```
TTTTTTTTTTTTTTT  TTTTTTTTTTTTTTT  DDDDDDDDDDDD     RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
TTTTTTTTTTTTTTT  TTTTTTTTTTTTTTT  DDDDDDDDDDDDD    RRRRRRRRRRRRR    VVV         VVV  RRRRRRRRRRRRR
TTTTTTTTTTTTTTT  TTTTTTTTTTTTTTT  DDDDDDDDDDDDD    RRRRRRRRRRRRR    VVV         VVV  RRRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRRRRRRRRRRR    VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRRRRRRRRRRR    VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRRRRRRRRRRR    VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRR   RRR       VVV         VVV  RRR   RRR
      TTT              TTT        DDD         DDD  RRR   RRR       VVV         VVV  RRR   RRR
      TTT              TTT        DDD         DDD  RRR   RRR       VVV         VVV  RRR   RRR
      TTT              TTT        DDD         DDD  RRR      RRR     VVV   VVV     RRR      RRR
      TTT              TTT        DDD         DDD  RRR      RRR     VVV   VVV     RRR      RRR
      TTT              TTT        DDD         DDD  RRR      RRR     VVV   VVV     RRR      RRR
      TTT              TTT        DDDDDDDDDDDD     RRR         RRR     VVV       RRR         RRR
      TTT              TTT        DDDDDDDDDDDDD    RRR         RRR     VVV       RRR         RRR
      TTT              TTT        DDDDDDDDDDDD     RRR         RRR     VVV       RRR         RRR
      TTT              TTT        DDDDDDDDDDDDD    RRR         RRR     VVV       RRR         RRR
      TTT              TTT        DDDDDDDDDDDD     RRR         RRR     VVV       RRR         RRR
```

```
Va
--
00
00
00
00
00
00
00
48
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
```

```
TTTTTTTTTT  TTTTTTTTTT  YY      YY    SSSSSSSS  YY      YY  NN      NN    CCCCCCCC  HH      HH
TTTTTTTTTT  TTTTTTTTTT  YY      YY    SSSSSSSS  YY      YY  NN      NN    CCCCCCCC  HH      HH
    TT          TT      YY      YY  SS          YY      YY  NN      NN  CC          HH      HH
    TT          TT      YY      YY  SS          YY      YY  NN      NN  CC          HH      HH
    TT          TT        YY  YY    SS            YY  YY    NNNN    NN  CC          HH      HH
    TT          TT        YY  YY    SS            YY  YY    NNNN    NN  CC          HH      HH
    TT          TT          YY        SSSSSS        YY      NN  NN  NN  CC          HHHHHHHHHH
    TT          TT          YY        SSSSSS        YY      NN  NN  NN  CC          HHHHHHHHHH
    TT          TT          YY              SS      YY      NN    NNNN  CC          HH      HH
    TT          TT          YY              SS      YY      NN    NNNN  CC          HH      HH
    TT          TT          YY              SS      YY      NN      NN  CC          HH      HH
    TT          TT          YY              SS      YY      NN      NN  CC          HH      HH
    TT          TT          YY      SSSSSSSS        YY      NN      NN    CCCCCCCC  HH      HH
    TT          TT          YY      SSSSSSSS        YY      NN      NN    CCCCCCCC  HH      HH

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     1          .TITLE  TTYSYNCH - THIS MODULE CONTAINS SYNCHRONIZATION ROUTINES FOR TTDRVR
0000     2          .IDENT  'V04-001'
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28 ;
0000    29 ; Author:
0000    30 ;
0000    31 ;       Michael I. Rosenblum 10/7/83
0000    32 ;
0000    33 ; Description:
0000    34 ;
0000    35 ;       THIS MODULE PROVIDES THE SET OF ROUTINES NECESSARY TO IMPLIMENT
0000    36 ; A SYNCHRONIZATION TECHNEQUE BASED ON IPL FOR THE VMS TERMINAL DRIVER
0000    37 ;
0000    38 ;--
0000    39 ;
0000    40 ; Edit History
0000    41 ;
0000    42 ;       V04-001 MIR1100         Michael I. Rosenblum            7-Sep-1984
0000    43 ;               Fix bug where R0 would be set to 1 if the ipl was already
0000    44 ;               below IPL$_SYNCH.
0000    45 ;               Fix bug that would let the ACB allocation routine run off
0000    46 ;               The end of the pool. Allocate more pool.
