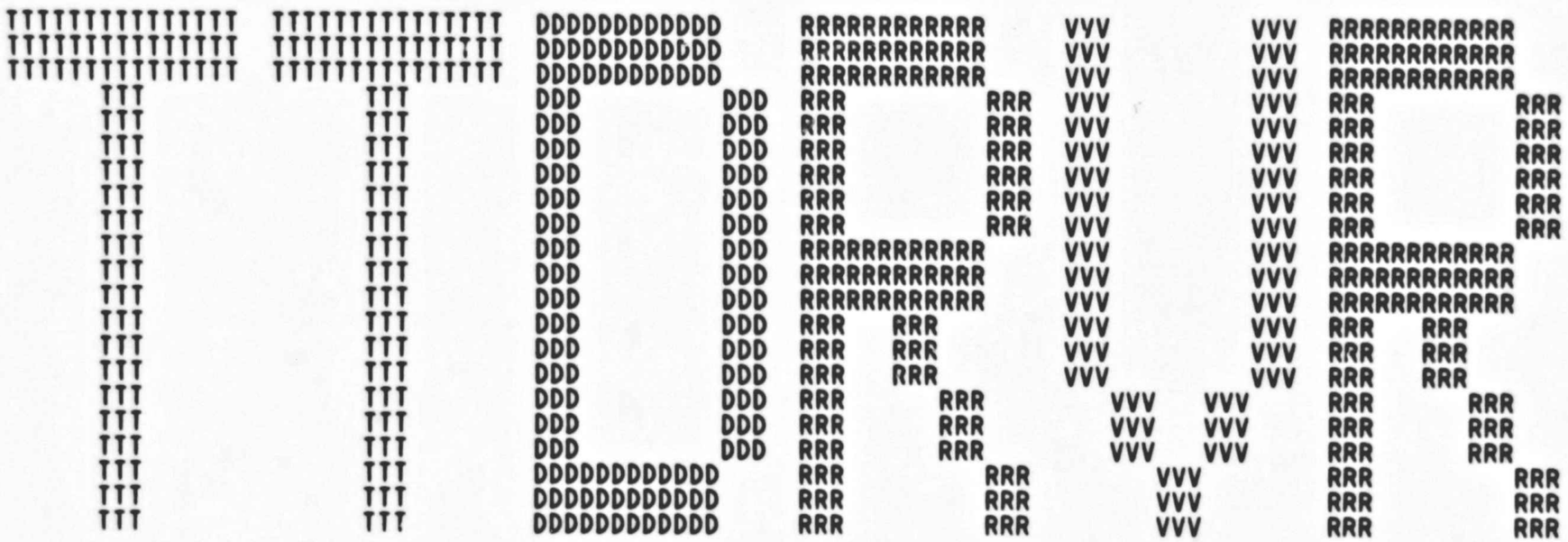
TTTTTTTTTTTTTTT TTTTTTTTTTTTTTT DDDDDDDDDDDD RRRRRRRRRRRR VVV VVV RRRRRRRRRRRR
TTTTTTTTTTTTTTT TTTTTTTTTTTTTTT DDDDDDDDDDDDD RRRRRRRRRRRR VVV VVV RRRRRRRRRRRR
TTTTTTTTTTTTTTT TTTTTTTTTTTTTTT DDDDDDDDDDDDD RRRRRRRRRRRR VVV VVV RRRRRRRRRRRR
TTT TTT DDD DDD RRR RRR VVV VVV RRR RRR
TTT TTT DDD DDD RRR RRR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

```
TTTTTTTTTT  TTTTTTTTTT  YY      YY   SSSSSSSS  UU      UU  BBBBBBBB
TTTTTTTTTT  TTTTTTTTTT  YY      YY   SSSSSSSS  UU      UU  BBBBBBBB
    TT          TT      YY      YY  SS         UU      UU  BB      BB
    TT          TT      YY      YY  SS         UU      UU  BB      BB
    TT          TT        YY  YY    SS         UU      UU  BB      BB
    TT          TT        YY  YY    SS         UU      UU  BB      BB
    TT          TT          YY        SSSSSS   UU      UU  BBBBBBBB
    TT          TT          YY        SSSSSS   UU      UU  BBBBBBBB
    TT          TT          YY              SS UU      UU  BB      BB
    TT          TT          YY              SS UU      UU  BB      BB
    TT          TT          YY              SS UU      UU  BB      BB
    TT          TT          YY              SS UU      UU  BB      BB
    TT          TT          YY       SSSSSSSS  UUUUUUUUUU  BBBBBBBB
    TT          TT          YY       SSSSSSSS  UUUUUUUUUU  BBBBBBBB

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

```
                        .TITLE  TTYSUB - Terminal driver miscellaneous subroutines
;MIR0001                .IDENT  'V04-002'
-1
                        ;
                        ;*****************************************************************************
                        ;*                                                                           *
                        ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
                        ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
                        ;*  ALL RIGHTS RESERVED.                                                     *
                        ;*                                                                           *
                        ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
                        ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
                        ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
                        ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
                        ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
                        ;*  TRANSFERRED.                                                             *
                        ;*                                                                           *
                        ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
                        ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
                        ;*  CORPORATION.                                                             *
                        ;*                                                                           *
                        ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
                        ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
                        ;*                                                                           *
                        ;*****************************************************************************
                        ;
                        ;++
                        ; FACILITY:
                        ;
                        ;       VAX/VMS TERMINAL DRIVER
                        ;
                        ; ABSTRACT:
                        ;
                        ;       THIS MODULE CONTAINS ROUTINES COMMON TO ALL TERMINAL DRIVER FUNCTIONS.
                        ;
                        ; AUTHOR:
                        ;
                        ;       R.HEINEN  11-AUG-1976
                        ;
                        ; Revision history:
                        ;
;MIR0001                ;       V04-002 MIR0001         Michael I. Rosenblum    5-Nov-1984
;MIR00C1                ;               Remove code added in V04-001 it is no longer necessary
;MIR0001                ;               Fix problem with disconnect where disconnection could
;MIR0001                ;               happen when a read completion was waiting on the queue.
;MIR0001                ;               This required a fork queue length check.
                        ;
                        ;       V04-001 MIR1100         Michael I. Rosenblum    7-Sep-1984
                        ;               Add code to catch fired hangup ast's.
                        ;
                        ;       V03-057 JLV0394         Jake VanNoy             13-AUG-1984
                        ;               Change exit path in read error such that condition code is
                        ;               is correctly on return to port driver.
                        ;
                        ;       V03-056 MIR0460         Michael I. Rosenblum            07-Aug-1984
```

```
0000     53 ;
0000     54 ;               Fix bug described in QAR 915 that points out a condition
0000     55 ;               Where the DMF hardware XON/XOFF handling get's out of synch
0000     56 ;               With the terminal driver's reconing.
0000     57 ;
0000     58 ;       V03-055 MIR0455         MICHAEL I. ROSENBLUM            20-JUN-1984
0000     59 ;               Add check to catch late rtimout ast's.
0000     60 ;               make detach cause a hangup if the line is set as such.
0000     61 ;
0000     62 ;       V03-054 RKS0054         RICK SPITZ                      10-APR-1984
0000     63 ;               Show Physical terminal UCB redirected when associated
0000     64 ;               with logical TT UCB. Remove characteristics logic on rebind
0000     65 ;               as it is done in the connect startio I/O action code.
0000     66 ;
0000     67 ;       V03-053 MIR0390         Michael I. Rosenblum            04-Apr-1984
0000     68 ;               Return the offset when a readverify read terminates
0000     69 ;               Due to timeout.  Terminate a Read verify read if
0000     70 ;               it is restarted by another io also return the correct
0000     71 ;               offset.
0000     72 ;
0000     73 ;       V03-052 MIR0370         Michael I. Rosenblum            20-Mar-1984
0000     74 ;               Remove local definition of TTY$V_ST_CTSLOW.
0000     75 ;
0000     76 ;       V03-051 RKS0051         RICK SPITZ                      15-MAR-1984
0000     77 ;
0000     78 ;               Cleanup modem disconnect routines and
0000     79 ;               add login timeout logic to modem processing.
0000     80 ;               Cleanup routine headers.
0000     81 ;
0000     82 ;       V03-050 RKS0050         RICK SPITZ                      05-MAR-1984
0000     83 ;               Enhance virtual terminal timeout code to delete
0000     84 ;               a virtual terminal when specified time period expires.
0000     85 ;
0000     86 ;       V03-048 MIR0310         Michael Rosenblum               09-Feb-1984
0000     87 ;               Fix bugs.
0000     88 ;               Clear breakthru classes on last deassign.
0000     89 ;               Maintnance functions shold not always return abort.
0000     90 ;               Add entry point to Transition to not check modem in
0000     91 ;               Devdepend so that the modem can be shutdown when
0000     92 ;               modem is turned off.
0000     93 ;
0000     94 ;       V03-047 MHB0100         Mark H. Bramhall                1-Feb-1984
0000     95 ;               Add the setting/resetting of DEV$M_DET in DEVCHAR2 during
0000     96 ;               LUCB unlinks/(re-)links.
0000     97 ;
0000     98 ;       V03-046 MIR0200         Michael I. Rosenblum            15-Oct-1983
0000     99 ;               make calls to EXE$FORK call TTY$SYNCH to allow the terminal
0000    100 ;               driver to provide different synchronization methods.
0000    101 ;               Change input IPL to the RTIMOU routine to allow the
0000    102 ;               class driver to provide it's own synchronization.
0000    103 ;
0000    104 ;       V03-045 MIR0082         Michael I. Rosenblum            19-Aug-1983
0000    105 ;               Add argument in R0 to the PORT DISCONNECT routine.
0000    106 ;
0000    107 ;       V03-044 MIR1080         Michael I. Rosenblum            11-Aug-1983
0000    108 ;               remove restriction that modem state table routines must
0000    109 ;               be forward references.
```

```
0000   110 ;       V03-043 MIR0080         Michael I. Rosenblum            28-Jul-1983
0000   111 ;               Reposition the routines in the module.
0000   112 ;
0000   113 ;       V03-042 MIR0070         Michael I. Rosenblum            13-Jul-1983
0000   114 ;               Re-write the code that handles modem timers to allow
0000   115 ;               any number of lines of modem control per controler.
0000   116 ;               Make the UNLINK routine turn around reads with timers
0000   117 ;               rather than requeueing them.
0000   118 ;               change setup ucb to  not setup the ucb if there are
0000   119 ;               channels assigned.
0000   120 ;               Cause transition when called from the ports to turn
0000   121 ;               a call to INIT with channels assigned to a SHUTDOWN.
0000   122 ;               This will blow the processes away when a power fail
0000   123 ;               happens.
0000   124 ;
0000   125 ;       V03-041 MIR0051         Michael I. Rosenblum            23-Jun-1983
0000   126 ;               Add TTY$NOTIFY service.  Cause break on NON-autobaud terminals
0000   127 ;               to login.  Remove unnecessary jsb's and replace with bsbw's.
0000   128 ;
0000   129 ;       V03-040 RKS0040         RICK SPITZ                      8-JUN-1983
0000   130 ;               ADD SUPPORT FOR CLONING DETACHED TERMINAL UCB AND LOGIC
0000   131 ;               TO L!NK AND UNLINK LOGICAL AND PHYSICAL UCBs.
0000   132 ;               RESTRUCTURE CANCEL I/O CODE TO BE MORE MODULAR.
0000   133 ;               MOVE UCB$V_TT_HANGUP INTO LUCB (EXCEPT FOR CANCEL_RESET SPECIAL CASE
0000   134 ;
0000   135 ;               REMOVE STOP BROADCASTS CODE SINCE THEY ARE NOW BREAKTHRU WRITES
0000   136 ;
0000   137 ;       V03-039 MIR0050         Michael I. Rosenblum            11-May-1983
0000   138 ;               Remove code that specail cases broadcasts.
0000   139 ;               Remove TTY$STOP2
0000   140 ;
0000   141 ;       V03-038 MIR0041         Michael I. Rosenblum            29-Apr-1983
0000   142 ;               Add code to allow autobaud to work with parity, autoparity
0000   143 ;               for even parity checking.  Change autobaud code to
0000   144 ;               require a readable <CR> before invoking LOGINOUT.
0000   145 ;
0000   146 ;       V03-037 MIR0032         Michael I. Rosenblum            05-Apr-1983
0000   147 ;               Change control character echoing representation.
0000   148 ;
0000   149 ;       V03-036 MIR0031         Michael I. Rosenblum            01-Apr-1983
0000   150 ;               add code to disable the transmiting of characters with
0000   151 ;               parity errors to the user.
0000   152 ;
0000   153 ;       V03-035 RKS0035         RICK SPITZ                      14-MAR-1983
0000   154 ;               ADD SUPPORT FOR LOGICAL UCB
0000   155 ;
0000   156 ;       V03-034 MIR8026         Michael I. Rosenblum            14-Mar-1983
0000   157 ;               Fix problem with initial string offsets and orgional
0000   158 ;               cursor position
0000   159 ;
0000   160 ;       V03-033 MIR0026         Michael I. Rosenblum            01-Mar-1983
0000   161 ;               Clear out the size of the recall buffer when creating
0000   162 ;               the typeahead buffer.
0000   163 ;
0000   164 ;       V03-032 MIR0024         Michael I. Rosenblum            28-Jan-1983
0000   165 ;               Change code to reflect the new read buffer.
0000   166 ;
```

