C 8

```
  000000000   PPPPPPPPPPPP    CCCCCCCCCCCC   000000000   MMM         MMM
  000000000   PPPPPPPPPPPP    CCCCCCCCCCCC   000000000   MMM         MMM
  000000000   PPPPPPPPPPPP    CCCCCCCCCCCC   000000000   MMM         MMM
000       000 PPP      PPP  CCC            000       000 MMMMMM   MMMMMM
000       000 PPP      PPP  CCC            000       000 MMMMMM   MMMMMM
000       000 PPP      PPP  CCC            000       000 MMMMMM   MMMMMM
000       000 PPP      PPP  CCC            000       000 MMM   MMM   MMM
000       000 PPP      PPP  CCC            000       000 MMM   MMM   MMM
000       000 PPP      PPP  CCC            000       000 MMM   MMM   MMM
000       000 PPPPPPPPPPPP  CCC            000       000 MMM         MMM
000       000 PPPPPPPPPPPP  CCC            000       000 MMM         MMM
000       000 PPPPPPPPPPPP  CCC            000       000 MMM         MMM
000       000 PPP           CCC            000       000 MMM         MMM
000       000 PPP           CCC            000       000 MMM         MMM
000       000 PPP           CCC            000       000 MMM         MMM
000       000 PPP           CCC            000       000 MMM         MMM
000       000 PPP           CCC            000       000 MMM         MMM
000       000 PPP           CCC            000       000 MMM         MMM
  000000000   PPP             CCCCCCCCCCCC   000000000   MMM         MMM
  000000000   PPP             CCCCCCCCCCCC   000000000   MMM         MMM
  000000000   PPP             CCCCCCCCCCCC   000000000   MMM         MMM
```

```
_$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
```

```
**FILE**ID**OPERUTIL                              K 16






  OOOOOO    PPPPPPPP    EEEEEEEEEE  RRRRRRRR    UU      UU  TTTTTTTTTT    IIIIII    LL
  OOOOOO    PPPPPPPP    EEEEEEEEEE  RRRRRRRR    UU      UU  TTTTTTTTTT    IIIIII    LL
OO      OO  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
OO      OO  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
OO      OO  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
OO      OO  PP      PP  EE          RR      RR  UU      UU      TT          II      LL
OO      OO  PPPPPPPP    EEEEEEEE    RRRRRRRR    UU      UU      TT          II      LL
OO      OO  PPPPPPPP    EEEEEEEE    RRRRRRRR    UU      UU      TT          II      LL
OO      OO  PP          EE          RR  RR      UU      UU      TT          II      LL
OO      OO  PP          EE          RR  RR      UU      UU      TT          II      LL
OO      OO  PP          EE          RR    RR    UU      UU      TT          II      LL              ....
OO      OO  PP          EE          RR    RR    UU      UU      TT          II      LL              ....
  OOOOOO    PP          EEEEEEEEEE  RR      RR  UUUUUUUUUU      TT        IIIIII    LLLLLLLLLL      ....
  OOOOOO    PP          EEEEEEEEEE  RR      RR  UUUUUUUUUU      TT        IIIIII    LLLLLLLLLL      ....


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
```

```
0 MODULE  OPC$OPERUTIL    (
0                           LANGUAGE (BLISS32),
0                           IDENT = 'V04-000'
0                         ) =
0
0 !*****************************************************************************
0 !*                                                                           *
0 !*   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0 !*   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0 !*   ALL RIGHTS RESERVED.                                                    *
0 !*                                                                           *
0 !*   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0 !*   ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0 !*   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0 !*   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0 !*   OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0 !*   TRANSFERRED.                                                            *
0 !*                                                                           *
0 !*   THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0 !*   AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0 !*   CORPORATION.                                                            *
0 !*                                                                           *
0 !*   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0 !*   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0 !*                                                                           *
0 !*                                                                           *
0 !*****************************************************************************
0
0 !++
0 ! FACILITY:
0 !
0 !       OPCOM
0 !
0 ! ABSTRACT:
0 !
0 !       This module contains the general utility routines used
0 !       to manipulate operator control blocks.  These routines
0 !       are used freely throughout OPCOM's request handlers.
0 !
0 ! Environment:
0 !
0 !       VAX/VMS operating system.
0 !
0 ! Author:
0 !
0 !       Steven T. Jeffreys
0 !
0 ! Creation date:
0 !
0 !       March 10, 1981
0 !
0 ! Revision historv:
0 !
0 !       V03-004 CWH3169         CW Hobbs                        5-May-1984
0 !               Second pass for cluster-wide OPCOM:
0 !                - Use queued brkthru mechanism to send messages.
0 !                - Add DVI$_ code to SHARE_FULL_DEVNAME calls.
```

```
0 !
0 !             V03-003 CWH3003         CW Hobbs                        16-Sep-1983
0 !                     Add a flag that a disable is in progress to prevent recursive
0 !                     disables.  Change $BRDCST to $BRKTHRU.  Comment out incomplete
0 !                     code for mailboxes as operators.
0 !
0 !             V03-002 CWH3002         CW Hobbs                        30-Jul-1983
0 !                     Various and sundry things to make OPCOM distributed
0 !                     across the cluster.
0 !
0 ! V03-001               STJ3034         Steven T. Jeffreys,     06-Oct-1982
0 !                     Check for dial-in terminal.  Treat it as a remote terminal.
0 !
0 ! V02-002               STJ0165         Steven T. Jeffreys,     08-Feb-1982
0 !                     Make references to library routines use general addressing mode.
0 !
0 !--
0
1 BEGIN                                                 ! Start of OPERUTIL
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 LIBRARY 'LIB$:OPCOMLIB';
1
1 FORWARD ROUTINE
1               CHECK_OPER_COVERAGE : NOVALUE,          ! Check operator coverage on requests
1               FIND_OPERATOR,                          ! Find a given operator RQCB
1               IMPLICIT_DISABLE,                       ! Check for implict disable of an operator
1               NOTIFY_LISTED_OPERATORS,                ! Notify the operators on the given list
1               NOTIFY_OPERATOR,                        ! Send a message to an operator
1               OPERUTIL_CLM_IMP_DISABLE : NOVALUE,     ! Implicit disable from another node
1               UPD_OPER_CONTEXT,                       ! Update operator context
1               VALID_OPERATOR;                         ! Check for valid operator device
1
1 BUILTIN
1               INSQUE,                                 ! Insert entry onto a queue
1               REMQUE;                                 ! Remove entry from a queue
1
1 EXTERNAL
1               GLOBAL_STATUS : BITVECTOR [32],
1               LCL_NOD : $ref_bblock,
1               LCL_CSID;
1
1 EXTERNAL ROUTINE
1               CLUSUTIL_SYSTEMID_EQUAL : JSB_ROR1,
1               DUMP_LOG_FILE,
1               REPLYBRD_BRKTHRU_QUEUE : NOVALUE;       ! Queue a $brkthru I/O
```

```
1 GLOBAL ROUTINE CHECK_OPER_COVERAGE (OCD) : NOVALUE =
1
1 !++
1 ! Functional descripton:
1 !
1 !       This routine will check all outstanding requests queued to
1 !       a given OCD for proper operator coverage.  Any request that
1 !       no longer has operator coverage will be canceled.  The requestor
1 !       will receive a NOPERATOR cancelation message.  No operators are
1 !       notified, since none are interested in the request.  The cancelation
1 !       is, however, logged.
1 !
1 ! Input:
1 !
1 !       OCD     : Address of an OCD
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implicit Output:
1 !
1 !       None.
1 !
1 ! Side Effects:
1 !
1 !       None.
1 !
1 ! Routine Value
1 !
1 !       None.
1 !--
1
2 BEGIN                                             ! Start of CHECK_OPER_COVERAGE
2
2 MAP
2       OCD             : $ref_bblock;              ! OCD data structure
2
2 EXTERNAL ROUTINE
2       DEALLOCATE_RQCB : NOVALUE,                  ! Dispose of an RQCB
2       FORMAT_MESSAGE,                             ! Format a message
2       LOG_MESSAGE,                                ! Log an event
2       SEND_REPLY;                                 ! Send a reply to reply mailbox
2
2 LOCAL
2       MESSAGE_VECTOR  : VECTOR [2,LONG],          ! Message info
2       MCB             : $ref_bblock,              ! MCB data structure
2       RQST_COUNT      : LONG,                     ! Count of outstanding requests
2       RQST            : $ref_bblock,              ! Pointer to current request RQCB
2       NEXT_RQST       : LONG;                     ! Pointer to next request RQCB
2 !
2 ! Set up the message info vector.
2 !
```

```
;  162       0161  2 MESSAGE_VECTOR [0] = OPC$_NOPERATOR;                       ! Set message code
;  163       0162  2 MESSAGE_VECTOR [1] = 0;                                    ! Set # of FAO arguements
;  164       0163  2 !
;  165       0164  2 ! Set up for the search loop.
;  166       0165  2 !
;  167       0166  2 NEXT_RQST = .OCD [OCD_L_RQSTFLINK];                        ! Get address of next RQCB
;  168       0167  2 RQST_COUNT = .OCD [OCD_W_RQSTCOUNT];                       ! Get count of requests
;  169       0168  2 WHILE (.RQST_COUNT GTR 0) DO
;  170       0169  3     BEGIN
;  171       0170  3     !
;  172       0171  3     ! Compare the request attention mask against the operator
;  173       0172  3     ! intererst mask for this OCD.  If there are no common
;  174       0173  3     ! bits, then the request does not have any operator coverage
;  175       0174  3     ! and must be canceled.
;  176       0175  3     !
;  177       0176  3     RQST = .NEXT_RQST;                                     ! Get address of request RQCB
;  178       0177  3     NEXT_RQST = .RQST [RQCB_L_FLINK];                      ! Get address of next RQCB
;  179       0178  4     IF  ((.RQST [RQCB_L_ATTNMASK1] AND .OCD [OCD_L_ATTNMASK1]) EQL 0)
;  180       0179  4     AND ((.RQST [RQCB_L_ATTNMASK2] AND .OCD [OCD_L_ATTNMASK2]) EQL 0)
;  181       0180  3     THEN
;  182       0181  4         BEGIN
;  183       0182  4         !
;  184       0183  4         ! Cancel the request.  This entails removing it from the OCD
;  185       0184  4         ! request queue, sending the cancel notice to the requestor,
;  186       0185  4         ! and deallocating the request RQCB.
;  187       0186  4         !
;  188       0187  4         REMQUE (.RQST, RQST);                              ! Dequeue the request
;  189       0188  4         OCD [OCD_W_RQSTCOUNT] = .OCD [OCD_W_RQSTCOUNT] - 1;
;  190       0189  4         FORMAT_MESSAGE (.RQST, MESSAGE_VECTOR);
;  191       0190  4         SEND_REPLY (.RQST);
;  192       0191  4         LOG_MESSAGE (.RQST);
;  193       0192  4         DEALLOCATE_RQCB (.RQST);
;  194       0193  3         END;
;  195       0194  3     RQST_COUNT = .RQST_COUNT - 1;
;  196       0195  2     END;
;  197       0196  2
;  198       0197  1 END;                                                       ! End of CHECK_OPER_COVERAGE


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

                                                                  .EXTRN  GLOBAL_STATUS, LCL_NOD
                                                                  .EXTRN  LCL_CSID, CLUSUTIL_SYSTEMID_EQUAL
                                                                  .EXTRN  DUMP_LOG_FILE, REPLYBRD_BRKTHRU_QUEUE
                                                                  .EXTRN  DEALLOCATE_RQCB
                                                                  .EXTRN  FORMAT_MESSAGE, LOG_MESSAGE
                                                                  .EXTRN  SEND_REPLY

                                                                  .PSECT  $CODE$,NOWRT,2

                                           001C 00000             .ENTRY  CHECK_OPER_COVERAGE, Save R2,R3,R4             ; 0104
                        5E          04  C2 00002                  SUBL2   #4, SP                                         ;
                           00058061 8F  DD 00005                  PUSHL   #360545                                        ; 0161
                                 04 AE  D4 0000B                  CLRL    MESSAGE_VECTOR+4                               ; 0162
                        50       04 AC  D0 0000E                  MOVL    OCD, R0                                        ; 0166
                        53       3C A0  D0 00012                  MOVL    60(R0), NEXT_RQST                              ;
```

```
                 54         3A   A0  3C 00016          MOVZWL   58(R0), RQST_COUNT                                  ; 0167
                                 44  15 0001A 1$:      BLEQ     3$                                                  ; 0168
                 52              53  D0 0001C          MOVL     NEXT_RQST, RQST                                     ; 0176
                 53              62  D0 0001F          MOVL     (RQST), NEXT_RQST                                   ; 0177
                 50         04   AC  D0 00022          MOVL     OCD, R0                                             ; 0178
         48      A0         5C   A2  D3 00026          BITL     92(RQST), 72(R0)                                    ;
                                 2F  12 0002B          BNEQ     2$                                                  ;
         4C      A0         60   A2  D3 0002D          BITL     96(RQST), 76(R0)                                    ; 0179
                                 28  12 00032          BNEQ     2$                                                  ;
                 52              62  0F 00034          REMQUE   (RQST), RQST                                        ; 0187