0000    47 ;
0000    48 ;       V03-001 MIR0320         Michael I. Rosenblum            17-Feb-1984
0000    49 ;               Fix bug in the loading of R2 by ttylock we must make sure
0000    50 ;               That we are in physical context.
0000    51 ;
```

```
                     0000     53          $FKBDEF
                     0000     54          $ACBDEF
                     0000     55          $UCBDEF
                     0000     56          $TTYDEFS
                     0000     57          $TTYDEF
                     0000     58          $TTDEF
                     0000     59          $TT2DEF
                     0000     60          $IPLDEF
                     0000     61
                 00000000     62          .PSECT  $$$115_DRIVER,LONG
                     0000     63
                     0000     64          .sbttl  TTY$SYNCH - common routine that provides necessary synchronization
                     0000     65 ;++
                     0000     66 ; EXE$FORK - replaces the functionality of EXE$FORK
                     0000     67 ;
                     0000     68 ; Description:
                     0000     69 ;
                     0000     70 ;       This routine is the basis of synchronization for the terminal class
                     0000     71 ; driver.  For normal controlers  this is implimented by a normal fork call
                     0000     72 ; for controlers that whish to run in process context specail kernal AST's
                     0000     73 ; may be used, in user mode the lock manager may be used.
                     0000     74 ;
                     0000     75 ;
                     0000     76 ; INPUTS:
                     0000     77 ;
                     0000     78 ;       00(SP) = RETURN ADDRESS OF CALLER.
                     0000     79 ;       04(SP) = PROCESS PID TO USE FOR THIS FORK
                     0000     80 ;       08(SP) = RETURN ADDRESS OF CALLER'S CALLER.
                     0000     81 ;
                     0000     82 ;       R5 = ADDRESS OF FORK BLOCK.
                     0000     83 ;
                     0000     84 ; OUTPUTS:
                     0000     85 ;
                     0000     86 ;       ***TBS***
                     0000     87 ;-
                     0000     88 TTY$SYNCH::                                  ;CREATE FORK PROCESS
                     0000     89
          02   0B A5   91     0000     90          CMPB    FKB$B_FIPL(R5),#2       ; ARE WE FORKING TO IPL2?
                 09   13     0004     91          BEQL    DEL_SPKNL               ; YES THEN DELIVER THINGS DIFFERENTLY
             6E   8E   D0     0006     92          MOVL    (SP)+,(SP)              ; NORMAL FORK THEN REMOVE PID FROM STACK
       00000000'GF   17     0009     93          JMP     g^EXE$FORK              ; THEN FORK
                     000F     94 DEL_SPKNL:
          10 A5   53   7D     000F     95          MOVQ    R3,FKB$L_FR3(R5)        ; SAVE FORK R3 AND R4
                0C A5 8ED0   0013     96          POPL    FKB$L_FPC(R5)           ; GET A FORK PC
                   53 8ED0   0017     97          POPL    R3                      ; GET PID
                   07   12     001A     98          BNEQ    5$                      ; IS THERE A PID
          53   00010001 8F   D0     001C     99          MOVL    #^X10001,R3             ; NO THEN USE THE SWAPPER
                     0023    100 5$:
00000B4D'EF   00000B2D'EF   DE     0023    101          MOVAL   TTY$FORK_POOL_END-TTY$K_FXT_LENGTH,TTY$FORK_POOL_END
          54   000001ED'EF   9E     002E    102          MOVAB   TTY$FORK_POOL,R4        ; GET THE ADDRESS OF THE FORK POOL
           14 64   00   E3     0035    103 10$:     BBCS    #0,(R4),20$             ; SET THE BUSY BIT  AND EXIT WITH THIS ADDRE
54 00000020'8F   00000B4D'EF   F1     0039    104          ACBL    TTY$FORK_POOL_END,#TTY$K_FXT_LENGTH,R4,10$; MOVE TO THE NEXT BLOCK
                         FFEE   0045
          FFFFFFFF'FF   D5     0047    105          TSTL    @-1
                     004D    106 20$:
             54   04   C0     004D    107          ADDL    #4,R4                   ; MOVE PAST THE LOCAL STATUS WORD
           14 A4   55   D0     0050    108          MOVL    R5,ACB$L_ASTPRM(R4)     ; KEEP THE REAL FORK BLOCK FOR LATER USE
```

```
                                55    DD  0054   109          PUSHL   R5                              ; SAVE THIS FOR LATER
                             55 54    D0  0056   110          MOVL    R4,R5                           ; MAKE R5 THE ACB