```
0000   167 ;       V03-031 MIR0022         Michael I. Rosenblum            18-Jan-1983
0000   168 ;               Move common code from port drivers into class jacket routines.
0000   169 ;               Change  references to UCB$B_ERTCNT to UCB$W_TT_UNITBIT to
0000   170 ;               allow more maintainability.
0000   171 ;
0000   172 ;       V03-030 MIR0017         Michael I. Rosenblum            05-Jan-1983
0000   173 ;               Move common powerfail code into this module, and change
0000   174 ;               the ATTENTION routine to use this common code.  Add
0000   175 ;               TTY$POWERACTION routine as a class survice to the port
0000   176 ;               drivers that will setup everything for postive action
0000   177 ;               when a powerfail occurs.
0000   178 ;
0000   179 ;       V03-029 MIR0015         Michael I. Rosenblum            20-Dec-1982
0000   180 ;               Change fork dispatching code to use state dispatch
0000   181 ;               Class entry vectors:
0000   182 ;                       CLASS_FORK - Schedules a fork interupt for the
0000   183 ;                                    port driver
0000   184 ;                       CLASS_DISCONNECT - Performs cleanup then delivers
0000   185 ;                                          DCL's hangup ast.
0000   186 ;               Add class survice:
0000   187 ;                       CLASS_MODEM_DIS - Forces modem shutdown transision
0000   188 ;               Make CLASS_DISCONNECT fork to FIPL
0000   189 ;               Add class jacket routines for all port functions.
0000   190 ;
0000   191 ;       V03-028 MIR0014         Michael I. Rosenblum            17-Dec-1982
0000   192 ;               Add CLASS_XON and CLASS_XOFF to remove some of the
0000   193 ;               code duplicated in all of the port drivers.
0000   194 ;
0000   195 ;       V03-027 MIR0013         Michael I. Rosenblum            16-Dec-1982
0000   196 ;               Fix up refferences to new ucb structure
0000   197 ;
0000   198 ;       V03-026 MIR0011         Michael I. Rosenblum            18-Nov-1982
0000   199 ;               Remove HOLDSCREEN code.
0000   200 ;               Change RESTARTIO to use new EDITREAD state and the
0000   201 ;               new MULTIECHO functionality.
0000   202 ;
0000   203 ;       V03-025 MIR0010         Michael I Rosenblum     9-Nov-1982
0000   204 ;               Moved the count of characters in the typeahead buffer
0000   205 ;               from the UCB into the typeahead structure.
0000   206 ;
0000   207 ;       V03-024 RKS0024         RICK SPITZ              8-NOV-1982
0000   208 ;               PREVENT PURGE TYPEAHEAD ROUTINE FROM DESTROYING CONTENTS
0000   209 ;               OF R4 (ADDRESS OF TYPEAHAED STRUCTURE)
0000   210 ;
0000   211 ;       V03-023 RKS0023         RICK SPITZ              04-OCT-1982
0000   212 ;               CORRECT ATTENTION ROUTINE TO CALL START OUTPUT
0000   213 ;               AFTER CALLING RESUME FOR BROADCAST. THIS FIXES A PROBLEM
0000   214 ;               WITH DMF CONTROLLERS HANGING SHUTDOWN IN MWAIT
0000   215 ;
0000   216 ;       V03-022 RKS0022         RICK SPITZ              23-SEP-1982
0000   217 ;               ABORT PORT OUTPUT IN MODEM HANGUP SERVICE ROUTINE.
0000   218 ;               THIS PREVENTS A TERMINAL IN CONTROL S STATE FROM
0000   219 ;               KEEPING A PROCESS RUNNING LOGINOUT AROUND UNTIL
0000   220 ;               NEXT CONTROL Q IS ISSUED.
0000   221 ;
0000   222 ;               MOVE LOGOUT MODEM ACTION ROUTINE CALL FROM SHUTDOWN
0000   223 ;               STATE TO SHUT1. THIS GUARANTEES THAT A TIMER INTERRUPT
```

```
0000   224 ;                   OCCURS BETWEEN MODEM TRANSITION AND DELIVERY OF ANY
0000   225 ;                   ATTENTION AST. IT IS NEEDED TO KEEP FROM
0000   226 ;                   CALLING COM$DELATTNAST
0000   227 ;                   AT DEVICE IPL (IN THE CASE OF CONTROLLERS WHICH
0000   228 ;                   INTERRUPT ON MODEM TRANSITIONS).
0000   229 ;
0000   230 ;       V03-021 KDM0002         Kathleen D. Morse       28-Jun-1982
0000   231 ;                   Added $PRDEF.
0000   232 ;
0000   233 ;
0000   234 ;--
```

```
                        .SBTTL  Declarations

;
; EXTERNAL SYMBOLS
;

        $ACBDEF                         ;
        $CANDEF                         ; DEFINE CANCEL DEFINITIONS
        $CRBDEF                         ; DEFINE CRB
        $IDBDEF                         ; DEFINE IDB OFFSETS
        $DCDEF                          ; DEFINE DEVICE CODES
        $DEVDEF                         ; DEFINE DEV STATUS AND DEVCHAR BITS
        $DPTDEF                         ; DEFINE DPT OFFSETS
        $DYNDEF                         ; DEFINE DYNAMIC REGION TYPES
        $IODEF                          ; DEFINE I/O FUNCTION CODES
        $IPLDEF                         ; DEFINE INTERRUPT PRIORITY LEVELS
        $IRPDEF                         ; DEFINE IRP
        $MSGDEF                         ; DEFINE MESSAGE TYPES
        $PCBDEF                         ; DEFINE PCB
        $PRDEF                          ; DEFINE PROCESSOR REGISTERS
        $SSDEF                          ; DEFINE SYSTEM STATUS CODES
        $UCBDEF                         ; DEFINE UCB
        $TTDEF                          ; DEFINE TERMINAL CHARACTERISTICS
        $TT2DEF                         ; DEFINE TERMINAL CHARACTERISTICS
        $TTYDEF                         ; DEFINE TERMINAL DRIVER SYMBOLS
        $VECDEF                         ; DEFINE CRB VECTOR
        $TTYMACS                        ; DEFINE TERMINAL MACROS
        $TTYDEFS                        ; DEFINE TERMINAL DEFINITIONS
        $TTYMODEM                       ; DEFINE TERMINAL MODEM SYMBOLS
        $TQEDEF                         ; TIMER QUEUE DEFINITIONS

        .PSECT  $$$115_DRIVER,QUAD      ; DEFINE NON-PAGED PSECT
```

```
                     0000   270          .SBTTL  TTY$ABORT_IO - ABORT I/O
                     0000   271 ;++
                     0000   272 ;
                     0000   273 ; TTY$ABORT_IO - Cancel I/O as a result of ABORT out-of-band.
                     0000   274 ;
                     0000   275 ; INPUTS:
                     0000   276 ;       R2 - STATE VECTOR
                     0000   277 ;       R5 - UCB
                     0000   278 ;
                     0000   279 ;--
                     0000   280
                     0000   281 TTY$ABORT_IO::
                     0000   282
        0368   30    0000   283          BSBW    TTY$PURGE_AHEAD         ; PURGE THE TYPEAHEAD
                     0003   284
51  00CC C5   9E    0003   285          MOVAB   UCB$L_TT_WFLINK(R5),R1  ; Get start of write queue.
    53   51   D0    0008   286          MOVL    R1,R3                   ; Copy it to check for end.
                     000B   287 30$:
    53   63   D0    000B   288          MOVL    TTY$L_WB_FLINK(R3),R3   ; Get next entry.
    51   53   D1    000E   289          CMPL    R3,R1                   ; At end of queue?
         13   13    0011   290          BEQL    50$                     ; Yes. Branch forward.
 54   24 A3   D0    0013   291          MOVL    TTY$L_WB_IRP(R3),R4     ; Get associated IRP address.
         F2   13    0017   292          BEQL    30$                     ; If no IRP, it's a broadcast,
                     0019   293                                          ; leave it in the queue.
 53   04 A3   D0    0019   294          MOVL    TTY$L_WB_BLINK(R3),R3   ; Get entry's backward link.
 54   00 B3   0F    001D   295          REMQUE  @TTY$L_WB_FLINK(R3),R4  ; Remove the link.
       FFDC'  30    0021   296          BSBW    TTY$WRITEPOST           ; Queue it for completion
         E5   11    0024   297          BRB     30$                     ; Try for next link.
                     0026   298
                     0026   299 50$:                                    ; Stop the current I/Os.
                     0026   300          IF_NOT_STATE -                  ; Branch forward if no read is
                     0026   301                  READ,60$                ; currently in progress.
                     002A   302          CLR_STATE -                     ; Clear escape bits.
                     002A   303                  <ESC,BADESC>
 53   78 A5   D0    0030   304          MOVL    UCB$L_SVAPTE(R5),R3     ; GET THE PACKET ADDRESS
 54   58 A5   D0    0034   305          MOVL    UCB$L_IRP(R5),R4        ; Get address of IRP.
                     0038   306          IF_STATE EOL,55$                ; IF END OF LINE THEN ALREADY SUBTRACTED
 51   3A A4   9B    003C   307          MOVZBW  IRP$L_MEDIA+2(R4),R1    ; GET THE LENGTH OF THE TERMINATOR
3C A3    51   A2    0040   308          SUBW    R1,TTY$W_RB_TXTOFF(R3)  ; AND FIXUP THE COUNT
                     0044   309 55$:
      38 A4   D4    0044   310          CLRL    IRP$L_MEDIA(R4)         ; Clear terminators.
       FFB6'  30    0047   311          BSBW    TTY$READONE             ; Complete the read.
                     004A   312
                     004A   313 60$:                                    ; Check for a write to complete.
    00DC C5   B5    004A   314          TSTW    UCB$W_TT_MULTILEN(R5)   ; Test for non-zero
         04   13    004E   315          BEQL    65$                     ; Skip if zero
                     0050   316          SET_STATE MULTI                 ; Set multiecho here because
                     0054   317                                          ;  READONE will clear it if called
                     0054   318 65$:     IF_NOT_STATE -                  ;Branch forward if no write is
                     0054   319                  WRITE,70$              ;in progress.
        09C7   30    0058   320          BSBW    TTY$ABORT               ;Abort any current output activity
        09FE   30    005B   321          BSBW    TTY$RESUME              ;Restart output
53  00D4 C5   D0    005E   322          MOVL    UCB$L_TT_WRTBUF(R5),R3  ; Get address of write packet.
      7C A5   B0    0063   323          MOVW    UCB$W_BOFF(R5),-        ; Put status code in the
      28 A3          0066   324                  TTY$W_WB_STATUS(R3)     ; packet.
       FF95'  30    0068   325          BSBW    TTY$WRITEDONE           ; Complete the I/O.
                     006B   326
```

```
                                I 7
     006B   327 70$:
 05  006B   328         RSB                 ; And echo the control char.
```

```
                        006C   330          .SBTTL  TTY$AUTOBAUD -  SENSE AND SET BAUD RATE
                        006C   331
                        006C   332 ;++
                        006C   333 ; TTY$AUTOBAUD - Sense and Set baud rate
                        006C   334 ;
                        006C   335 ; Functional description:
                        006C   336 ;
                        006C   337 ; This routine is called from the TTYCHARI module when a job controller
                        006C   338 ; character has been detected. It assumes the character typed was a carriage
                        006C   339 ; return. The default sampling speed is 9600. If the terminal is set at baud
                        006C   340 ; rates 1200 - 9600, the algorithm will detect the proper baud rate on one
                        006C   341 ; character and set to it. If the baud rate is other than that, READERROR will
                        006C   342 ; detect a framing error and set the sampling rate to 600. A second carriage
                        006C   343 ; return then sampled at 600 baud can map the 50-600 baud range. Using the 9600
                        006C   344 ; or 600 baud table the algorithm matches the character read to a specific baud
                        006C   345 ; rate. No split speed detection is allowed.
                        006C   346 ;
                        006C   347 ; Inputs:
                        006C   348 ;
                        006C   349 ;       IPL = Device IPL
                        006C   350 ;       R5 = UCB address
                        006C   351 ;
                        006C   352 ; Outputs:
                        006C   353 ;
                        006C   354 ;       Baud rate can change.
                        006C   355 ;       R3 will be changed to <CR>, or a space
                        006C   356 ;       R4 is destroyed.
                        006C   357 ;       R5 is preserved.
                        006C   358 ;--
                        006C   359
                        006C   360 TTY$AUTOBAUD::
                        006C   361
      00B0 C5   D1      006C   362          CMPL    UCB$L_TT_RDUE(R5),-
00000000'GF             0070   363                  G^EXE$GL_ABSTIM             ; compare to current
            4E   13     0075   364          BEQL    50$                         ; branch if too soon
                        0077   365
   00 53    07   E5     0077   366          BBCC    #7,R3,10$                   ; Clear high order bit
                        007B   367
                        007B   368 ; check baud rate
                        007B   369
00000000'EF   9E        007B   370 10$:     MOVAB   TTY$AB_9600,R4              ; set 9600 baud table
                  54    0081
00F4 C5   0F   91       0082   371          CMPB    #TT$C_BAUD_9600,UCB$W_TT_SPEED(R5) ; speed at 9600 ?
            16   13     0087   372          BEQL    20$                         ; branch if yes
                        0089   373
00000000'EF   9E        0089   374          MOVAB   TTY$AB_600,R4               ; set 600 baud table
                  54    008F
00F4 C5   07   91       0090   375          CMPB    #TT$C_BAUD_600,UCB$W_TT_SPEED(R5) ; speed at 600 ?
            08   13     0095   376          BEQL    20$                         ; branch if no
                        0097   377
      54    0F   9A     0097   378          MOVZBL  #TT$C_BAUD_9600,R4          ; Wrong baud baud rate for sampling
         002C   30     009A   379          BSBW    SET_SPEED                   ; Set to 9600
            26   11     009D   380          BRB     50$                         ; Exit
                        009F   381
                        009F   382 ; now test character
                        009F   383
      84    53   91     009F   384 20$:     CMPB    R3,(R4)+                    ; input same as table ?
```

```
                08   13   00A2    385          BEQL    30$                     ; branch to set speed if yes
                54   D6   00A4    386          INCL    R4                      ; increment past baud rate
                64   95   00A6    387          TSTB    (R4)                    ; end of list
                F5   18   00A8    388          BGEQ    20$                     ; loop if >= 0
                19   11   00AA    389          BRB     50$                     ; no match
                          00AC    390
                          00AC    391 ; Set speed, zero speed+1 to indicate not split speed
                          00AC    392
              54    64   9B   00AC    393 30$:     MOVZBW  (R4),R4                 ; set baud rate
                    18   10   00AF    394          BSBB    SET_SPEED               ; Set line speed
                          00B1    395
            64 A5   01   AA   00B1    396          BICW    #UCB$M_TIM,UCB$W_STS(R5) ; Clear timeout expected
                          00B5    397          CLR_STATE <AUTOP>               ; Clear AUTOBAUD pending
              53    0D   9A   00B9    398          MOVZBL  #TTY$C_CR,R3            ; Set carriage return as input
        00000000'GF    D0   00BC    399          MOVL    G^EXE$GL_ABSTIM,-
             00B0 C5         00C2    400                  UCB$L_TT_RDUE(R5)       ; Set time this path taken
                          00C5    401
                          00C5    402 40$:
                          00C5    403 ;
                          00C5    404 ; No appropriate mapping was found, make sure that character isn't a terminator
                          00C5    405 ;
              53    20   9A   00C5    406 50$:     MOVZBL  #TTY$C_BLANK,R3         ; Invalidate character
                    05   00C8    407          RSB
                          00C9    408
                          00C9    409 SET_SPEED:
        00F4 C5    54   9B   00C9    410          MOVZBW  R4,UCB$W_TT_SPEED(R5)   ; set speed, zero out speed+1
                 09A4   30   00CE    411          BSBW    TTY$SET_LINE            ; Set line speed
                    05   00D1    412          RSB                             ; Return
```

