L 12

```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
SSS              YYY         YYY  SSS
SSS              YYY         YYY  SSS
SSS              YYY         YYY  SSS
SSS                 YYY   YYY     SSS
SSS                 YYY   YYY     SSS
SSS                 YYY   YYY     SSS
   SSSSSSSSS           YYY           SSSSSSSSS
   SSSSSSSSS           YYY           SSSSSSSSS
   SSSSSSSSS           YYY           SSSSSSSSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
```

```
_$

Ps
--

YZ

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$
```

```
**FILE**ID**UCBCREDEL                                        C  7
```

```
UU      UU    CCCCCCCC  BBBBBBBB      CCCCCCCC  RRRRRRRR    EEEEEEEEEE  DDDDDDDD    EEEEEEEEEE  LL
UU      UU    CCCCCCCC  BBBBBBBB      CCCCCCCC  RRRRRRRR    EEEEEEEEEE  DDDDDDDD    EEEEEEEEEE  LL
UU      UU  CC          BB      BB  CC          RR      RR  EE          DD      DD  EE          LL
UU      UU  CC          BB      BB  CC          RR      RR  EE          DD      DD  EE          LL
UU      UU  CC          BB      BB  CC          RR      RR  EE          DD      DD  EE          LL
UU      UU  CC          BB      BB  CC          RR      RR  EE          DD      DD  EE          LL
UU      UU  CC          BBBBBBBB    CC          RRRRRRRR    EEEEEEEE    DD      DD  EEEEEEEE    LL
UU      UU  CC          BBBBBBBB    CC          RRRRRRRR    EEEEEEEE    DD      DD  EEEEEEEE    LL
UU      UU  CC          BB      BB  CC          RR  RR      EE          DD      DD  EE          LL
UU      UU  CC          BB      BB  CC          RR  RR      EE          DD      DD  EE          LL
UU      UU  CC          BB      BB  CC          RR    RR    EE          DD      DD  EE          LL                ....
UU      UU  CC          BB      BB  CC          RR    RR    EE          DD      DD  EE          LL                ....
UUUUUUUUUU    CCCCCCCC  BBBBBBBB      CCCCCCCC  RR      RR  EEEEEEEEEE  DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL      ....
UUUUUUUUUU    CCCCCCCC  BBBBBBBB      CCCCCCCC  RR      RR  EEEEEEEEEE  DDDDDDDD    EEEEEEEEEE  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
```

Table of contents

```
0000            .TITLE  UCBCREDEL General UCB Creation/Deletion Routines
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ; R. O. Weber  14-SEP-1982
0000    ;
0000    ; Permanently present paged and non-paged routines for UCB creatation,
0000    ; deletion, and ancilliary related tasks.
0000    ;
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-013 LMP0304         L. Mark Pilant,         22-Aug-1984  8:51
0000    ;               Fix stack alignment problem introduced in LMP0302.
0000    ;
0000    ;       V03-012 LMP0302         L. Mark Pilant,         10-Aug-1984  14:45
0000    ;               Use a special kernel AST routine to delete the ACL segments
0000    ;               associated with a UCB.
0000    ;
0000    ;       V03-011 LMP0275         L. Mark Pilant,         12-Jul-1984  20:44
0000    ;               Initialize the ACL info in the ORB to be a null descriptor
0000    ;               list rather than an empty queue.  This avoids the overhead
0000    ;               of locking and unlocking the ACL mutex, only to find out
0000    ;               that the ACL was empty.
0000    ;
0000    ;       V03-010 RAS0300         Ron Schaefer            2-May-1984
0000    ;               Change unit number limit in IOC$CLONE_UCB to be 9999
0000    ;               so that cluster device names will fit in 15 characters.
0000    ;
0000    ;       V03-009 TMK0001         Todd M. Katz            26-Apr-1984
0000    ;               Remove the $LOGDEF logical name definitions.
0000    ;
0000    ;       V03-008 LMP0221         L. Mark Pilant,         31-Mar-1984  9:07
0000    ;               Add support for the Object's Rights Block (ORB).
```

```
0000  ;
0000  ;         V03-007 LMP0185         L. Mark Pilant,         23-Jan-1984  12:52
0000  ;                 Add support for ACLs on devices.
0000  ;
0000  ;         V03-006 ROW0216         Ralph O. Weber          27-AUG-1983
0000  ;                 Correct two incorrect uses of R5 as the UCB address in
0000  ;                 IOC$LINK_UCB.  R2 contains the UCB address in that routine.
0000  ;                 Also remove one instruction from the setup for IOC$LINK_UCB
0000  ;                 and IOC$SEVER_UCB.  Also change R2 usage in IOC$CREDIT_UCB to
0000  ;                 R5.
0000  ;
0000  ;         V03-005 KDM0076         Kathleen D. Morse       25-Aug-1983
0000  ;                 Fix incorrect use of R2 to be R5, in ROW0204 change.
0000  ;
0000  ;         V03-004 ROW0204         Ralph O. Weber           5-AUG-1983
0000  ;                 Change IOC$DEBIT_UCB and IOC$CREDIT_UCB to test for DEV$M_CLU
0000  ;                 being set and non-fatal bugcheck if it is.  This coincides
0000  ;                 with moving UCB$L_CPID to overlay UCB$L_LOCKID.  The later
0000  ;                 field is used only when DEV$M_CLU is set.  Therefore, testing
0000  ;                 the bit insures correct use of the overlayed field.
0000  ;
0000  ;                 Also remove IOC$DELMBX.  This has been the intention all along
0000  ;                 and now that V3.4 has shipped its no longer needed for
0000  ;                 compatibility.  Correct spelling error in .TITLE
0000  ;
0000  ;         V03-003 DMW4062         DMWalp                  23-Jun-1983
0000  ;                 Changed LOG$xxx references to LNM$xxx
0000  ;
0000  ;         V03-002 ROW0182         Ralph O. Weber          15-APR-1983
0000  ;                 Change IOC$SEVER_UCB to overwrite UCB$L_LINK of the severed
0000  ;                 UCB with minus one.  This assists wildcard GETDVI with context
0000  ;                 verification between wildcard operation calls.
0000  ;
0000  ;         V03-001 ROW0164         Ralph O. Weber          25-FEB-1983
0000  ;                 Make several bug fixes including:
0000  ;                 o Fix IOC$COPY_UCB to actually preserve R3.
0000  ;                 o Change BSB, RSB in IOC$FREE_UCB to BR.
0000  ;                 o Fix comments to indicate that IOC$CREDIT_UCB must be entered
0000  ;                   at IPL$_ASTDEL.
0000  ;                 o Optimize IOC$CREDIT_UCB to skip elevated IPL code when
0000  ;                   UCB$L_CPID equals zero.