                 50         04   AC  D0 00037          MOVL     OCD, R0                                             ; 0188
                            3A   A0  B7 0003B          DECW     58(R0)                                              ;
                          4004   8F  BB 0003E          PUSHR    #^M<R2,SP>                                          ; 0189
            0000G   CF           02  FB 00042          CALLS    #2, FORMAT_MESSAGE                                  ;
                                 52  DD 00047          PUSHL    RQST                                                ; 0190
            0000G   CF           01  FB 00049          CALLS    #1, SEND_REPLY                                      ;
                                 52  DD 0004E          PUSHL    RQST                                                ; 0191
            0000G   CF           01  FB 00050          CALLS    #1, LOG_MESSAGE                                     ;
                                 52  DD 00055          PUSHL    RQST                                                ; 0192
            0000G   CF           01  FB 00057          CALLS    #1, DEALLOCATE_RQCB                                 ;
                                 54  D7 0005C 2$:      DECL     RQST_COUNT                                          ; 0194
                                 BA  11 0005E          BRB      1$                                                  ; 0168
                                     04 00060 3$:      RET                                                          ; 0197

; Routine Size:  97 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 GLOBAL ROUTINE FIND_OPERATOR (RQCB, BLOCK) =
1
1 !++
1 ! Functional description:
1 !
1 !       This routine will scan through the list(s) of operators
1 !       known by OPCOM, and return the address of the operator
1 !       RQCB if it is found.
1 !
1 ! Input:
1 !
1 !       RQCB    : Address of an RQCB that describes the operator
1 !                         device that is being sought.
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       BLOCK   : Contains the address of a longword to receive
1 !                 the address of the known operator RQCB.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       If the operator is found, then the RQCB is provided
1 !       with a pointer to the OCD.
1 !
1 ! Routine value:
1 !
1 !       TRUE    : If the operator is known to OPCOM
1 !       FALSE   : If the operator is not known to OPCOM
1 !--
1
2 BEGIN                                         ! Start of FIND_OPERATOR
2
2 MAP
2       RQCB            : $ref_bblock;          ! RQCB data structure
2
2 EXTERNAL ROUTINE
2       IMPLICIT_DISABLE;                       ! Check for implicit disable
2
2 EXTERNAL LITERAL
2       MIN_SCOPE,                              ! Minimum scope value
2       MAX_SCOPE;                              ! Maximum scope value
2
2 EXTERNAL
2       OCD_VECTOR      : VECTOR;               ! Pointer to OCD structure
2
2 LOCAL
2       OCD             : $ref_bblock,          ! OCD data structure
2       OPER_RQCB       : $ref_bblock,          ! Operator RQCB structure
2       OCD_INDEX       : LONG;                 ! Index into OCD_VECTOR
```

```
	OCD_COUNT		: LONG,			! Count of OCDs in the OCD list
	OPER_COUNT		: LONG,			! Count of operators in OCD list
	FOUND			: LONG;			! Boolean loop control

!
! Scan through the list of all known operators,
! looking for a match on the device name.
! The scan is started on the lowest privileged
! operator class and proceeds to the highest.
!
.BLOCK = 0;						! Zero the output parameter
FOUND = FALSE;						! Assume not found
OCD_INDEX = MAX_SCOPE;					! Set higest (lowest privileged) scope value
WHILE (.OCD_INDEX GEQ MIN_SCOPE) AND (NOT .FOUND) DO
    BEGIN
    !
    ! Scan the OCD list for each class of operator.
    !
    OCD = .OCD_VECTOR [(.OCD_INDEX - 1)*2];		! Get OCD address
    OCD_COUNT = .OCD_VECTOR [(.OCD_INDEX - 1)*2+1];	! Get count of known operators of this scope
    WHILE (NOT .FOUND) AND (.OCD_COUNT GTR 0) DO
	BEGIN
	!
	! Scan the operator list for each OCD.
	!
	OPER_COUNT = .OCD [OCD_W_OPERCOUNT];		! Get the count of operators in the list
	OPER_RQCB  = .OCD [OCD_L_OPERFLINK];		! Get pointer to first operator in the list
	WHILE (.OPER_COUNT GTR 0) AND (NOT .FOUND) DO
	    BEGIN
	    !
	    ! Examine the device name for each operator in the list.
	    ! Compare the operator device names for equality.
	    ! Both device names are assumed to be in the DDCU format.
	    !
	    IF CH$EQL	(.OPER_RQCB [RQCB_L_OPER_LEN],
			 .OPER_RQCB [RQCB_L_OPER_PTR],
			 .RQCB [RQCB_L_OPER_LEN],
			 .RQCB [RQCB_L_OPER_PTR],
			 0
			)
	    THEN
		BEGIN
		FOUND = TRUE;				! The operator is known to OPCOM
		.BLOCK = .OPER_RQCB;			! Save the RQCB address
		RQCB [RQCB_L_OCD] = .OCD;		! Save the OCD address
		END
	    ELSE
		BEGIN
		OPER_RQCB = .OPER_RQCB [RQCB_L_FLINK];	! Get link to next operator RQCB
		OPER_COUNT = .OPER_COUNT - 1;		! Decrement operator count
		END;
	    END;
	OCD = .OCD [OCD_L_FLINK];			! Get address of next OCD
	OCD_COUNT = .OCD_COUNT - 1;			! Decrement OCD count
	END;
    OCD_INDEX = .OCD_INDEX - 1;				! Decrement OCD_INDEX
    END;
```

```
;   314        0312  2
;   315        0313  2  !
;   316        0314  2  ! If the operator was found, make sure
;   317        0315  2  ! it has not been implicitly disabled.
;   318        0316  2  !
;   319        0317  2  IF .FOUND
;   320        0318  2  THEN
;   321        0319  2      FOUND = NOT (IMPLICIT_DISABLE (.OPER_RQCB));
;   322        0320  2  RETURN (.FOUND);                                       ! Return status of search
;   323        0321  2
;   324        0322  1  END;                                                   ! End of FIND_OPERATOR



                                                            .EXTRN  IMPLICIT_DISABLE
                                                            .EXTRN  MIN_SCOPE, MAX_SCOPE
                                                            .EXTRN  OCD_VECTOR

                                     07FC 00000             .ENTRY  FIND_OPERATOR, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0198
                                                                    R9,R10                                        :
                              08   BC   D4 00002            CLRL    @BLOCK                                        ; 0265
                                   58   D4 00005            CLRL    FOUND                                         ; 0266
                      54 00000000G 8F   D0 00007            MOVL    #MAX_SCOPE, OCD_INDEX                         ; 0267
                      56        04 AC   D0 0000E            MOVL    RQCB, R6                                      ; 0291
           00000000G  8F           54   D1 00012  1$:       CMPL    OCD_INDEX, #MIN_SCOPE                         ; 0268
                                   54   19 00019            BLSS    7$                                            :
                      54           58   E8 0001B            BLBS    FOUND, 8$                                     :
           50         54           01   78 0001E            ASHL    #1, OCD_INDEX, R0                             ; 0273
                      55     0000GCF40  D0 00022            MOVL    OCD_VECTOR-8[R0], OCD                         :
                      5A     0000GCF40  D0 00028            MOVL    OCD_VECTOR-4[R0], OCD_COUNT                   ; 0274
                      3A           58   E8 0002E  2$:       BLBS    FOUND, 6$                                     ; 0275
                                   38   15 00031            BLEQ    6$                                            :
                      59        46 A5   3C 00033            MOVZWL  70(OCD), OPER_COUNT                           ; 0280
                      57        50 A5   D0 00037            MOVL    80(OCD), OPER_RQCB                            ; 0281
                                   59   D5 0003B  3$:       TSTL    OPER_COUNT                                    ; 0282
                                   25   15 0003D            BLEQ    5$                                            :
                      22           58   E8 0003F            BLBS    FOUND, 5$                                     :
   7C   A6     00     0080  D7    7C A7   2D 00042            CMPC5   124(OPER_RQCB), @128(OPER_RQCB), #0, -        ; 0289
                              0080 D6      0004B                    124(R6), @128(R6)                             :
                                   0D   12 0004E            BNEQ    4$                                            :
                      58           01   D0 00050            MOVL    #1, FOUND                                     ; 0297
                      08   BC      57   D0 00053            MOVL    OPER_RQCB, @BLOCK                             ; 0298
                      24   A6      55   D0 00057            MOVL    OCD, 36(R6)                                   ; 0299
                                   DE   11 0005B            BRB     3$                                            ; 0289
                      57           67   D0 0005D  4$:       MOVL    (OPER_RQCB), OPER_RQCB                        ; 0303
                                   59   D7 00060            DECL    OPER_COUNT                                    ; 0304
                                   D7   11 00062            BRB     3$                                            ; 0282
                      55           65   D0 00064  5$:       MOVL    (OCD), OCD                                    ; 0307
                                   5A   D7 00067            DECL    OCD_COUNT                                     ; 0308
                                   C3   11 00069            BRB     2$                                            ; 0275
                                   54   D7 0006B  6$:       DECL    OCD_INDEX                                     ; 0310
                                   A3   11 0006D            BRB     1$                                            ; 0268
                      0A           58   E9 0006F  7$:       BLBC    FOUND, 9$                                     ; 0317
                                   57   DD 00072  8$:       PUSHL   OPER_RQCB                                     ; 0319
                0000G CF           01   FB 00074            CALLS   #1, IMPLICIT_DISABLE                          :
                      58           50   D2 00079            MCOML   R0, FOUND                                     :
                      50           58   D0 0007C  9$:       MOVL    FOUND, R0                                     ; 0320
```

```
                                   04 0007F           RET                                              ; 0322

; Routine Size:  128 bytes,    Routine Base:  $CODE$ + 0061
```

```
1 GLOBAL ROUTINE IMPLICIT_DISABLE (OPER_RQCB) =
1
1 !++
1 ! Functional description:
1 !
1 !       This routine will determine if an operator device has
1 !       been implicitly disabled.  That is, if the operator device
1 !       is no longer marked as an operator.  The OPR bit is cleared
1 !       when the last channel to a non-allocated device has been
1 !       released, or when a device is deallocated.  The OPR bit will
1 !       be reset if the operator is a "permanent" operator.
1 !
1 ! Input:
1 !
1 !       OPER_RQCB       : Address of an operator RQCB
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       If the operator has been implicitly disabled, and is not
1 !       a permanent operator, then the operator will be disabled
1 !       without a disable message being sent to the operator.
1 !
1 ! Routine value:
1 !
1 !       TRUE    : If the operator is disabled
1 !       FALSE   : If the operator is still enabled
1 !--
1
2 BEGIN                                           ! Start of IMPLICIT_DISABLE
2
2 MAP
2       OPER_RQCB       : $ref_bblock;          ! Operator RQCB structure
2
2 EXTERNAL ROUTINE
2       CHECK_OPER_COVERAGE,                    ! Check coverage for requests
2       CLUSMSG_RQCB_SEND,                      ! Tell the cluster about something
2       DEALLOCATE_RQCB : NOVALUE,              ! Dispose of an RQCB
2       UPD_OPER_CONTEXT;                       ! Update an operator context
2
2 LOCAL
2       LOST_COVERAGE   : LONG,                 ! Boolean
2       DEV_CHAR        : $bblock [DIB$K_LENGTH],! Device characteristics buffer
2       CHAR_DESC       : $desc_block,          ! Dev. char. buffer descriptor
2       OCD             : $ref_bblock,          ! OCD data structure
2       DISABLED        : LONG,                 ! Boolean
```

```
2           ARG_LIST          : VECTOR [3];             ! Arguement list for EXE$SETOPR
2
2   !
2   ! Do not implicitly disable operators on other nodes
2   !
2   IF .GLOBAL_STATUS [GBLSTS_K_IN_VAXcluster]
2   THEN
2       IF NOT CLUSUTIL_SYSTEMID_EQUAL (OPER_RQCB [RQCB_T_SYSTEMID], LCL_NOD [NOD_T_NODE_SYSTEMID])
2       THEN
2           RETURN FALSE;                               ! Not disabled
2   !
2   ! If the operator has a disable in progress, then do not try another one.  This is just in case
2   ! any routine that we call tries to notify operators.
2   !
2   IF .OPER_RQCB [OPRSTS_V_IMPDISABLE]
2   THEN
2           RETURN TRUE;
2   !
2   ! Create a descriptor for the characteristics buffer and
2   ! get the operator device characteristics.
2   !
2   DISABLED = FALSE;                                   ! Assume operator not disabled
2   CHAR_DESC [0,0,32,0] = DIB$K_LENGTH;                ! Set buffer length
2   CHAR_DESC [DSC$A_POINTER] = DEV_CHAR;               ! Set buffer address
3   IF NOT ($GETDEV (DEVNAM=OPER_RQCB [RQCB_L_OPER_LEN], PRIBUF=CHAR_DESC))
2   THEN
2       DISABLED = TRUE;                                ! Device no longer exists
2   !
2   ! Check the OPR bit.  Reset it if this is a permanent operator.
2   !
3   IF NOT (.$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_OPR])
2   THEN
2       IF .$bblock [OPER_RQCB [RQCB_L_RQ_OPTIONS], OPC$V_PERMOPER]
2       THEN
3           BEGIN
3           !
3           ! Reset the OPR bit in the device UCB.
3           !
3           ARG_LIST [0] = 2;                           ! Set number of arguements
3           ARG_LIST [1] = OPER_RQCB [RQCB_L_OPER_LEN];
