTURE_TYPE value
;   136          0135  2
;   137          0136  2 LOCAL
;   138          0137  2         SCB             : $ref_bblock,          ! SCB structure
```

```
;  139          0138  2          BLOCK           : $ref_bblock,          ! Block of VM allocated
;  140          0139  2          SIZE            : LONG;                 ! Size of the VM block
;  141          0140  2          STATUS          : LONG;
;  142          0141  2
;  143          0142  2
;  144          0143  2 !
;  145          0144  2 ! Make sure DS_TYPE is a legal value.
;  146          0145  2 !
;  147          0146  3 IF (.DS_TYPE LSS MIN_DS_TYPE) OR (.DS_TYPE GTR MAX_DS_TYPE)
;  148          0147  2 THEN
;  149          0148  2     $signal_stop (SS$_BADPARAM);
;  150          0149  2 !
;  151          0150  2 ! Get the structure info from the appropriate SCB.
;  152          0151  2 ! Allocate the structure from the look aside list if possible.
;  153          0152  2 !
;  154          0153  2 SCB = .SCB_TABLE [.DS_TYPE - 1];
;  155          0154  2 IF .SCB [SCB_W_LAL_COUNT] GTR 0
;  156          0155  2 THEN
;  157          0156  3     BEGIN
;  158          0157  3     REMQUE (.SCB [SCB_L_FLINK], BLOCK);
;  159          0158  3     SCB [SCB_W_LAL_COUNT] = .SCB [SCB_W_LAL_COUNT] - 1;
;  160          0159  3     END
;  161          0160  2 ELSE
;  162          0161  3     BEGIN
;  163          0162  3     !
;  164          0163  3     ! The look aside list was empty.
;  165          0164  3     ! Allocate and zero a block of memory via OPC$GET_VM.
;  166          0165  3     ! LAL blocks do not have to be zeroed, as it is done
;  167          0166  3     ! when the blocks are put on the list.
;  168          0167  3     !
;  169          0168  3     SIZE = .SCB [SCB_W_SIZE];
;  170          0169  4     IF NOT (STATUS = OPC$GET_VM (SIZE, BLOCK))
;  171          0170  3     THEN
;  172          0171  3         $signal_stop (.STATUS);
;  173          0172  3     CH$FILL (0, .SIZE, .BLOCK);
;  174          0173  3     !
;  175          0174  3     ! Fill in the structure dependent information.
;  176          0175  3     !
;  177          0176  3     BLOCK [HDR_L_FLINK] = BLOCK [HDR_L_FLINK];
;  178          0177  3     BLOCK [HDR_L_BLINK] = BLOCK [HDR_L_FLINK];
;  179          0178  3     BLOCK [HDR_W_SIZE]  = .SCB [SCB_W_SIZE];
;  180          0179  3     BLOCK [HDR_B_TYPE]  = .DS_TYPE;
;  181          0180  2     END;
;  182          0181  2
;  183          0182  2 !
;  184          0183  2 ! Set the cluster-wide data structure sequence number.
;  185          0184  2 !
;  186          0185  2 SCB [SCB_L_SEQNUM] = CLUSUTIL_NEXT_SEQUENCE (); ! Get a unique id and save it in the SCB
;  187          0186  2 BLOCK [HDR_L_SEQNUM] = .SCB [SCB_L_SEQNUM];     ! Mark the block field (read-only, nobody else should touch)
;  188          0187  2 BLOCK [HDR_L_IDENT] = .SCB [SCB_L_SEQNUM];      ! Also save as the ident field (read/write, allocate doesn't
;  189          0188  2 BLOCK [HDR_L_CSID] = .LCL_CSID;                 ! Zero means a local structure
;  190          0189  2 IF (BLOCK [HDR_L_NOD] = .LCL_NOD) NEQ 0         ! If in a cluster, set the local systemid field
;  191          0190  2 THEN
;  192          0191  3     BEGIN
;  193          0192  3     BLOCK [HDR_L_SYSTEMIDL] = .LCL_NOD [NOD_L_NODE_SYSTEMIDL];
;  194          0193  3     BLOCK [HDR_W_SYSTEMIDH] = .LCL_NOD [NOD_W_NODE_SYSTEMIDH];
;  195          0194  2     END;
```

```
;  196          0195  2 !
;  197          0196  2 ! Set the address of the data structure in the output cell.
;  198          0197  2 !
;  199          0198  2 .DS_ADDR = .BLOCK;
;  200          0199  2
;  201          0200  2 RETURN (TRUE);
;  202          0201  2
;  203          0202  1 END;                                    ! End of ALLOCATE_DS

                                                                .TITLE  OPC$ALLOCATE
                                                                .IDENT  \V04-000\

                                                                .EXTRN  CLUSUTIL_NEXT_SEQUENCE
                                                                .EXTRN  GLOBAL_STATUS, LCL_NOD
                                                                .EXTRN  LCL_CSID, SCB_TABLE
                                                                .EXTRN  SS$_BADPARAM, MIN_DS_TYPE
                                                                .EXTRN  MAX_DS_TYPE  LIB$STOP
                                                                .EXTRN  OPC$GET_VM