0000  ;
```

```
0000  ;
0000  ; MACRO LIBRARY CALLS
0000  ;
0000
0000          $ACBDEF                                 ;DEFINE ACB OFFSETS
0000          $CRBDEF                                 ;DEFINE CRB OFFSETS
0000          $DDBDEF                                 ;DEFINE DDB OFFSETS
0000          $DEVDEF                                 ;DEFINE DEVICE CHARACTERISTICS FLAGS
0000          $DYNDEF                                 ;DEFINE STRUCTURE CODES
0000          $IPLDEF                                 ;DEFINE INTERRUPT PRIORITY LEVELS
0000          $JIBDEF                                 ;DEFINE JIB OFFSETS
0000          $ORBDEF                                 ;DEFINE OBJECT'S RIGHTS BLOCK OFFSETS
0000          $PCBDEF                                 ;DEFINE PCB OFFSETS
0000          $PRIDEF                                 ;DEFINE PRIORITY BOOST VALUES
0000          $PRDEF                                  ;DEFINE PROCESSOR REGISTERS
0000          $SSDEF                                   DEFINE SYSTEM STATUS VALUES
0000          $UCBDEF                                 ;DEFINE UCB OFFSETS
0000
0000  ;
0000  ; LOCAL SYMBOLS
0000  ;
```

```
                        0000         .SBTTL  IOC$CHKUCBQUOTA - Check create UCB quota
                        0000         .SBTTL  IOC$CHKMBXQUOTA - Check create mailbox quota
                        0000  ;+
                        0000  ;  IOC$CHKUCBQUOTA - Check create UCB quota
                        0000  ;  IOC$CHKMBXQUOTA - Check create mailbox quota
                        0000  ;
                        0000  ;  FUNCTIONAL DESCRIPTION
                        0000  ;
                        0000  ;      Test byte I/O count quota of process whose PCB address is in R4 for
                        0000  ;      sufficient quota to create the UCB whose template is pointed to by
                        0000  ;      R5.  IOC$CHKMBXQUOTA tests a quota requirement with low order word
                        0000  ;      of R8 summed to the UCB quota requirement.
                        0000  ;
                        0000  ;  INPUTS
                        0000  ;
                        0000  ;      R4        PCB address
                        0000  ;      R5        Template UCB address
                        0000  ;      R8        Additional quota charge (IOC$CHKMBXQUOTA only)
                        0000  ;
                        0000  ;  OUTPUTS
                        0000  ;
                        0000  ;      R0        SS$_NORMAL process has sufficient quota
                        0000  ;                SS$_EXBYTLM process does not have sufficient quota
                        0000  ;                SS$_BADPARAM quota charge overflow; <UCB$W_SIZE + R8> gt 65535
                        0000  ;-
                        0000
                    00000000         .PSECT  Y$EXEPAGED
                        0000
                        0000  IOC$CHKMBXQUOTA::
                        0000
       7E    58    3C   0000         MOVZWL  R8, -(SP)                 ; Save additional quota charge value.
             02    11   0003         BRB     CHKQUOTA                  ; Branch to common quota checking code.
                        0005
                        0005  IOC$CHKUCBQUOTA::
                        0005
             7E    D4   0005         CLRL    -(SP)                     ; Zero additional quota charge value.
                        0007
                        0007  CHKQUOTA:
                        0007
    6E    08 A5    A0   0007         ADDW2   UCB$W_SIZE(R5), (SP)      ; Sum UCB size and extra quota charge.
             16    1F   000B         BCS     80$                       ; Branch if that overflowed a word.
6E  00000100 8F    C0   000D         ADDL2   #256, (SP)                ; Add more to allow process deletion.
  50    0080 C4    D0   0014         MOVL    PCB$L_JIB(R4), R0         ; Get JIB address.
     20 A0   8E    D1   0019         CMPL    (SP)+, JIB$L_BYTCNT(R0)   ; Enough bytes to safisfy requirements?
             08    1A   001D         BGTRU   90$                       ; Branch if not enough bytes.
       50    01    3C   001F         MOVZWL  #SS$_NORMAL, R0           ; Setup success status
                   05   0022         RSB                               ; and return.
                        0023
       50    14    3C   0023  80$:   MOVZWL  #SS$_BADPARAM, R0         ; Setup quota charged overflow status
                   05   0026         RSB                               ; and return.
  50    2A14 8F    3C   0027  90$:   MOVZWL  #SS$_EXBYTLM, R0          ; Setup insufficient quota status
                   05   002C         RSB                               ; and return.
```

```
002D          .SBTTL  IOC$CLONE_UCB - Copy and link a new UCB
002D    ;+
002D    ;  IOC$CLONE_UCB - Copy and link a new UCB
002D    ;
002D    ;  FUNCTIONAL DESCRIPTION
002D    ;
002D    ;      Copy a template UCB and link it.  This is a combination of
002D    ;      IOC$COPY_UCB and IOC$LINK_UCB.  The unit number is determined adding
002D    ;      one to UCB$W_UNIT_SEED in the template UCB.  If that unit number
002D    ;      exists, the seed value is incremented and the link operation is
002D    ;      repeated.
002D    ;
002D    ;      N.B.  The UCB is not added to the list of UCBs for this controller
002D    ;      kept in the IDB.
002D    ;
002D    ;      N.B.  This routine will loop forever if all UCBs between 1 and 9999
C02D    ;      are in use.
002D    ;
002D    ;  INPUTS
002D    ;
002D    ;      R5        Template UCB address
002D    ;       UCB$W_UNIT_SEED(R5) seed unit number value
002D    ;
002D    ;      I/O database locked for write access
002D    ;
002D    ;      IPL less than or equal to IPL$_MAILBOX
002D    ;
002D    ;  OUTPUTS
002D    ;
002D    ;      R0        SS$_NORMAL UCB cloning successful
002D    ;                SS$_INSFMEM insufficient non-paged pool to copy UCB
002D    ;      R1        Address of UCB following this one in the list
002D    ;      R2        Destination UCB address
002D    ;      R3        Address of UCB preceding this one in the list
002D    ;      R4        Preserved
002D    ;      R5        Source UCB address
002D    ;
002D    ;      CRB$W_REFC( UCB$L_CRB(R2) ) incremented
002D    ;      UCB$W_UNIT_SEED(R5) <== UCB$W_UNIT(R2) ; unit number of the new UCB
002D    ;
002D    ;      The following initialization is performed on the destination UCB:
002D    ;         UCB$L_FQFL    <== addr( UCB$L_FQFL )
002D    ;         UCB$L_FQBL    <== addr( UCB$L_FQFL )
002D    ;         UCB$L_FPC     <== 0
002D    ;         UCB$L_FR3     <== 0
002D    ;         UCB$L_FR4     <== 0
002D    ;         UCB$W_BUFQUO  <== 0
002D    ;         UCB$L_LINK    <== addr( next UCB in list ) or zero
002D    ;         UCB$L_IOQFL   <== addr( UCB$L_IOQFL )
002D    ;         UCB$L_IOQBL   <== addr( UCB$L_IOQFL )
002D    ;         UCB$W_UNIT    <== unit number
002D    ;         UCB$W_CHARGE  <== UCB$W_SIZE
002D    ;         UCB$W_REFC    <== 1
002D    ;         UCB$L_STS     <== UCB$M_ONLINE
002D    ;         UCB$W_DEVSTS  <== 0
002D    ;         UCB$L_OPCNT   <== 0
002D    ;         UCB$L_SVAPTE  <== 0
```

```
                           002D  ;          UCB$W_BOFF    <== 0
                           002D  ;          UCB$W_BCNT    <== 0
                           002D  ;          UCB$L_ORB     <== addr( ORB )
                           002D  ;
                           002D  ;       The following initialization is performed on the destination ORB:
                           002D  ;          ORB$L_OWNER   <== 0
                           002D  ;          ORB$L_ACL_MUTEX <== ^X0000FFFF
                           002D  ;          ORB$B_FLAGS   <== ORB$M_PROT_16
                           002D  ;          ORB$W_PROT    <== 0
                           002D  ;          ORB$L_ACL_COUNT <== 0
                           002D  ;          ORB$L_ACL_DESC  <== 0
                           002D  ;          ORB$R_MIN_CLASS <== first longword 0
                           002D  ;-
                           002D
                       00000000          .PSECT  WIONONPAGED
                           0000
                           0000  IOC$CLONE_UCB::
                           0000
                 24   10   0000          BSBB    IOC$COPY_UCB             ; Make a copy of the template UCB.