3           ARG_LIST [2] = ON;                          ! Set bit state
4           IF NOT $CMKRNL (ROUTIN=EXE$SETOPR, ARGLST=ARG_LIST)
3           THEN
3               DISABLED = TRUE;
3           END
2       ELSE
2           DISABLED = TRUE;
2   !
2   ! If the operator is disabled, then remove it from the operator list.
2   ! Do not notify the operator of the disable.  After doing the disable,
2   ! check to see if any requests have lost operator coverage.
2   !
2   IF .DISABLED
2   THEN
3       BEGIN
3       !
3       ! The rqcb is tainted, let everybody know
```

```
0437 3       !
0438 3       OPER_RQCB [OPRSTS_V_IMPDISABLE] = TRUE;
0439 3       !
0440 3       ! Tell the cluster to disable this operator
0441 3       !
0442 3       CLUSMSG_RQCB_SEND (-1, CLM__IMP_DISABLE, .OPER_RQCB);
0443 3       LOST_COVERAGE = UPD_OPER_CONTEXT    (TRUE,                   ! Do the disable
0444 3                                            .OPER_RQCB [RQCB_L_ATTNMASK1],
0445 3                                            .OPER_RQCB [RQCB_L_ATTNMASK2],
0446 3                                            .OPER_RQCB
0447 3                                           );
0448 3       REMQUE (.OPER_RQCB, OPER_RQCB);             ! Remove RQCB from operator list
0449 3       OCD = .OPER_RQCB [RQCB_L_OCD];              ! Get OCD address
0450 3       OCD [OCD_W_OPERCOUNT] = .OCD [OCD_W_OPERCOUNT] - 1;
0451 3       DEALLOCATE_RQCB (.OPER_RQCB);               ! Dispose of the RQCB
0452 3       !
0453 3       ! If operator coverage was lost due to the disable, check all
0454 3       ! outstanding reuquests queued to this OCD for operator coverage.
0455 3       ! All requests that no longer have operator coverage will be canceled.
0456 3       !
0457 3       IF .LOST_COVERAGE
0458 3       THEN
0459 3           CHECK_OPER_COVERAGE (.OCD);
0460 2       END;
0461 2
0462 2   RETURN (.DISABLED);                             ! Return the routine value
0463 2
0464 1   END;                                            ! End of IMPLICIT_DISABLE


                                                        .EXTRN  CLUSMSG_RQCB_SEND
                                                        .EXTRN  UPD_OPER_CONTEXT
                                                        .EXTRN  SYS$GETDEV, EXE$SETOPR
                                                        .EXTRN  SYS$CMKRNL

                                  001C 00000            .ENTRY  IMPLICIT_DISABLE, Save R2,R3,R4                  ; 0323
                  5E     FF78  CE  9E 00002             MOVAB   -136(SP), SP                                     ;
                  18    0000G  CF  E9 00007             BLBC    GLOBAL_STATUS+1, 1$                              ; 0385
  51      0000G   CF 00000050  8F  C1 0000C             ADDL3   #80, LCL_NOD, R1                                 ; 0387
  50         04   AC           1C  C1 00016             ADDL3   #28, OPER_RQCB, R0                               ;
                            0000G  30 0001B             BSBW    CLUSUTIL_SYSTEMID_EQUAL                          ;
                  03           50  E8 0001E             BLBS    R0, 1$                                           ;
                            00A2   31 00021             BRW     7$                                               ;
                  52      04   AC  D0 00024  1$:        MOVL    OPER_RQCB, R2                                    ; 0394
  04         78   A2           03  E1 00028             BBC     #3, 120(R2), 2$                                  ;
                  50           01  D0 0002D             MOVL    #1, R0                                           ; 0396
                               04     00030             RET                                                      ;
                               53  D4 00031  2$:        CLRL    DISABLED                                         ; 0401
             0C   AE      74   8F  9A 00033             MOVZBL  #116, CHAR_DESC                                  ; 0402
             10   AE      14   AE  9E 00038             MOVAB   DEV_CHAR, CHAR_DESC+4                            ; 0403
                               7E  7C 0003D             CLRQ    -(SP)                                            ; 0404
                          14   AE  9F 0003F             PUSHAB  CHAR_DESC                                        ;
                               7E  D4 00042             CLRL    -(SP)                                            ;
                          7C   A2  9F 00044             PUSHAB  124(R2)                                          ;
      00000000G   00           05  FB 00047             CALLS   #5, SYS$GETDEV                                   ;
                  03           50  E8 0004E             BLBS    R0, 3$                                           ;
```

```
                    53                 01  D0 00051          MOVL    #1, DISABLED                         ; 0406
                              14       AE  95 00054 3$:      TSTB    DEV_CHAR                             ; 0410
                                       26  19 00057          BLSS    5$                                   ;
          1E        58    A2           01  E1 00059          BBC     #1, 88(R2), 4$                       ; 0412
                          6E           02  D0 0005E          MOVL    #2, ARG_LIST                         ; 0418
                    04    AE      7C   A2  9E 00061          MOVAB   124(R2), ARG_LIST+4                  ; 0419
                    08    AE           01  D0 00066          MOVL    #1, ARG_LIST+8                       ; 0420
                                       5E  DD 0006A          PUSHL   SP                                   ; 0421
                          00000000G    00  9F 0006C          PUSHAB  EXE$SETOPR                           ;
             00000000G    00           02  FB 00072          CALLS   #2, SYS$CMKRNL                       ;
                          03           50  E8 00079          BLBS    R0, 5$                               ;
                          53           01  D0 0007C 4$:      MOVL    #1, DISABLED                         ; 0426
                          40           53  E9 0007F 5$:      BLBC    DISABLED, 6$                         ; 0432
                    78    A2           08  88 00082          BISB2   #8, 120(R2)                          ; 0438
                                       52  DD 00086          PUSHL   R2                                   ; 0442
                                       0A  DD 00088          PUSHL   #10                                  ;
                          7E           01  CE 0008A          MNEGL   #1, -(SP)                            ;
                 0000G    CF           03  FB 0008D          CALLS   #3, CLUSMSG_RQCB_SEND                ;
                                       52  DD 00092          PUSHL   R2                                   ; 0446
                          7E      5C   A2  7D 00094          MOVQ    92(R2), -(SP)                        ; 0444
                                       01  DD 00098          PUSHL   #1                                   ; 0443
                 0000G    CF           04  FB 0009A          CALLS   #4, UPD_OPER_CONTEXT                 ;
                          54           50  D0 0009F          MOVL    R0, LOST_COVERAGE                    ;
                    04    AC           62  0F 000A2          REMQUE  (R2), OPER_RQCB                      ; 0448
                          50      04   AC  D0 000A6          MOVL    OPER_RQCB, R0                        ; 0449
                          52      24   A0  D0 000AA          MOVL    36(R0), OCD                          ;
                                  46   A2  B7 000AE          DECW    70(OCD)                              ; 0450
                                       50  DD 000B1          PUSHL   R0                                   ; 0451
                 0000G    CF           01  FB 000B3          CALLS   #1, DEALLOCATE_RQCB                  ;
                          07           54  E9 000B8          BLBC    LOST_COVERAGE, 6$                    ; 0457
                                       52  DD 000BB          PUSHL   OCD                                  ; 0459
                 0000G    CF           01  FB 000BD          CALLS   #1, CHECK_OPER_COVERAGE              ;
                          50           53  D0 000C2 6$:      MOVL    DISABLED, R0                         ; 0462
                                           04 000C5          RET                                          ;
                                       50  D4 000C6 7$:      CLRL    R0                                   ; 0464
                                           04 000C8          RET                                          ;

; Routine Size:  201 bytes,    Routine Base:  $CODE$ + 00E1
```

```
1 GLOBAL ROUTINE NOTIFY_LISTED_OPERATORS (RQST_RQCB) =
1
1 !++
1 ! Functional description:
1 !
1 !       This routine will traverse a list of operators
1 !       (pointed to by the OCD pointed to by the request RQCB)
1 !       and send the message associated with the RQCB to all
1 !       operators who are enabled to receive the message.
1 !
1 ! Input:
1 !
1 !       RQST_RQCB       : Address of a request RQCB
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       The message will be sent to the interested operators.
1 !
1 ! Side effects:
1 !
1 !       As part of sending the message, the operators are checked
1 !       to see if they have been implicitly disabled.  If so, they
1 !       are removed from the operator list.
1 !
1 ! Routine value:
1 !
1 !       TRUE    : If at least one operator was notified.
1 !       FALSE   : If no operators were notified.
1 !--
1
2 BEGIN                                                 ! Start of NOTIFY_LISTED_OPERATORS
2
2 MAP
2       RQST_RQCB       : $ref_bblock;                  ! Request RQCB structure
2
2 EXTERNAL ROUTINE
2       IMPLICIT_DISABLE,                               ! Check for implicit disable
2       NOTIFY_OPERATOR;                                ! Send a message to a giiven operator
2
2 LOCAL
2       OCD             : $ref_bblock,                  ! OCD data structure
2       SAVED_MCB       : LONG,                         ! Address of an MCB
2       OPER_COUNT      : LONG,                         ! Count of operators on list
2       CURRENT_OPER    : $ref_bblock,                  ! Current operator RQCB
2       NEXT_OPER       : $ref_bblock,                  ! Next operator RQCB
2       NOTIFIED        : LONG;                         ! Boolean
2
2 NOTIFIED = FALSE;                                     ! Assume no operator notified
2 !
```

```
2 ! Check the request to see if NOBRD is specified.  If it is, and the requestor
2 ! has the proper privileges, return failure without sending the message.
2 !
2 IF .$bblock [RQST_RQCB [RQCB_L_OPTIONS], OPC$V_NOBRD]
2 THEN
3     IF (.$bblock [RQST_RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER])
4     OR ((.RQST_RQCB [RQCB_B_SCOPE] EQL OPC$K_GROUP) AND ((.$bblock [RQST_RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER
3                                                          (.$bblock [RQST_RQCB [RQCB_L_PRIVMASK1], PRV$V_GROU
2 THEN
2     RETURN (FALSE);
2 OCD = .RQST_RQCB [RQCB_L_OCD];                    ! Get OCD address
2 IF .OCD EQL 0
2 THEN
3     BEGIN
3     LOCAL
3         DESC : VECTOR [2, LONG];
3     DESC [0] = RQCB_K_SIZE;
3     DESC [1] = .RQST_RQCB;
3     DUMP_LOG_FILE (DESC, %ASCID 'OCD address is zero in RQCB');
3     RETURN (FALSE);
2     END;
2 OPER_COUNT = .OCD [OCD_W_OPERCOUNT];              ! Get count of operators
2 NEXT_OPER = .OCD [OCD_L_OPERFLINK];               ! Get address of next operator in list
2 WHILE (.OPER_COUNT GTR 0) DO
3     BEGIN
3     !
3     ! Link to the next operator RQCB.  We have to keep the address
3     ! of the next operator RQCB in case this one evaporates as a
3     ! side effect of IMPLICT_DISABLE.
3     !
3     CURRENT_OPER = .NEXT_OPER;
3     NEXT_OPER = .CURRENT_OPER [RQCB_L_FLINK];
3     !
3     ! Check the request attention mask against the operator's
3     ! enable mask.  If an bits in common, then notify the operator.
3     ! The message is also sent if a special status bit is set.
3     ! This is an internal hack used to force message output.
3     !
4     IF ((.RQST_RQCB [RQCB_L_ATTNMASK1] AND .CURRENT_OPER [RQCB_L_ATTNMASK1]) NEQ 0)
4     OR ((.RQST_RQCB [RQCB_L_ATTNMASK2] AND .CURRENT_OPER [RQCB_L_ATTNMASK2]) NEQ 0)
4     OR (.RQST_RQCB [HDR_V_BRD])
3     THEN
4         IF NOT (IMPLICIT_DISABLE (.CURRENT_OPER))
3         THEN
4             BEGIN
4             !
4             ! Send the message to the operator.  The MCB from the RQST_RQCB is
4             ! reused to avoid the overhead of creating a new MCB for each operator.
4             !
4             SAVED_MCB = .CURRENT_OPER [RQCB_L_MCB];
4             CURRENT_OPER [RQCB_L_MCB] = .RQST_RQCB [RQCB_L_MCB];
4             IF NOTIFY_OPERATOR (.CURRENT_OPER)
4             THEN
4                 NOTIFIED = TRUE;              ! An operator was notified
4             CURRENT_OPER [RQCB_L_MCB] = .SAVED_MCB;
3             END;
3     OPER_COUNT = .OPER_COUNT - 1;                 ! Decrement the operator count
```

```
;  583            0579  2        END;
;  584            0580  2
;  585            0581  2  RETURN (.NOTIFIED);                                     ! Return routine value
;  586            0582  2
;  587            0583  1  END;                                                    ! End of NOTIFY_LISTED_OPERATORS


                                                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

20  73  69  20  73  73  65  72  64  64  61  20  44  43  4F  00000 P.AAB:   .ASCII  \OCD address is zero in RQCB\<0>              ;
    00  42  43  51  52  20  6E  69  20  6F  72  65  7A  0000F                                                                      ;
                                            010E001B  0001C P.AAA:   .LONG   17694747                                             ;
                                            00000000'  00020          .ADDRESS P.AAB                                               ;

                                                                     .EXTRN  NOTIFY_OPERATOR

                                                                     .PSECT  $CODE$,NOWRT,2