```
                     00D2   414          .SBTTL  TTY$CANCELIO - CANCEL I/O ON TERMINAL
                     00D2   415
                     00D2   416 ;++
                     00D2   417 ; TTY$CANCELIO - CANCEL I/O, DEASSIGN, AND DEALLOCATE
                     00D2   418 ;
                     00D2   419 ; FUNCTIONAL DESCRIPTION:
                     00D2   420 ;
                     00D2   421 ; THIS ROUTINE IS ENTERED TO STOP CURRENT I/O  ON A TERMINAL UNIT.
                     00D2   422 ;
                     00D2   423 ; THE ASSOCIATED CHANNEL'S READ OR WRITE I/O IS CANCELED ALONG WITH
                     00D2   424 ; A FLUSH OF CONTROL-C AST REQUESTS.
                     00D2   425 ; IF THIS OPERATION IS DONE ON A UNIT WITH A ZERO REFERENCE COUNT, THE CONTROL
                     00D2   426 ; Y ASTS ARE FLUSHED, THE TYPEAHEAD BUFFER IS DEALLOCATED AND THE UNIT
                     00D2   427 ; IS INITIALIZED FOR THE NEXT USER.
                     00D2   428 ;
                     00D2   429 ; This routine always clears the control-O state bit, and also clears
                     00D2   430 ; the control-S state bit if the UCB reference count is zero.
                     00D2   431 ;
                     00D2   432 ; INPUTS:
                     00D2   433 ;
                     00D2   434 ;       R2 = NEGATIVE OF THE CHANNEL NUMBER
                     00D2   435 ;       R3 = CURRENT I/O PACKET ADDRESS
                     00D2   436 ;       R4 = PCB OF CANCELING PROCESS
                     00D2   437 ;       R5 = UCB ADDRESS
                     00D2   438 ;
                     00D2   439 ; OUTPUTS:
                     00D2   440 ;
                     00D2   441 ;       R4,R5 ARE PRESERVED
                     00D2   442 ;--
                     00D2   443
                     00D2   444 TTY$CANCELIO::                          ; CANCELIO TTY USAGE
          53    52   D0   00D2   445          MOVL    R2,R3               ; SAVE THE CHANNEL NUMBER
             FF28'   30   00D5   446          BSBW    TTY$LOCK            ; SETUP IPL AND REGISTERS
                     00D8   447
          02D0 8F    BB   00D8   448          PUSHR   #^M<R4,R6,R7,R9>
          56    53   D0   00DC   449          MOVL    R3,R6               ; COPY CHANNEL NUMBER
    53    58 A5    D0   00DF   450          MOVL    UCB$L_IRP(R5),R3    ; GET THE CURRENT IRP
          59    55   D0   00E3   451          MOVL    R5,R9               ; SAVE LOGICAL UCB ADDRESS
 55   00A0 C5    D0   00E6   452          MOVL    UCB$L_TL_PHYUCB(R5),R5 ; GET THE PHYSICAL UCB
                03   12   00EB   453          BNEQ    5$                  ; CHECK AGAIN AFTER LOCKING THE DAT BASE
                0115 31   00ED   454          BRW     CANCEL_DETACH
 00000227'EF    9F   00F0   455 5$:      PUSHAB  CANCEL_DONE             ; EXIT TO HERE WHEN FINISHED
                     00F6   456 ;
                     00F6   457 ; Empty the write queue.
                     00F6   458 ;
                     00F6   459
 51   00CC C5    9E   00F6   460          MOVAB   UCB$L_TT_WFLINK(R5),R1 ; Get first buffer in queue.
          57    51   D0   00FB   461          MOVL    R1,R7               ; Copy buffer address.
                     00FE   462
                     00FE   463 10$:                                    ; Empty write queue.
          57    67   D0   00FE   464          MOVL    TTY$L_WB_FLINK(R7),R7 ; Get next buffer.
          51    57   D1   0101   465          CMPL    R7,R1               ; Is this the end of queue?
                3$   13   0104   466          BEQL    40$                 ; Yes. Branch past loop.
    50    24 A7    D0   0106   467          MOVL    TTY$L_WB_IRP(R7),R0 ; Get address of associated IRP.
                F2   13   010A   468          BEQL    10$                 ; Branch if broadcast buffer
                     010C   469                                         ; (leave these in queue).
                04   E0   010C   470          BBS     #IRP$V_VIRTUAL,-    ; Don't dequeue virtual I/O
```

```
                        010E   471                     IRP$W_STS(R0),10$       ; packet.
      0C A0   D1        0111   472             CMPL    IRP$L_PID(R0),-         ; Compare packet's PID with
      60 A4             0114   473                     PCB$L_PID(R4)           ; current process' PID.
         E6   12        0116   474             BNEQ    10$                     ; No match. Go to next packet.
28 A0    56   B1        0118   475             CMPW    R6,IRP$W_CHAN(R0)       ; Does the channel match?
         E0   12        011C   476             BNEQ    10$                     ; No. Go to next packet.
         00   E1        011E   477             BBC     #IRP$V_BUFIO,-          ; Branch on direct I/O.
   04 2A A0             0120   478                     IRP$W_STS(R0),20$
         02   AA        0123   479             BICW    #IRP$M_FUNC,-           ; Otherwise, clear function of
      2A A0             0125   480                     IRP$W_STS(R0)           ; buffered I/O.
                        0127   481
                        0127   482 20$:                                       ; Set up I/O completion.
    0830 8F   3C        0127   483             MOVZWL  #SS$_CANCEL,-           ; Return CANCEL error status
      7C A5             012B   484                     UCB$W_BOFF(R5)          ; code.
         54   DD        012D   485             PUSHL   R4                      ; Save R4
57    04 A7   D0        012F   486             MOVL    TTY$L_WB_BLINK(R7),R7   ; Get backward link.
54    00 B7   0F        0133   487             REMQUE  @TTY$L_WB_FLINK(R7),R4  ; Remove buffer from queue.
      FEC6'   30        0137   488             BSBW    TTY$WRITEPOST           ; QUEUE IT FOR COMPLETION
         54 8ED0        013A   489             POPL    R4                      ; Restore R4
         BF   11        013D   490             BRB     10$                     ; Get next buffer in queue.
                        013F   491
                        013F   492 ;
                        013F   493 ; See if a read is in progress. Cancel it if it was initiated by the
                        013F   494 ; canceling process.
                        013F   495 ;
                        013F   496
                        013F   497 40$:
                        013F   498
                        013F   499                                             ; Check for read in progress.
         08   E1        013F   500             BBC     #UCB$V_BSY,-            ; Branch if no read in progress.
      29 64 A9          0141   501                     UCB$W_STS(R9),50$
0C A3    60 A4   D1      0144   502             CMPL    PCB$L_PID(R4),IRP$L_PID(R3); PID MATCH?
         22   12        0149   503             BNEQ    50$                     ; Branch if no match.
   28 A3    56   B1      014B   504             CMPW    R6,IRP$W_CHAN(R3)       ; CHANNEL MATCH?
         1C   12        014F   505             BNEQ    50$                     ; Branch if no match.
                        0151   506             IF_NOT_STATE -                  ; Proceed if this is not a read.
                        0151   507                     READ,50$
                        0155   508             CLR_STATE -                     ; Clear escape bits.
                        0155   509                     <ESC,BADESC>
         10   BB        015B   510             PUSHR   #^M<R4>                 ; Save the PCB address.
   54    58 A5   D0      015D   511             MOVL    UCB$L_IRP(R5),R4        ; Get address of IRP.
         38 A4   D4      0161   512             CLRL    IRP$L_MEDIA(R4)         ; Clear terminators.
         2C      3C      0164   513             MOVZWL  #SS$_ABORT,-            ; Set up abort status code.
         7C A5           0166   514                     UCB$W_BOFF(R5)
          FE95'  30      0168   515             BSBW    TTY$READONE             ; Complete the read.
                        016B   516
                        016B   517 45$:                                       ; Remember to restore register.
         10   BA        016B   518             POPR    #^M<R4>                 ; Restore PCB address.
                        016D   519
                        016D   520 50$:                                       ; Check for a write in progress.
                        016D   521             IF_NOT_STATE -                  ; If no write request is in
                        016D   522                     WRITE,60$               ; progress, branch.
57    00D4 C5   D0      0171   523             MOVL    UCB$L_TT_WRTBUF(R5),R7  ; Get address of write buffer.
   50   24 A7   D0      0176   524             MOVL    TTY$L_WB_IRP(R7),R0     ; Get associated IRP.
         25   13        017A   525             BEQL    60$                     ; no irp then don't cancel
      0C A0   D1        017C   526             CMPL    IRP$L_PID(R0),-         ; See if packet's PID matches
      60 A4             017F   527                     PCB$L_PID(R4)           ; process' PID.
```

```
                 1E   12  0181   528          BNEQ    60$                             ; Branch if no match.
          28 A0  56   B1  0183   529          CMPW    R6,IRP$W_CHAN(R0)               ; See if channels match.
                 18   12  0187   530          BNEQ    60$                             ; No. Don't stop this write.
               0896   30  0189   531          BSBW    TTY$ABORT                       ; Abort any current output activity
                 2C   3C  018C   532          MOVZWL  #SS$_ABORT,-                    ; Set abort status in the
               7C A5          018E   533                  UCB$W_BOFF(R5)              ; UCB status word.
                 10   BB  0190   534          PUSHR   #^M<R4>                         ; Save the PCB address.
       53  00D4 C5   D0  0192   535          MOVL    UCB$L_TT_WRTBUF(R5),R3          ; Get address of write packet.
               7C A5   B0  0197   536          MOVW    UCB$W_BOFF(R5),-                ; Put status code in the
               28 A3          019A   537                  TTY$W_WB_STATUS(R3)         ; packet.
                FE61'  30  019C   538          BSBW    TTY$WRITEDONE                   ; Complete the I/O.
                 10   BA  019F   539          POPR    #^M<R4>                         ; Restore PCB address.
                          01A1   540
                          01A1   541 ;
                          01A1   542 ;       PROCESS CONTROL AND OUT OF BAND ASTS
                          01A1   543 ;
                          01A1   544
               002D   30  01A1   545 60$:     BSBW    CANCEL_AST                      ; CANCEL OUTSTANDING ASTS
                          01A4   546
                          01A4   547 ;
                          01A4   548 ; SEE IF THIS IS THE LAST DEASSIGN
                          01A4   549 ;
               5C A9  B5  01A4   550          TSTW    UCB$W_REFC(R9)                  ; REF COUNT 0?
                 10   12  01A7   551          BNEQ    65$
                          01A9   552
                          01A9   553
          64 A9  01   8A  01A9   554          BICB    #UCB$M_TIM,UCB$W_STS(R9); RESET HANGUP TIMER IN LUCB
                          01AD   555
                 3C   E4  01AD   556          BBSC    #TTY$V_SX_RECONNECT,-
          12 00B8 C5         01AF   557                  UCB$Q_TT_STATE(R5),70$
                          01B3   558
               0079   30  01B3   559          BSBW    CANCEL_MODEM                    ; SIGNAL HANGUP IF NEEDED
               009E   30  01B6   560          BSBW    CANCEL_RESET                    ; RESET PHYSICAL UCB
                          01B9   561 65$:
       52  00B8 C5   DE  01B9   562          MOVAL   UCB$Q_TT_STATE(R5),R2           ; RESTORE STATE ADDRESS
                          01BE   563          CLR_STATE       <CTRLO>                 ; Clear control-O state always.
                FE3B'  31  01C2   564          BRW     TTY$STARTOUTPUT                 ; RESTART ANY STOPPED OUTPUT
                          01C5   565
                          01C5   566
                          01C5   567 70$:
                 54   DD  01C5   568          PUSHL   R4                              ; PRESERVE R4
              54   05   D0  01C7   569          MOVL    #TTY$V_FD_LINK,R4               ; SCHEDULE LINK FORK FUNCTION
               045E   30  01CA   570          BSBW    TTY$CRE_FORK
                 54 8ED0  01CD   571          POPL    R4
                    05   01D0   572          RSB
```

```
                        01D1    574
                        01D1    575 ;
                        01D1    576 ;       CANCEL CONTROL AND OUT OF BAND ASTS
                        01D1    577 ;
                        01D1    578 ;       INPUTS: R9 - LUCB
                        01D1    579 ;
                        01D1    580
                        01D1    581 CANCEL_AST:
                        01D1    582 ;
                        01D1    583 ; RELEASE THE CONTROL C AST BLOCKS
                        01D1    584 ;
57   0094 C9    9E      01D1    585         MOVAB   UCB$L_TL_CTRLC(R9),R7   ; ADDRESS THE CONTROL C ENABLE LIST
          52    D4      01D6    586         CLRL    R2                      ; SPECIFY NULL MASK
 00000000'GF    16      01D8    587         JSB     G^COM$FLUSHATTNS        ; FLUSH THE LIST
                        01DE    588 ;
                        01DE    589 ; FLUSH OUT OF BAND ASTS
                        01DE    590 ;
57   009C C9    9E      01DE    591         MOVAB   UCB$L_TL_BANDQUE(R9),R7 ; ADDRESS THE OUT OF BAND LIST
52   0098 C9    DE      01E3    592         MOVAL   UCB$L_TL_OUTBAND(R9),R2 ; GET CURRENT SUMMARY MASK
 00000000'GF    16      01E8    593         JSB     G^COM$FLOSHCTRLS        ; DO IT
                        01EE    594 ;
                        01EE    595 ; FLUSH CONTROL Y AST QUEUE
                        01EE    596 ;
     01    58    B1      01EE    597         CMPW    R8,#CAN$C_DASSGN        ; IS THIS A DEASSIGN?
           11    12      01F1    598         BNEQ    10$                     ; NO, SKIP
57   0090 C9    9E      01F3    599         MOVAB   UCB$L_TL_CTRLY(R9),R7   ; ADDRESS LIST HEAD FOR CONTROL Y ENABLE
          52    D4      01F8    600         CLRL    R2                      ; SPECIFY NULL MASK
 00000000'GF    16      01FA    601         JSB     G^COM$FLUSHATTNS        ; FLUSH LIST
     00A4 C5    D4      0200    602         CLRL    UCB$L_TL_CTLPID(R5)     ; RESET CONTROLLING PID
                        0204    603 10$:
                 05     0204    604         RSB
                        0205    605
                        0205    606
```

```
                     0205   608
                     0205   609 ;
                     0205   610 ;       PROCESS CANCEL FOR DETACHED LOGICAL UCB.
                     0205   611 ;       THIS ROUTINE CANCELS ANY INTERNAL STRUCTURES FOR
                     0205   612 ;       A DETACHED LUCB.
                     0205   613 ;
                     0205   614 ;       INPUTS: R9 - LUCB
                     0205   615 ;
                     0205   616
                     0205   617 CANCEL_DETACH:
        55    59     D0  0205   618          MOVL    R9,R5                           ; RESTORE LOGICAL UCB ADDRESS
          FFC6     30  0208   619          BSBW    CANCEL_AST                      ; PROCESS OUTSTANDING CONTROL ASTS
        5C A5      B5  020B   620          TSTW    UCB$W_REFC(R5)                  ; CHECK FOR LAST DEASSIGN
           12      12  020E   621          BNEQ    10$
    64 A9  01      8A  0210   622          BICB    #UCB$M_TIM,UCB$W_STS(R9)        ; RESET HANGUP TIMER IN LUCB
        6C A9      D4  0214   623          CLRL    UCB$L_DUETIM(R9)                ; RESET HANGUP TIMEOUT VALUE IN LUCB
50      0084 C5    D0  0217   624          MOVL    UCB$L_PDT(R5),R0                ; CHECK FOR PENDING REBIND FUNCTION
           04      13  021C   625          BEQL    10$                             ; NONE
      0084 C0      D4  021E   626          CLRL    UCB$L_PDT(R0)                   ; SIGNAL THAT THIS UCB IS DELETED
      02D0 8F      BA  0222   627 10$:     POPR    #^M<R4,R6,R7,R9>                ; Restore registers.
                   05  0226   628          RSB                                     ; RETURN
                     0227   629
                     0227   630
```