              20 50   E9   0002          BLBC    R0, 90$                  ; Skip the rest if that failed.
    54 A2   65   01   A1   0005          ADDW3   #1, UCB$W_UNIT_SEED(R5), - ; Build first possible unit number
                           000A                  UCB$W_UNIT(R2)             ; for the new UCB.
 270F 8F    54 A2   B1   000A  30$:      CMPW    UCB$W_UNIT(R2),#9999     ; Over the limit?
                 04   1B   0010          BLEQU   40$                      ; okay
       54 A2     01   B0   0012          MOVW    #1,UCB$W_UNIT(R2)        ; Reset unit number
               0094   30   0016  40$:    BSBW    IOC$LINK_UCB             ; Attempt to link to UCB.
              05 50   E8   0019          BLBS    R0, 70$                  ; Branch if link successful.
              54 A2   B6   001C          INCW    UCB$W_UNIT(R2)           ; Else increment unit number
                 E9   11   001F          BRB     30$                      ; and try again.
       65     54 A2   B0   0021  70$:    MOVW    UCB$W_UNIT(R2), -        ; Save final unit number as
                           0025                  UCB$W_UNIT_SEED(R5)      ; next seed value.
                      05   0025  90$:    RSB                              ; Then return.
    0026          .SBTTL  IOC$COPY_UCB - Copy a given UCB
    0026      ;+
    0026      ; IOC$COPY_UCB - Copy a given UCB
    0026      ;
    0026      ; FUNCTIONAL DESCRIPTION
    0026      ;
    0026      ;     Non-paged pool sufficient to accommodate the template UCB is
    0026      ;     allocated.  The template UCB is copied to the newly allocated pool.
    0026      ;     The template copy UCB is initialized as shown below.
    0026      ;
    0026      ; INPUTS
    0026      ;
    0026      ;     R5        Template UCB address
    0026      ;
    0026      ;     I/O database locked for write access
    0026      ;
    0026      ;     IPL less than or equal to IPL$_MAILBOX
    0026      ;
    0026      ; OUTPUTS
    0026      ;
    0026      ;     R0        SS$_NORMAL UCB copy successful
    0026      ;               SS$_INSFMEM insufficient non-paged pool to copy UCB
    0026      ;     R1        Destroyed
    0026      ;     R2        Destination UCB address
    0026      ;     R3        Preserved
    0026      ;     R4        Preserved
    0026      ;     R5        Source UCB address
    0026      ;
    0026      ;     The following initialization is performed on the destination UCB:
    0026      ;        UCB$L_FQFL   <== addr( UCB$L_FQFL )
    0026      ;        UCB$L_FQBL   <== addr( UCB$L_FQFL )
    0026      ;        UCB$L_FPC    <== 0
    0026      ;        UCB$L_FR3    <== 0
    0026      ;        UCB$L_FR4    <== 0
    0026      ;        UCB$W_BUFQUO <== 0
    0026      ;        UCB$L_IOQFL  <== addr( UCB$L_IOQFL )
    0026      ;        UCB$L_IOQBL  <== addr( UCB$L_IOQFL )
    0026      ;        UCB$W_CHARGE <== UCB$W_SIZE
    0026      ;        UCB$W_REFC   <== 1
    0026      ;        UCB$L_STS    <== UCB$M_ONLINE
    0026      ;        UCB$W_DEVSTS <== 0
    0026      ;        UCB$L_OPCNT  <== 0
    0026      ;        UCB$L_SVAPTE <== 0
    0026      ;        UCB$W_BOFF   <== 0
    0026      ;        UCB$W_BCNT   <== 0
    0026      ;        UCB$L_ORB    <== addr( ORB )
    0026      ;
    0026      ;     The following initialization is performed on the destination ORB:
    0026      ;        ORB$L_ACL_MUTEX <== ^X0000FFFF
    0026      ;        ORB$B_FLAGS  <== ORB$M_PROT_16
    0026      ;        ORB$W_PROT   <== 0
    0026      ;        ORB$L_ACL_COUNT <== 0
    0026      ;        ORB$L_ACL_DESC  <== 0
    0026      ;        ORB$R_MIN_CLASS <== first longword 0
    0026      ;-
    0026
00000026          .PSECT  WIONONPAGED
```

```
                    0026
                    0026  IOC$COPY_UCB::                                     ; Copy a given UCB
           53   DD  0026            PUSHL   R3                               ; Save caller's R3.
  53    1C A5   D0  0028            MOVL    UCB$L_ORB(R5), R3                ; Get prototype ORB address
  50    08 A3   3C  002C            MOVZWL  ORB$W_SIZE(R3), R0               ; Get size of ORB
  51    08 A5   3C  0030            MOVZWL  UCB$W_SIZE(R5), R1               ; Get size of block to allocate.
     53    51   D0  0034            MOVL    R1, R3                           ; Save original size of UCB for later
     51    50   C0  0037            ADDL2   R0, R1                           ; Make ORB adjacent to UCB
        FFC3'   30  003A            BSBW    EXE$ALONONPAGED                  ; Allocate block from nonpaged memory.
        66 50   E9  003D            BLBC    R0, 40$                          ; Branch if allocation failure.
           3C   BB  0040            PUSHR   #^M<R2,R3,R4,R5>                 ; Save registers.
62  65  08 A5   28  0042            MOVC3   UCB$W_SIZE(R5), (R5), (R2)            ; Copy given UCB to new UCB.
           3C   BA  0047            POPR    #^M<R2,R3,R4,R5>                 ; Restore registers.
1C A2  52  53   C1  0049            ADDL3   R3, R2, UCB$L_ORB(R2)            ; Set ORB address
       62  52   D0  004E            MOVL    R2, UCB$L_FQFL(R2)               ; Initialize new UCB fork queue
    04 A2  52   D0  0051            MOVL    R2, UCB$L_FQBL(R2)               ; listhead.
                    0055            ASSUME  UCB$L_FR3    EQ <UCB$L_FPC + 4>
                    0055            ASSUME  UCB$W_BUFQUO EQ <UCB$L_FR4 + 4>
                    0055            ASSUME  UCB$W_SRCADDR  EQ <UCB$L_FR4 + 6>
        0C A2   7C  0055            CLRQ    UCB$L_FPC(R2)                    ; Clear fork context information, byte
        14 A2   7C  0058            CLRQ    UCB$L_FR4(R2)                    ; count quota charge, and protection.
  4C A2 4C A2   9E  005B            MOVAB   UCB$L_IOQFL(R2), UCB$L_IOQFL(R2) ; Init I/O queue listhead.