                          0C A5 53    D0  0059   111          MOVL    R3,ACB$L_PID(R5)                ; PUT IN THE PID
                          0B A5 30    90  005D   112          MOVB    #ACB$M_PRAST!ACB$M_NODELETE,ACB$B_RMOD(R5); MAKE THIS A SPECAIL KERN
                    10 A5 000000CD'EF 9E  0061   113          MOVAB   L1,ACB$L_AST(R5)                ; GET THE AST ADDRESS
                    18 A5 000000DB'EF 9E  0069   114          MOVAB   FREE_ACB,ACB$L_KAST(R5); A PIGGY BACK AST TO DEASSIGN THE AASTBLK
                                52    DD  0071   115          PUSHL   R2                              ; SAVE R2 AS WELL AS R5
                          000000E8'EF 9F  0073   116          PUSHAB  RESTR5                          ; RESTORE R5 AFTER QUEUEING THE FORK
                       54 00000000'8F DB  0079   117          MFPR    #PR$_IPL,R4                     ; GET THE CURRENT IPL
                             53 55    D0  0080   118          MOVL    R5,R3                           ; SAVE THE ACB ADDRESS
                             08 54    D1  0083   119          CMPL    R4,#IPL$_SYNCH                  ; ARE WE AT SYNC OR BELOW?
                                34    15  0086   120          BLEQ    60$
                                          0088   121 ;
                                          0088   122 ; we need a fork block to get to IPL queueast before we can queue an ast
                                          0088   123 ;
                       54 000001ED'EF     9E  0088   124          MOVAB   TTY$FORK_POOL,R4                ; GET THE ADDRESS OF THE FORK POOL
                          14 64    00     E3  008F   125 40$:     BBCS    #0,(R4),50$                     ; SET THE BUSY BIT  AND EXIT WITH THIS ADDRE
54 00000020'8F 00000B4D'EF     F1  0093   126          ACBL    TTY$FORK_POOL_END,#TTY$K_FXT_LENGTH,R4,40$; MOVE TO THE NEXT BLOCK
                                FFEE  009F
                          FFFFFFFF'FF D5  00A1   127          TSTL    @-1
                             54    04  C0  00A7   128 50$:     ADDL    #4,R4                           ; MOVE BEAOND THE STATUS LONGWORD
                          0B A4    06  90  00AA   129          MOVB    #IPL$_QUEUEAST,FKB$B_FIPL(R4);
                             55    54  D0  00AE   130          MOVL    R4,R5                           ; SETUP THIS ADDRESS TO FORK ON
                                       00B1   131
                          00000000'GF 16  00B1   132          JSB     g^EXE$FORK                      ; CALL OURSELFS WITH A NON-IPL 2 IPL
                                       00B7   133 ;
                                       00B7   134 ; When the fork returns then queue the ast
                                       00B7   135 ;
                    00 FC A5    00     E5  00B7   136          BBCC    #0,-4(R5),60$                   ; FREE THE FORK BLOCK
                          55    53     D0  00BC   137 60$:     MOVL    R3,R5
                                52     D4  00BF   138          CLRL    R2                              ; NO PRIORITY BOOST NECESSARY
                                50     DD  00C1   139          PUSHL   R0
                       00000000'GF     16  00C3   140          JSB     G^SCH$QAST                      ; QUEUE ST
                                50 8ED0    00C9   141          POPL    R0
                                   05      00CC   142          rsb
                                           00CD   143 ;
                                           00CD   144 ; RETURN HERE AFTER THE AST IS FIRED
                                           00CD   145 ;
                                   003C    00CD   146 L1:      .WORD   ^M<R2,R3,R4,R5>                 ; SAVE SOME REGISTERS
                             55 04 AC      D0  00CF   147          MOVL    4(AP),R5                        ; RESTORE THE REAL FORK BLOCK ADDRESS
                                           00D3   148
                             53 10 A5      7D  00D3   149 30$:     MOVQ    FKB$L_FR3(R5),R3                ; THE FORK BLOCK AND REGISTERS THEN
                                           00D7   150                                                     ; RETURN
                                0C B5      16  00D7   151          JSB     @FKB$L_FPC(R5)                  ; GOTO THE ROUTINE
                                   04      00DA   152          RET                                     ; RETURN TO THE SYSTEM
                                           00DB   153 ;
                                           00DB   154 ; CALLED AS A PIGGYBACK AST TO THE NORMAL AST