                                                                .PSECT  $CODE$,NOWRT,2

                                        00FC 00000              .ENTRY  ALLOCATE_DS, Save R2,R3,R4,R5,R6,R7     ; 0081
                                     5E 08  C2 00002            SUBL2   #8, SP                                  ;
                    00000000G    8F    04  AC  D1 00005        CMPL    DS_TYPE, #MIN_DS_TYPE                   ; 0146
                                           0A  19 00008        BLSS    1$                                      ;
                    00000000G    8F    04  AC  D1 0000A        CMPL    DS_TYPE, #MAX_DS_TYPE                   ;
                                           08  15 00012        BLEQ    2$                                      ;
                              00000000G  8F  DD 00014 1$:       PUSHL   #SS$_BADPARAM                           ; 0148
                                           2B  11 00017        BRB     4$                                      ;
                                 50    04  AC  D0 00019 2$:     MOVL    DS_TYPE, R0                             ; 0153
                                 57  0000GCF 40  D0 0001D       MOVL    SCB_TABLE-4[R0], SCB                    ;
                                       02  A7  B5 00021        TSTW    2(SCB)                                  ; 0154
                                           09  13 00024        BEQL    3$                                      ;
                                 6E    08  B7  0F 00026        REMQUE  @8(SCB), BLOCK                          ; 0157
                                       02  A7  B7 0002A        DECW    2(SCB)                                  ; 0158
                                           35  11 0002D        BRB     6$                                      ; 0154
                           04    AE          67  3C 0002F 3$:  MOVZWL  (SCB), SIZE                             ; 0168
                                             5E  DD 00033      PUSHL   SP                                      ; 0169
                                       08    AE  9F 00035      PUSHAB  SIZE                                    ;
                          0000G    CF          02  FB 00038    CALLS   #2, OPC$GET_VM                          ;
                                   0A          50  E8 0003D    BLBS    STATUS, 5$                              ;
                                               50  DD 00040    PUSHL   STATUS                                  ; 0171
                    00000000G      00          01  FB 00042 4$:  CALLS   #1, LIB$STOP                          ;
                                               04  00049      RET                                             ;
                                   56          6E  D0 0004A 5$:  MOVL    BLOCK, R6                             ; 0172
04    AE                    0C               6E  00  2C 0004D    MOVC5   #0, (SP), #0, SIZE, (R6)                ;
                                               66      00051                                                   ;
                                   66          56  D0 00052    MOVL    R6, (R6)                                ; 0176
                             04    A6          56  D0 00055    MOVL    R6, 4(R6)                               ; 0177
                             08    A6          67  B0 00059    MOVW    (SCB), 8(R6)                            ; 0178
                             0A    A6    04    AC  90 0005D    MOVB    DS_TYPE, 10(R6)                         ; 0179
                          0000G    CF          00  FB 00062 6$:  CALLS   #0, CLUSUTIL_NEXT_SEQUENCE            ; 0185
                             04    A7          50  D0 00067    MOVL    R0, 4(SCB)                              ;
                                   50          6E  D0 0006B    MOVL    BLOCK, R0                               ; 0186
                             0C    A0    04    A7  D0 0006E    MOVL    4(SCB), 12(R0)                          ;
                             10    A0    04    A7  D0 00072    MOVL    4(SCB), 16(R0)                          ; 0187
```

```
                14    A0      0000G   CF   D0 00084          MOVL    LCL_CSID, 20(R0)                ; 0188
                      51      0000G   CF   D0 0008A          MOVL    LCL_NOD, R1                     ; 0189
                18    A0              51   D0 0008F          MOVL    R1, 24(R0)                      ;
                                      0A   13 00093          BEQL    7$                              ;
                1C    A0        50    A1   D0 00095          MOVL    80(R1), 28(R0)                  ; 0192
                20    A0        54    A1   B0 0009A          MOVW    84(R1), 32(R0)                  ; 0193
                08    BC              50   D0 0009F 7$:      MOVL    R0, @DS_ADDR                    ; 0198
                      50              01   D0 000A3          MOVL    #1, R0                          ; 0200
                                           04 000A6          RET                                     ; 0202

; Routine Size:  167 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  205     0203  1 GLOBAL ROUTINE CREATE_OCD (SCOPE, UIC, BLOCK) : NOVALUE =
;  206     0204  1
;  207     0205  1 !++
;  208     0206  1 ! Functional description:
;  209     0207  1 !
;  210     0208  1 !       This routine contains the specializedlogic to crea e and initialize
;  211     0209  1 !       an OCD and to insert it onto the appropriate OCD l.st.  The OCD list
;  212     0210  1 !       depends on the SCOPE.
;  213     0211  1 !
;  214     0212  1 ! Input:
;  215     0213  1 !
;  216     0214  1 !       SCOPE   : The sccpe of the OCD.
;  217     0215  1 !       UIC     : The UIC of the OCD.
;  218     0216  1 !
;  219     0217  1 ! Implicit Input:
;  220     0218  1 !
;  221     0219  1 !       None.
;  222     0220  1 !
;  223     0221  1 ! Output:
;  224     0222  1 !
;  225     0223  1 !       BLOCK   : The address of a longword to receive the
;  226     0224  1 !                 address of the new OCD.
;  227     0225  1 !
;  228     0226  1 ! Implict output:
;  229     0227  1 !
;  230     0228  1 !       None.
;  231     0229  1 !
;  232     0230  1 ! Side effects:
;  233     0231  1 !
;  234     0232  1 !       None.
;  235     0233  1 !
;  236     0234  1 ! Routine value:
;  237     0235  1 !
;  238     0236  1 !       None.
;  239     0237  1 !--
;  240     0238  1
;  241     0239  2 BEGIN                                           ! Start of CREATE_OCD
;  242     0240  2
;  243     0241  2 EXTERNAL LITERAL
;  244     0242  2         OCD_K_TYPE;                             ! Structure type
;  245     0243  2
;  246     0244  2 EXTERNAL ROUTINE
;  247     0245  2         ALLOCATE_DS;                            ! Allocate a data structure
;  248     0246  2
;  249     0247  2 EXTERNAL
;  250     0248  2         OCD_VECTOR      : VECTOR;               ! Pointers to OCD list heads
;  251     0249  2
;  252     0250  2 LOCAL
;  253     0251  2         STATUS          : LONG,
;  254     0252  2         OCD             : $ref_bblock;          ! OCD data structure
;  255     0253  2
;  256     0254  2 !
;  257     0255  2 ! NOTE: This routine is incomplete, and must later be enhanced
;  258     0256  2 !       to create and link in a default LCB as well.
;  259     0257  2 !
;  260     0258  3 IF NOT (ALLOCATE_DS (OCD_K_TYPE, OCD))          ! Allocate a new OCD
;  261     0259  2 THEN
```