  50 A2 4C A2   9E  0060            MOVAB   UCB$L_IOQFL(R2), UCB$L_IOQBL(R2)
  56 A2 08 A2   B0  0065            MOVW    UCB$W_SIZE(R2), -                ; Initialize byte count quota charge.
                    006A                    UCB$W_CHARGE(R2)
    5C A2  01   B0  006A            MOVW    #1, UCB$W_REFC(R2)               ; Initialize reference count.
    64 A2  10   3C  006E            MOVZWL  #UCB$M_ONLINE, -                 ; Init device independent status.
                    0072                    UCB$L_STS(R2)
        68 A2   B4  0072            CLRW    UCB$W_DEVSTS(R2)                 ; Clear device dependent status.
        70 A2   D4  0075            CLRL    UCB$L_OPCNT(R2)                  ; Clear operations completed count.
                    0078            ASSUME  UCB$W_BOFF EQ <UCB$l_SVAPTE + 4>
                    0078            ASSUME  UCB$W_BCNT EQ <UCB$L_SVAPTE + 6>
        78 A2   7C  0078            CLRQ    UCB$L_SVAPTE(R2)                 ; Clear SVAPTE, byte offset, and count.
                    007B
                    007B  ; Now that the UCB has been initialized, it is time for the ORB.
                    007B
  53    1C A2   D0  007B            MOVL    UCB$L_ORB(R2), R3                ; Get the address of the new ORB
           3C   BB  007F            PUSHR   #^M<R2,R3,R4,R5>                 ; Save registers.
  54    1C A5   D0  0081            MOVL    UCB$L_ORB(R5), R4                ; Get address of the prototype ORB
63  64  08 A4   28  0085            MOVC3   ORB$W_SIZE(R4), (R4), (R3)            ; Copy given ORB to new ORB.
           3C   BA  008A            POPR    #^M<R2,R3,R4,R5>                 ; Restore registers.
04 A3  FFFF 8F  3C  008C            MOVZWL  #-1, ORB$L_ACL_MUTEX(R3)              ; Set initial mutex value
    0B A3  01   90  0092            MOVB    #ORB$M_PROT_16, ORB$B_FLAGS(R3) ; SOGW protection word
        18 A3   B4  0096            CLRW    ORB$W_PROT(R3)                   ; Set all access to everybody
                    0099
                    0099            ASSUME  ORB$L_ACL_DESC EQ ORB$L_ACL_COUNT+4
                    0099
        28 A3   7C  0099            CLRQ    ORB$L_ACL_COUNT(R3)              ; Null initial ACL
        30 A3   D4  009C            CLRL    ORB$R_MIN_CLASS(R3)              ; No classification supplied
     50    01   3C  009F            MOVZWL  #SS$_NORMAL, R0                  ; Set success completion status.
        53 8ED0     00A2  10$:      POPL    R3                               ; Restore caller's R3.
           05       00A5            RSB                                      ; Return.
                    00A6
   50   0124 8F 3C  00A6  40$:      MOVZWL  #SS$_INSFMEM, R0                 ; Set insufficient memory status.
           F5   11  00AB            BRB     10$                              ; Return.
```

```
                            00AD          .SBTTL  IOC$LINK_UCB - Link UCB to DDB chain
                            00AD    ;+
                            00AD    ; IOC$LINK_UCB - Link UCB to DDB chain
                            00AD    ;
                            00AD    ; FUNCTIONAL DESCRIPTION
                            00AD    ;
                            00AD    ;       Search UCB list pointed to by DDB referenced in input UCB and link
                            00AD    ;       input UCB into list in ascending unit number order.  Count UCB in
                            00AD    ;       number of UCBs referencing CRB pointed to by UCB.  The UCB is not
                            00AD    ;       added to the list of UCBs for this controller kept in the IDB.
                            00AD    ;
                            00AD    ;       N.B.  The UCB is not added to the list of UCBs for this controller
                            00AD    ;       kept in the IDB.
                            00AD    ;
                            00AD    ; INPUTS
                            00AD    ;
                            00AD    ;       R2        Address of UCB to be linked
                            00AD    ;                 UCB$L_DDB(R2) Address of DDB on which UCB will be hung
                            00AD    ;                 UCB$W_UNIT(R2) Unit number for UCB
                            00AD    ;                 UCB$L_CRB(R2) Address of CRB which UCB will be counted as a
                            00AD    ;                         referencer
                            00AD    ;
                            00AD    ;       I/O database locked for write access
                            00AD    ;
                            00AD    ; OUTPUTS
                            00AD    ;
                            00AD    ;       R0        SS$_NORMAL ==> Link operation successful
                            00AD    ;                 SS$_OPINCOMPL ==> Link operation failed due to presence of UCB
                            00AD    ;                         with same unit number
                            00AD    ;       R1        Address of UCB following this one in the list
                            00AD    ;       R2        Address of this UCB
                            00AD    ;       R3        Address of UCB preceding this one in the list
                            00AD    ;
                            00AD    ;       CRB$W_REFC( UCB$L_CRB(R2) ) incremented
                            00AD    ;-
                            00AD
                        000000AD          .PSECT  WIONONPAGED
                            00AD
                            00AD    IOC$LINK_UCB::
                            00AD
51   28 A2   2C       C3    00AD          SUBL3   #<UCB$L_LINK-DDB$L_UCB>, -
                            00B2                  UCB$L_DDB(R2), R1          ; Get address of first UCB link.
        53   51       D0    00B2    20$:  MOVL    R1, R3                     ; Save address of previous UCB.
     51   30 A3       D0    00B5          MOVL    UCB$L_LINK(R3), R1         ; Get address of next UCB.
             09       13    00B9          BEQL    50$                        ; 0 ==> end-of-list reached; go insert.
  54 A1   54 A2       B1    00BB          CMPW    UCB$W_UNIT(R2), UCB$W_UNIT(R1) ; Compare unit numbers.
             F0       1A    00C0          BGTRU   20$                        ; If new GT list, continue search.
             13       13    00C2          BEQL    90$                        ; If new EQ list, declare error.
     30 A2   51       D0    00C4    50$:  MOVL    R1, UCB$L_LINK(R2)         ; Else, link UCB.  Forward link new UCB.
     30 A3   52       D0    00C8          MOVL    R2, UCB$L_LINK(R3)         ; Forward link previous UCB.
     50   24 A2       D0    00CC          MOVL    UCB$L_CRB(R2), R0          ; Get CRB address.
          0C A0       B6    00D0          INCW    CRB$W_REFC(R0)             ; Increment CRB reference count.
        50   01       3C    00D3          MOVZWL  #SS$_NORMAL, R0            ; Set successful link status
                      05    00D6          RSB                                ; and return
                            00D7
   50    02D4 8F      3C    00D7    90$:  MOVZWL  #SS$_OPINCOMPL, R0         ; Set link failed status.