                                              00FC 00000             .ENTRY  NOTIFY_LISTED_OPERATORS, Save R2,R3,R4,R5,-  ; 0465
                                                                             R6,R7
                      5E           08  C2 00002             SUBL2   #8, SP                                               ;
                      57               D4 00005             CLRL    NOTIFIED                                             ; 0520
                      53     04    AC  D0 00007             MOVL    RQST_RQCB, R3                                        ; 0525
          14    54    A3           01  E1 0000B             BBC     #1, 84(R3), 1$                                       ;
          7C    32    A3           02  E0 00010             BBS     #2, 50(R3), 8$                                       ; 0527
                      02     53    A3  91 00015             CMPB    83(R3), #2                                           ; 0528
                      09               12 00019             BNEQ    1$                                                   ;
          71    32    A3           02  E0 0001B             BBS     #2, 50(R3), 8$                                       ;
                      6D     31    A3  E8 00020             BLBS    49(R3), 8$                                           ; 0529
                      52     24    A3  D0 00024 1$:         MOVL    36(R3), OCD                                          ; 0532
                      16               12 00028             BNEQ    2$                                                   ; 0533
                      6E     94    8F  9A 0002A             MOVZBL  #148, DESC                                           ; 0538
                04    AE           53  D0 0002E             MOVL    R3, DESC+4                                           ; 0539
                           0000'   CF  9F 00032             PUSHAB  P.AAA                                                ; 0540
                             04    AE  9F 00036             PUSHAB  DESC                                                 ;
              0000G   CF           02  FB 00039             CALLS   #2, DUMP_LOG_FILE                                    ;
                      51               11 0003E             BRB     8$                                                   ; 0541
                      55     46    A2  3C 00040 2$:         MOVZWL  70(OCD), OPER_COUNT                                  ; 0543
                      54     50    A2  D0 00044             MOVL    80(OCD), NEXT_OPER                                   ; 0544
                                   55  D5 00048 3$:         TSTL    OPER_COUNT                                           ; 0545
                                   41  15 0004A             BLEQ    7$                                                   ;
                      52           54  D0 0004C             MOVL    NEXT_OPER, CURRENT_OPER                              ; 0552
                      54           62  D0 0004F             MOVL    (CURRENT_OPER), NEXT_OPER                            ; 0553
                5C    A2     5C    A3  D3 00052             BITL    92(R3), 92(CURRENT_OPER)                             ; 0560
                                   0C  12 00057             BNEQ    4$                                                   ;
                60    A2     60    A3  D3 00059             BITL    96(R3), 96(CURRENT_OPER)                             ; 0561
                                   05  12 0005E             BNEQ    4$                                                   ;
          24    28    A3           01  E1 00060             BBC     #1, 40(R3), 6$                                       ; 0562
                                   52  DD 00065 4$:         PUSHL   CURRENT_OPER                                         ; 0564
              0000G   CF           01  FB 00067             CALLS   #1, IMPLICIT_DISABLE                                 ;
                      1A           50  E8 0006C             BLBS    R0, 6$                                               ;
                      56     6C    A2  D0 0006F             MOVL    108(CURRENT_OPER), SAVED_MCB                         ; 0571
                6C    A2     6C    A3  D0 00073             MOVL    108(R3), 108(CURRENT_OPER)                           ; 0572
                                   52  DD 00078             PUSHL   CURRENT_OPER                                         ; 0573
              0000G   CF           01  FB 0007A             CALLS   #1, NOTIFY_OPERATOR                                  ;
                      03           50  E9 0007F             BLBC    R0, 5$                                               ;
```

```
                   57          01  D0 00082         MOVL    #1, NOTIFIED                                   ; 0575
          6C       A2          56  D0 00085 5$:     MOVL    SAVED_MCB, 108(CURRENT_OPER)                   ; 0576
                               55  D7 00089 6$:     DECL    OPER_COUNT                                     ; 0578
                               BB  11 0008B         BRB     3$                                             ; 0545
                   50          57  D0 0008D 7$:     MOVL    NOTIFIED, R0                                   ; 0581
                                   04 00090         RET                                                    ;
                               50  D4 00091 8$:     CLRL    R0                                             ; 0583
                                   04 00093         RET                                                    ;

; Routine Size:  148 bytes,    Routine Base:  $CODE$ + 01AA
```

```
1 GLOBAL ROUTINE NOTIFY_OPERATOR (RQCB) =
1
1 !++
1 ! Functional description:
1 !
1 !       This routine will send a message to an operator,
1 !       be it a terminal or a mailbox.
1 !
1 ! Input:
1 !
1 !       RQCB            : Address of an operator RQCB
1 !
1 ! Implicit Input:
1 !
1 !       The RQCB points to an MCB that describes the message.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       A message is sent to the operator.
1 !
1 ! Side effects:
1 !
1 !       If the operator device is a mailbox, the message
1 !       may be truncated if the mailbox buffer size is not
1 !       large enough to hold the entire message.
1 !
1 ! Routine value:
1 !
1 !       TRUE            : If success
1 !       <anything else> : If the message could not be sent
1 !--
2 BEGIN                                          ! Start of NOTIFY_OPERATOR
2
2 MAP
2       RQCB            : $ref_bblock;           ! Operator RQCB structure
2
2 LOCAL
2       OCD             : $ref_bblock,           ! OCD data structure
2       MSG_SIZE        : WORD,                  ! Size of message to operator
2       MBX_CHANNEL     : WORD,                  ! Channel to operator mailbox
2       IOSB            : $bblock [8],           ! I/O status block
2       MCB             : $ref_bblock,           ! MCB data structure
2       STATUS          : LONG;
2
2 !
2 ! If there is no MCB connected to the RQCB, then return an error status.
2 !
2 MCB = .RQCB [RQCB_L_MCB];
2 IF .MCB EQL 0
2 THEN
2     RETURN (FALSE);
2 !
2 ! Check the request to see if NOBRD is specified.  If it is, and the requestor
```

```
  2 ! has the proper privileges, return failure without sending the message.
  2 !
  2 IF .$bblock [RQCB [RQCB_L_OPTIONS], OPC$V_NOBRD]
  2 THEN
  3     IF (.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER])
  4     OR ((.RQCB [RQCB_B_SCOPE] EQL OPC$K_GROUP) AND ((.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER]) OR
  3                                                    (.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_GROUP])))
  2 THEN
  2     RETURN (FALSE);
  2 !
  2 ! If the operator is on another node, then pretend that we notified the operator.  OPCOMRQST uses the
  2 ! value to determine if a request can be fielded.
  2 !
  2 IF .GLOBAL_STATUS [GBLSTS_K_IN_VAXcluster]
  2 THEN
  2     IF NOT CLUSUTIL_SYSTEMID_EQUAL (RQCB [RQCB_T_SYSTEMID], LCL_NOD [NOD_T_NODE_SYSTEMID])
  2     THEN
  2         RETURN (TRUE);
  2 !
  2 ! Send the message.  How it is sent depends on the device type.
  2 !
  2 IF .RQCB [OPRSTS_V_TRM]
  2 OR .RQCB [OPRSTS_V_REMTRM]
  2 THEN
  3     BEGIN
  3     !
  3     ! The operator device is a terminal or remote terminal.
  3     ! Send the message via $BRKTHRU
  3     !
  3     REPLYBRD_BRKTHRU_QUEUE (
  3                         MCB [MCB_L_TEXTLEN],      ! Message to send
  3                         RQCB [RQCB_L_OPER_LEN],   ! Target (operator device name)
  3                         BRK$C_DEVICE,             ! Type of target
  3                         32,                       ! Carriage control
  3                         0,                        ! Flags
  3                         BRK$C_OPCOM,              ! Type of requestor
  3                         0,0,0,0,0);               ! No completion routine or arguments
  3     RETURN 1;
  3     END
  2 ELSE
  2     IF .RQCB [OPRSTS_V_MBX]
  2     THEN
  3         BEGIN
  3         !
  3         ! The operator device is a mailbox.
  3         ! Send the message via $QIO.  If the mailbox is
  3         ! too small, truncate the message to fit.
  3         !
  3         MSG_SIZE = .MCB [MCB_L_TEXTLEN];                ! Assume mailbox big enough
  3         IF .MSG_SIZE GTR .RQCB [RQCB_W_MBXSIZE]         ! Is message to big?
  3         THEN
  3             MSG_SIZE = .RQCB [RQCB_W_MBXSIZE];          ! Yes, truncate message
  3
P 4         IF NOT (STATUS = $ASSIGN (CHAN = MBX_CHANNEL,   ! Assign a channel to the  operator device
P 4                                   DEVNAM = RQCB [RQCB_L_OPER_LEN]
  4                                  ))
  3         THEN
```

```
;  703          0698  3                 RETURN (.STATUS);
;  704          0699  3
;  705        P 0700  4             IF (STATUS = $QIOW (FUNC = (IO$_WRITEVBLK OR IO$M_NOW), ! Send the message
;  706        P 0701  4                                 CHAN = .MBX_CHANNEL,
;  707        P 0702  4                                 IOSB = IOSB,
;  708        P 0703  4                                 P1   = .MCB [MCB_L_TEXTPTR],
;  709        P 0704  4                                 P2   = .MSG_SIZE
;  710          0705  4                                ))
;  711          0706  3             THEN
;  712          0707  3                 STATUS = .IOSB [0,0,16,0];                  ! Get actual I/O operation status
;  713          0708  3
;  714          0709  3             $DASSGN (CHAN = .MBX_CHANNEL);                  ! Deassign channel to operator device
;  715          0710  3             RETURN (.STATUS);                               ! Return the appropriate status
;  716          0711  3             END;
;  717          0712  2     !
;  718          0713  2     ! If we get this far, it means that the device is not a
;  719          0714  2     ! legal operator device, and that the message cannot be sent.
;  720          0715  2     ! Return an error status.
;  721          0716  2     !
;  722          0717  2     RETURN (FALSE);
;  723          0718  2
;  724          0719  1     END;                                                    ! End of NOTIFY_OPERATOR


                                                                        .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                                        .EXTRN  SYS$DASSGN

                                              001C 00000                .ENTRY  NOTIFY_OPERATOR, Save R2,R3,R4              ; 0584
                         5E            0C      C2 00002                SUBL2   #12, SP                                     ;
                         52       04   AC      D0 00005                MOVL    RQCB, R2                                    ; 0635
                         53       6C   A2      D0 00009                MOVL    108(R2), MCB                                ;
                                       03      12 0000D                BNEQ    2$                                          ; 0636
                                     00B6      31 0000F  1$:           BRW     11$                                         ;
              14       54   A2         01      E1 00012  2$:           BBC     #1, 84(R2), 3$                              ; 0643
              F3       32   A2         02      E0 00017                BBS     #2, 50(R2), 1$                              ; 0645
                            02    53   A2      91 0001C                CMPB    83(R2), #2                                  ; 0646
                                       09      12 00020                BNEQ    3$                                          ;
              E8       32   A2         02      E0 00022                BBS     #2, 50(R2), 1$                              ;
                            E4    31   A2      E8 00027                BLBS    49(R2), 1$                                  ; 0647
                            14  0000G  CF      E9 0002B  3$:           BLBC    GLOBAL_STATUS+1, 4$                         ; 0654
              51     0000G  CF 00000050  8F    C1 00030                ADDL3   #80, LCL_NOD, R1                            ; 0656
                            50    1C   A2      9E 0003A                MOVAB   28(R2), R0                                  ;
                                    0000G      30 0003E                BSBW    CLUSUTIL_SYSTEMID_EQUAL                     ;
                            20         50      E9 00041                BLBC    R0, 6$                                      ;
                            05    78   A2      E8 00044  4$:           BLBS    120(R2), 5$                                 ; 0662
              1B       78   A2         01      E1 00048                BBC     #1, 120(R2), 7$                             ; 0663
                                       7E      7C 0004D  5$:           CLRQ    -(SP)                                       ; 0672
                                       7E      7C 0004F                CLRQ    -(SP)                                       ;
                            7E         07      7D 00051                MOVQ    #7, -(SP)                                   ;
                            7E         20      7D 00054                MOVQ    #32, -(SP)                                  ;