```
;  262          0260  2      RETURN (.STATUS);
;  263          0261  2 OCD [OCD_L_FLINK] = OCD [OCD_L_FLINK];          ! Init OCD list head
;  264          0262  2 OCD [OCD_L_BLINK] = OCD [OCD_L_FLINK];          !
;  265          0263  2 OCD [OCD_L_RQSTFLINK] = OCD [OCD_L_RQSTFLINK];  ! Init request list head
;  266          0264  2 OCD [OCD_L_RQSTBLINK] = OCD [OCD_L_RQSTFLINK];  !
;  267          0265  2 OCD [OCD_L_OPERFLINK] = OCD [OCD_L_OPERFLINK];  ! Init operator list head
;  268          0266  2 OCD [OCD_L_OPERBLINK] = OCD [OCD_L_OPERFLINK];  !
;  269          0267  2 OCD [OCD_B_SCOPE] = .SCOPE;                     ! Set the OCD SCOPE
;  270          0268  2 OCD [OCD_L_UIC] = .UIC;                         ! Set the OCD UIC
;  271          0269  2 OCD [OCD_L_NOTIFYMASK1] = -1;                   ! Broadcast all messages
;  272          0270  2 OCD [OCD_L_NOTIFYMASK2] = -1;                   !
;  273          0271  2 !
;  274          0272  2 ! Imsert the OCD in the appropriate OCD list.  If the list is empty,
;  275          0273  2 ! then this OCD will be the list head, and the OCD_VECTOR must be
;  276          0274  2 ! made to point to it.  After inserting the OCD, increment the OCD count
;  277          0275  2 ! for this list.
;  278          0276  2 !
;  279          0277  2 IF .OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2+1] EQL 0
;  280          0278  2 THEN
;  281          0279  2     OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2] = .OCD! Make OCD_VECTOR point to the OCD
;  282          0280  2 ELSE
;  283          0281  2     INSQUE (.OCD, .OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2]);
;  284          0282  2 OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2+1] = .OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2+1] + 1;
;  285          0283  2 .BLOCK = .OCD;                                  ! Return OCD address
;  286          0284  2
;  287          0285  1 END;                                           ! End of CREATE_OCD
; INFO#250              L1:0260
; Referenced LOCAL symbol STATUS is probably not initialized

                                                .EXTRN  OCD_K_TYPE, OCD_VECTOR

                                        001C 00000      .ENTRY  CREATE_OCD, Save R2,R3,R4       ; 0203
                        54      0000G   CF   9E 00002   MOVAB   OCD_VECTOR-4, R4                ;
                        52              04   C2 00007   SUBL2   #4, SP                         ;
                                        5E   DD 0000A   PUSHL   SP                              ; 0258
                                00000000G 8F DD 0000C   PUSHL   #OCD_K_TYPE                     ;
                    0000G   CF          02   FB 00012   CALLS   #2, ALLOCATE_DS                 ;
                            66          50   E9 00017   BLBC    R0, 3$                       ;
                            50          6E   D0 0001A   MOVL    OCD, R0                      ; 0261
                            60          50   D0 0001D   MOVL    R0, (R0)                     ;
                        04  A0          50   D0 00020   MOVL    R0, 4(R0)                    ; 0262
                        3C  A0      3C  A0   9E 00024   MOVAB   60(R0), 60(R0)               ; 0263
                        40  A0      3C  A0   9E 00029   MOVAB   60(R0), 64(R0)               ; 0264
                        50  A0      50  A0   9E 0002E   MOVAB   80(R0), 80(R0)               ; 0265
                        54  A0      50  A0   9E 00033   MOVAB   80(R0), 84(R0)               ; 0266
                        0B  A0      04  AC   90 00038   MOVB    SCOPE, 11(R0)                ; 0267
                        24  A0      08  AC   D0 0003D   MOVL    UIC, 36(R0)                  ; 0268
                        2C  A0          01   CE 00042   MNEGL   #1, 44(R0)                   ; 0269
                        30  A0          01   CE 00046   MNEGL   #1, 48(R0)                   ; 0270
                            52      0B  A0   9A 0004A   MOVZBL  11(R0), R2                   ; 0277
                51          52          01   78 0004E   ASHL    #1, R2, R1                   ;
                                      6441   D5 00052   TSTL    OCD_VECTOR-4[R1]             ;
                                        0B   12 00055   BNEQ    1$                           ;
                51          52          01   78 00057   ASHL    #1, R2, R1                   ; 0279
                        FC  A441        50   D0 0005B   MOVL    R0, OCD_VECTOR-8[R1]         ;
```