                                           00DB   155 ;
                                           00DB   156 FREE_ACB:
                    00 FC A5    00     E5  00DB   157          BBCC    #0,-4(R5),30$                   ; NOW CLEAR THE BUSY BIT
55 00000B6D'EF              DE  00E0   158 30$:     MOVAL   TTY$NODELACB,R5                 ; MAKE SURE WE DO NOT DELETE THIS ACB
                                   05      00E7   159          RSB                                     ; NOW RETURN
                                           00E8   160 ;
                                           00E8   161 ; RESTORE USED REGISTERS BEFORE RETURNING TO THE HIGHER LEVEL
                                           00E8   162 ;
                                           00E8   163 RESTR5:                                         ; RESTORE R5 AFTER CALLING FORK
                                52 8ED0    00E8   164          POPL    R2                              ; RESTORE R2
```

```
                55 8ED0  00EB    165          POPL    R5
                    05   00EE    166          RSB
```

```
                                00EF   168          .SBTTL  TTY$LOCK - SETUP IPL AND REGISTERS
                                00EF   169
                                00EF   170 ;++
                                00EF   171 ; TTY$LOCK - SETUP IPL AND REGISTER CO-ROUTINE
                                00EF   172 ;
                                00EF   173 ; FUNCTIONAL DESCRIPTION:
                                00EF   174 ;
                                00EF   175 ; THIS IS A CO-ROUTINE THAT DISABLES INTERRUPTS TO THE IPL IN UCB$B_DIPL
                                00EF   176 ; AND SETS UP A POINTER TO THE UNIT STATE VECTOR.
                                00EF   177 ;
                                00EF   178 ; SUBSEQUENT RETURN CAUSES IPL TO BE RETURNED.
                                00EF   179 ;
                                00EF   180 ; INPUTS:
                                00EF   181 ;
                                00EF   182 ;       R5 = UCB ADDRESS
                                00EF   183 ;
                                00EF   184 ; OUTPUTS:
                                00EF   185 ;
                                00EF   186 ;       R1 IS DESTROYED.
                                00EF   187 ;
                                00EF   188 ;       R2 = ADDRESS OF THE UNIT STATE VECTOR
                                00EF   189 ;       R5 = UCB ADDRESS
                                00EF   190 ;--
                                00EF   191
                                00EF   192 TTY$LOCK::                               ; SETUP IPL AND REGISTERS
                     51   6E    D0   00EF   193          MOVL    (SP),R1            ; GET RETURN ADDRESS
                6E   00000000'8F   DB   00F2   194          MFPR    #PR$_IPL,(SP)      ; GET THE CURRENT IPL
                     6E   5E A5   91   00F9   195          CMPB    UCB$B_DIPL(R5),(SP)  ; If already at device IPL or
                                04   1B   00FD   196          BLEQU   10$                ; higher, branch forward.
                                00FF   197          SETIPL  UCB$B_DIPL(R5)     ; Else, raise to device IPL.
                          02    5E A5   91   0103   198 10$:     CMPB    UCB$B_DIPL(R5),#IPL$_ASTDEL; ARE WE AT ASTDEL OR GOING THERE?
                                16   13   0107   199          BEQL    20$                ; YES THEN HANDLE SPECAILLY
                       52   00A0 C5   D0   0109   200 30$:     MOVL    UCB$L_TL_PHYUCB(R5),R2  ; MAKE SURE WE ARE IN PHYSICAL CONTEXT
                       52   00B8 C2   9E   010E   201          MOVAB   UCB$Q_TT_STATE(R2),R2   ; SETUP STATE VECTOR POINTER
                                61   16   0113   202          JSB     (R1)               ; CALL CALLER BACK
                          02    5E A5   91   0115   203          CMPB    UCB$B_DIPL(R5),#IPL$_ASTDEL; ARE WE AT ASTDEL OR GOING THERE?
                                03   13   0119   204          BEQL    40$                ; YES THEN HANDLE SPECAILLY
                                011B   205          ENBINT                             ; ENABLE INTERRUPTS
                                05   011E   206 40$:     RSB                        ;
                                011F   207
                                011F   208 20$:
     00000B4D'EF   00000B2D'EF   DE   011F   209          MOVAL   TTY$FORK_POOL_END-TTY$K_FXT_LENGTH,TTY$FORK_POOL_END
                          8E    02   D1   012A   210          CMPL    #IPL$_ASTDEL,(SP)+ ; ARE WE ALREADY AT ASTDEL?