                                       01      DD 00057                PUSHL   #1                                          ;
                                  7C   A2      9F 00059                PUSHAB  124(R2)                                     ;
                                  30   A3      9F 0005C                PUSHAB  48(MCB)                                     ; 0671
                     0000G  CF         0B      FB 0005F                CALLS   #11, REPLYBRD_BRKTHRU_QUEUE                 ; 0672
                            50         01      D0 00064  6$:           MOVL    #1, R0                                      ; 0678
                                               04 00067                RET                                                 ;
```

```
            5B        78  A2        02  E1 00068 7$:    BBC     #2, 120(R2), 11$                         ; 0681
                          54    30  A3  B0 0006D        MOVW    48(MCB), MSG_SIZE                        ; 0689
                    7A    A2        54  B1 00071        CMPW    MSG_SIZE, 122(R2)                        ; 0690
                                    04  1B 00075        BLEQU   8$                                       ;
                          54    7A  A2  B0 00077        MOVW    122(R2), MSG_SIZE                        ; 0692
                                    7E  7C 0007B 8$:    CLRQ    -(SP)                                    ; 0696
                                08  AE  9F 0007D        PUSHAB  MBX_CHANNEL                              ;
                                7C  A2  9F 00080        PUSHAB  124(R2)                                  ;
                00000000G  00       04  FB 00083        CALLS   #4, SYS$ASSIGN                           ;
                           52       50  D0 0008A        MOVL    R0, STATUS                               ;
                           34       52  E9 0008D        BLBC    STATUS, 10$                              ;
                                    7E  7C 00090        CLRQ    -(SP)                                    ; 0705
                                    7E  7C 00092        CLRQ    -(SP)                                    ;
                           7E       54  3C 00094        MOVZWL  MSG_SIZE, -(SP)                          ;
                                34  A3  DD 00097        PUSHL   52(MCB)                                  ;
                                    7E  7C 0009A        CLRQ    -(SP)                                    ;
                                24  AE  9F 0009C        PUSHAB  IOSB                                     ;
                           7E   70  8F  9A 0009F        MOVZBL  #112, -(SP)                              ;
                           7E   28  AE  3C 000A3        MOVZWL  MBX_CHANNEL, -(SP)                       ;
                                    7E  D4 000A7        CLRL    -(SP)                                    ;
                00000000G  00       0C  FB 000A9        CALLS   #12, SYS$QIOW                            ;
                           52       50  D0 000B0        MOVL    R0, STATUS                               ;
                           04       52  E9 000B3        BLBC    STATUS, 9$                               ;
                           52   04  AE  3C 000B6        MOVZWL  IOSB, STATUS                             ; 0707
                           7E       6E  3C 000BA 9$:    MOVZWL  MBX_CHANNEL, -(SP)                       ; 0709
                00000000G  00       01  FB 000BD        CALLS   #1, SYS$DASSGN                           ;
                           50       52  D0 000C4 10$:   MOVL    STATUS, R0                               ; 0710
                                        04 000C7        RET                                              ;
                                    50  D4 000C8 11$:   CLRL    R0                                       ; 0719
                                        04 000CA        RET                                              ;

; Routine Size:  203 bytes,    Routine Base:  $CODE$ + 023E
```

```
GLOBAL ROUTINE OPERUTIL_CLM_IMP_DISABLE (BUFFER_DESC : $ref_bblock, CLM : $ref_bblock, LEN) : NOVALUE =

!++
! Functional description:
!
!       This routine processes an implicit disable request from another node.
!
! Input:
!
!       BUFFER_DESC -   pointer to message from remote node, including $SNDOPR header
!       CLM -           pointer to CLMRQCB structure
!       LEN -           length of LEN
!
! Implicit Input:
!
!       None.
!
! Output:
!
!       None.
!
! Implict output:
!
!       None.
!
! Side effects:
!
!       If the operator has been implicitly disabled, and is not
!       a permanent operator, then the operator will be disabled
!       without a disable message being sent to the operator.
!
! Routine value:
!
!       TRUE    : If the operator is disabled
!       FALSE   : If the operator is still enabled
!--

BEGIN                                           ! Start of OPERUTIL_CLM_IMP_DISABLE

EXTERNAL ROUTINE
        CLUSMSG_CONV_CLM_RQCB,                  ! Convert message to RQCB
        CHECK_OPER_COVERAGE,                    ! Check coverage for requests
        DEALLOCATE_RQCB : NOVALUE,              ! Dispose of an RQCB
        DUMP_LOG_FILE,                          ! Place random string in log
        UPD_OPER_CONTEXT;                       ! Update an operator context

LOCAL
        FOUND           : LONG,                 ! Found the operator
        LOST_COVERAGE   : LONG,                 ! Boolean
        DEV_CHAR        : $bblock [DIB$K_LENGTH],! Device characteristics buffer
        CHAR_DESC       : $desc_block,          ! Dev. char. buffer descriptor
        OCD             : $ref_bblock,          ! OCD data structure
        OPER_RQCB       : $ref_bblock,          ! RQCB data structure
        RQCB            : $ref_bblock,          ! RQCB data structure
        DISABLED        : LONG,                 ! Boolean
        ARG_LIST        : VECTOR [3];           ! Arguement list for EXE$SETOPR
```

```
2 !
2 ! Check the version number of the message.  If the message is from any other version,
2 ! simply ignore it.
2 !
2 IF .CLM [CLM_B_DS_VERSION] NEQ CLMRQCB_K_DS_VERSION
2 THEN
2     RETURN DUMP_LOG_FILE (.BUFFER_DESC, %ASCID 'CLM__OPRENABLE mismatch');
2 !
2 ! Allocate an RQCB and convert the message RQCB into the new RQCB
2 !
2 IF NOT CLUSMSG_CONV_CLM_RQCB (.CLM, RQCB)
2 THEN
2     RETURN DUMP_LOG_FILE (.BUFFER_DESC, ascid_INVALIDRQCB);
2 !
2 ! See if the operator is already known to OPCOM.  This entails scanning down the appropriate operator
2 ! list and comparing the device names for equality.  FIND_OPERATOR will set RQCB [RQCB_L_OCD] if it
2 ! finds a match.
2 !
2 FOUND = FIND_OPERATOR (.RQCB, OPER_RQCB);
2 DEALLOCATE_RQCB (.RQCB);                            ! Don't need this any more
2 !
2 ! The operator has been disabled on the remote node, remove it from the operator list.
2 ! Do not notify anyone of the disable.  After doing the disable,
2 ! check to see if any requests have lost operator coverage.
2 !
2 IF .FOUND
2 THEN
3     BEGIN
3     LOST_COVERAGE = UPD_OPER_CONTEXT    (TRUE,                   ! Do the disable
3                                          .OPER_RQCB [RQCB_L_ATTNMASK1],
3                                          .OPER_RQCB [RQCB_L_ATTNMASK2],
3                                          .OPER_RQCB
3                                         );
3     REMQUE (.OPER_RQCB, OPER_RQCB);                 ! Remove RQCB from operator list
3     OCD = .OPER_RQCB [RQCB_L_OCD];                  ! Get OCD address
3     OCD [OCD_W_OPERCOUNT] = .OCD [OCD_W_OPERCOUNT] - 1;
3     DEALLOCATE_RQCB (.OPER_RQCB);                   ! Dispose of the RQCB
3     !
3     ! If operator coverage was lost due to the disable, check all
3     ! outstanding reuquests queued to this OCD for operator coverage.
3     ! All requests that no longer have operator coverage will be canceled.
3     !
3     IF .LOST_COVERAGE
3     THEN
3         CHECK_OPER_COVERAGE (.OCD);
2     END;
2
2 RETURN;                                             ! Return the routine value
2
1 END;                                                ! End of OPERUTIL_CLM_IMP_DISABLE


                                                      .PSECT  $PLIT$,NOWRT,NOEXE,2

20 45 4C 42 41 4E 45 52 50 4F 5F 5F 4D 4C 43 00024 P.AAD:  .ASCII  \CLM__OPRENABLE mismatch\<0>          ;
                  00 68 63 74 61 6D 73 69 6D 00033                                                      ;
```

```
                          010E0017  0003C P.AAC:   .LONG    17694743                                        ;
                          00000000' 00040          .ADDRESS P.AAD                                           ;

                                                   .EXTRN   CLUSMSG_CONV_CLM_RQCB
                                                   .EXTRN   ASCID_INVALIDRQCB

                                                   .PSECT   $CODE$,NOWRT,2

                             000C 00000            .ENTRY   OPERUTIL_CLM_IMP_DISABLE, Save R2,R3     ; 0720
           5E      FF70   CE   9E 00002            MOVAB    -144(SP), SP                               ;
           52        08   AC   D0 00007            MOVL     CLM, R2                                    ; 0781
           02        02   A2   91 0000B            CMPB     2(R2), #2                                  ;
                          06   13 0000F            BEQL     1$                                         ;
                  0000'   CF   9F 00011            PUSHAB   P.AAC                                      ; 0783
                          10   11 00015            BRB      2$                                         ;
                   4004   8F   BB 00017 1$:        PUSHR    #^M<R2,SP>                                 ; 0787
     0000G CF             02   FB 0001B            CALLS    #2, CLUSMSG_CONV_CLM_RQCB                  ;
           0D             50   E8 00020            BLBS     R0, 3$                                     ;
                  0000G   CF   9F 00023            PUSHAB   ASCID_INVALIDRQCB                          ; 0789
                     04   AC   DD 00027 2$:        PUSHL    BUFFER_DESC                                ;
     0000G CF             02   FB 0002A            CALLS    #2, DUMP_LOG_FILE                          ;
                               04 0002F            RET                                                 ;
                     04   AE   9F 00030 3$:        PUSHAB   OPER_RQCB                                  ; 0795
                     04   AE   DD 00033            PUSHL    RQCB                                       ;
      FD1D CF             02   FB 00036            CALLS    #2, FIND_OPERATOR                          ;
           52             50   D0 0003B            MOVL     R0, FOUND                                  ;
                          6E   DD 0003E            PUSHL    RQCB                                       ; 0796
     0000G CF             01   FB 00040            CALLS    #1, DEALLOCATE_RQCB                        ;
           34             52   E9 00045            BLBC     FOUND, 4$                                  ; 0802
           52        04   AE   D0 00048            MOVL     OPER_RQCB, R2                              ; 0808
                          52   DD 0004C            PUSHL    R2                                         ;
           7E        5C   A2   7D 0004E            MOVQ     92(R2), -(SP)                              ; 0806
                          01   DD 00052            PUSHL    #1                                         ; 0805
     0000G CF             04   FB 00054            CALLS    #4, UPD_OPER_CONTEXT                       ;
           53             50   D0 00059            MOVL     R0, LOST_COVERAGE                          ;
        04 AE             62   0F 0005C            REMQUE   (R2), OPER_RQCB                            ; 0810
           50        04   AE   D0 00060            MOVL     OPER_RQCB, R0                              ; 0811
           52        24   A0   D0 00064            MOVL     36(R0), OCD                                ;
                     46   A2   B7 00068            DECW     70(OCD)                                    ; 0812
                          50   DD 0006B            PUSHL    R0                                         ; 0813
     0000G CF             01   FB 0006D            CALLS    #1, DEALLOCATE_RQCB                        ;
           07             53   E9 00072            BLBC     LOST_COVERAGE, 4$                          ; 0819
                          52   DD 00075            PUSHL    OCD                                        ; 0821
     0000G CF             01   FB 00077            CALLS    #1, CHECK_OPER_COVERAGE                    ;
                               04 0007C 4$:        RET                                                 ; 0826

; Routine Size:  125 bytes,    Routine Base:  $CODE$ + 0309
```

```
1 GLOBAL ROUTINE UPD_OPER_CONTEXT (DISABLE, MASK1, MASK2, RQCB) =
1
1 !++
1 ! Functional description:
1 !
1 !       Update the OCD count vector for each bit present in the bit mask.
1 !       The count will be decremented for a DISABLE, incremented for an ENABLE.
1 !       Also update the OCD operator intrest mask, and the corresponding intrest
1 !       mask in the operator RQCB.  This must be done in two loops, due to
1 !       BLISS's inability to cope with a bitmask of more then 32 elements.
1 !       Also note that the code could be more compact, but I traded that
1 !       for readability.
1 !
1 ! Input:
1 !
1 !       DISABLE : A boolean value that declares whether this is an ENABLE or DISABLE.
1 !       MASK1   : The first 32 bits of an operator attention mask.
1 !       MASK2   : The second 32 bits of an operator attention mask.
1 !       RQCB    : Address of an operator RQCB.
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       The operator context contained in the RQCB
1 !       and the appropriate OCD is updated.
1 !
1 ! Side effects:
1 !
1 !       None.
1 !
1 ! Routine value:
1 !
1 !       TRUE    : If an element of the countvector went to 0
1 !       FALSE   : If no element of the countvector went to 0
1 !--
1
2 BEGIN                                         ! Start of UPD_OPER_CONTEXT