```
                     0227   632
                     0227   633
                     0227   634 CANCEL_DONE:
55   59    D0        0227   635          MOVL    R9,R5                   ; RESTORE LOGICAL UCB ADDRESS
02D0 8F    BA        022A   636          POPR    #^M<R4,R6,R7,R9>        ; Restore registers.
           05        022E   637          RSB                             ; RETURN
                     022F   638
                     022F   639
                     022F   640
```

```
                     022F   642 ;
                     022F   643 ;       FINAL MODEM PROCESSING FOR LAST DEASSIGN
                     022F   644 ;
                     022F   645 ;
                     022F   646 ;       INPUTS: R5 - PUCB
                     022F   647 ;
                     022F   648 CANCEL_MODEM:
                     022F   649
                     022F   650 ;
                     022F   651 ; CHECK FOR HANGUP
                     022F   652 ;
          02   E1   022F   653         BBC     #TT2$V_HANGUP,-
 03 48 A5              0231   654                 UCB$L_DEVDEPND2(R5),15$ ; BRANCH IF LOGOUT/NOHANGUP
      0C92   30   0234   655         BSBW    CLASS_MODEM_DIS         ; RESET SPEED AND UCB
                     0237   656 15$:
                     0237   657
                     0237   658 ;
                     0237   659 ; CHECK FOR NEED TO REACTIVATE MODEM PROCESSING
                     0237   660 ;
                     0237   661
0E 44 A5   15   E0   0237   662         BBS     #TT$V_MODEM,UCB$L_DEVDEPEND(R5),20$     ; CURRENT STATE IS MODEM
 00C4 C5   15   E1   023C   663         BBC     #TT$V_MODEM,UCB$L_TT_DECHAR(R5),25$     ; NO CHANGE
                14   0241
      51   00   9A   0242   664         MOVZBL  #MODEM$C_INIT,R1                        ; NEED TO RESTART MODEM
      08F4   30   0245   665         BSBW    TRANSITION
        0C   11   0248   666         BRB     25$
                     024A   667
                     024A   668 20$:
 00C4 C5   15   E0   024A   669         BBS     #TT$V_MODEM,UCB$L_TT_DECHAR(R5),25$     ; NO CHANGE
                06   024F
      51   01   9A   0250   670         MOVZBL  #MODEM$C_SHUTDWN,R1                     ; SHUT DOWN MODEM
      08E6   30   0253   671         BSBW    TRANSITION
                     0256   672 25$:
                05   0256   673         RSB
                     0257   674
```

```
                     0257   676 ;
                     0257   677 ;       RESET PHYSICAL UCB AND MAKE IT AVAILABLE FOR FUTURE USE
                     0257   678 ;
                     0257   679 ;
                     0257   680 ;       INPUTS: R5 - PUCB
                     0257   681 ;
                     0257   682 CANCEL_RESET:
                     0257   683
    68 A5    08   AA 0257   684         BICW    #UCB$M_TT_HANGUP,UCB$W_DEVSTS(R5); RESET HANGUP FLAG
                     025B   685                                                 ; NOTE THIS IS CORRECTLY REFERENCED
                     025B   686                                                 ; BUT IS NORMALLY MANIPULATED IN THE
                     025B   687 ;
                     025B   688 ;       DELETE TYPEAHEAD BUFFER
                     025B   689 ;
 50  00E4 C5    D0 025B   690         MOVL    UCB$L_TT_TYPAHD(R5),R0  ; GET ADDRESS OF TYPE AHEAD
          0A    13 0260   691         BEQL    10$                     ; NONE
      00E4 C5   D4 0262   692         CLRL    UCB$L_TT_TYPAHD(R5)
00000000'GF     16 0266   693         JSB     G^COM$DRVDEALMEM        ; DEALLOCATE TYPE AHEAD BUFFER
        00FC    30 026C   694 10$:     BSBW    TTY$PURGE_AHEAD         ; PURGE TYPE AHEAD BUFFER
                     026F   695
                     026F   696 ;
                     026F   697 ; SET UP DISCONNECT SIGNAL TYPE FOR PORT
                     026F   698 ;
      7E    01   D0 026F   699         MOVL    #1,-(SP)                ; INDICATE NOHANGUP
            02   E1 0272   700         BBC     #TT2$V_HANGUP,-
    02 48 A5       0274   701                 UCB$L_DEVDEPND2(R5),15$ ; BRANCH IF LOGOUT/NOHANGUP
            6E   D4 0277   702         CLRL    (SP)                    ; ASSUME HANGUP
                     0279   703 15$:
                     0279   704                                         ; OF LAST DEASSIGN
                     0279   705 ;
                     0279   706 ; RESTORE PERM CHARACTERISTICS
                     0279   707 ;
                     0279   708
      00A8 C5    7C 0279   709         CLRQ    UCB$Q_TL_BRKTHRU(R5)    ; RESET ALL THE BREAKTHRU CLASSES
        02E8    30 027D   710         BSBW    TTY$SETUP_UCB           ; SETUP UCB FIELDS, AND RESET PUCB
                     0280   711
00000000'GF     D0 0280   712         MOVL    G^EXE$GL_ABSTIM,-
      00B0 C5       0286   713                 UCB$L_TT_RDUE(R5)       ; SET TIME TO "TURN OFF" AUTOBAUD
        07E9    30 0289   714         BSBW    TTY$SET_LINE            ; RESET SPEED AND UCB
                     028C   715
00 68 A5    00   E5 028C   716         BBCC    #UCB$V_JOB,UCB$W_DEVSTS(R5),30$; RESET JOB CONTROLLER OWNERSHIP BIT
                     0291   717 30$:
 7E   0122 C5    BO 0291   718         MOVW    UCB$W_TT_PRTCTL(R5),-(SP); SAVE THE PORT CONTROL WORD
      0040 8F    AA 0296   719         BICW    #TTY$M_PC_XOFENA,UCB$W_TT_PRTCTL(R5); MAKE SURE THAT THE CONTROL
      0122 C5       029A
                     029D   720                                         ; AND THE DRIVER AGREE ON XOFF SETTING
        07BC    30 029D   721         BSBW    TTY$RESUME              ; CONTINUE ANY PORT OUTPUT
0122 C5    8E   B0 02A0   722         MOVW    (SP)+,UCB$W_TT_PRTCTL(R5)
                     02A5   723
 52  00B8 C5    DE 02A5   724         MOVAL   UCB$Q_TT_STATE(R5),R2   ; RESTORE STATE ADDRESS
                     02AA   725         CLR_STATE       <CTRLO>         ; Clear control-O state always.
                     02AE   726
          50 8ED0   02AE   727         POPL    R0                      ; GET HANGUP INFO AS ARGUMENT TO
        0781    30 02B1   728         BSBW    TTY$DISCONNECT          ; INFORM THE PORT
                05   02B4   729         RSB
                     02B5   730
                     02B5   731
```

```
                     02B5    733          .SBTTL  TTY$SETUP_READ - Set up UCB and states for a read
                     02B5    734
                     02B5    735 ;++
                     02B5    736 ; TTY$SETUP_READ - get ready to initiate a read operation
                     02B5    737 ;
                     02B5    738 ; Functional description:
                     02B5    739 ;
                     02B5    740 ;       This routine copies IRP fields into the UCB, disables mailbox,
                     02B5    741 ;       sets up a timed-out read, cancels control-O, purges the type
                     02B5    742 ;       ahead buffer, etc. as instructed by the IRP.
                     02B5    743 ;
                     02B5    744 ;       This routine is called by both TTY$STARTIO to start a read, and
                     02B5    745 ;       TTY$RESTARTIO to restart a read after a powerfailure or after
                     02B5    746 ;       a write completes.
                     02B5    747 ;
                     02B5    748 ; Inputs:
                     02B5    749 ;
                     02B5    750 ;       R2      - address of the UCB state bits
                     02B5    751 ;       R3      - address of the IRP
                     02B5    752 ;       R5      - address of the UCB
                     02B5    753 ;
                     02B5    754 ; Outputs:
                     02B5    755 ;
                     02B5    756 ;       R0, R2, R3, and R5 must be preserved.
                     02B5    757 ;
                     02B5    758 ;       R1,R4 destroyed.
                     02B5    759 ;
                     02B5    760 ;--
                     02B5    761
                     02B5    762 TTY$SETUP_READ::
54   2C A3    D0     02B5    763          MOVL    IRP$L_SVAPTE(R3),R4     ; Get buffered block address.
  00FC C5    B0     02B9    764          MOVW    UCB$W_TT_CURSOR(R5),-   ; Save the current cursor
    3A A4           02BD    765                  TTY$W_RB_CPZORG(R4)     ; position.
                     02BF    766
                     02BF    767 ;
                     02BF    768 ; Check to see if mailbox should be disabled.
                     02BF    769 ;
                     02BF    770
0D   4A A4    91     02BF    771          CMPB    TTY$A_RB_PRM(R4),#TTY$C_CR; IS THE FIRST CHARACTER A <CR>
        03    12     02C3    772          BNEQ    5$                      ; NO THEN LEAVE CPZORG HERE
    3A A4    B4     02C5    773          CLRW    TTY$W_RB_CPZORG(R4)     ; ELSE ZERO THE CURSOR POSITION
        0A    E1     02C8    774 5$:      BBC     #IO$V_DSABLMBX,-        ; Branch forward if disable
 15 20 A3           02CA    775                  IRP$W_FUNC(R3),10$      ; mailbox not requested.
                     02CD    776
51   00C0 C5    D0     02CD    777          MOVL    UCB$L_TT_LOGUCB(R5),R1  ; GET LUCB ADDRESS
 00010000 8F    C8     02D2    778          BISL    #TT$M_MBXDSABL,UCB$L_DEVDEPEND(R5)
      44 A5           02D8
 00010000 8F    C8     02DA    779          BISL    #TT$M_MBXDSABL,UCB$L_DEVDEPEND(R1)
      44 A1           02E0
                     02E2    780 ;
                     02E2    781 ; Set up read with time out if necessary.
                     02E2    782 ;
                     02E2    783 10$:
        07    E1     02E2    784          BBC     #IO$V_TIMED,-           ; Branch forward if read with
 1D 20 A3           02E4    785                  IRP$W_FUNC(R3),30$      ; timeout not requested.
00000000'EF    D0     02E7    786          MOVL    TTY$A_MAXTIME,-         ; Assume zero second timeout
    00B0 C5         02ED    787                  UCB$L_TT_RDUE(R5)       ; (meaning just empty typeahead
```

```
                        02F0   788
                        02F0   789          MOVZWL  TTY$W_RB_TIMOS(R4),R4   ; buffer into read buffer).
                        02F0   790                                          ; Get number of seconds to wait
                        02F4   791                                          ; between each character.
                        02F4   792          BEQL    20$                     ; Branch if none.
                        02F6   793          ADDL3   R4,G^EXE$GL_ABSTIM,-    ; Calculate and store the due
                        02F8                                                ; time.
                        02FD   794                  UCB$L_TT_RDUE(R5)
                        0300   795
                        0300   796 20$:
                        0300   797          BISW    #UCB$M_TT_TIMO,-        ; Set the "this is a read with
                        0302   798                  UCB$W_DEVSTS(R5)        ; timeout" bit.
                        0304   799
                        0304   800 ;
                        0304   801 ; Always turn off control-O state.
                        0304   802 ;
                        0304   803
                        0304   804 30$:
                        0304   805          CLR_STATE -                     ; Always clear control-O state.
                        0308   806                  <CTRLO>
                        0308   807
                        0308   808 ;
                        0308   809 ; Handle a readsync terminal.
                        0308   810 ;
                        0308   811
                        030A   812          BBC     #TT$V_READSYNC,-        ; Branch forward if terminal is
                        030D   813                  UCB$L_DEVDEPEND(R5),40$ ; not in readsync mode.
                        030D   814          IF_STATE -                      ; If in EOL state, don't issue
                        0319   815                  <TYPFUL,EOL>,40$        ; an XON code.
                        031C   816          BSBW    TTY$XON                 ; Send XON
                        031C   817
                        031C   818 40$:
                        031C   819          RSB                             ; Return to caller.
```

```
                     031D   820          .SBTTL  TTY$NOTIFY - NOTIFY JOB CONTROLER OF LOGIN IF NECESSARY
                     031D   821 ;++
                     031D   822 ; TTY$NOTIFY - NOTIFY THE JOB CONTROLER OF LOGIN PENDING IF NECESSARY
                     031D   823 ;
                     031D   824 ; FUCNTIONAL DESCRIPTION:
                     031D   825 ;
                     031D   826 ; THIS ROUTINE WILL VALIDATE A REQUEST TO NOTIFY THE JOB CONTROLLER
                     031D   827 ; OF A LOGIN REQUEST. IF THERE IS A TYPEAHEAD BUFFER, THE JOB CONTROLER
                     031D   828 ; HAS NOT ALREADY BEEN NOTIFIED AND TERMINALS ARE ENABLED A FORK IS
                     031D   829 ; QUEUED TO PERFORM THE FUNCTION.
                     031D   830 ;
                     031D   831 ; INPUTS:
                     031D   832 ;
                     031D   833 ;       R5 = PHYSICAL UCB ADDRESS
                     031D   834 ;
                     031D   835 ; OUTPUTS:
                     031D   836 ;
                     031D   837 ;       ALL REGISTERS SAVED
                     031D   838 ;--
                     031D   839 TTY$NOTIFY::
              50     DD 031D   840          PUSHL   R0                      ; SAVE A SCRATCH REGISTER
  43 44 A5    02     E0 031F   841          BBS     #TT$V_NOTYPEAHD,UCB$L_DEVDEPEND(R5),100$; IF SLAVED TERMINAL, IGNORE
   50   00C0   C5     D0 0324   842          MOVL    UCB$L_TT_LOGUCB(R5),R0  ; GET LOGICAL UCB ADDRESS
         5C   A0     B5 0329   843          TSTW    UCB$W_REFC(R0)          ; UNIT REF COUNT 0?
         11     13     13 032C   844          BEQL    10$                     ; IF EQL THEN JOB CONTROLLER POSSIBILITY
         60   A0     D5 032E   845          TSTL    UCB$L_AMB(R0)           ; USER ASSOCIATED MAILBOX?
         34     13     13 0331   846          BEQL    100$                    ; IF EQL THEN NO
  2F 44 A5    1C     E0 0333   847          BBS     #TT$V_MBXDSABL,UCB$L_DEVDEPEND(R5),100$; BR IF NOT ENABLED
  2A 68 A5    02     E0 0338   848          BBS     #UCB$V_TT_NOTIF,UCB$W_DEVSTS(R5),100$; BR IF ALREADY NOTIFIED
         1E     11     11 033D   849          BRB     15$                     ; CONTINUE IN COMMON
  23 68 A5    00     E0 033F   850 10$:     BBS     #UCB$V_JOB,UCB$W_DEVSTS(R5),100$; IF ALREADY NOTIFIED THEN GO
  1E 38 A5    06     E0 0344   851          BBS     #DEV$V_SPL,UCB$L_DEVCHAR(R5),100$;IGNORE IF DEVICE IS SPOOLED
    00000000'GF     9E 0349   852          MOVAB   G^OPA$UCB0,R0           ; ADDRESS CONSOLE UCB
              50     034F
         55   50     D1 0350   853          CMPL    R0,R5                   ; IS THIS THE CONSOLE
              08     13 0353   854          BEQLU   15$                     ; IF EQL THEN SKIP TERMINAL ENABLED TEST
  00000068'GF     B5 0355   855          TSTW    G^SYS$GL_JOBCTLMB+UCB$W_DEVSTS; TERMINALS ENABLED FOR JOBCTLR?
              0A     15 035B   856          BLEQ    100$                    ; IF LEQ THEN DISMISS
              18     BB 035D   857 15$:     PUSHR   #^M<R3,R4>              ; SAVE THESE REGISTERS OVER THE FORK
         54   00     D0 035F   858          MOVL    #TTY$V_FD_UNSOL,R4      ; ASK FOR UNSOLICITED DATA FORK
            02C6     30 0362   859          BSBW    TTY$CRE_FORK            ; FORK TO SEND MESSAGE
              18     BA 0365   860 20$:     POPR    #^M<R3,R4>
                     0367   861
              50 8ED0   0367   862 100$:    POPL    R0
                 05     036A   863          RSB
```