                      05    00DC          RSB                                ; and return.
```

```
                     00DD          .SBTTL  IOC$DEBIT_UCB - Charge process quotas for created UCB
                     00DD  ;+
                     00DD  ;  IOC$DEBIT_UCB - Charge process quotas for created UCB
                     00DD  ;
                     00DD  ;  FUNCTIONAL DESCRIPTION
                     00DD  ;
                     00DD  ;      Charge the process whose PID is in R4 for the UCB whose address is
                     00DD  ;      in R2.
                     00DD  ;
                     00DD  ;  INPUTS
                     00DD  ;
                     00DD  ;      R2       Address of UCB to be debited from process quotas
                     00DD  ;               UCB$W_CHARGE(R2) Byte I/O byte count quota charge for the UCB
                     00DD  ;                        and its associated paraphernalia
                     00DD  ;      R4       Address of PCB for process to be charged for UCB
                     00DD  ;
                     00DD  ;      IPL equal to IPL$_ASTDEL
                     00DD  ;
                     00DD  ;  OUTPUTS
                     00DD  ;
                     00DD  ;      R0       Destroyed
                     00DD  ;      R1       Destroyed
                     00DD  ;
                     00DD  ;      For JIB pointed to by PCB
                     00DD  ;         JIB$L_BYTLM reduced by UCB$W_CHARGE
                     00DD  ;         JIB$L_BYTCNT reduced by UCB$W_CHARGE
                     00DD  ;         UCB$L_CPID (which is the same as UCB$L_DUETIM) <== JIB$L_MPID
                     00DD  ;-
                     00DD
                 0000002D          .PSECT  Y$EXEPAGED
                     002D
                     002D  IOC$DEBIT_UCB::
                     002D
                     002D          ASSUME  DEV$M_CLU EQ 1
     17 3C A2   E8   002D          BLBS    UCB$L_DEVCHAR2(R2), 90$ ; Branch if UCB$L_LOCKID is in use.
   50   56 A2   3C   0031          MOVZWL  UCB$W_CHARGE(R2), R0    ; Get amount to charge BYTLM quota.
 51   0080 C4   D0   0035          MOVL    PCB$L_JIB(R4), R1       ; Get JIB address.
   24 A1    50   C2   003A          SUBL2   R0, JIB$L_BYTLM(R1)     ; Reduce byte count limit.
   20 A1    50   C2   003E          SUBL2   R0, JIB$L_BYTCNT(R1)    ; Reduce byte count quota.
20 A2   54 A1   D0   0042          MOVL    JIB$L_MPID(R1), -       ; Save master PID charged in
                     0047                  UCB$L_CPID(R2)          ; charged UCB.
                05   0047          RSB
                     0048
                     0048  90$:    BUG_CHECK INCONSTATE            ; Non-fatal bugcheck if DEV$M_CLU set.
                05   004C          RSB                             ; Then continue, ignoring the debit
                     004D                                          ; request.
```

```
                      004D         .SBTTL  IOC$DELETE_UCB - Delete UCB if REFC eq 0
                      004D   ;+
                      004D   ;  IOC$DELETE_UCB - Delete UCB if REFC eq 0
                      004D   ;
                      004D   ;  FUNCTIONAL DESCRIPTION
                      004D   ;
                      004D   ;      Check UCB pointed to by R5 for possible deletion and if needed
                      004D   ;      delete it.  In order to be deleted, a UCB must 1) have UCB$W_REFC
                      004D   ;      equal to zero, and 2) have the UCB$V_DELETEUCB bit set in UCB$L_STS.
                      004D   ;      If UCB can be deleted, sever UCB linkage and return space occupied
                      004D   ;      by UCB to non-paged pool.  This is a combination of IOC$SEVER_UCB
                      004D   ;      and IOC$FREE_UCB.  The UCB is not removed from the list of UCBs
                      004D   ;      for this controller kept in the IDB.
                      004D   ;
                      004D   ;  INPUTS
                      004D   ;
                      004D   ;      R5        Address of UCB to be unlinked
                      004D   ;                UCB$L_DDB(R5) Address of DDB on which UCB is hung
                      004D   ;                UCB$L_CRB(R5) Address of CRB which counts UCB as a
                      004D   ;                        referencer
                      004D   ;
                      004D   ;      I/O database locked for write access
                      004D   ;
                      004D   ;  OUTPUTS
                      004D   ;
                      004D   ;      R0        Destroyed
                      004D   ;      R1        Destroyed
                      004D   ;
                      004D   ;      CRB$W_REFC( UCB$L_CRB(R5) ) decremented
                      004D   ;-
                      004D
                  000000DD         .PSECT  WIONONPAGED
                      00DD
                      00DD   IOC$DELETE_UCB::
                      00DD
         0F'AF   9F   00DD         PUSHAB  B^IOC$FREE_UCB          ; Setup to free UCB after severing it.
   05 64 A5  10   E1   00E0         BBC     #UCB$V_DELETEUCB, -     ; Is the delete UCB bit set?
                      00E5                 UCB$L_STS(R5), 70$      ; Branch if bit not set.
         5C A5   B5   00E5         TSTW    UCB$W_REFC(R5)          ; Is the reference count zero?
            03   13   00E8         BEQL    IOC$SEVER_UCB           ; Branch to sever UCB if count is zero.
                      00EA
                      00EA   70$:                                  ; UCB cannot be deleted.
            8E   D5   00EA         TSTL    (SP)+                   ; Pop IOC$FREE_UCB address from stack.
                 05   00EC         RSB                             ; Return without deleting UCB.
```

```
                                00ED   529         .SBTTL  IOC$SEVER_UCB - Unlink a UCB
                                00ED   530 ;+
                                00ED   531 ;  IOC$SEVER_UCB - Unlink a UCB
                                00ED   532 ;
                                00ED   533 ;      Remove UCB pointed to by R5 from UCB list pointed to by DDB
                                00ED   534 ;      referenced in UCB.  Reduce count of UCBs referencing CRB pointed
                                00ED   535 ;      to by UCB by one.  The UCB is not removed from the list of UCBs
                                00ED   536 ;      for this controller kept in the IDB.
                                00ED   537 ;
                                00ED   538 ;  INPUTS
                                00ED   539 ;
                                00ED   540 ;      R5         Address of UCB to be unlinked
                                00ED   541 ;                 UCB$L_DDB(R5) Address of DDB on which UCB is hung
                                00ED   542 ;                 UCB$L_CRB(R5) Address of CRB which counts UCB as a
                                00ED   543 ;                         referencer
                                00ED   544 ;
                                00ED   545 ;      I/O database locked for write access
                                00ED   546 ;
                                00ED   547 ;  OUTPUTS
                                00ED   548 ;
                                00ED   549 ;      R0         Destroyed
                                00ED   550 ;      R1         Destroyed
                                00ED   551 ;
                                00ED   552 ;      UCB$L_LINK(R5) <== -1
                                00ED   553 ;      CRB$W_REFC( UCB$L_CRB(R5) ) decremented
                                00ED   554 ;-
                                00ED   555
                            000000ED   556         .PSECT  WIONONPAGED
                                00ED   557
                                00ED   558 IOC$SEVER_UCB::
                                00ED   559
 50    28 A5    2C      C3      00ED   560         SUBL3   #<UCB$L_LINK-DDB$L_UCB>, -
                                00F2   561                 UCB$L_DDB(R5), R0           ; Get address of first UCB link.