2
2 MAP
2       RQCB            : $ref_bblock;          ! Operator RQCB
2
2 LOCAL
2       OCD             : $ref_bblock,          ! OCD data structure
2       K               : LONG,                 ! Index into enablecount vector
2       TRANSITION      : LONG,                 ! Boolean
2       ENABLE_MASK     : BITVECTOR [32],       ! ENABLE/DISABLE control bits
2       CHANGE_BITS1    : LONG,                 ! ditto
2       CHANGE_BITS2    : LONG;                 ! ibid
2
2 TRANSITION = FALSE;
```

```
2 IF .DISABLE
2 THEN
3     BEGIN
3     !
3     ! This is a DISABLE request.  Determine the bits to clear.
3     !
3     CHANGE_BITS1 = .RQCB [RQCB_L_ATTNMASK1] AND .MASK1;
3     CHANGE_BITS2 = .RQCB [RQCB_L_ATTNMASK2] AND .MASK2;
3     END
2 ELSE
3     BEGIN
3     !
3     ! This is an ENABLE request.  Determine the bits to set.
3     !
3     CHANGE_BITS1 = (NOT .RQCB [RQCB_L_ATTNMASK1]) AND .MASK1;
3     CHANGE_BITS2 = (NOT .RQCB [RQCB_L_ATTNMASK2]) AND .MASK2;
2     END;
2
2 !
2 ! Get the OCD address and do the update.
2 !
2 OCD = .RQCB [RQCB_L_OCD];                              ! Get OCD address
2 ENABLE_MASK = .CHANGE_BITS1;                           ! Get first 32 bits
2 INCR J FROM 0 TO 31 DO
2     IF .ENABLE_MASK [.J]
2     THEN
2         IF .DISABLE
2         THEN
3             BEGIN
3             RQCB [RQCB_L_ATTNMASK1] = .RQCB [RQCB_L_ATTNMASK1] AND (NOT (1^.J));
3             OCD [OCD_W_ENABLECOUNT (.J)] = .OCD [OCD_W_ENABLECOUNT (.J)] - 1;
4             IF (.OCD [OCD_W_ENABLECOUNT (.J)] EQL 0)
3             THEN
4                 BEGIN
4                 TRANSITION = TRUE;
4                 OCD [OCD_L_ATTNMASK1] = .OCD [OCD_L_ATTNMASK1] AND (NOT (1^.J));
3                 END;
3             END
2         ELSE
3             BEGIN
3             RQCB [RQCB_L_ATTNMASK1] = .RQCB [RQCB_L_ATTNMASK1] OR (1^.J);
3             OCD [OCD_W_ENABLECOUNT (.J)] = .OCD [OCD_W_ENABLECOUNT (.J)] + 1;
3             OCD [OCD_L_ATTNMASK1] = .OCD [OCD_L_ATTNMASK1] OR (1^.J);
2             END;
2
2 ENABLE_MASK = .CHANGE_BITS2;                          ! Get second 32 bits
2 INCR J FROM 0 TO 31 DO
2     IF .ENABLE_MASK [.J]
2     THEN
3         BEGIN
3         K = .J + 32;
3         IF .DISABLE
3         THEN
4             BEGIN
4             RQCB [RQCB_L_ATTNMASK2] = .RQCB [RQCB_L_ATTNMASK2] AND (NOT (1^.J));
4             OCD [OCD_W_ENABLECOUNT (.K)] = .OCD [OCD_W_ENABLECOUNT (.K)] - 1;
5             IF (.OCD [OCD_W_ENABLECOUNT (.K)] EQL 0)
```

```
;   948         0941  4                 THEN
;   949         0942  5                     BEGIN
;   950         0943  5                     TRANSITION = TRUE;
;   951         0944  5                     OCD [OCD_L_ATTNMASK2] = .OCD [OCD_L_ATTNMASK2] AND (NOT (1^.J));
;   952         0945  4                     END;
;   953         0946  4                 END
;   954         0947  3             ELSE
;   955         0948  4                 BEGIN
;   956         0949  4                 RQCB [RQCB_L_ATTNMASK2] = .RQCB [RQCB_L_ATTNMASK2] OR (1^.J);
;   957         0950  4                 OCD [OCD_W_ENABLECOUNT (.K)] = .OCD [OCD_W_ENABLECOUNT (.K)] + 1;
;   958         0951  4                 OCD [OCD_L_ATTNMASK2] = .OCD [OCD_L_ATTNMASK2] OR (1^.J);
;   959         0952  3                 END;
;   960         0953  2             END;
;   961         0954  2
;   962         0955  2 RETURN (.TRANSITION);
;   963         0956  2
;   964         0957  1 END;                                                   ! End of UPD_OPER_CONTEXT




                                           01FC 00000          .ENTRY  UPD_OPER_CONTEXT, Save R2,R3,R4,R5,R6,R7,R8 ; 0827
                                         58   D4 00002          CLRL    TRANSITION                                  ; 0883
                      50          10     AC   D0 00004          MOVL    RQCB, R0                                    ; 0890
                      53          5C     A0   9E 00008          MOVAB   92(R0), R3                                  ;
                      55          60     A0   9E 0000C          MOVAB   96(R0), R5                                  ; 0891
                      12          04     AC   E9 00010          BLBC    DISABLE, 1$                                 ; 0884
                      51          08     AC   D2 00014          MCOML   MASK1, CHANGE_BITS1                         ; 0890
        51            63                 51   CB 00018          BICL3   CHANGE_BITS1, (R3), CHANGE_BITS1            ;
                      56          0C     AC   D2 0001C          MCOML   MASK2, CHANGE_BITS2                         ; 0891
        56            65                 56   CB 00020          BICL3   CHANGE_BITS2, (R5), CHANGE_BITS2            ;
                                         0A   11 00024          BRB     2$                                          ; 0884
        51      08    AC                 63   CB 00026 1$:      BICL3   (R3), MASK1, CHANGE_BITS1                   ; 0898
        56      0C    AC                 65   CB 0002B          BICL3   (R5), MASK2, CHANGE_BITS2                   ; 0899
                      52          24     A0   D0 00030 2$:      MOVL    36(R0), OCD                                 ; 0905
                      57                 51   D0 00034          MOVL    CHANGE_BITS1, ENABLE_MASK                   ; 0906
                                         51   D4 00037          CLRL    J                                           ; 0907
        32            57                 51   E1 00039 3$:      BBC     J, ENABLE_MASK, 5$                          ; 0908
                      50          58 A241     3E 0003D          MOVAW   88(OCD)[J], R0                              ; 0914
                      18          04     AC   E9 00042          BLBC    DISABLE, 4$                                 ;
        54            01                 51   78 00046          ASHL    J, #1, R4                                   ; 0913
                      63                 54   CA 0004A          BICL2   R4, (R3)                                    ;
                                         60   B7 0004D          DECW    (R0)                                        ; 0914
                                         1E   12 0004F          BNEQ    5$                                          ; 0915
                      58                 01   D0 00051          MOVL    #1, TRANSITION                              ; 0918
        50            01                 51   78 00054          ASHL    J, #1, R0                                   ; 0919
                48    A2                 50   CA 00058          BICL2   R0, 72(OCD)                                 ;
                                         11   11 0005C          BRB     5$                                          ; 0910
        54            01                 51   78 0005E 4$:      ASHL    J, #1, R4                                   ; 0924
                      63                 54   C8 00062          BISL2   R4, (R3)                                    ;
                                         60   B6 00065          INCW    (R0)                                        ; 0925
        50            01                 51   78 00067          ASHL    J, #1, R0                                   ; 0926
                48    A2                 50   C8 0006B          BISL2   R0, 72(OCD)                                 ;
        C6            51                 1F   F3 0006F 5$:      AOBLEQ  #31, J, 3$                                  ; 0908
                      57                 56   D0 00073          MOVL    CHANGE_BITS2, ENABLE_MASK                   ; 0929
                                         51   D4 00076          CLRL    J                                           ; 0930
```

```
          36          57           51  E1 00078 6$:     BBC     J, ENABLE_MASK, 8$                            ; 0931
                      50      20   A1  9E 0007C         MOVAB   32(R1), K                                     ; 0934
                      53      58 A240  3E 00080         MOVAW   88(OCD)[K], R3                                ; 0939
                      18      04   AC  E9 00085         BLBC    DISABLE, 7$                                   ;
          54          01           51  78 00089         ASHL    J, #1, R4                                     ; 0938
                      65           54  CA 0008D         BICL2   R4, (R5)                                      ;
                                   63  B7 00090         DECW    (R3)                                          ; 0939
                                   1E  12 00092         BNEQ    8$                                            ; 0940
                      58           01  D0 00094         MOVL    #1, TRANSITION                                ; 0943
          53          01           51  78 00097         ASHL    J, #1, R3                                     ; 0944
                4C    A2           53  CA 0009B         BICL2   R3, 76(OCD)                                   ;
                                   11  11 0009F         BRB     8$                                            ; 0935
          54          01           51  78 000A1 7$:     ASHL    J, #1, R4                                     ; 0949
                      65           54  C8 000A5         BISL2   R4, (R5)                                      ;
                                   63  B6 000A8         INCW    (R3)                                          ; 0950
          53          01           51  78 000AA         ASHL    J, #1, R3                                     ; 0951
                4C    A2           53  C8 000AE         BISL2   R3, 76(OCD)                                   ;
          C2          51           1F  F3 000B2 8$:     AOBLEQ  #31, J, 6$                                    ; 0931
                      50           58  D0 000B6         MOVL    TRANSITION, R0                                ; 0955
                                       04 000B9         RET                                                   ; 0957

; Routine Size:  186 bytes,    Routine Base:  $CODE$ + 0386
```

```
1 GLOBAL ROUTINE VALID_OPERATOR (BUFFER_DESC, RQCB) =
1
1 !++
1 ! Functional description:
1 !
1 !       This routine will make sure that the device
1 !       specified in the user's request is capable
1 !       of being an operator device.  A side effect
1 !       of this routine is to create an operator device
1 !       name descriptor within the RQCB.  Note that
1 !       the operator device name is formatted in such
1 !       a way as to make for easy string compares in
1 !       the future.
1 !
1 ! Input:
1 !
1 !       BUFFER_DESC     : Address of string descriptor that points
1 !                         to the user's request message.
1 !       RQCB            : Address of an RQCB data structure.
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       A string descriptor of the validated operator device name
1 !       is created within the RQCB.
1 !
1 ! Routine value:
1 !
1 !       TRUE    : If the device is a valid operator device
1 !       FALSE   : If the device is not a valid operator device.
1 !--
1
2 BEGIN                                         ! Start of VALID_OPERATOR
2
2 MAP
2       BUFFER_DESC     : $ref_bblock,          ! User's request descriptor
2       RQCB            : $ref_bblock;          ! RQCB data structure
2
2 EXTERNAL
2       DEVICE_FAO      : $bblock;              ! FAO control string descriptor
2
2 EXTERNAL ROUTINE
2       SHARE_FULL_DEVNAME;                     ! Expand device name
2
2 LOCAL
2       ARG_LIST        : VECTOR [4],           ! Arguement list structure
```

```
2                 ARB               : $bblock [ARB$K_LENGTH],! Access rights block
2                 MSG               : $ref_bblock,          ! Pointer to user request
2                 DEV_CHAR          : $bblock [DIB$K_LENGTH],! Dev. char. buffer
2                 CHAR_DESC         : $desc_block,          ! Dev. char. buffer descriptor
2                 FULL_DESC         : $ref_bblock,          ! Descriptor for expanded name
2                 OPR_NAM_BUF       : $bblock [MAX_DEV_NAM],! Oper. device name buffer
2                 OPR_NAM_DESC      : $desc_block,          ! Oper. dev. name buffer descriptor
2                 STATUS            : LONG;
2   !
2   ! See if the requestor is issuing this request in another's behalf.
2   ! If, and the requestor does not have the privilege to do so, then
2   ! return FALSE.  Allow the request if the requestor has OPER pivilege,
2   ! or the GROUP field of the UIC's are the same and the requestor has
2   ! GROUP privilege.
2   !
2   IF .RQCB [RQCB_L_SENDERUIC] NEQ .RQCB [RQCB_L_UIC]
2   THEN
3       IF  (NOT .$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER])
3       THEN
3           IF NOT ((.$bblock [RQCB [RQCB_L_SENDERUIC], 2,0,16,0] EQL .$bblock [RQCB [RQCB_L_UIC], 2,0,16,0]) AN
3                   (.$bblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_GROUP]))
2           THEN
2               RETURN (FALSE);
2   !
2   ! Create a descriptor for the operator device name.
2   !
2   MSG = .BUFFER_DESC [DSC$A_POINTER] + OPC$K_COMHDRSIZ;
2   OPR_NAM_DESC [0,0,32,0] = .$bblock [MSG [OPC$T_OPRENABLE_OPR], 0,0,8,0];
2   OPR_NAM_DESC [DSC$A_POINTER] = MSG [OPC$T_OPRENABLE_OPR] + 1;
2   !
2   ! Create a buffer descriptor and get the device
2   ! characteristics of the operator device.
2   !
2   CHAR_DESC [0,0,32,0] = DIB$K_LENGTH;
2   CHAR_DESC [DSC$A_POINTER] = DEV_CHAR;
3   IF NOT (STATUS = $GETDEV (DEVNAM=OPR_NAM_DESC, PRIBUF=CHAR_DESC))
2   THEN
2       RETURN (.STATUS);                           ! There is no such device
2   !
2   ! Check the device type.  The device must be a
2   ! terminal, remote terminal, or mailbox.
2   !
2   IF (NOT .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_TRM]) AND
3      (NOT .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_MBX])
2   THEN
2       RETURN (FALSE);
2   !
2   ! If the device is a mailbox, then indicate such
2   ! and save the device buffer size.  The requestor
2   ! must have read and write access to the mailbox.
2   !
2   IF .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_MBX]
2   THEN
3       BEGIN
3       RETURN (FALSE);
3   !
3   ! The mailbox as operator implementation is not complete.  Tie off this code by
```