```
                                    0D  11 00060          BRB     2$                                      ;
                51          52      01  78 00062 1$:     ASHL    #1, R2, R1                              ; 0281
                            53  FC A441  DE 00066          MOVAL   OCD_VECTOR-8[R1], R3                    ;
                00      B3          60  0E 0006B          INSQUE  (R0), @0(R3)                            ;
                            51      6E  D0 0006F 2$:      MOVL    OCD, R1                                 ; 0282
                            50  0B  A1  9A 00072          MOVZBL  11(R1), R0                              ;
                            50      02  C4 00076          MULL2   #2, R0                                  ;
                                6440    D6 00079          INCL    OCD_VECTOR-4[R0]                        ;
                0C      BC          51  D0 0007C          MOVL    R1, @BLOCK                              ; 0283
                                        04 00080 3$:      RET                                             ; 0285

; Routine Size:  129 bytes,    Routine Base:  $CODE$ + 00A7
```

```
;  289     0286  1 GLOBAL ROUTINE DEALLOCATE_DS (DS_ADDR) =
;  290     0287  1
;  291     0288  1 !++
;  292     0289  1 ! Functional description:
;  293     0290  1 !
;  294     0291  1 !       This routine will deallocate a data structure.
;  295     0292  1 !       The data structure is zeroed and placed on the
;  296     0293  1 !       appropriate free list.
;  297     0294  1 !
;  298     0295  1 ! Input:
;  299     0296  1 !
;  300     0297  1 !       DS_ADDR = The address of the data structure.
;  301     0298  1 !
;  302     0299  1 ! Implicit Input:
;  303     0300  1 !
;  304     0301  1 !       The SCB data base.
;  305     0302  1 !
;  306     0303  1 ! Output:
;  307     0304  1 !
;  308     0305  1 !       None.
;  309     0306  1 !
;  310     0307  1 ! Implict output:
;  311     0308  1 !
;  312     0309  1 !       The SCB entry for this structure type is updated.
;  313     0310  1 !
;  314     0311  1 ! Side effects:
;  315     0312  1 !
;  316     0313  1 !       None.
;  317     0314  1 !
;  318     0315  1 ! Routine value:
;  319     0316  1 !
;  320     0317  1 !       TRUE             = The deallocate succeeded.
;  321     0318  1 !       <anything else> = The deallocate failed.
;  322     0319  1 !--
;  323     0320  1
;  324     0321  2 BEGIN                                           ! Start of ALLOCATE_DS
;  325     0322  2
;  326     0323  2 MAP
;  327     0324  2         DS_ADDR          : $ref_bblock;
;  328     0325  2
;  329     0326  2 EXTERNAL
;  330     0327  2         SCB_TABLE        : VECTOR;              ! SCB table of pointers
;  331     0328  2
;  332     0329  2 EXTERNAL LITERAL
;  333     0330  2         SS$_BADPARAM,                           ! System status value
;  334     0331  2         MIN_DS_TYPE,                            ! Min STRUCTURE_TYPE value
;  335     0332  2         MAX_DS_TYPE;                            ! Max STRUCTURE_TYPE value
;  336     0333  2
;  337     0334  2 LOCAL
;  338     0335  2         SCB              : $ref_bblock;         ! SCB structure
;  339     0336  2
;  340     0337  2 !
;  341     0338  2 ! Check input parameter.
;  342     0339  2 !
;  343     0340  3 IF (.DS_ADDR [HDR_B_TYPE] LSS MIN_DS_TYPE)
;  344     0341  3 OR (.DS_ADDR [HDR_B_TYPE] GTR MAX_DS_TYPE)
;  345     0342  3 OR (.DS_ADDR EQL 0)
```

```
;   346          0343  2 THEN
;   347          0344  2     RETURN (FALSE);
;   348          0345  2
;   349          0346  2 ! Zero the block and place it on the correct look aside list.
;   350          0347  2 ! The first 11 bytes of the data structure are not zeroed, because
;   351          0348  2 ! they contain info that will identify the structure.
;   352          0349  2 !
;   353          0350  2 SCB = .SCB_TABLE [.DS_ADDR [HDR_B_TYPE] - 1];    ! Get the SCB address
;   354          0351  2 CH$FILL (0, (.SCB [SCB_W_SIZE] - $BYTEOFFSET (HDR_B_SCOPE)), (.DS_ADDR + $BYTEOFFSET (HDR_B_SCOPE)));
;   355          0352  2 DS_ADDR [HDR_V_LAL] = TRUE;                      ! Mark this an LAL block
;   356          0353  2 INSQUE (DS_ADDR [HDR_L_FLINK], SCB [SCB_L_FLINK]);    ! Put structure on LAL
;   357          0354  2 SCB [SCB_W_LAL_COUNT] = .SCB [SCB_W_LAL_COUNT] + 1;    ! Incr. the LAL count
;   358          0355  2
;   359          0356  2 RETURN (TRUE);
;   360          0357  2
;   361          0358  1 END;                                    ! End of DEALLOCATE_DS