          51    50      D0      00F2   562 10$:    MOVL    R0, R1                      ; Save address of last UCB.
    50    30 A1         D0      00F5   563         MOVL    UCB$L_LINK(R1), R0          ; Get address of next UCB.
          55    50      D1      00F9   564         CMPL    R0,R5                       ; Do the UCB addresses match?
                F4      12      00FC   565         BNEQ    10$                         ; Branch and loop if no match.
   30 A1    30 A5       D0      00FE   566         MOVL    UCB$L_LINK(R5), UCB$L_LINK(R1) ; Else, remove UCB from UCB list.
    30 A5       01      CE      0103   567         MNEGL   #1,UCB$L_LINK(R5)           ; Invalidate severed UCB's forward link.
    50    24 A5         D0      0107   568         MOVL    UCB$L_CRB(R5), R0           ; Get CRB address.
          0C A0         B7      010B   569         DECW    CRB$W_REFC(R0)              ; Decrement CRB reference count.
                        05      010E   570         RSB
```

```
                           010F            .SBTTL  IOC$FREE_UCB - Free pool used by a UCB
                           010F      ;+
                           010F      ;  IOC$FREE_UCB - Free pool used by a UCB
                           010F      ;
                           010F      ;  FUNCTIONAL DESCRIPTION
                           010F      ;
                           010F      ;       Return to non-paged pool the space occupied by the UCB pointed to by
                           010F      ;       R5.
                           010F      ;
                           010F      ;  INPUTS
                           010F      ;
                           010F      ;       R5      UCB address
                           010F      ;
                           010F      ;       I/O database locked for write access
                           010F      ;
                           010F      ;  OUTPUTS
                           010F      ;
                           010F      ;       R0      Destroyed
                           010F      ;-
                           010F
                       0000010F            .PSECT  WIONONPAGED
                           010F
                           010F      IOC$FREE_UCB::
          7E    54    D0   010F            MOVL    R4, -(SP)               ; Save a register
      54    1C A5    D0   0112            MOVL    UCB$L_ORB(R5), R4       ; Get the address of the ORB
                5B    13   0116            BEQL    20$                     ; Skip following if no ORB present
   56 0B A4    01    E1   0118            BBC     #ORB$V_ACL_QUEUE, ORB$B_FLAGS(R4), 20$  ; Xfer if ACL not a queue
                           011D
                           011D      ; If there are no ACL segments, life is simple.
                           011D
      50    28 A4    9E   011D            MOVAB   ORB$L_ACLFL(R4), R0     ; Get addr of ACL queue head
         50    60    D1   0121            CMPL    (R0), R0                ; Is the queue empty?
                4D    13   0124            BEQL    20$                     ; Xfer if so, nothing to do here
                           0126
                           0126      ; Since there are ACL segments, it will be necessary to fire off a special
                           0126      ; kernel AST to the SWAPPER process to delete them.  This is because IOC$FREE_UCB
                           0126      ; may be called above IPL 2.  With ACL segments living in paged pool, this would
                           0126      ; not be a friendly gesture.
                           0126
      7E    2C A4    D0   0126            MOVL    ORB$L_ACLBL(R4), -(SP)  ; Save addr of last segment
      50    28 A4    0F   012A            REMQUE  ORB$L_ACLFL(R4), R0     ; Separate ORB from ACL segments
         7E    54    7D   012E            MOVQ    R4, -(SP)               ; Save R4 & UCB address
         7E    53    D0   0131            MOVL    R3, -(SP)               ; Save some more registers
         7E    51    7D   0134            MOVQ    R1, -(SP)
         51    24    D0   0137            MOVL    #ACB$C_LENGTH+8, R1     ; Size of the block to get
             FEC3'   30   013A            BSBW    EXE$ALONONPAGED         ; Get block for special kernel AST
         55    52    D0   013D            MOVL    R2, R5                  ; Copy block addr to right register
      0A A5    02    90   0140            MOVB    #DYN$C_ACB, ACB$B_TYPE(R5)      ; Set structure type
      08 A5    51    B0   0144            MOVW    R1, ACB$W_SIZE(R5)      ; Set structure size
 0C A5   00000000'EF  D0   0148            MOVL    SCH$GL_SWPPID, ACB$L_PID(R5)    ; Set target process PID
    0B A5    80 8F    90   0150            MOVB    #ACB$M_KAST, ACB$B_RMOD(R5)     ; Special kernel AST
 18 A5   00000195'EF  9E   0155            MOVAB   60$, ACB$L_KAST(R5)     ; Set address of AST routine
    14 BE    1C A5    0E   015D            INSQUE  ACB$C_LENGTH(R5), @20(SP)       ; Add ACL segments to AST block
         52    02    D0   0162            MOVL    #PRI$_RESAVL, R2        ; Set priority increment
             FE98'   30   0165            BSBW    SCH$QAST                ; Fire off special kernel AST
         51    8E    7D   0168            MOVQ    (SP)+, R1               ; Restore saved registers
         53    8E    D0   016B            MOVL    (SP)+, R3
```

```
        54    8E   7D  016E   629          MOVQ    (SP)+, R4
              8E   D5  0171   630          TSTL    (SP)+                   ; Final cleanup of the stack
                       0173   631
                       0173   632 ; Now that the ACL segments have been taken care of, delete the ORB and UCB.
                       0173   633
     50    08 A5   3C  0173   634 20$:     MOVZWL  UCB$W_SIZE(R5), R0      ; Note size of the current UCB
        50    55   C0  0177   635          ADDL2   R5, R0                  ; Calc end of the UCB
        54    50   D1  017A   636          CMPL    R0, R4                  ; Is this where the ORB lives?
              08   13  017D   637          BEQL    40$                     ; Xfer if so, diddle the UCB size
        50    54   D0  017F   638          MOVL    R4, R0                  ; Else setup ORB address to deallocate
           FE7B'   30  0182   639          BSBW    COM$DRVDEALMEM          ; Release the ORB
              05   11  0185   640          BRB     50$                     ; And now for the UCB
  08 A5    08 A4   A0  0187   641 40$:     ADDW2   ORB$W_SIZE(R4), UCB$W_SIZE(R5)  ; Release ORB also
        54    8E   D0  018C   642 50$:     MOVL    (SP)+, R4               ; Restore saved register
        50    55   D0  018F   643          MOVL    R5, R0                  ; Setup UCB address to deallocate.
           FE6B'   31  0192   644          BRW     COM$DRVDEALMEM          ; Deallocate the UCB and return to
                       0195   645                                          ; caller.
                       0195   646
                       0195   647 ; Here is the special kernel AST routine used to deallocate the ACL segments
                       0195   648 ; associated with the UCB being vaporized.