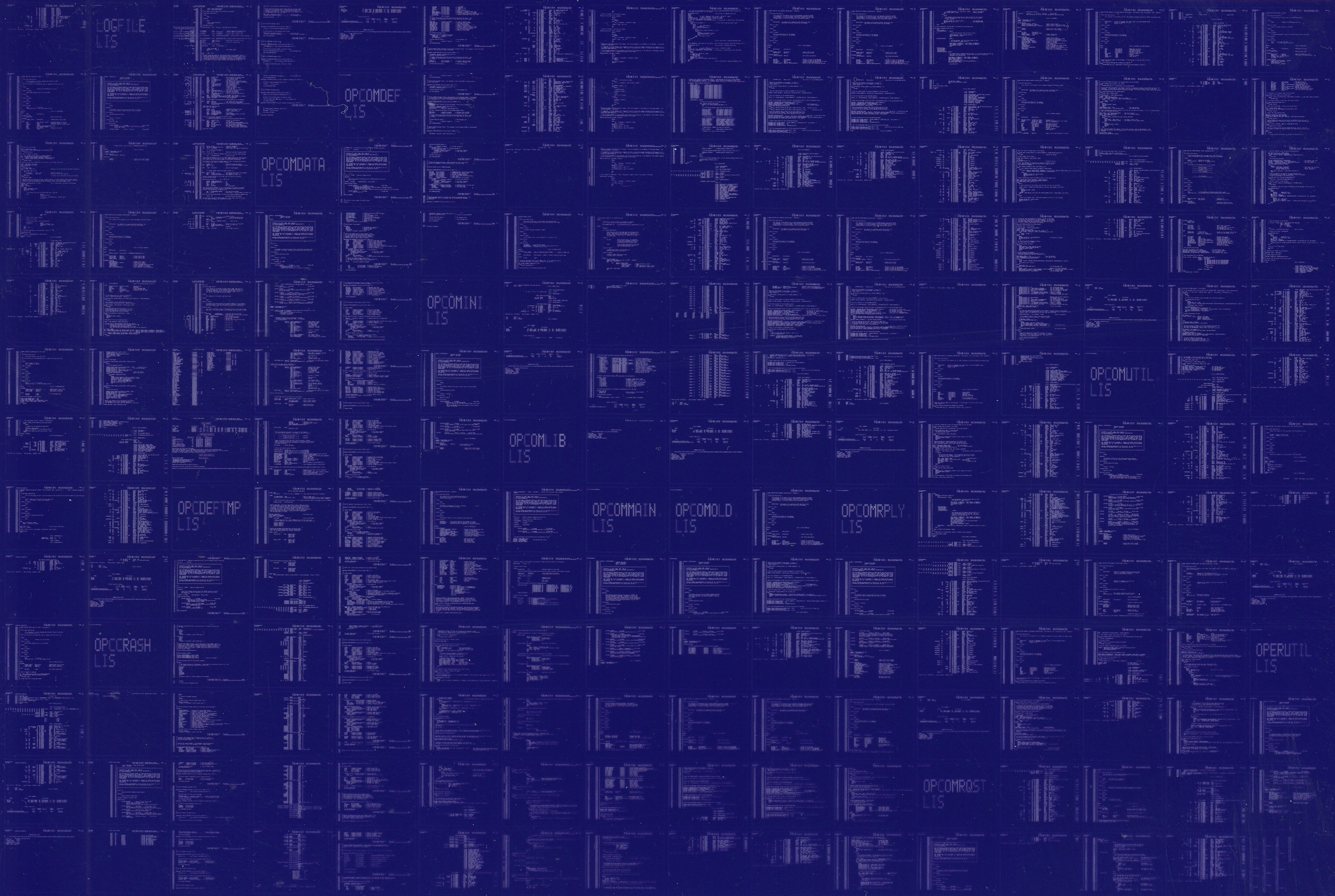
```
  3 ! commenting it out.
  3 !
  3 !     RQCB [OPRSTS_V_MBX] = TRUE;                               ! Mark OPER as MBX
  3 !     RQCB [RQCB_W_MBXSIZE] = .DEV_CHAR [DIB$W_DEVBUFSIZ];! Save MBX size
  3 !     !
  3 !     ! The following code is a workaround until a GETACCess
  3 !     ! system service can be written.  Check for R/W access.
  3 !
  3 !     CH$FILL (0, ARB$K_LENGTH, ARB);                          ! Fill with blanks
  3 !     (ARB [ARB$Q_PRIV])   = .RQCB [RQCB_L_ATTNMASK1];          ! Build a dummy ARB
  3 !     (ARB [ARB$Q_PRIV]+4) = .RQCB [RQCB_L_ATTNMASK2];          !
  3 !     ARB [ARB$L_UIC]      = .RQCB [RQCB_L_UIC];                !
  3 !     ARG_LIST [0] = 3;                                         ! Build an argument list
  3 !     ARG_LIST [1] = ARB;                                           ! Address of ARB
  3 !     ARG_LIST [2] = .DEV_CHAR [DIB$W_VPROT];                   ! Volume protection mask
  3 !     ARG_LIST [2] = .DEV_CHAR [DIB$L_OWNUIC];                  ! Volume owner
  3 !     IF NOT (STATUS = $CMKRNL (ROUTIN=EXE$CHKRDACCES, ARGLST=ARG_LIST))
  3 !     OR NOT (STATUS = $CMKRNL (ROUTIN=EXE$CHKWRTACCES, ARGLST=ARG_LIST))
  3 !     THEN
  3 !         RETURN (.STATUS);                                     ! No R/W access
  2     END;
  2 !
  2 !
  2 ! If the device is terminal, mark it as such.  If it is
  2 ! a remote terminal or a dial-in terminal, then mark it
  2 ! as a remote terminal.
  2 !
  2 ! NOTE: THE METHOD OF DETERMINING IF A TERMINAL IS
  2 !       A REMOTE TERMINAL MAY CHANGE OVER TIME.
  2 !
  2 !
  2 IF .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_TRM]
  2 THEN
  2     IF .$bblock [DEV_CHAR [DIB$L_DEVCHAR], DEV$V_MNT]
  2     OR .$bblock [DEV_CHAR [DIB$L_DEVDEPEND], TT$V_MODEM]
  2     THEN
  2         RQCB [OPRSTS_V_REMTRM] = TRUE
  2     ELSE
  2         RQCB [OPRSTS_V_TRM] = TRUE;
  2 !
  2 ! Format the operator device name from the info
  2 ! in the device characteristics buffer.  All operator
  2 ! devices known to OPCOM have their operator device
  2 ! names formatted here, so that they are in a consistant
  2 ! format.
  2 !
  2 OPR_NAM_DESC [0,0,32,0] = MAX_DEV_NAM;                    ! Create an output string descriptor
  2 OPR_NAM_DESC [DSC$A_POINTER] = OPR_NAM_BUF;               !
P 3 IF NOT (STATUS = $FAO   (DEVICE_FAO,                      ! Format the operator device name
P 3                          OPR_NAM_DESC,                    !
P 3                          OPR_NAM_DESC,                    !
P 3                          DEV_CHAR + .DEV_CHAR [DIB$W_DEVNAMOFF],
P 3                          .DEV_CHAR [DIB$W_UNIT]
  3                         ))
  2 THEN
  2     RETURN (.STATUS);
  2 !
```

```
; 1137        1129  2  ! Expand the name to the full name
; 1138        1130  2  !
; 1139        1131  2  FULL_DESC = SHARE_FULL_DEVNAME (OPR_NAM_DESC, DVI$_FULLDEVNAM);
; 1140        1132  2  RQCB [RQCB_L_OPER_LEN] = .FULL_DESC [DSC$W_LENGTH];
; 1141        1133  2  !
; 1142        1134  2  ! Create a string descriptor for the formatted
; 1143        1135  2  ! operator device name, within the RQCB.
; 1144        1136  2  !
; 1145        1137  3  IF NOT (STATUS = OPC$GET_VM (RQCB [RQCB_L_OPER_LEN], RQCB [RQCB_L_OPER_PTR]))
; 1146        1138  2  THEN
; 1147        1139  2      RETURN (.STATUS);
; 1148        1140  2  !
; 1149        1141  2  ! Copy the operator device name to the new buffer.
; 1150        1142  2  !
; 1151        1143  2  CH$MOVE (.RQCB [RQCB_L_OPER_LEN],
; 1152        1144  2           .FULL_DESC [DSC$A_POINTER],
; 1153        1145  2           .RQCB [RQCB_L_OPER_PTR]
; 1154        1146  2          );
; 1155        1147  2  RETURN (TRUE);
; 1156        1148  2
; 1157        1149  1  END;                                                   ! End of VALID_OPERATOR


                                                                    .EXTRN  DEVICE_FAO, SHARE_FULL_DEVNAME
                                                                    .EXTRN  SYS$FAO, OPC$GET_VM

                                         003C 00000                 .ENTRY  VALID_OPERATOR, Save R2,R3,R4,R5             ; 0958
                  5E      FEB4   CE   9E 00002                     MOVAB   -332(SP), SP                                  ;
                  52        08   AC   D0 00007                     MOVL    RQCB, R2                                      ; 1030
                  51        38   A2   9E 0000B                     MOVAB   56(R2), R1                                    ;
                  50        68   A2   9E 0000F                     MOVAB   104(R2), R0                                   ;
                  60             61   D1 00013                     CMPL    (R1), (R0)                                    ;
                                 13   13 00016                     BEQL    3$                                            ;
     0E      32   A2             02   E0 00018                     BBS     #2, 50(R2), 3$                                ; 1032
             02   A0        02   A1   B1 0001D                     CMPW    2(R1), 2(R0)                                  ; 1034
                                 03   13 00022                     BEQL    2$                                            ;
                               00BB   31 00024 1$:                 BRW     9$                                            ;
                  F9        31   A2   E9 00027 2$:                 BLBC    49(R2), 1$                                    ; 1035
                  50        04   AC   D0 0002B 3$:                 MOVL    BUFFER_DESC, R0                               ; 1041
     50      04   A0             26   C1 0002F                     ADDL3   #38, 4(R0), MSG                               ;
                  6E        1A   A0   9A 00034                     MOVZBL  26(MSG), OPR_NAM_DESC                         ; 1042
             04   AE        1B   A0   9E 00038                     MOVAB   27(R0), OPR_NAM_DESC+4                        ; 1043
             48   AE        74   8F   9A 0003D                     MOVZBL  #116, CHAR_DESC                               ; 1048
             4C   AE        50   AE   9E 00042                     MOVAB   DEV_CHAR, CHAR_DESC+4                         ; 1049
                                 7E   7C 00047                     CLRQ    -(SP)                                         ; 1050
                            50   AE   9F 00049                     PUSHAB  CHAR_DESC                                     ;
                                 7E   D4 0004C                     CLRL    -(SP)                                         ;
                            10   AE   9F 0004E                     PUSHAB  OPR_NAM_DESC                                  ;
       00000000G  00             05   FB 00051                     CALLS   #5, SYS$GETDEV                                ;
                  54             50   D0 00058                     MOVL    R0, STATUS                                    ;
                  74             54   E9 0005B                     BLBC    STATUS, 7$                                    ;
     7F      50   AE             02   E1 0005E                     BBC     #2, DEV_CHAR, 9$                              ; 1057
     7A      52   AE             04   E0 00063                     BBS     #4, DEV_CHAR+2, 9$                            ; 1066
     14      50   AE             02   E1 00068                     BBC     #2, DEV_CHAR, 6$                              ; 1103
     05      52   AE             03   E0 0006D                     BBS     #3, DEV_CHAR+2, 4$                            ; 1105
     06      5A   AE             05   E1 00072                     BBC     #5, DEV_CHAR+10, 5$                           ; 1106
```

```
                          78     A2              02  88 00077 4$:      BISB2   #2, 120(R2)                                   ; 1108
                                                 04  11 0007B          BRB     6$                                            ;
                          78     A2              01  88 0007D 5$:      BISB2   #1, 120(R2)                                   ; 1110
                                 6E       40     8F  9A 00081 6$:      MOVZBL  #64, OPR_NAM_DESC                             ; 1118
                          04     AE       08     AE  9E 00085          MOVAB   OPR_NAM_BUF, OPR_NAM_DESC+4                   ; 1119
                                 7E       5C     AE  3C 0008A          MOVZWL  DEV_CHAR+12, -(SP)                            ; 1125
                                 50       62     AE  3C 0008E          MOVZWL  DEV_CHAR+14, R0                               ;
                                          54   AE40  9F 00092          PUSHAB  DEV_CHAR[R0]                                  ;
                                          08     AE  9F 00096          PUSHAB  OPR_NAM_DESC                                  ;
                                          0C     AE  9F 00099          PUSHAB  OPR_NAM_DESC                                  ;
                                       0000G     CF  9F 0009C          PUSHAB  DEVICE_FAO                                    ;
                   00000000G     00              05  FB 000A0          CALLS   #5, SYS$FAO                                   ;
                                 54              50  D0 000A7          MOVL    R0, STATUS                                    ;
                                 25              54  E9 000AA          BLBC    STATUS, 7$                                    ;
                                 7E       E8     8F  9A 000AD          MOVZBL  #232, -(SP)                                   ; 1131
                                          04     AE  9F 000B1          PUSHAB  OPR_NAM_DESC                                  ;
                       0000G     CF              02  FB 000B4          CALLS   #2, SHARE_FULL_DEVNAME                        ;
                                 53              50  D0 000B9          MOVL    R0, FULL_DESC                                 ;
                          7C     A2              63  3C 000BC          MOVZWL  (FULL_DESC), 124(R2)                          ; 1132
                                       0080      C2  9F 000C0          PUSHAB  128(R2)                                       ; 1137
                                          7C     A2  9F 000C4          PUSHAB  124(R2)                                       ;
                       0000G     CF              02  FB 000C7          CALLS   #2, OPC$GET_VM                                ;
                                 54              50  D0 000CC          MOVL    R0, STATUS                                    ;
                                 04              54  E8 000CF          BLBS    STATUS, 8$                                    ;
                                 50              54  D0 000D2 7$:      MOVL    STATUS, R0                                    ; 1139
                                                     04 000D5          RET                                                   ;
          0080     D2     04     B3       7C     A2  28 000D6 8$:      MOVC3   124(R2), @4(FULL_DESC), @128(R2)              ; 1145
                                 50              01  D0 000DE          MOVL    #1, R0                                        ; 1147
                                                     04 000E1          RET                                                   ;
                                                 50  D4 000E2 9$:      CLRL    R0                                            ; 1149
                                                     04 000E4          RET                                                   ;

; Routine Size:  229 bytes,    Routine Base:  $CODE$ + 0440


; 1158          1150  1
; 1159          1151  1 END                                                ! End of OPERUTIL
; 1160          1152  0 ELUDOM




;                                 PSECT SUMMARY
;
;       Name                    Bytes                        Attributes
;
;  $CODE$                        1317  NOVEC,NOWRT,  RD , EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                          68  NOVEC,NOWRT,  RD ,NOEXE,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        32          0        1000        00:01.8
;  _$255$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1         633        52          8          43        00:00.9




;                                    COMMAND QUALIFIERS

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

; Size:            1317 code + 68 data bytes
; Run Time:          00:30.6
; Elapsed Time:      01:39.0
; Lines/CPU Min:       2259
; Lexemes/CPU-Min:  19902
; Memory Used:  157 pages
; Compilation Complete
```