                                21   12   012D   211          BNEQ    60$                ; NO THEN WE MUST FORK
                                51   DD   012F   212          PUSHL   R1                 ; KEEP THE RETURN LOCATION
                          51    2C A5   3C   0131   213          MOVZWL  UCB$L_PID(R5),R1   ; GET THE PID INDEX
                                03   12   0135   214          BNEQ    50$
                             51   01   D0   0137   215          MOVL    #1,R1              ; NO PID USE SWAPPER
                   7E   00000000'GF   D0   013A   216 50$:     MOVL    g^sch$gl_pcbvec,-(sp)  ; get the location of the pcb
              00000000'GF   8E41   D1   0141   217          CMPL    (sp)+[R1],G^SCH$GL_CURPCB  ; ARE WE CURRENTLY ACTIVE
                                07   12   0149   218          BNEQ    70$                ; YES THEN NO FORK NECESSARY
                                51 8ED0   014B   219          POPL    R1                 ; RESTORE R1
                                B9   11   014E   220          BRB     30$
                                51   DD   0150   221 60$:     PUSHL   R1                 ; SAVE THE RETURN ADDRESS
                    51   000001ED'EF   9E   0152   222 70$:     MOVAB   TTY$FORK_POOL,R1   ; GET THE ADDRESS OF THE FORK POOL
                          14 61   00   E3   0159   223 200$:    BBCS    #0,(R1),250$       ; SET THE BUSY BIT  AND EXIT WITH THIS ADDRE
51   00000020'8F   00000B4D'EF   F1   015D   224          ACBL    TTY$FORK_POOL_END,#TTY$K_FXT_LENGTH,R1,200$; MOVE TO THE NEXT BLOCK
```

```
                                FFEE  0169
                 FFFFFFFF'FF     D5  016B   225              TSTL    @-1
                      51  04     C0  0171   226 250$:         ADDL    #4,R1                   ; MOVE BEAOND THE STATUS LONGWORD
                   10 A1  53     7D  0174   227              MOVQ    R3,FKB$L_FR3(R1)        ; SAVE R3 AND R4
                      53  51     D0  0178   228              MOVL    R1,R3                   ; KEEP THE FORK BLOCK
                          54  8ED0  017B   229              POPL    R4                      ; AND THE RETURN ADDRESS
                 000001E0'EF     9F  017E   230              PUSHAB  REST_REGS               ; RESTOR THE NECESSARY REGISTERS
                   7E   2C  A5   D0  0184   231              MOVL    UCB$L_PID(R5),-(SP)     ; PUT THE PID ON THE STACK
                           07    12  0188   232              BNEQ    80$                     ; NO PID
              6E   00010001  8F   D0  018A   233              MOVL    #^X10001,(SP)           ; THEN USE SWAPPER
                                    0191   234 80$:
              51   000001ED'EF     9E  0191   235              MOVAB   TTY$FORK_POOL,R1        ; GET THE ADDRESS OF THE FORK POOL
                   14 61     00    E3  0198   236 90$:         BBCS    #0,(R1),T00$            ; SET THE BUSY BIT  AND EXIT WITH THIS ADDRE
51   00000020'8F   00000B4D'EF     F1  019C   237              ACBL    TTY$FORK_POOL_END,#TTY$K_FXT_LENGTH,R1,90$; MOVE TO THE NEXT BLOCK
                                FFEE  01A8
                 FFFFFFFF'FF     D5  01AA   238              TSTL    @-1
                      51  04     C0  01B0   239 100$:         ADDL    #4,R1                   ; MOVE BEAOND THE STATUS LONGWORD
                   18 A1  55     D0  01B3   240              MOVL    R5,ACB$L_KAST(R1)       ; PLACE R5 IN AN UNUSED LOCATION
              0B A1   0B  A5     90  01B7   241              MOVB    UCB$B_FIPL(R5),FKB$B_FIPL(R1); SETUP THE FORK IPL
                      55  51     D0  01BC   242              MOVL    R1,R5                   ; MAKE THIS THE FORK BLOCK
                        FE3E     30  01BF   243              BSBW    TTY$SYNCH               ; AND SYNCHRONIZE
                      51  55     D0  01C2   244              MOVL    R5,R1                   ; GET THE FORK BLOCK ADDRESS
                   55   18  A1   D0  01C5   245              MOVL    ACB$L_KAST(R1),R5       ; GET THE UCB ADDRESS
              00 FC A1   00     E4  01C9   246              BBSC    #0,-4(R1),130$          ; NOW CLEAR THE BUSY BIT
                      52  53     D0  01CE   247 130$:         MOVL    R3,R2                   ; GET THE AUXILLARY BLOCK