                       0195   649
        7E    53   D0  0195   650 60$:     MOVL    R3, -(SP)               ; Save a register
     50    1C B5   0F  0198   651 70$:     REMQUE  @ACB$C_LENGTH(R5), R0   ; Remove ACL segment
              05   1D  019C   652          BVS     80$                     ; Xfer if no more
           FE5F'   30  019E   653          BSBW    EXE$DEAPAGED            ; Else deallocate the segment
              F5   11  01A1   654          BRB     70$                     ; And try for another
        53    8E   D0  01A3   655 80$:     MOVL    (SP)+, R3               ; Restore a register
        50    55   D0  01A6   656          MOVL    R5, R0                  ; And now to deallocate the
           FE54'   31  01A9   657          BRW     EXE$DEANONPAGED         ; AST control block
```

```
                                  01AC              .SBTTL  IOC$CREDIT_UCB - Return UCB charged quotas
                                  01AC      ;+
                                  01AC      ;  IOC$CREDIT_UCB - Return UCB charged quotas
                                  01AC      ;
                                  01AC      ;  FUNCTIONAL DESCRIPTION
                                  01AC      ;
                                  01AC      ;       Credit the process with the PID stored in UCB$L_CPID(R5) for the
                                  01AC      ;       UCB charges associated with the UCB whose address is in R5.  If
                                  01AC      ;       UCB$L_CPID equals zero or the process pointed to by UCB$L_CPID does
                                  01AC      ;       not exist, make no process quota changes.
                                  01AC      ;
                                  01AC      ;  INPUTS
                                  01AC      ;
                                  01AC      ;       R5       Address of UCB to be credited to process quotas
                                  01AC      ;                UCB$L_CPID(R5) Process ID of process to which quota usage is to
                                  01AC      ;                        be credited
                                  01AC      ;                UCB$W_CHARGE(R5) Byte I/O byte count quota charge for the UCB
                                  01AC      ;                        and its associated paraphernalia
                                  01AC      ;
                                  01AC      ;        IPL equal to IPL$_ASTDEL
                                  01AC      ;
                                  01AC      ;  OUTPUTS
                                  01AC      ;
                                  01AC      ;       R0       Destroyed
                                  01AC      ;       R1       Destroyed
                                  01AC      ;
                                  01AC      ;       For JIB pointed to by PCB
                                  01AC      ;          JIB$L_BYTLM increased by UCB$W_CHARGE
                                  01AC      ;          JIB$L_BYTCNT increased by UCB$Q_CHARGE
                                  01AC      ;          UCB$L_CPID (which is the same as UCB$L_DUETIM) <== 0
                                  01AC      ;-
                                  01AC
                              0000004D              .PSECT  Y$EXEPAGED
                                  004D
                                  004D      IOC$CREDIT_UCB::
                                  004D
                                  004D              ASSUME  DEV$M_CLU EQ 1
           3B 3C A5         E8    004D              BLBS    UCB$L_DEVCHAR2(R5), 90$ ; Branch if UCB$L_LOCKID is in use.
        51    20 A5         3C    0051              MOVZWL  UCB$L_CPID(R5), R1      ; Get charged PID index.
                 30         13    0055              BEQL    40$                     ; Branch if none.
                                  0057              DSBINT  70$                     ; Block scheduler database changes.
   51   00000000'FF41       D0    0061              MOVL    @SCH$GL_PCBVEC[R1], R1  ; Get PCB address.
     20 A5    60 A1         D1    0069              CMPL    PCB$L_PID(R1), UCB$L_CPID(R5)   ; Is PID correct?
                 14         12    006E              BNEQ    30$                     ; Branch if no longer the right PID.
       51   0080 C1         D0    0070              MOVL    PCB$L_JIB(R1), R1       ; Get JIB address.
        50    56 A5         3C    0075              MOVZWL  UCB$W_CHARGE(R5), R0    ; Get charged amount.
        24 A1    50         C0    0079              ADDL2   R0, JIB$L_BYTLM(R1)     ; Return byte count limit.
        20 A1    50         C0    007D              ADDL2   R0, JIB$L_BYTCNT(R1)    ; Return byte count quota.
              20 A5         D4    0081              CLRL    UCB$L_CPID(R5)          ; Zero charged PID.
                                  0084      30$:    ENBINT                          ; Restore privious IPL.
                    05            0087      40$:    RSB                             ; Return
                                  0088
              00000008            0088      70$:    .LONG   IPL$_SYNCH                      ; Construct used to temporarily
                                  008C              ASSUME  <. - IOC$CREDIT_UCB> LE 512     ; lock less than a page at
                                  008C                                                      ; elevated IPL.
                                  008C
                                  008C      90$:    BUG_CHECK INCONSTATE            ; Non-fatal bugcheck if DEV$M_CLU set.
```

```
 05  0090   716         RSB                                      ; Then continue, ignoring the credit
     0091   717                                                  ; request.
```

```
                         0091          .SBTTL  IOC$CREATE_UCB - CREATE MAILBOX OR NETWORK UCB
                         0091  ;+
                         0091  ; IOC$CREATE_UCB - CREATE MAILBOX OR NETWORK UCB
                         0091  ;
                         0091  ; THIS ROUTINE IS CALLED TO CPEATE A MAILBOX OR NETWORK UCB AND LINK IT INTO
                         0091  ; THE I/O DATABASE.
                         0091  ;
                         0091  ; INPUTS:
                         0091  ;
                         0091  ;       R4 = CURRENT PROCESS PCB ADDRESS.
                         0091  ;       R5 = ADDRESS OF CLONE UCB.
                         0091  ;
                         0091  ;       I/O DATABASE IS LOCKED FOR WRITE ACCESS.
                         0091  ;
                         0091  ; OUTPUTS:
                         0091  ;
                         0091  ;       RO LOW BIT CLEAR INDICATES FAILURE TO ALLOCATE UCB.
                         0091  ;
                         0091  ;               RO = SS$_INSFMEM - INSUFFICIENT MEMORY TO ALLOCATE MAILBOX
                         0091  ;                       OR NETWORK UCB.
                         0091  ;
                         0091  ;       RO LOW BIT SUCCESS INDICATES SUCCESSFUL CREATION.
                         0091  ;
                         0091  ;               R2 = ADDRESS OF ALLOCATED UCB.
                         0091  ;
                         0091  ;       CONTROL IS RETURNED WITH I/O DATABASE STILL LOCKED FOR WRITE ACCESS.
                         0091  ;
                         0091  ;       This is a temporary replacement for the V3.x UCB creation routine
                         0091  ;       (found in IOSUBPAGD).  This routine will be removed when development
                         0091  ;       of V3.x compatible software which dynamically creates and deletes
                         0091  ;       UCBs is concluded.
                         0091  ;-
                         0091
                     00000091          .PSECT  Y$EXEPAGED
                         0091
                         0091  IOC$CREATE_UCB::                          ;CREATE MAILBOX OR NETWORK UCB
                         0091
           FF6C'   30    0091          BSBW    IOC$CLONE_UCB             ; Clone a copy of the UCB.
         16 50     E9    0094          BLBC    R0, 90$                   ; Branch if clone failed.