                                                  00FC 00000          .ENTRY  DEALLOCATE_DS, Save R2,R3,R4,R5,R6,R7      ; 0286
                                          57  04  AC  D0 00002        MOVL    DS_ADDR, R7                                ; 0340
00000000G   8F      0A  A7          08          00  ED 00006        CMPZV   #0, #8, 10(R7), #MIN_DS_TYPE               ;
                                                39  19 00010        BLSS    1$                                         ;
00000000G   8F      0A  A7          08          00  ED 00012        CMPZV   #0, #8, 10(R7), #MAX_DS_TYPE               ; 0341
                                                2D  14 0001C        BGTR    1$                                         ;
                                                57  D5 0001E        TSTL    R7                                         ; 0342
                                                29  13 00020        BEQL    1$                                         ;
                                          50        0A  A7  9A 00022        MOVZBL  10(R7), R0                         ; 0350
                                          56  0000GCF   40  D0 00026        MOVL    SCB_TABLE-4[R0], SCB               ;
                                          50            66  3C 0002C        MOVZWL  (SCB), R0                          ; 0351
                                          50            0B  C2 0002F        SUBL2   #11, R0                            ;
        50              00                6E            00  2C 00032        MOVC5   #0, (SP), #0, R0, 11(R7)           ;
                                                    0B  A7      00037                                                   ;
                                    28  A7                  01  88 00039        BISB2   #1, 40(R7)                         ; 0352
                                          50        08  A6  9E 0003D        MOVAB   8(R6), R0                          ; 0353
                                          60            67  0E 00041        INSQUE  (R7), (R0)                         ;
                                                    02  A6  B6 00044        INCW    2(SCB)                             ; 0354
                                          50            01  D0 00047        MOVL    #1, R0                             ; 0356
                                                        04  0004A        RET                                        ;
                                                50      D4  0004B 1$:    CLRL    R0                                 ; 0358
                                                        04  0004D        RET                                        ;

; Routine Size:  78 bytes,    Routine Base:  $CODE$ + 0128
```

```
;  363     0359  1 GLOBAL ROUTINE DEALLOCATE_MCB (MCB) : NOVALUE =
;  364     0360  1
;  365     0361  1 !++
;  366     0362  1 ! Functional description:
;  367     0363  1 !
;  368     0364  1 !       This routine contains the specialized logic to
;  369     0365  1 !       deallocate an MCB.
;  370     0366  1 !
;  371     0367  1 ! Input:
;  372     0368  1 !
;  373     0369  1 !       MCB     : Address of an MCB.
;  374     0370  1 !
;  375     0371  1 ! Implicit Input:
;  376     0372  1 !
;  377     0373  1 !       None.
;  378     0374  1 !
;  379     0375  1 ! Output:
;  380     0376  1 !
;  381     0377  1 !       None.
;  382     0378  1 !
;  383     0379  1 ! Implict output:
;  384     0380  1 !
;  385     0381  1 !       None.
;  386     0382  1 !
;  387     0383  1 ! Side effects:
;  388     0384  1 !
;  389     0385  1 !       None.
;  390     0386  1 !
;  391     0387  1 ! Routine value:
;  392     0388  1 !
;  393     0389  1 !       None.
;  394     0390  1 !--
;  395     0391  1
;  396     0392  2 BEGIN                                           ! Start of DEALLOCATE_MCB
;  397     0393  2
;  398     0394  2 MAP
;  399     0395  2         MCB             : $ref_bblock;
;  400     0396  2
;  401     0397  2 EXTERNAL ROUTINE
;  402     0398  2         DEALLOCATE_DS;                          ! Deallocate a data structure
;  403     0399  2
;  404     0400  2 EXTERNAL LITERAL
;  405     0401  2         MCB_K_TYPE,                             ! MCB structure type
;  406     0402  2         SS$_BADPARAM;                           ! System status code
;  407     0403  2
;  408     0404  2 LOCAL
;  409     0405  2         RQCB            : $ref_bblock;
;  410     0406  2
;  411     0407  2 !
;  412     0408  2 ! If block address is 0, then there is no work to do.
;  413     0409  2 !
;  414     0410  2 IF .MCB EQL 0
;  415     0411  2 THEN
;  416     0412  2     RETURN;
;  417     0413  2 !
;  418     0414  2 ! Make sure this is an MCB.
;  419     0415  2 !
```

```
;  420          0416  3 IF (.MCB [MCB_B_TYPE] NEQ MCB_K_TYPE)
;  421          0417  2 THEN
;  422          0418  2     RETURN;
;  423          0419  2 !
;  424          0420  2 ! If the MCB points to any message text, deallocate
;  425          0421  2 ! the text buffer.
;  426          0422  2 !
;  427          0423  2 IF .MCB [MCB_L_TEXTPTR] NEQ 0
;  428          0424  2 THEN
;  429          0425  3     BEGIN
;  430          0426  3     OPC$FREE_VM (MCB [MCB_L_TEXTLEN], MCB [MCB_L_TEXTPTR]);
;  431          0427  3     MCB [MCB_L_TEXTPTR] = 0;
;  432          0428  2     END;
;  433          0429  2 !
;  434          0430  2 ! Deallocate the MCB.  As an added precaution,
;  435          0431  2 ! explicitly zero the RQCB backpointer.
;  436          0432  2 !
;  437          0433  2 RQCB = .MCB [MCB_L_RQCB];                       ! Get back pointer to RQCB
;  438          0434  3 IF NOT (.RQCB EQL 0)
;  439          0435  2 THEN
;  440          0436  2     RQCB [RQCB_L_MCB] = 0;                      ! Break link from RQCB to MCB
;  441          0437  2 MCB [MCB_L_RQCB] = 0;                           ! Break link from MCB to RQCB
;  442          0438  2 DEALLOCATE_DS (.MCB);
;  443          0439  2
;  444          0440  1 END;                                            ! End of DEALLOCATE_MCB