                      51  54     D0  01D1   248              MOVL    R4,R1                   ; AND THE RETURN ADDRESS
                   53   10  A2   7D  01D4   249              MOVQ    FKB$L_FR3(R2),R3        ; RESTORE R3 AND R4
              00 FC A2   00     E4  01D8   250              BBSC    #0,-4(R2),140$          ; FREE THE AUXILLIARY BLOCK
                        FF29     31  01DD   251 140$:         BRW     30$
                                    01E0   252 REST_REGS:
                   53   10  A5   7D  01E0   253              MOVQ    FKB$L_FR3(R5),R3        ; RESTORE THE REGISTERS
                   55   18  A5   D0  01E4   254              MOVL    ACB$L_KAST(R5),R5       ; GET R5 BACK
                   53   10  A3   7D  01E8   255              MOVQ    FKB$L_FR3(R3),R3        ; RESTORE THE REGISTERS
                                05  01EC   256              RSB                             ; AND RETURN
```

```
00000020  01ED   258 TTY$K_FXT_LENGTH=ACB$K_LENGTH+4
          01ED   259 TTY$FORK_POOL:
          01ED   260 .REPEAT 75
          01ED   261         .BLKB   TTY$K_FXT_LENGTH
0000020D  01ED   262 .ENDR
          0B4D   263
          0B4D   264 TTY$FORK_POOL_END:
00000B6D  0B4D   265         .BLKB   TTY$K_FXT_LENGTH
          0B6D   266 TTY$NODELACB:
00000B78  0B6D   267         .BLKB   ACB$B_RMOD
       20 0B78   268         .BYTE   ACB$M_NODELETE
00000B8A  0B79   269         .BLKB   ACB$K_LENGTH-ACB$B_RMOD
          0B8A   270         .END
```

```
ACB$B_RMOD                      = 0000000B
ACB$K_LENGTH                    = 0000001C
ACB$L_AST                       = 00000010
ACB$L_ASTPRM                    = 00000014
ACB$L_KAST                      = 00000018
ACB$L_PID                       = 0000000C
ACB$M_NODELETE                  = 00000020
ACB$M_PKAST                     = 00000010
DEL_SPKNL                         0000000F R      02
EXE$FORK                          ********   X    02
FKB$B_FIPL                      = 0000000B
FKB$L_FPC                       = 0000000C
FKB$L_FR3                       = 00000010
FREE_ACB                          000000DB R      02
IPL$_ASTDEL                     = 00000002
IPL$_QUEUEAST                   = 00000006
IPL$_SYNCH                      = 00000008
L1                                000000CD R      02
PR$_IPL                           ********   X    02
RESTR5                            000000E8 R      02
REST_REGS                         000001E0 R      02
SCH$GL_CURPCB                     ********   X    02
SCH$GL_PCBVEC                     ********   X    02
SCH$QAST                          ********   X    02
TTY$FORK_POOL                     000001ED R      02
TTY$FORK_POOL_END                 00000B4D R      02
TTY$K_FXT_LENGTH                = 00000020
TTY$LOCK                          000000EF RG     02
TTY$NODELACB                      00000B6D R      02
TTY$SYNCH                         00000000 RG     02
UCB$B_DIPL                      = 0000005E
UCB$B_FIPL                      = 0000000B
UCB$L_PID                       = 0000002C
UCB$L_TL_PHYUCB                 = 000000A0
UCB$Q_TT_STATE                  = 000000B8
```

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
$$$115_DRIVER	00000B8A (2954.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.04	00:00:00.31
Command processing	126	00:00:00.36	00:00:01.01
Pass 1	341	00:00:07.92	00:00:16.63
Symbol table sort	0	00:00:01.30	00:00:02.98
Pass 2	68	00:00:01.41	00:00:03.14

Symbol table output	6	00:00:00.04	00:00:00.04
Psect synopsis output	1	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	574	00:00:11.10	00:00:24.14

The working set limit was 1350 pages.
64181 bytes (126 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1203 non-local and 22 local symbols.
270 source lines were read in Pass 1, producing 15 object records in Pass 2.
27 pages of virtual memory were used to define 26 macros.

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

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	17
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	22

1336 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

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