```
                        036B    865          .SBTTL TTY$PURGE_AHEAD - PURGE TYPEAHEAD BUFFER
                        036B    866
                        036B    867 ;++
                        036B    868 ; TTY$PURGE_AHEAD - PURGE THE CONTENTS OF THE TYPEAHEAD BUFFER
                        036B    869 ;
                        036B    870 ; FUNCTIONAL DESCRIPTION:
                        036B    871 ;
                        036B    872 ; THIS ROUTINE RESETS THE TYPEAHEAD BUFFER POINTERS TO REFLECT NO
                        036B    873 ; DATA PRESENT.
                        036B    874 ;
                        036B    875 ; INPUTS:
                        036B    876 ;
                        036B    877 ;       R2 = ADDRESS OF THE UNIT STATE VECTOR
                        036B    878 ;       R5 = UCB ADDRESS
                        036B    879 ;
                        036B    880 ; OUTPUTS:
                        036B    881 ;
                        036B    882 ;       R4 = ADDRESS OF TYPEAHEAD BUFFER STRUCTURE
                        036B    883 ;       R2,R3,R5 ARE PRESERVED.
                        036B    884 ;--
                        036B    885
                        036B    886 TTY$PURGE_AHEAD::                           ; PURGE TYPEAHEAD
 54    00E4 C5    D0    036B    887          MOVL    UCB$L_TT_TYPAHD(R5),R4     ; GET TYPE AHEAD BUFFER ADDRESS
              1A    13    0370    888          BEQL    55$                        ; IF EQL THEN NONE
           0C A4    B4    0372    889          CLRW    TTY$W_TA_INAHD(R4)         ; SET NO DATA IN TYPEAHEAD BUFFER
 64    0118 C4    9E    0375    890          MOVAB   TTY$L_TA_DATA(R4),TTY$L_TA_PUT(R4); RESET PUT POINTER
       0118 C4    9E    037A    891          MOVAB   TTY$L_TA_DATA(R4),TTY$L_TA_GET(R4); RESET GET POINTER
           04 A4          037E
07 04 A2    0C    E5    0380    892 50$:     BBCC    #TTY$V_ST_TYPFUL,4(R2),55$; TYPE AHEAD FULL AND INPUT STOPPED?
                          0385    893          CLR_STATE <OVRFLO>                 ; RESET OVERFLOW CONDITION
            0746    30    0389    894          BSBW    TTY$XON                    ; SEND XON
                    05    038C    895 55$:     RSB
```

```
                     038D   897          .SBTTL  TTY$READERROR - Line error occured.
                     038D   898 ;++
                     038D   899 ; TTY$READERROR - READ ERROR OCCURED ON LINE
                     038D   900 ;
                     038D   901 ; Functional description:
                     038D   902 ;
                     038D   903 ; This routine is called from a terminal port driver to report
                     038D   904 ; a read error on a line. It completes the read with error in the
                     038D   905 ; event a read is active, or just returns if no read is active.
                     038D   906 ;
                     038D   907 ; Inputs:
                     038D   908 ;
                     038D   909 ;       IPL = Device IPL
                     038D   910 ;       R5 = UCB address
                     038D   911 ;
                     038D   912 ; Outputs:
                     038D   913 ;
                     038D   914 ;       R0,R1,R2,R3 are destroyed.
                     038D   915 ;       R4,R5 are preserved.
                     038D   916 ;--
                     038D   917          .ENABLE LSB
                     038D   918
                     038D   919 TTY$READERROR::
            11   BB  038D   920          PUSHR   #^M<R0,R4>                      ; SAVE REGISTERS
    54  01F4 8F   3C  038F   921          MOVZWL  #SS$_PARITY,R4                  ; ASSUME PARITY
    52  00B8 C5   9E  0394   922          MOVAB   UCB$B_TT_STATE(R5),R2           ; ADDRESS STATE DATA
      09 53   0C   E0  0399   923          BBS     #12,R3,10$                      ; BRANCH IF PARITY ERROR
      7E 53   0D   E0  039D   924          BBS     #13,R3,TTY$FRAMERROR            ; BRANCH IF FRAME ERROR
    54  0838 8F   3C  03A1   925          MOVZWL  #SS$_DATAOVERUN,R4              ; OVERUN ERROR
    51  00C0 C5   D0  03A6   926 10$:     MOVL    UCB$L_TT_LOGUCB(R5),R1          ; Get logical UCB  address
         5C A1   B5  03AB   927          TSTW    UCB$W_REFC(R1)                  ; Terminal in use ?
            27   12  03AE   928          BNEQ    12$                             ; Branch if yes
 22 48 A5   01   E1  03B0   929          BBC     #TT2$V_AUTOBAUD,UCB$L_DEVDEPND2(R5),12$; Check AUTOBAUD
         53   0D   91  03B5   930          CMPB    #TTY$C_CR,R3                    ; IS THIS CHARACTER A <CR>
            06   13  03B8   931          BEQL    11$                             ; YES THEN DO AUTOPARITY
          FCAF   30  03BA   932          BSBW    TTY$AUTOBAUD                    ; YES THEN JUMP THERE
          0042   31  03BD   933          BRW     15$                             ; RETURN TO THE USER
                     03C0   934
 00F8 C5    06   E5  03C0   935 11$:     BBCC    #UCB$V_TT_PARTY,UCB$B_TT_PARITY(R5),70$; CLEAR PARITY
                08  03C5
 00F8 C5    02   E0  03C6   936          BBS     #UCB$V_TT_USERFRAME,UCB$B_TT_PARITY(R5),70$; DON'T CHANGE THE FRAME
                05  03CB
                     03CC   937                                                  ; IF THE USER SPECIFYED IT.
 00F8 C5    18   88  03CC   938          BISB    #UCB$M_TT_LEN,UCB$B_TT_PARITY(R5); SET EIGHT BIT FRAME SIZE
          06A1   30  03D1   939 70$:     BSBW    TTY$SET_LINE                    ; THEN TELL THE PORT
          002B   31  03D4   940          BRW     15$                             ; NOW EXIT
                     03D7   941
                     03D7   942 12$:     IF_NOT_STATE    READ,90$                ; SKIP IF NO READ ACTIVE
 00F8 C5    01   E1  03DB   943          BBC     #UCB$V_TT_DISPARERR,UCB$B_TT_PARITY(R5),13$
                09  03E0
            54   D1  03E1   944          CMPL    R4,#SS$_PARITY                  ; IS THIS A PARITY ERROR
  000001F4 8F       03E3
            18   13  03E8   945          BEQL    15$                             ; YES THEN IGNORE IT
    7C A5   54   B0  03EA   946 13$:     MOVW    R4,UCB$W_BOFF(R5)               ; SET ERROR CODE TO RETURN
          FC0F'  30  03EE   947          BSBW    TTY$READDONE                    ; COMPLETE THE REQUEST
            0F   11  03F1   948          BRB     15$
                     03F3   949
```

```
                54   D1  03F3    950 90$:    CMPL    R4,#SS$_DATAOVERUN              ; IS THIS A DATA OVERRUN
00000838        8F       03F5
                06   12  03FA    951         BNEQ    15$                             ; NO THEN EXIT
                         03FC    952         SET_STATE <TYPFUL,OVRFLO>               ; YES THEN MAKE SURE IT
                         0402    953                                                 ; IS SIGNALED
                         0402    954 15$:
                11   BA  0402    955         POPR    #^M<R0,R4>                      ; RESTORE REGISTERS
            010B C5  94  0404    956         CLRB    UCB$B_TT_OUTYPE(R5)             ; ASSUME NO CHARACTER TO OUTPUT
                53   95  0408    957         TSTB    R3                              ; TEST FOR DATA
                05   13  040A    958         BEQL    16$                             ; BRANCH IF ZERO
010B C5         01   90  040C    959         MOVB    #1,UCB$B_TT_OUTYPE(R5)          ; SET CHARACTER VALID
                05   0411    960 16$:    RSB                                         ; RETURN TO PORT
                         0412    961
                         0412    962 18$:
54      01F4    8F   3C  0412    963         MOVZWL  #SS$_PARITY,R4                  ; SET ERROR TYPE
                06   E1  0417    964         BBC     #TT$V_PARITY,-
          E5 00F8 C5           0419    965                 UCB$B_TT_PARITY(R5),15$ ; IF NOT ENABLED, THEN DROP
                87   11  041D    966         BRB     10$                             ; DROP CHARACTER, NO ERROR
```

```
                        041F   968          .SBTTL  TTY$FRAMERROR - FRAME ERROR ON A READ
                        041F   969 ;++
                        041F   970 ; TTY$FRAMERROR - FRAME READ ERROR OCCURED ON LINE
                        041F   971 ;
                        041F   972 ; Functional description:
                        041F   973 ;
                        041F   974 ; This routine is branched to from a TTY$READERROR to report a frame
                        041F   975 ; read error on a line. It compares time against the
                        041F   976 ; last autobaud attempt, and if a "sufficient" (avg 0.5 seconds) has
                        041F   977 ; passed, the sampling baud rate is toggled between 9600 and 600.
                        041F   978 ; The toggling to 600 baud will time out in less than 3 seconds.
                        041F   979 ;
                        041F   980 ; Inputs:
                        041F   981 ;
                        041F   982 ;       R0,R4 saved on stack
                        041F   983 ;       IPL = Device IPL
                        041F   984 ;       R5 = UCB address
                        041F   985 ;
                        041F   986 ; Outputs:
                        041F   987 ;
                        041F   988 ;       Baud rate may change.
                        041F   989 ;       R4,R5 are preserved.
                        041F   990 ;--
                        041F   991 TTY$FRAMERROR:
                        041F   992
                        041F   993 ; Check if we should attempt autobaud on this line
                        041F   994
    54   00C0 C5   D0   041F   995          MOVL    UCB$L_TT_LOGUCB(R5),R4  ; Get logical UCB  address
           5C A4   B5   0424   996          TSTW    UCB$W_REFC(R4)          ; Terminal in use ?
              0F   13   0427   997          BEQL    40$                     ; Branch if yes
                        0429   998
E4 48 A5      10   E1   0429   999          BBC     #TT2$V_SECURE,UCB$L_DEVDEPND2(R5),18$   ; NOT SECURE THEN
                        042E  1000                                          ; GIVE ERROR
              53   95   042E  1001          TSTB    R3                      ; NOT NULL DATA THEN NOT A BREAK
              E0   12   0430  1002          BNEQ    18$                     ; SO GIVE AN ERROR
            0A9F   30   0432  1003          BSBW    TTY$CLASS_DISCONNECT    ; ELSE SIGNAL HANGUP STATUS
            FFCA   31   0435  1004          BRW     15$                     ; THEN RETURN NOTHING
                        0438  1005
0A 48 A5      01   E0   0438  1006 40$:     BBS     #TT2$V_AUTOBAUD,UCB$L_DEVDEPND2(R5),43$ ; Check AUTOBAUD
              53   95   043D  1007          TSTB    R3                      ; BREAK?
              D1   12   043F  1008          BNEQ    18$                     ; NO THEN GIVE ERROR
            FED9   30   0441  1009          BSBW    TTY$NOTIFY              ; LET THE JOB CONTROLER KNOW ABOUT
            FFBB   31   0444  1010          BRW     15$                     ; THE BREAK
                        0447  1011
                        0447  1012 ; check time to see that it's been "awhile".
                        0447  1013 ; note that you want to throw away multiple-character, single-keystroke
                        0447  1014 ; characters after the first. This check gives between 0 and one second.
                        0447  1015 ; It doesn't seem necessary to make it be at least one second.
                        0447  1016 43$:
    00B0 C5   D1   0447  1017          CMPL    UCB$L_TT_RDUE(R5),-
00000000'GF           044B  1018                  G^EXE$GL_ABSTIM         ; check against current
              C0   13   0450  1019          BEQL    18$                     ; exit if equal
                        0452  1020
                        0452  1021 ; make sure there is no valid data in R3, if so, leave it for TTY$AUTOBAUD and
                        0452  1022 ; don't toggle baud rate
                        0452  1023
010B C5       01   90   0452  1024          MOVB    #1,UCB$B_TT_OUTYPE(R5)  ; ASSUME GOOD DATA
```

```
                53   95  0457  1025          TSTB    R3                          ; test
                34   12  0459  1026          BNEQ    30$                         ; Branch if data there
          010B  C5   94  045B  1027          CLRB    UCB$B_TT_OUTYPE(R5)         ; BUT WE FOUND NO GOOD DATA
                                045F  1028
                                045F  1029  ; time is OK - continue by checking baud rate
                                045F  1030
                54   0F  9A  045F  1031          MOVZBL  #TT$C_BAUD_9600,R4          ; Assume set to 9600 baud
                     0F  91  0462  1032          CMPB    #TT$C_BAUD_9600,-
          00F4  C5             0464  1033                  UCB$W_TT_SPEED(R5)          ; At 9600 ?
                     1A  12  0467  1034          BNEQ    20$                         ; no, set 9600
                                0469  1035  ;
                                0469  1036  ; Toggle to 600 baud for 2 to 3 seconds
                                0469  1037  ;
                54   07  9A  0469  1038          MOVZBL  #TT$C_BAUD_600,R4           ; otherwise, set 600 baud
                                046C  1039  80$:    SET_STATE -
                                046C  1040                  <AUTOP>                     ; Set state autobaud pending
                                0470  1041          TIMSET  2
                                0483  1042
              FC43   30  0483  1043  20$:    BSBW    SET_SPEED                   ; Set line speed
  00000000'GF      D0  0486  1044          MOVL    G^EXE$GL_ABSTIM,-
          00B0  C5             048C  1045                  UCB$L_TT_RDUE(R5)           ; set time
              FF70   31  048F  1046  30$:    BRW     15$                         ; and exit via main module
                                0492  1047
                                0492  1048
                                0492  1049          .DISABLE LSB
```