                                                        .EXTRN  MCB_K_TYPE, OPC$FREE_VM

                                                0004 00000      .ENTRY  DEALLOCATE_MCB, Save R2                 ; 0359
                                    52      04  AC   D0 00002      MOVL    MCB, R2                             ; 0410
                                                32   13 00006      BEQL    3$                                  ;
00000000G  8F       0A      A2              08      00   ED 00008  CMPZV   #0, #8, 10(R2), #MCB_K_TYPE         ; 0416
                                                26   12 00012      BNEQ    3$                                  ;
                                            34  A2   D5 00014      TSTL    52(R2)                              ; 0423
                                                0E   13 00017      BEQL    1$                                  ;
                                            34  A2   9F 00019      PUSHAB  52(R2)                              ; 0426
                                            30  A2   9F 0001C      PUSHAB  48(R2)                              ;
                                0000G       CF      02   FB 0001F  CALLS   #2, OPC$FREE_VM                     ;
                                            34  A2   D4 00024      CLRL    52(R2)                              ; 0427
                                    50      24  A2   D0 00027 1$:  MOVL    36(R2), RQCB                        ; 0433
                                                03   13 0002B      BEQL    2$                                  ; 0434
                                            6C  A0   D4 0002D      CLRL    108(RQCB)                           ; 0436
                                            24  A2   D4 00030 2$:  CLRL    36(R2)                              ; 0437
                                                52   DD 00033      PUSHL   R2                                  ; 0438
                                0000G       CF      01   FB 00035  CALLS   #1, DEALLOCATE_DS                   ;
                                                    04   0003A 3$:  RET                                         ; 0440

; Routine Size:  59 bytes,    Routine Base:  $CODE$ + 0176
```

```
;  446          0441  1 GLOBAL ROUTINE DEALLOCATE_OCD (OCD) : NOVALUE =
;  447          0442  1
;  448          0443  1 !++
;  449          0444  1 ! Functional description:
;  450          0445  1 !
;  451          0446  1 !       This routine contains the specialized logic to deallocate
;  452          0447  1 !       an OCD to the appropriate free list.
;  453          0448  1 !
;  454          0449  1 ! Input:
;  455          0450  1 !
;  456          0451  1 !       OCD     : Address of an OCD
;  457          0452  1 !
;  458          0453  1 ! Implicit Input:
;  459          0454  1 !
;  460          0455  1 !       None.
;  461          0456  1 !
;  462          0457  1 ! Output:
;  463          0458  1 !
;  464          0459  1 !       None.
;  465          0460  1 !
;  466          0461  1 ! Implict output:
;  467          0462  1 !
;  468          0463  1 !       None.
;  469          0464  1 !
;  470          0465  1 ! Side effects:
;  471          0466  1 !
;  472          0467  1 !       None.
;  473          0468  1 !
;  474          0469  1 ! Routine value:
;  475          0470  1 !
;  476          0471  1 !       None.
;  477          0472  1 !--
;  478          0473  1
;  479          0474  2 BEGIN                                           ! Start of DEALLOCATE_OCD
;  480          0475  2
;  481          0476  2 MAP
;  482          0477  2         OCD             : $ref_bblock;          ! OCD structure
;  483          0478  2
;  484          0479  2 EXTERNAL ROUTINE
;  485          0480  2         DEALLOCATE_DS;                          ! Deallocate a data structure
;  486          0481  2
;  487          0482  2 EXTERNAL LITERAL
;  488          0483  2         OCD_K_TYPE;                             ! OCD structure type
;  489          0484  2
;  490          0485  2 EXTERNAL
;  491          0486  2         OCD_VECTOR      : VECTOR;               ! Pointers to OCDs
;  492          0487  2
;  493          0488  2 LOCAL
;  494          0489  2         TEMP            : LONG;                 ! Used to receive an address
;  495          0490  2
;  496          0491  2 !
;  497          0492  2 ! Make sure that the data structure is an OCD.
;  498          0493  2 !
;  499          0494  3 IF (.OCD EQL 0) OR (.OCD [OCD_B_TYPE] NEQ OCD_K_TYPE)
;  500          0495  2 THEN
;  501          0496  2     RETURN;
;  502          0497  2 !
```

```
;  503          0498  2 ! If the OCD is still busy, do not deallocate it.
;  504          0499  2 !
;  505          0500  3 IF (.OCD [OCD_W_OPERCOUNT] NEQ 0)
;  506          0501  3 OR (.OCD [OCD_W_RQSTCOUNT] NEQ 0)
;  507          0502  3 OR (.OCD [OCD_L_LCB] NEQ 0)
;  508          0503  2 THEN
;  509          0504  2     RETURN;
;  510          0505  2 !
;  511          0506  2 ! Decrement the OCD count of the appropriate location in
;  512          0507  2 ! the OCD vector and remove the OCD from the OCD list.
;  513          0508  2 ! If the count went to zero, then the pointer to the first
;  514          0509  2 ! OCD in the OCD vector must be zeroed.
;  515          0510  2 !
;  516          0511  2 REMQUE  (OCD [OCD_L_FLINK], TEMP);
;  517          0512  2 OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2+1] = .OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2+1] - 1;
;  518          0513  2 IF .OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2+1] EQL 0
;  519          0514  2 THEN
;  520          0515  2     OCD_VECTOR [(.OCD [OCD_B_SCOPE]-1)*2] = 0;
;  521          0516  2 !
;  522          0517  2 ! Release the OCD.
;  523          0518  2 !
;  524          0519  2 DEALLOCATE_DS (.OCD);
;  525          0520  2 RETURN;
;  526          0521  2
;  527          0522  1 END;                                    ! End of DEALLOCATE_OCD