                         0097                                            ; Then do the things that IOC$COPY_UCB
                         0097                                            ; did that IOC$CLONE_UCB does not do.
    50    1C A2    D0    0097          MOVL    UCB$L_ORB(R2),R0          ; Get the address of the ORB
  60    00BC C4    D0    009B          MOVL    PCB$L_UIC(R4), -          ;   Insert creator UIC.
                         00A0                  ORB$L_OWNER(R0)
  50    0080 C4    D0    00A0          MOVL    PCB$L_JIB(R4), RC         ;   Get JIB address.
20 A2     54 A0    D0    00A5          MOVL    JIB$L_MPID(R0), -         ;   Store master PID as creator.
                         00AA                  UCD$L_CPID(R2)
     50    01      3C    00AA          MOVZWL  #SS$_NORMAL, R0           ; Indicate that function succeeded.
                   05    00AD  90$:    RSB                               ; Return
                         00AE
                         00AE          .END
```

```
ACB$B_RMOD                        = 0000000B                  UCB$L_CPID                        = 00000020
ACB$B_TYPE                        = 0000000A                  UCB$L_CRB                         = 00000024
ACB$C_LENGTH                      = 0000001C                  UCB$L_DDB                         = 00000028
ACB$L_KAST                        = 00000018                  UCB$L_DEVCHAR2                    = 0000003C
ACB$L_PID                         = 0000000C                  UCB$L_FPC                         = 0000000C
ACB$M_KAST                        = 00000080                  UCB$L_FQBL                        = 00000004
ACB$W_SIZE                        = 00000008                  UCB$L_FQFL                        = 00000000
BUG$_INCONSTATE                     ********    X     02      UCB$L_FR3                         = 00000010
CHKQUOTA                            00000007 R        02      UCB$L_FR4                         = 00000014
COM$DRVDEALMEM                      ********    X     03      UCB$L_IOQBL                       = 00000050
CRB$W_REFC                        = 0000000C                  UCB$L_IOQFL                       = 0000004C
DDB$L_UCB                         = 00000004                  UCB$L_LINK                        = 00000030
DEV$M_CLU                         = 00000001                  UCB$L_OPCNT                       = 00000070
DYN$C_ACB                         = 00000002                  UCB$L_ORB                         = 0000001C
EXE$ALONONPAGED                     ********    X     03      UCB$L_STS                         = 00000064
EXE$DEANONPAGED                     ********    X     03      UCB$L_SVAPTE                      = 00000078
EXE$DEAPAGED                        ********    X     03      UCB$M_ONLINE                      = 00000010
IOC$CHKMBXQUOTA                     00000000 RG       02      UCB$V_DELETEUCB                   = 00000010
IOC$CHKUCBQUOTA                     00000005 RG       02      UCB$W_BCNT                        = 0000007E
IOC$CLONE_UCB                       00000000 RG       03      UCB$W_BOFF                        = 0000007C
IOC$COPY_UCB                        00000026 RG       03      UCB$W_BUFQUO                      = 00000018
IOC$CREATE_UCB                      00000091 RG       02      UCB$W_CHARGE                      = 00000056
IOC$CREDIT_UCB                      0000004D RG       02      UCB$W_DEVSTS                      = 00000068
IOC$DEBIT_UCB                       0000002D RG       02      UCB$W_REFC                        = 0000005C
IOC$DELETE_UCB                      000000DD RG       03      UCB$W_SIZE                        = 00000008
IOC$FREE_UCB                        0000010F RG       03      UCB$W_SRCADDR                     = 0000001A
IOC$LINK_UCB                        000000AD RG       03      UCB$W_UNIT                        = 00000054
IOC$SEVF _UCB                       000000ED RG       03      UCB$W_UNIT_SEED                   = 00000000
IPL$_SYNCH                        = 00000008
JIB$L_BYTCNT                      = 00000020
JIB$L_BYTLM                       = 00000024
JIB$L_MPID                        = 00000054
ORB$B_FLAGS                       = 0000000B
ORB$L_ACLBL                       = 0000002C
ORB$L_ACLFL                       = 00000028
ORB$L_ACL_COUNT                   = 00000028
ORB$L_ACL_DESC                    = 0000002C
ORB$L_ACL_MUTEX                   = 00000004
ORB$L_OWNER                       = 00000000
ORB$M_PROT_16                     = 00000001
ORB$R_MIN_CLASS                   = 00000030
ORB$V_ACL_QUEUE                   = 00000001
ORB$W_PROT                        = 00000018
ORB$W_SIZE                        = 00000008
PCB$L_JIB                         = 00000080
PCB$L_PID                         = 00000060
PCB$L_UIC                         = 000000BC
PR$_IPL                           = 00000012
PRI$_RESAVL                       = 00000002
SCH$GL_PCBVEC                       ********    X     02
SCH$GL_SWPPID                       ********    X     03
SCH$QAST                            ********    X     03
SS$_BADPARAM                      = 00000014
SS$_EXBYTLM                       = 00002A14
SS$_INSFMEM                       = 00000124
SS$_NORMAL                        = 00000001
SS$_OPINCOMPL                     = 000002D4
```

```
                                             +----------------+
                                             ! Psect synopsis !
                                             +----------------+

PSECT name                  Allocation          PSECT No.  Attributes
----------                  ----------          ---------  ----------
.  ABS  .                   00000000  (     0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                       00000000  (     0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
Y$EXEPAGED                  000000AE  (   174.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
WIONONPAGED                 000001AC  (   428.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                          +------------------------+
                                          ! Performance indicators !
                                          +------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   30    00:00:00.05    00:00:01.77
Command processing              106    00:00:00.48    00:00:04.74
Pass 1                          370    00:00:13.01    00:00:43.50
Symbol table sort                 0    00:00:02.11    00:00:06.73
Pass 2                          146    00:00:02.91    00:00:12.82
Symbol table output              10    00:00:00.10    00:00:00.14
Psect synopsis output             2    00:00:00.03    00:00:00.23
Cross-reference output            0    00:00:00.00    00:00:00.00
Assembler run totals            666    00:00:18.69    00:01:09.93

The working set limit was 1650 pages.
76372 bytes (150 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1385 non-local and 25 local symbols.
769 source lines were read in Pass 1, producing 16 object records in Pass 2.
24 pages of virtual memory were used to define 23 macros.

                                         +--------------------------+
                                         ! Macro library statistics !
                                         +--------------------------+

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                     13
_$255$DUA28:[SYSLIB]STARLET.MLB;2                   7
TOTALS (all libraries)                             20

1505 GETS were required to define 20 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:UCBCREDEL/OBJ=OBJ$:UCBCREDEL MSRC$:UCBCREDEL/UPDATE=(ENH$:UCBCREDEL)+EXECML$/LIB
```

0389 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

WRTMFYPAG
LIS

UCBCREDEL
LIS

USRVECTOR
LIS

SYSVECTOR
LIS

SYSINIT
LIS

VERSION
LIS

SYSINI

UPCASEDAT
LIS

SYSINIT
MAP

SYSWAIT
LIS

TIMESCHDL
LIS