```
                     0492  1051          .SBTTL  TTY$RESTARTIO - RESTART CURRENT I/O
                     0492  1052
                     0492  1053 ;
                     0492  1054 ;++
                     0492  1055 ; TTY$RESTARTIO - RESTART CURRENT I/O
                     0492  1056 ;
                     0492  1057 ; FUNCTIONAL DESCRIPTION:
                     0492  1058 ;
                     0492  1059 ;       This routine restarts an interrupted read or write request. Such
                     0492  1060 ;       requests may have been interrupted due to a powerfail or
                     0492  1061 ;       to coincident read and write requests.
                     0492  1062 ;
                     0492  1063 ; INPUTS:
                     0492  1064 ;
                     0492  1065 ;       R2 = ADDRESS OF THE UNIT STATE VECTOR
                     0492  1066 ;       R5 = UNIT UCB ADDRESS
                     0492  1067 ;
                     0492  1068 ; OUTPUTS:
                     0492  1069 ;
                     0492  1070 ;       R2,R5 ARE PRESERVED.
                     0492  1071 ;--
                     0492  1072
                     0492  1073 TTY$RESTARTIO::                             ; RESTART THE I/O
                     0492  1074          IF_NOT_STATE WRITE,10$             ; IF NOT WRITE THEN CONTINUE
                     0496  1075          IF_STATE WRTALL,30$                ; If passall mode, no special
                     049A  1076                                             ; restart is required.
54   00D4 C5    D0   049A  1077          MOVL    UCB$L_TT_WRTBUF(R5),R4     ; Get address of write buffer.
       0580     30   049F  1078          BSBW    TTY$ABORT                  ; Abort any output activity
     2A A4      B4   04A2  1079          CLRW    TTY$W_WB_BCNT(R4)          ; No. Clear output byte count.
     30 A4      9E   04A5  1080          MOVAB   TTY$L_WB_DATA(R4),-        ; And reset the start of data
     1C A4                04A8  1081                  TTY$L_WB_NEXT(R4)      ; pointer.
                05   04AA  1082          RSB                                ; Return.
                     04AB  1083
                     04AB  1084 10$:
                     04AB  1085          IF_NOT_STATE READ,30$              ; BR IF READ NOT IN PROGRESS
53     58 A5    D0   04AF  1086          MOVL    UCB$L_IRP(R5),R3           ; Get address of IRP.
                     04B3  1087          IF_STATE RDVERIFY,40$              ; NOTHING SPECIAL FOR READ VERIFY
         FDFB    30   04B7  1088          BSBW    TTY$SETUP_READ             ; Setup for the read operation.
                     04BA  1089          IF_STATE -                         ; Continue without special
                     04BA  1090                  PASALL,20$                 ; handling if PASSALL
                     04BE  1091          IF_NOT_STATE -                     ; or NOECHO ...
                     04BE  1092                  NOECHO,15$                 ;
1D 48 A5     00   E1   04C2  1093          BBC     #TT2$V_LOCALECHO,-         ; without local echo.
                     04C7  1094                  UCB$L_DEVDEPND2(R5),20$    ;
                     04C7  1095 15$:     IF_NOT_STATE -                     ; Also no special handling if
                     04C7  1096                  REFRSH,30$                 ; REFRESH state is not set.
       0001'CF   9E   04CB  1097          MOVAB   W^TTY$A_CTRLU+1,-          ; GET A <CR> TO MULTIECHO
       00D8 C5      04CF  1098                  UCB$L_TT_MULTI(R5)
00DC C5     01   B0   04D2  1099          MOVW    #1,UCB$W_TT_MULTILEN(R5)       ; AND ONLY 1 CHARACTER HERE.
                     04D7  1100          CLR_STATE -
                     04D7  1101                  <CTRLR,SKIPCRLF>
                     04DB  1102          SET_STATE -                        ; Set the control-R and
                     04DB  1103                  <EDITREAD,MULTI,SKIPLF>; send a linefeed and then control-r the read
                     04E4  1104 20$:
            0D   E0  04E4  1105          BBS     #IO$V_REFRESH,-            ; If the read specified a
       04 20 A3      04E6  1106                  IRP$W_FUNC(R3),30$         ; refresh, don't cancel it.
                     04E9  1107          CLR_STATE -                        ; Otherwise, cancel refresh
```

```
                     04E9  1108                    REFRSH                          ; state.
                     04ED  1109
                     04ED  1110 30$:
                  05 04ED  1111                    RSB                             ; CONTINUE
              02D4 8F  B0 04EE  1112 40$:          MOVW    #SS$_OPINCOMPL,UCB$W_BOFF(R5) ; Set I/O completion status
                7C A5     04F2
           54   78 A5  D0 04F4  1113               MOVL    UCB$L_SVAPTE(R5),R4     ; GET THE READ BUFFER ADDRESS
                38 A3  D4 04F8  1114               CLRL    IRP$L_MEDIA(R3)         ; ZERO ALL OTHER READ VERIFY FIELDS
                30 A4  90 04FB  1115               MOVB    TTY$W_RB_LINOFF(R4),-   ; PUT THE INDEX INTO THE IOSB
                3B A3     04FE  1116                       IRP$L_MEDIA+3(R3)
                 FAFD'    31 0500  1117               BRW     TTY$READONE             ; Go complete the read
                          0503  1118
```

```
                     0503  1120          .SBTTL  TTY$RTIMOU - READ TIMED OUT ENTRY
                     0503  1121
                     0503  1122 ;++
                     0503  1123 ; TTY$RTIMOU - Read timed out entry (from time dependent schedular)
                     0503  1124 ;
                     0503  1125 ; Functional description:
                     0503  1126 ;
                     0503  1127 ; This routine is called from the time dependent schedular via the UCB
                     0503  1128 ; offset UCB$L_TT_RTIMOU whenever a timed read times out.  It completes
                     0503  1129 ; the read with time SS$_TIMEOUT error and, then, continues terminal
                     0503  1130 ; processing.
                     0503  1131 ;
                     0503  1132 ; Inputs:
                     0503  1133 ;
                     0503  1134 ;       IPL = IPL$_POWER
                     0503  1135 ;       R5 = UCB address
                     0503  1136 ;
                     0503  1137 ; Outputs:
                     0503  1138 ;
                     0503  1139 ;       R0,R1,R2,R3,R4 are destroyed.
                     0503  1140 ;       R5 is preserved.
                     0503  1141 ;--
                     0503  1142
                     0503  1143 TTY$RTIMOU:                                 ; Read timed out entry
                     0503  1144          SETIPL  #IPL$_TIMER                ; REDUCE IPL TO BEFORE LOCKING THE DATA BASE
                FAF7' 30   0506  1145          BSBW    TTY$LOCK           ; MAKE SURE THE UCB IS LOCKED
          52 00B8 C5  9E   0509  1146          MOVAB   UCB$Q_TT_STATE(R5),R2 ; Address state vector
                     050E  1147          IF_NOT_STATE READ,20$              ; IF THE AST IS LATE (I.E. THE READ IS
                     0512  1148                                             ; NO LONGER ACTIVE) THEN EXIT.
              022C 8F  B0   0512  1149          MOVW    #SS$_TIMEOUT,UCB$W_BOFF(R5) ; Set I/O completion status
                 7C A5
                     0516
                     0518  1150          IF_NOT_STATE RDVERIFY,5$           ; NOTHING SPECIAL FOR OTHER THAN READ VERIFY
          53 58 A5   D0   051C  1151          MOVL    UCB$L_IRP(R5),R3       ; GET THE IRP ADDRESS IN R3
          54 78 A5   D0   0520  1152          MOVL    UCB$L_SVAPTE(R5),R4    ; GET THE READ BUFFER ADDRESS
             38 A3   D4   0524  1153          CLRL    IRP$L_MEDIA(R3)        ; ZERO ALL OTHER READ VERIFY FIELDS
             30 A4   90   0527  1154          MOVB    TTY$W_RB_LINOFF(R4),-  ; PUT THE INDEX INTO THE IOSB
             3B A3        052A  1155                  IRP$L_MEDIA+3(R3)
              FAD1' 30   052C  1156 5$:      BSBW    TTY$READONE            ; Go complete the read
              FACE' 31   052F  1157 10$:     BRW     TTY$STARTOUTPUT        ; then startup any waiting activity
                    05   0532  1158 20$:     RSB
                     0533  1159
```

```
                     0533  1161          .SBTTL TTY$VT_TIMEOUT - DETACHED TERMINAL TIMEOUT
                     0533  1162
                     0533  1163 ;++
                     0533  1164 ; TTY$VT_TIMEOUT - DETACHED TERMINAL TIMEOUT ROUTINE
                     0533  1165 ;
                     0533  1166 ; FUNCTIONAL DESCRIPTION:
                     0533  1167 ;
                     0533  1168 ; THIS ROUTINE IS CALLED BY THE SYSTEM TIMER SERVICE WHEN
                     0533  1169 ; A LUCB HAS BEEN DETACHED FOR A SPECIFIED PERIOD OF TIME
                     0533  1170 ; WITHOUT BEING REATTACHED. A HANGUP AST IS FIRED FOR THAT
                     0533  1171 ; LUCB TO GET THE PROCESS KILLED. ALL PENDING IO ON THE INPUT QUEUE
                     0533  1172 ; IS COMPLETED WITH SS$_HANGUP STATUS. IN THIS STATE THERE SHOULD BE
                     0533  1173 ; NO CURRENT READ OR WRITE. LAST DEASSIGN CODE IN CANCEL
                     0533  1174 ; WILL CLEAN UP THE LUCB, AND DEASSIGN WILL DELETE IT BACK TO POOL.
                     0533  1175 ;
                     0533  1176 ; INPUTS:
                     0533  1177 ;       INTERLOCKED AT DIPL
                     0533  1178 ;       R5 = LUCB ADDRESS
                     0533  1179 ;
                     0533  1180 ;--
             0002'   0533  1181          .WORD   TTY$VT_TIMEOUT-.
                     0535  1182 TTY$VT_TIMEOUT:
          56   DD   0535  1183          PUSHL   R6                      ; SAVE SCRATCH REGISTER
     56   55   D0   0537  1184          MOVL    R5,R6                   ; LUCB REQUIRED IN R6
53   0084 C6   D0   053A  1185          MOVL    UCB$L_PDT(R6),R3        ; GET ADDRESS OF ANY REBINDING PUCB
          04   13   053F  1186          BEQL    5$                      ; NONE
     0084 C3   D4   0541  1187          CLRL    UCB$L_PDT(R3)           ; CANCEL ANY PENDING REBIND OPERATION
          10   AA   0545  1188 5$:       BICW    #UCB$M_ONLINE,-         ; SET LUCB OFFLINE TO PREVENT ANY MORE IO
        64 A6        0547  1189                  UCB$W_STS(R6)
       019B  30      0549  1190          BSBW    DISC_FIREAST            ; SIGNAL HANGUP AST TO OWNER
53     4C B6   0F    054C  1191 10$:     REMQUE  @UCB$L_IOQFL(R6),R3     ; DRAIN IO QUEUE
          12   1D   0550  1192          BVS     20$                     ; NO MORE PACKETS QUEUED
     02CC 8F   3C    0552  1193          MOVZWL  #SS$_HANGUP,-           ; FILL IN COMPLETION STATUS
        38 A3        0556  1194                  IRP$L_MEDIA(R3)
2A A3   02   AA      0558  1195          BICW    #IRP$M_FUNC,IRP$W_STS(R3) ; RESET ANY BUFFERED READS
00000000'GF   16     055C  1196          JSB     G^COM$POST              ; AND COMPLETE THE IO
          E8   11    0562  1197          BRB     10$
                     0564  1198
          56 8ED0   0564  1199 20$:     POPL    R6
                05   0567  1200          RSB
                     0568  1201
                     0568  1202
```

```
                                0568  1204          .SBTTL  TTY$SETUP_UCB - SETUP TERMINAL UCB
                                0568  1205
                                0568  1206 ;++
                                0568  1207 ; TTY$SETUP_UCB - SETUP TERMINAL UCB
                                0568  1208 ;
                                0568  1209 ; FUNCTIONAL DESCRIPTION:
                                0568  1210 ;
                                0568  1211 ; THIS ROUTINE IS ENTERED DURING SYSTEM STARTUP TO SET UP TERMINAL UCBS.
                                0568  1212 ; THE UCB IS ZEROED EXCEPT FOR THE SPEED AND FILL COUNTS. THE CURSOR IS
                                0568  1213 ; SET TO 1 TO FORCE A CR-LF. THE HOLDING TANK IS INVALIDATED. THE FORK BLOCK
                                0568  1214 ; IS INITIALIZED.
                                0568  1215 ;
                                0568  1216 ; If the write queue is not yet initialized (indicated by zeroes in the
                                0568  1217 ; listhead), initialize the queue.
                                0568  1218 ;
                                0568  1219 ; Initialize the read timed out dispatch (UCB$L_TT_RTIMOU).
                                0568  1220 ;
                                0568  1221 ; INPUTS:
                                0568  1222 ;
                                0568  1223 ;       R5 = UCB ADDRESS
                                0568  1224 ;
                                0568  1225 ; OUTPUTS:
                                0568  1226 ;
                                0568  1227 ;       R5 = UCB ADDRESS
                                0568  1228 ;
                                0568  1229 ;       R0-R4 ARE PRESERVED.
                                0568  1230 ;--
                                0568  1231
                                0568  1232 TTY$SETUP_UCB::                          ; SET UP THE UCB (PORT INTERFACE)
                     50   DD    0568  1233          PUSHL   R0
           50   00C0 C5   D0    056A  1234          MOVL    UCB$L_TT_LOGUCB(R5),R0  ; GET THE LOGICAL UCB ADDRESS
                     09   13    056F  1235          BEQL    3$                      ; NONE THEN INIT THE UCB
                  5C A0   B5    0571  1236          TSTW    UCB$W_REFC(R0)          ; IS THIS UCB ACTIVE?
                     04   13    0574  1237          BEQL    3$                      ; NO THEN INIT IT
                     50 8ED0    0576  1238          POPL    R0                      ; IF ACTIVE THEN RETURN NORMALY
                        05      0579  1239          RSB
                                057A  1240 3$:
           00C0 C5      55   D0  057A  1241          MOVL    R5,UCB$L_TT_LOGUCB(R5)  ; INIT LOGICAL UCB ADDRESS
           00A0 C5      55   D0  057F  1242          MOVL    R5,UCB$L_TL_PHYUCB(R5)  ; INIT PHYSICAL UCB
-1          00040000    8F   C8  0584  1244          BISL    #DEV$M_AVL,-            ; SHOW PUCB AVALABLE FOR USE
                 38 A5          058A  1245                  UCB$L_DEVCHAR(R5)
            00000100    8F   CA  058C  1246          BICL    #DEV$M_RED,UCB$L_DEVCHAR2(R5) ; SHO PUCB NOT REDIRECTED
                 3C A5          0592
                                0594  1247
           00A4 C5      D4   D4  0594  1248          CLRL    UCB$L_TL_CTLPID(R5)     ; CLEAN OUT A LEFT OVER PID
           00FC C5      01   B0  0598  1249          MOVW    #1,UCB$W_TT_CURSOR(R5)  ; SET BAD CURSOR TO FORCE CRLF
           0108 C5           B4  059D  1250          CLRW    UCB$W_TT_HOLD(R5)       ; INIT HOLDING TANK
           064D'CF           9E  05A1  1251          MOVAB   W^TTY$FORK_ADDR,UCB$L_FPC(R5); SET UP FORK ADDRESS
                 0C A5          05A5
           00E8 C5      D0   D0  05A7  1252          MOVL    UCB$W_TT_DESPEE(R5),UCB$W_TT_SPEED(R5); SET UP TEMP CHARS
           00F4 C5               05AB
           00EC C5      90   90  05AE  1253          MOVB    UCB$B_TT_DEPARI(R5),UCB$B_TT_PARITY(R5);SET UP DEFAULT PARITY
           00F8 C5               05B2
           00F0 C5      D0   D0  05B5  1254          MOVL    UCB$B_TT_DETYPE(R5),UCB$B_DEVTYPE(R5);
                 41 A5          05B9
                                05BB  1255
            FFFFDFFF    8F   CB  05BB  1256          BICL3   #^C<TT$M_REMOTE>,UCB$L_DEVDEPEND(R5),R0; SAVE CURRENT REMOTE STATE
```