                                        0004 00000          .ENTRY  DEALLOCATE_OCD, Save R2                 ; 0441
                                50      04    AC  D0 00002  MOVL    OCD, R0                                 ; 0494
                                                41  13 00006  BEQL    2$                                      ;
00000000G  8F       0A  A0              08      00  ED 00008  CMPZV   #0, #8, 10(R0), #OCD_K_TYPE            ;
                                                35  12 00012  BNEQ    2$                                      ;
                                            46  A0  B5 00014  TSTW    70(R0)                                  ; 0500
                                                30  12 00017  BNEQ    2$                                      ;
                                            3A  A0  B5 00019  TSTW    58(R0)                                  ; 0501
                                                2B  12 0001C  BNEQ    2$                                      ;
                                            34  A0  D5 0001E  TSTL    52(R0)                                  ; 0502
                                                26  12 00021  BNEQ    2$                                      ;
                                51              60  0F 00023  REMQUE  (R0), TEMP                              ; 0511
                                52      04    AC  D0 00026  MOVL    OCD, R2                                 ; 0512
                                51      0B    A2  9A 0002A  MOVZBL  11(R2), R1                              ;
                    50          51              01  78 0002E  ASHL    #1, R1, R0                              ;
                                        0000GCF 40  D7 00032  DECL    OCD_VECTOR-4[R0]                        ;
                                                09  12 00037  BNEQ    1$                                      ; 0513
                    50          51              01  78 00039  ASHL    #1, R1, R0                              ; 0515
                                        0000GCF 40  D4 0003D  CLRL    OCD_VECTOR-8[R0]                        ;
                                                52  DD 00042  1$:     PUSHL   R2                              ; 0519
                          0000G CF              01  FB 00044          CALLS   #1, DEALLOCATE_DS               ;
                                                    04 00049  2$:     RET                                     ; 0522

; Routine Size:  74 bytes,    Routine Base:  $CODE$ + 01B1
```

```
;  529     0523  1 GLOBAL ROUTINE DEALLOCATE_RQCB (RQCB) : NOVALUE =
;  530     0524  1
;  531     0525  1 !++
;  532     0526  1 ! Functional description:
;  533     0527  1 !
;  534     0528  1 !       This routine contains the specialized logic to
;  535     0529  1 !       deallocate an RQCB and any data structures that
;  536     0530  1 !       that may be linked to it.
;  537     0531  1 !
;  538     0532  1 ! Input:
;  539     0533  1 !
;  540     0534  1 !       RQCB    :  Address of an RQCB.
;  541     0535  1 !
;  542     0536  1 ! Implicit Input:
;  543     0537  1 !
;  544     0538  1 !       None.
;  545     0539  1 !
;  546     0540  1 ! Output:
;  547     0541  1 !
;  548     0542  1 !       None.
;  549     0543  1 !
;  550     0544  1 ! Implict output:
;  551     0545  1 !
;  552     0546  1 !       None.
;  553     0547  1 !
;  554     0548  1 ! Side effects:
;  555     0549  1 !
;  556     0550  1 !       None.
;  557     0551  1 !
;  558     0552  1 ! Routine value:
;  559     0553  1 !
;  560     0554  1 !       None.
;  561     0555  1 !--
;  562     0556  1
;  563     0557  2 BEGIN                                           ! Start of DEALLOCATE_RQCB
;  564     0558  2
;  565     0559  2 MAP
;  566     0560  2         RQCB            : $ref_bblock;
;  567     0561  2
;  568     0562  2 EXTERNAL ROUTINE
;  569     0563  2         DEALLOCATE_DS,                          ! Deallocate a data structure
;  570     0564  2         DEALLOCATE_MCB  : NOVALUE;              ! Deallocate an MCB
;  571     0565  2
;  572     0566  2 EXTERNAL LITERAL
;  573     0567  2         RQCB_K_TYPE;                            ! RQCB structure type
;  574     0568  2
;  575     0569  2 !
;  576     0570  2 ! If no block specified, then return.
;  577     0571  2 !
;  578     0572  2 IF .RQCB EQL 0
;  579     0573  2 THEN
;  580     0574  2     RETURN;
;  581     0575  2 !
;  582     0576  2 ! Make sure this is a RQCB.
;  583     0577  2 !
;  584     0578  3 IF (.RQCB [RQCB_B_TYPE] NEQ RQCB_K_TYPE)
;  585     0579  2 THEN
```