0290 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


LOGFILE
LIS

OPCOMDEF
LIS

OPCOMDATA
LIS

OPCOMINI
LIS

OPCOMUTIL
LIS

OPCOMLIB
LIS

OPCDEFTMP
LIS

OPCOMMAIN
LIS

OPCOMOLD
LIS

OPCOMRPLY
LIS

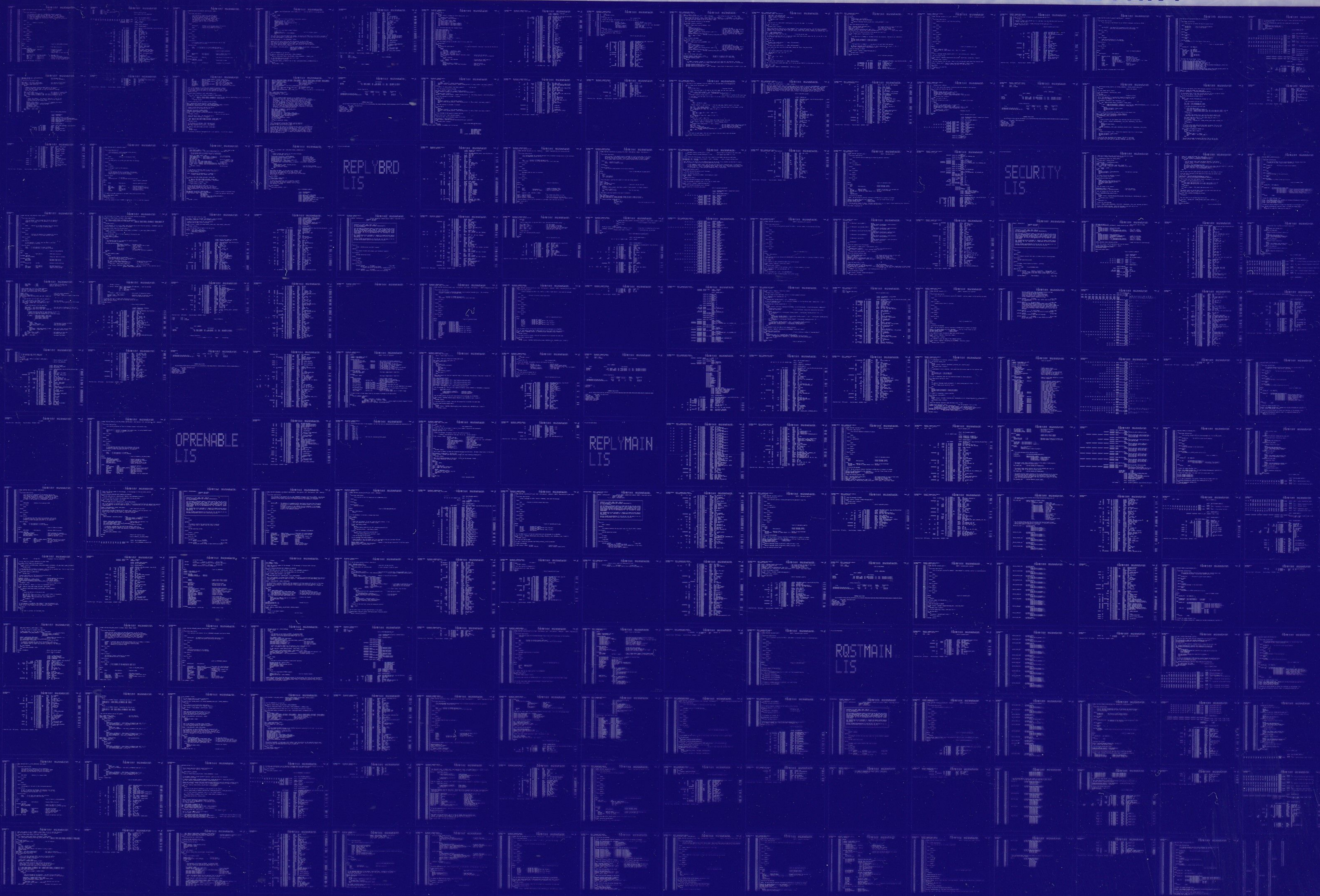
OPCCRASH
LIS

OPERUTIL
LIS

OPCOMRQST
LIS

0291 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



REPLYBRD
LIS

SECURITY
LIS

OPRENABLE
LIS

REPLYMAIN
LIS

RQSTMAIN
LIS