```
                50    44 A5          05C1
          00002000    8F    CB   05C4  1257          BICL3   #TT$M_REMOTE,UCB$L_TT_DECHAR(R5),UCB$L_DEVDEPEND(R5);
                00C4 C5            05CA
                   44 A5           05CD
             44 A5    50    C8   05CF  1258          BISL    R0,UCB$L_DEVDEPEND(R5)  ; KEEP PREVIOUS SETTING
          44 A5    FC 8F    8B   05D3  1259          BICB3   #^C<TT$M_PASSALL!TT$M_NOECHO>,UCB$L_DEVDEPEND(R5),R0; GET PASSALL AN
                   50              05D8
                                   05D9  1260
                                   05D9  1261 ;       Set up default DMA characteristics
                                   05D9  1262
                00CB C5    D0   05D9  1263          MOVL    UCB$L_TT_DECHA1(R5),UCB$L_DEVDEPND2(R5) ; SET UP DEFAULT CHAR WORD 2
                   48 A5           05DD
       1E 48 A5    06    E1   05DF  1264          BBC     #TT2$V_DMA,UCB$L_DEVDEPND2(R5),8$       ; SKIP IF NOT SET
        0122 C5    02    E0   05E4  1265          BBS     #TTY$V_PC_DMAAVL,UCB$W_TT_PRTCTL(R5),5$ ; DMA FEATURE AVAILABLE
                              13   05E9
          00000040 8F    CA   05EA  1266          BICL    #TT2$M_DMA,UCB$L_TT_DECHA1(R5)          ; SHOW DMA NOT AVAILABLE
                00C8 C5           05F0
          00000040 8F    CA   05F3  1267          BICL    #TT2$M_DMA,UCB$L_DEVDEPND2(R5)          ; IF DMA IS NOT AVAILABLE
                   48 A5           05F9
                      05    11   05FB  1268          BRB     8$
                                   05FD  1269 5$:
        0122 C5    02    A8   05FD  1270          BISW    #TTY$M_PC_DMAENA,UCB$W_TT_PRTCTL(R5)    ; ENABLE DMA USAGE
                                   0602  1271 8$:
                                   0602  1272
        02   02    50    F0   0602  1273          INSV    R0,#TTY$V_ST_PASALL,#2,UCB$Q_TT_STATE+4(R5); RESET CURRENT STATE
                00BC C5           0606
                      50 8ED0   0609  1274          POPL    R0
                                   060C  1275
                                   060C  1276 ;
                                   060C  1277 ; Initialize the write queue.
                                   060C  1278 ;
                                   060C  1279
        00CC C5    D5   060C  1280          TSTL    UCB$L_TT_WFLINK(R5)     ; Is the queue initialized?
                   0E   12   0610  1281          BNEQ    10$                     ; Yes. Don't redo it.
        00CC C5    9E   0612  1282          MOVAB   UCB$L_TT_WFLINK(R5),-   ; Initialize queue to point to
        00CC C5         0616  1283                  UCB$L_TT_WFLINK(R5)     ; itself in both directions.
        00CC C5    9E   0619  1284          MOVAB   UCB$L_TT_WFLINK(R5),-   ; Now the backward link.
        00D0 C5         061D  1285                  UCB$L_TT_WBLINK(R5)
                                   0620  1286
                                   0620  1287 10$:
                                   0620  1288
                                   0620  1289 ;
                                   0620  1290 ; Initialize read timed out dispatch.
                                   0620  1291 ;
                                   0620  1292
        FEDF CF    9E   0620  1293          MOVAB   TTY$RTIMOU,UCB$L_TT_RTIMOU(R5) ; Set read timed out dispatch
        00B4 C5         0624
                                   0627  1294
                      05   0627  1295          RSB                             ; Return.
```

```
                     0628  1297
                     0628  1298          .SBTTL  TTY$CLASS_FORK - PORT DRIVER FORK ROUTINE
                     0628  1299 ;++
                     0628  1300 ; TTY$CLASS_FORK
                     0628  1301 ;
                     0628  1302 ; DESCRIPTION
                     0628  1303 ;
                     0628  1304 ;          WILL CAUSE A FORK TO BE CREATED FOR A PORT DRIVER.
                     0628  1305 ; WHEN THE FORK IS CREATED THE DRIVER IS CALLED BACK AT IT'S PORT_FORKRET
                     0628  1306 ; DISPATCH VECTOR.
                     0628  1307 ;
                     0628  1308 ; INPUTS:
                     0628  1309 ;
                     0628  1310 ;          R2 = UNIT STATE VECTOR
                     0628  1311 ;          R5 = UCB ADDRESS
                     0628  1312 ;
                     0628  1313 ; OUTPUTS:
                     0628  1314 ;          R4,R3 DESTROYED
                     0628  1315 ;          R2,R5 ARE PRESERVED
                     0628  1316 ;--
                     0628  1317 TTY$CLASS_FORK::                             ; CREATE A FORK FOR THE TERMINAL PORT
54   03   D0         0628  1318          MOVL    #TTY$V_FD_PORTFORK,R4      ; SET THE PORT FORK BIT.
                     062B  1319                                             ; AND FALL INTO CREATING THE FORK
```

```
                        062B  1321          .SBTTL  TTY$CRE_FORK - General purpose driver fork routine
                        062B  1322 ;++
                        062B  1323 ; TTY$CRE_FORK - CREATE FORK ROUTINE
                        062B  1324 ;
                        062B  1325 ; FUNCTIONAL DESCRIPTION:
                        062B  1326 ;
                        062B  1327 ; THIS ROUTINE FORKS IN THE UCB FOR EITHER OF TWO REASONS:
                        062B  1328 ;
                        062B  1329 ;       1. UNSOLICITED DATA - SEND MESSAGE TO OWNER
                        062B  1330 ;       2. NO TYPEAHEAD BUFFER - ALLOCATE ONE
                        062B  1331 ;
                        062B  1332 ; INPUTS:
                        062B  1333 ;
                        062B  1334 ;       R2 = UNIT STATE VECTOR ADDRESS
                        062B  1335 ;       R3 = CHARACTER TO BUFFER FOR TYPEAHEAD NEEDED
                        062B  1336 ;       R4 = FORK REASON STATE BIT
                        062B  1337 ;       R5 = UCB ADDRESS
                        062B  1338 ;
                        062B  1339 ; OUTPUTS:
                        062B  1340 ;
                        062B  1341 ;       R3,R4 ARE DESTROYED
                        062B  1342 ;       R2,R5 ARE PRESERVED
                        062B  1343 ;--
                        062B  1344 TTY$CRE_FORK::                                   ; CREATE FORK
15 14 A5    06    E2    062B  1345          BBSS    #TTY$V_FD_BUSY,UCB$L_FR4(R5),20$; ARE WE BUSY
00 14 A5    54    E2    0630  1346          BBSS    R4,UCB$L_FR4(R5),3$     ; SET THE APROPRIATE BIT
   54    14 A5    D0    0635  1347 3$:      MOVL    UCB$L_FR4(R5),R4        ; AND GET THE BIT INTO R4
         2C A5    DD    0639  1348 5$:      PUSHL   UCB$L_PID(R5)           ; SAVE THE PID ON THE STACK
         4D'AF    9F    063C  1349          PUSHAB  B^TTY$FORK_ADDR         ; SET FORK ROUTINE ADDRESS
   00000000'EF    17    063F  1350          JMP     TTY$SYNCH               ; CREATE FORK PROCESS
                        0645  1351 ;
                        0645  1352 ; FORK ALREADY ACTIVE
                        0645  1353 ;
00 14 A5    54    E2    0645  1354 20$:     BBSS    R4,UCB$L_FR4(R5),25$    ; SET THE APROPRIATE BIT
                  05    064A  1355 25$:     RSB                             ; AND RETURN TO THE CALLER
```

```
                        064B  1357
                        064B  1358          .SBTTL  FORK DISPATCHER
                0357'   064B  1359          .WORD   ATTENTION-.                 ; SET UP ATTENTION INTERRUPT
                        064D  1360  TTY$FORK_ADDR::
                        064D  1361  ;
                        064D  1362  ; ATTENTION FORK PROCESS
                        064D  1363  ;
 52   00B8 C5    9E      064D  1364  10$:     MOVAB   UCB$Q_TT_STATE(R5),R2       ; ADDRESS UNIT STATE
                        0652  1365
                        0652  1366          DSBINT  UCB$B_DIPL(R5)              ; DISABLE INTERUPTS FOR THE CHECK
      07    00    EA      0659  1367          FFS     #0,#TTY$V_FD_BUSY+1,UCB$L_FR4(R5),R4; FIND THE FIRST BIT SET
   54    14 A5
00 14 A5   54    E4      065C  1368          BBSC    R4,UCB$L_FR4(R5),15$        ; CLEAR THAT BIT
                        065F  1369  15$:     ENBINT                              ; ENABLE INTERUPTS
                        0664  1370
        E3 AF    DF      0667  1371          PUSHAL  10$                         ; SAVE A RETURN ADDRESS
                        0667  1372          CASE    R4,TYPE=B,<-                ; AND DISPATCH TO THE NECESSARY ROUTINE
                        066A  1373          UNSOL -
                        066A  1374          GETAHD -
                        066A  1375          DISCONNECT -
                        066A  1376          PORTFORK -
                        066A  1377          UNLINKFORK -
                        066A  1378          LINKFORK -
                        066A  1379          FORKEXIT>
                        067C  1380
                        067C  1381  FORKEXIT:
           54 8ED0      067C  1382          POPL    R4                          ; POP OFF  THE ADDRESS OF THE DISPATCHER
              05        067F  1383          RSB                                 ; THEN RETURN
                        0680  1384
                        0680  1385  PORTFORK:
 51   0118 C5    D0      0680  1386          MOVL    UCB$L_TT_PORT(R5),R1        ; GET THE PORT ADDRESS
        34 B1    17      0685  1387          JMP     @PORT_FORKRET(R1)           ; AND CALL THE PORTS FORK ADDRESS
                        0688  1388
                        0688  1389
```

```
                     0688  1391          .SBTTL  FORK TO DISCONNECT THE TERMINAL AND DELIVER THE HANGUP AST
                     0688  1392 ;++
                     0688  1393 ;DISCONNECT
                     0688  1394 ;
                     0688  1395 ; Descrpition:
                     0688  1396 ;       Delivers all AST's, aborts all IO and then sends a terminal hangup
                     0688  1397 ; message to the job controler.
                     0688  1398 ;
                     0688  1399 ; Inputs:
                     0688  1400 ;       R2 = Unit state vector
                     0688  1401 ;       R5 = PUcb address
                     0688  1402 ;
                     0688  1403 ; Outputs:
                     0688  1404 ;
                     0688  1405 ;       R5 = PRESERVED
                     0688  1406 ;--
                     0688  1407 DISCONNECT:
          0040 8F    BB  0688  1408          PUSHR   #^M<R6>
     56   00C0 C5    D0  068C  1409          MOVL    UCB$L_TT_LOGUCB(R5),R6  ; GET LOGICAL UCB ADDRESS
                     0691  1410
                     0691  1411          DSBINT  UCB$B_DIPL(R5)
              0387   30  0698  1412          BSBW    TTY$ABORT               ; ABORT ANY OUTPUT,
              03BE   30  069B  1413          BSBW    TTY$RESUME              ; AND RESET CONTROL S
                     069E  1414          ENBINT
                     06A1  1415
  00 44 A5    0D    E5  06A1  1416          BBCC    #TT$V_REMOTE,UCB$L_DEVDEPEND(R5),10$    ;RESET REMOTE TERMINAL
  44 A6   44 A5     D0  06A6  1417 10$:     MOVL    UCB$L_DEVDEPEND(R5),UCB$L_DEVDEPEND(R6) ;UPDATE MODIFIED FIELD
                     06AB  1418
  08 48 A5    11    E1  06AB  1419          BBC     #TT2$V_DISCONNECT,UCB$L_DEVDEPND2(R5),15$;SKIP IF DISCONNECT NOT AVA
        56    55    D1  06B0  1420          CMPL    R5,R6                           ; MAKE SURE ITS REASONABLE
              03    13  06B3  1421          BEQL    15$                             ; NO, LUCB/PUCB THE SAME
           0028   31  06B5  1422          BRW     50$
                     06B8  1423
                     06B8  1424 15$:
           002C   30  06B8  1425          BSBW    DISC_FIREAST                    ; FIRE HANGUP AST
        02CC 8F    3C  06BB  1426          MOVZWL  #SS$_HANGUP,UCB$W_BOFF(R5)      ; SET THE RETURN STATUS
          7C A5        06BF
           0056   30  06C1  1427          BSBW    DISC_STOPIO                     ; STOP CURRENT I/O
                     06C4  1428
                     06C4  1429
        54    06    D0  06C4  1430          MOVL    #MSG$_TRMHANGUP,R4      ; SET MESSAGE TYPE
        55    56    D0  06C7  1431          MOVL    R6,R5                   ; SWITCH TO LUCB CONTEXT
     53    60 A5    D0  06CA  1432          MOVL    UCB$L_AMB(R5),R3        ; GET ASSOC MAILBOX
              06    13  06CE  1433          BEQL    20$                     ; IF EQL THEN NONE
  00000000'GF      16  06D0  1434          JSB     G^EXE$SNDEVMSG          ; INSERT MAILBOX MESSAGE
                     06D6  1435
  55    00A0 C5    D0  06D6  1436 20$:     MOVL    UCB$L_TL_PHYUCB(R5),R5  ; SWITCH BACK TO PUCB CONTEXT
        0040 8F    BA  06DB  1437          POPR    #^M<R6>
                     05  06DF  1438          RSB
                     06E0  1439
                     06E0  1440 50$:
        0040 8F    BA  06E0  1441          POPR    #^M<R6>                 ; RESTORE SAVED REGISTERS
           0161   31  06E4  1442          BRW     UNLINKFORK              ; AND DO UNLINK OPERATION
```