```
;  586          0580  2      RETURN;
;  587          0581  2  !
;  588          0582  2  ! If there is an MCB linked to the RQCB then deallocate it.
;  589          0583  2  !
;  590          0584  2  IF .RQCB [RQCB_L_MCB] NEQ 0
;  591          0585  2  THEN
;  592          0586  3      BEGIN
;  593          0587  3      DEALLOCATE_MCB (.RQCB [RQCB_L_MCB]);
;  594          0588  3      RQCB [RQCB_L_MCB] = 0;
;  595          0589  2      END;
;  596          0590  2  !
;  597          0591  2  ! If the operator name descriptor is valid, deallocate
;  598          0592  2  ! the block of memory it points to.
;  599          0593  2  !
;  600          0594  2  IF .RQCB [RQCB_L_OPER_PTR] NEQ 0
;  601          0595  2  THEN
;  602          0596  3      BEGIN
;  603          0597  3      OPC$FREE_VM (RQCB [RQCB_L_OPER_LEN], RQCB [RQCB_L_OPER_PTR]);
;  604          0598  3      RQCB [RQCB_L_OPER_PTR] = 0;
;  605          0599  2      END;
;  606          0600  2  !
;  607          0601  2  ! If the request/reply text descriptor is valid,
;  608          0602  2  ! deallocate the block of memory it points to.
;  609          0603  2  !
;  610          0604  2  IF .RQCB [RQCB_L_TEXT_PTR] NEQ 0
;  611          0605  2  THEN
;  612          0606  3      BEGIN
;  613          0607  3      OPC$FREE_VM (RQCB [RQCB_L_TEXT_LEN], RQCB [RQCB_L_TEXT_PTR]);
;  614          0608  3      RQCB [RQCB_L_TEXT_PTR] = 0;
;  615          0609  2      END;
;  616          0610  2  !
;  617          0611  2  ! Deallocate the RQCB.
;  618          0612  2  !
;  619          0613  2  DEALLOCATE_DS (.RQCB);
;  620          0614  2
;  621          0615  1  END;                                   ! End of DEALLOCATE_RQCB


                                                        .EXTRN  RQCB_K_TYPE

                                                 000C 00000     .ENTRY  DEALLOCATE_RQCB, Save R2,R3                     ; 0523
                                      52    04   AC  D0 00002     MOVL    RQCB, R2                                    ; 0572
                                                 4E  13 00006     BEQL    4$                                          ;
00000000G  8F     0A  A2              08         00  ED 00008     CMPZV   #0, #8, 10(R2), #RQCB_K_TYPE                ; 0578
                                                 42  12 00012     BNEQ    4$                                          ;
                                            6C   A2  D5 00014     TSTL    108(R2)                                     ; 0584
                                                 0B  13 00017     BEQL    1$                                          ;
                                            6C   A2  DD 00019     PUSHL   108(R2)                                     ; 0587
                                  0000G     CF       01  FB 0001C     CALLS   #1, DEALLOCATE_MCB                      ;
                                            6C   A2  D4 00021     CLRL    108(R2)                                     ; 0588
                                      53    0080 C2  9E 00024 1$:  MOVAB   128(R2), R3                                 ; 0594
                                                 63  D5 00029     TSTL    (R3)                                        ;
                                                 0C  13 0002B     BEQL    2$                                          ;
                                                 53  DD 0002D     PUSHL   R3                                          ; 0597
                                            7C   A2  9F 0002F     PUSHAB  124(R2)                                     ;
                                  0000G     CF       02  FB 00032     CALLS   #2, OPC$FREE_VM                         ;
```

```
                                 63  D4 00037      CLRL    (R3)                          ; 0598
          53     0088            C2  9E 00039 2$:  MOVAB   136(R2), R3                   ; 0604
                                 63  D5 0003E      TSTL    (R3)                          ;
                                 0D  13 00040      BEQL    3$                            ;
                                 53  DD 00042      PUSHL   R3                            ; 0607
                 0084            C2  9F 00044      PUSHAB  132(R2)                       ;
      0000G  CF                  02  FB 00048      CALLS   #2, OPC$FREE_VM               ;
                                 63  D4 0004D      CLRL    (R3)                          ; 0608
                                 52  DD 0004F 3$:  PUSHL   R2                            ; 0613
      0000G  CF                  01  FB 00051      CALLS   #1, DEALLOCATE_DS             ;
                                     04 00056 4$:  RET                                   ; 0615

; Routine Size:  87 bytes,    Routine Base:  $CODE$ + 01FB


;  622          0616  1
;  623          0617  1 END                                     ! End of ALLOCATE
;  624          0618  0 ELUDOM



;                                 PSECT SUMMARY
;
;       Name                      Bytes                     Attributes
;
;  $CODE$                            594  NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)


;                            Library Statistics
;
;                                       -------- Symbols --------      Pages      Processing
;       File                            Total    Loaded   Percent      Mapped     Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1         18619        3         0       1000      00:01.8
;  _$255$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1   633       48         7         43      00:00.8


; Information:   1
; Warnings:      0
; Errors:        0


;                                 COMMAND QUALIFIERS

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

; Size:          594 code + 0 data bytes
; Run Time:         00:14.9
; Elapsed Time:     00:53.6
```

```
; Lines/CPU Min:     2481
; Lexemes/CPU-Min: 17626
; Memory Used:  98 pages
; Compilation Complete
```

NPARSE
LIS

REQUEST
MAP

OPCOM

REPLY
MAP

NMLXFER
LIS

OPCCRASH
MAP

NMLV2STA
LIS

OPCOMLIB
B32

ALLOCATE
LIS

NMLZERO
LIS

OPCOMDEF
SDL

OPCOM
MAP

OPCDEFTMP
SDL

CANCEL
LIS