```
                                06E7  1444 ;
                                06E7  1445 ;       FIRE HANGUP AST
                                06E7  1446 ;
                                06E7  1447 ;       THIS ROUTINE DELIVERS ALL OUTSTANDING ^Y ASTS WITH
                                06E7  1448 ;       HANGUP STATUS TO REQUEST THE PROCESS BE RUN DOWN. IN THE
                                06E7  1449 ;       EVENT NONE ARE CURRENTLY QUEUED, THEN A FLAG IS SET TO
                                06E7  1450 ;       SIGNAL THIS CONDITION WHEN THE NEXT ^Y AST IS QUEUED.
                                06E7  1451 ;       *NOTE* THIS ROUTINE MAY NOT UTILIZE ANY FIELDS IN THE PUCB
                                06E7  1452 ;       AS IT IS ALSO USED BY DISCONNECTED LOGICAL UCB'S.
                                06E7  1453 ;
                                06E7  1454 ;       INPUTS:
                                06E7  1455 ;               R6 - LUCB
                                06E7  1456 ;       OUTPUTS:
                                06E7  1457 ;               R5,R6 PRESERVED
                                06E7  1458
                                06E7  1459 DISC_FIREAST:
                                06E7  1460              DSBINT  UCB$B_DIPL(R6)                  ; INTERLOCK TO DEVICE IPL
                          54    0090 C6   9E  06EE    1461      MOVAB   UCB$L_TL_CTRLY(R6),R4   ; ADDRESS CONTROL Y AST LIST
                          50      54   D0  06F3    1462      MOVL    R4,R0                   ; COPY LIST ADDRESS
-1                                60   D5  06F6    1464      TSTL    (R0)                    ;LIST HEAD EMPTY?
                                  09   12  06F8    1465      BNEQ    5$                      ;NO,PROCESS HANGUP ASTS
                                5C A6  B5  06FA    1466      TSTW    UCB$W_REFC(R6)          ;IS THIS UNIT OWNED OR ACTIVE?
                                       06FD    1467                                      ;(OR IS IT LOGGED OUT)
                                  04   13  06FD    1468      BEQL    5$                      ;NOT ACTIVE,SO SKIP FLAG SETTING
                                  08   A8  06FF    1469      BISW    #UCB$M_TT_HANGUP,-
                                68 A6          0701    1470              UCB$W_DEVSTS(R6)        ;SET HANGUP FLAG TO NOTIFY NEXT CTRL-Y
                                       0703    1471                                      ;ATTENTION AST WHEN IT IS QUEUED.
                          50    60   D0  0703    1472 5$:  MOVL    (R0),R0                 ; GET NEXT ENTRY
                                  08   13  0706    1473      BEQL    10$                     ; IF EQL THEN DONE
-1                        02CC  8F   3C  0708    1475      MOVZWL  #SS$_HANGUP,ACB$L_KAST+4(R0); INSERT NEW PARAM FOR AST
                            1C  A0         070C
                                F3   11  070E    1476      BRB     5$                      ; CONTINUE UNTIL DONE
                00000000'GF     16  0710    1477 10$: JSB     G^COM$DELATTNAST        ; DELIVER THE AST'S
                                       0716    1478      ENBINT
                                05     0719    1479      RSB
                                       071A    1480
                                       071A    1481
```

```
                        071A  1483
                        071A  1484 ;       ABORT CURRENT AND QUEUED READS AND WRITES
                        071A  1485 ;
                        071A  1486 ;       THIS ROUTINE IS USED TO ABORT INTERNALLY QUEUED WRITES AND
                        071A  1487 ;       ANY ACTIVE READ. IT RUNS IN THE CONTEXT OF THE PUCB.
                        071A  1488 ;
                        071A  1489 ;       INPUT:  R5 - PUCB
                        071A  1490 ;       OUPUT:  R5 - PRESERVED
                        071A  1491
                        071A  1492 DISC_STOPIO:
                        071A  1493          DSBINT  UCB$B_DIPL(R5)              ; INTERLOCK TO DEVICE IPL
52   00BB C5    9E      0721  1494          MOVAB   UCB$Q_TT_STATE(R5),R2       ; SET UP LUCB STATE POINTER
51   00CC C5    9E      0726  1495          MOVAB   UCB$L_TT_WFLINK(R5),R1      ; Get start of write queue.
     53    51   D0      072B  1496          MOVL    R1,R3                       ; Copy it to check for end.
                        072E  1497
                        072E  1498 10$:
     53    63   D0      072E  1499          MOVL    TTY$L_WB_FLINK(R3),R3       ; Get next entry.
     51    53   D1      0731  1500          CMPL    R3,R1                       ; At end of queue?
           0D   13      0734  1501          BEQL    15$                         ; Yes. Branch forward.
                        0736  1502                                              ; leave it in the queue.
53    04 A3    D0      0736  1503          MOVL    TTY$L_WB_BLINK(R3),R3       ; Get entry's backward link.
54    00 B3    0F      073A  1504          REMQUE  @TTY$L_WB_FLINK(R3),R4      ; Remove the link.
        F8BF'  30      073E  1505          BSBW    TTY$WRITEPOST               ; Queue it for completion
           EB  11      0741  1506          BRB     10$                         ; Try for next link.
                        0743  1507
                        0743  1508 15$:                                        ; Stop the current I/Os.
                        0743  1509          IF_NOT_STATE -                      ; Branch forward if no read is
                        0743  1510                  READ,20$                    ; currently in progress.
                        0747  1511          CLR_STATE -                         ; Clear escape bits.
                        0747  1512                  <ESC,BADESC>
54    58 A5    D0      074D  1513          MOVL    UCB$L_IRP(R5),R4            ; Get address of IRP.
      38 A4    D4      0751  1514          CLRL    IRP$L_MEDIA(R4)             ; Clear terminators.
        F8A9'  30      0754  1515          BSBW    TTY$READONE                 ; Complete the read.
                        0757  1516 20$:                                        ; Check for a write to complete.
                        0757  1517          IF_NOT_STATE -                      ; Branch forward if no write is
                        0757  1518                  WRITE,30$                   ; in progress.
53    00D4 C5    D0      075B  1519          MOVL    UCB$L_TT_WRTBUF(R5),R3      ; Get address of write packet.
        7C A5    B0      0760  1520          MOVW    UCB$W_BOFF(R5),-            ; Put status code in the
      28 A3             0763  1521                  TTY$W_WB_STATUS(R3)         ; packet.
        F898'  30      0765  1522          BSBW    TTY$WRITEDONE               ; Complete the I/O.
                        0768  1523 30$:
                        0768  1524          ENBINT
           05           0768  1525          RSB
                        076C  1526
```

```
                     076C  1528          .SBTTL  FORK TO CHECK FOR TYPEAHEAD BUFFER FETCH
                     076C  1529 ;++
                     076C  1530 ; GETAHD
                     076C  1531 ;
                     076C  1532 ; Description:
                     076C  1533 ;       Will create a typeahead buffer for the user, then insert
                     076C  1534 ; the character that caused the buffer to be allocated and cause
                     076C  1535 ; that character to be processed.
                     076C  1536 ;
                     076C  1537 ; Inputs:
                     076C  1538 ;
                     076C  1539 ;       R2 = UNIT STATE VECTOR
                     076C  1540 ;       R5 = UCB ADDRESS
                     076C  1541 ;       UCB$L_FR3 = CHARACTER THAT CAUSED TYPEAHEAD BUFFER TO BE CREATED
                     076C  1542 ;
                     076C  1543 ; Outputs:
                     076C  1544 ;
                     076C  1545 ;       R1,R4 are Destroyed
                     076C  1546 ;       R2, R5 are Preserved
                     076C  1547 ;--
                     076C  1548 GETAHD:
     00E4 C5   D5    076C  1549          TSTL    UCB$L_TT_TYPAHD(R5)     ; ALREADY ALLOCATED?
           64   12    0770  1550          BNEQ    100$                    ; IF NEQ THEN YES
           07   E1    0772  1551          BBC     #TT2$V_ALTYPEAHD,-      ; SKIP IF NORMAL SIZE
         09 48 A5    0774  1552                  UCB$L_DEVDEPND2(R5),72$
00000000'GF   3C    0777  1553          MOVZWL  G^TTY$GW_ALTYPAHD,R1    ; ALLOCATE THE TYPEAHEAD BUFFER
           51        077D
           07   11    077E  1554          BRB     73$
00000000'GF   3C    0780  1555 72$:     MOVZWL  G^TTY$GW_TYPAHDSZ,R1    ; ALLOCATE THE TYPEAHEAD BUFFER
           51        0786
00000118 8F   C0    0787  1556 73$:     ADDL    #TTY$L_TA_DATA,R1       ; PLUS SPACE FOR HEADER
           51        078D
00000000'GF   16    078E  1557          JSB     G^EXE$ALONONPAGED       ; ALLOCATE THE TYPEAHEAD BUFFER
        54   52    0794  1558          MOVL    R2,R4                   ; COPY BLOCK ADDRESS
          3C 50    0797  1559          BLBC    R0,100$                 ; IF FAILURE THEN TOUGH
     08 A4   51    079A  1560          MOVW    R1,TTY$W_TA_SIZE(R4)    ; SET UP SIZE
  10 A4    6441    079E  1561          MOVAB   (R4)[R1],TTY$L_TA_END(R4); SET UP END POINTER
     0A A4    14   07A3  1562          MOVB    #DYN$C_TYPAHD,TTY$B_TA_TYPE(R4);
           14 A4    B4   07A7  1563          CLRW    TTY$W_TA_RCLSIZ(R4)     ; CLEAR OUT THE RECALL SIZE
                     07AA  1564          DSBINT  UCB$B_DIPL(R5)          ; LOCK TERMINAL DATA BASE
     00E4 C5   54    D0    07B1  1565          MOVL    R4,UCB$L_TT_TYPAHD(R5)  ; STORE ADDRESS
     52   00B8 C5    9E    07B6  1566          MOVAB   UCB$Q_TT_STATE(R5),R2   ; RESTORE THE UNIT STATE VECTOR ADDRESS
             FBAD    30    07BB  1567          BSBW    TTY$PURGE_AHEAD         ; SET UP THE BLOCK
           10 A5     90    07BE  1568          MOVB    UCB$L_FR3(R5),TTY$L_TA_DATA(R4); INSERT DATA
          0118 C4          07C1
                08   13    07C4  1569          BEQL    80$                     ; SKIP INSERT IF NULL
                64   D6    07C6  1570          INCL    TTY$L_TA_PUT(R4)        ; POINT TO NEXT PLACE
             0C A4   B6    07C8  1571          INCW    TTY$W_TA_INAHD(R4)      ; AND BUMP THE COUNT
               F832'  30    07CB  1572          BSBW    TTY$STARTOUTPUT         ; BEGIN NEXT STATE
00 14 A5       01    E4    07CE  1573 80$:     BBSC    #TTY$V_FD_GETAHD,UCB$L_FR4(R5),90$; CLEAN THE BIT OUT AGAIN
                     07D3  1574 90$:     ENBINT                          ; LOWER IPL
                     05    07D6  1575 100$:    RSB                             ; THEN RETURN
```

```
                        07D7  1577          .SBTTL LINKFORK - RECONNECT LUCB/PUCB
                        07D7  1578
                        07D7  1579 ;++
                        07D7  1580 ; LINKFORK - LINK LOGICAL AND PHYSICAL UCB
                        07D7  1581 ;
                        07D7  1582 ; FUNCTIONAL DESCRIPTION:
                        07D7  1583 ;
                        07D7  1584 ;       THIS FORK ROUTINE IS QUEUED TO LINK A PUCB AND LUCB.
                        07D7  1585 ;       THE LUCB ADDRESS TO BE LINKED TO IS CONTAINED IN PUCB
                        07D7  1586 ;       OFFSET UCB$L_PDT. THIS ROUTINE RESETS BSY IN UCB$W_STS
                        07D7  1587 ;       AND RESTARTS PENDING IRP'S IN THE I/O QUEUE.
                        07D7  1588 ;
                        07D7  1589 ; INPUTS:
                        07D7  1590 ;
                        07D7  1591 ;       R5 = PUCB ADDRESS
                        07D7  1592 ;
                        07D7  1593 ;--
                        07D7  1594 LINKFORK:
  54    0084 C5   D0    07D7  1595          MOVL    UCB$L_PDT(R5),R4                        ; GET SPECIFIED LUCB
0B A4     0B A5   90    07DC  1596          MOVB    UCB$B_FIPL(R5),UCB$B_FIPL(R4)           ; FORK IPL
5E A4     5E A5   90    07E1  1597          MOVB    UCB$B_DIPL(R5),UCB$B_DIPL(R4)           ; DEVICE IPL
   64 A4     01   8A    07E6  1598          BICB    #UCB$M_TIM,UCB$W_STS(R4)                ; AND TURN OFF TIMER BIT IN
                        07EA  1599
 00020000 8F     C8     07EA  1600          BISL    #TT2$M_DISCONNECT,UCB$L_DEVDEPND2(R5)   ; SET DISCONNECT BIT
       48 A5            07F0
                        07F2  1601                                                          ; BECAUSE WE ARE DISCONNECTA
44 A4     44 A5   D0    07F2  1602          MOVL    UCB$L_DEVDEPEND(R5),UCB$L_DEVDEPEND(R4) ; MAKE SURE LUCB IS
48 A4     48 A5   D0    07F7  1603          MOVL    UCB$L_DEVDEPND2(R5),UCB$L_DEVDEPND2(R4) ; IMAGE COPY OF PUCB
                        07FC  1604
00A0 C4     55    D0    07FC  1605          MOVL    R5,UCB$L_TL_PHYUCB(R4)                  ; LINK LUCB TO PUCB
00C0 C5     54    D0    0801  1606          MOVL    R4,UCB$L_TT_LOGUCB(R5)                  ; LINK PUCB TO LUCB
00A0 C5     55    D0    0806  1607          MOVL    R5,UCB$L_TL_PHYUCB(R5)                  ; SHOW PUCB=PUCB
     0084 C4     D4    080B  1608          CLRL    UCB$L_PDT(R4)                           ; RESET REBIND FIELDS
     0084 C5     D4    080F  1609          CLRL    UCB$L_PDT(R5)                           ; RESET REBIND FIELDS
                        0813  1610
  3C A4     02   CA    0813  1611          BICL    #DEV$M_DET,UCB$L_DEVCHAR2(R4)           ; LUCB NOW NOT DETACHED
 00040000 8F     CA     0817  1612          BICL    #DEV$M_AVL,-                            ; SHOW PUCB NOT AVALABLE FOR
       38 A5            081D  1613                  UCB$L_DEVCHAR(R5)
 00000100 8F     C8     081F  1614          BISL    #DEV$M_RED,UCB$L_DEVCHAR2(R5)           ; SHO PUCB REDIRECTED
       3C A5            0825
                        0827  1615
     55    54    D0    0827  1616          MOVL    R4,R5                                   ; SWITCH TO LUCB CONTEXT
  53    4C B5    0F    082A  1617          REMQUE  @UCB$L_IOQFL(R5),R3                     ; GET A PACKET OFF QUEUE
             0C   1D    082E  1618          BVS     10$                                     ; QUEUE EMPTY
00000842'EF     DF    0830  1619          PUSHAL  20$                                     ; SAVE RETURN ADDRESS
00000000'GF     17    0836  1620          JMP     G^IOC$INITIATE                          ; START THE PACKET
     0100 8F    AA    083C  1621 10$:     BICW    #UCB$M_BSY,UCB$W_STS(R5)                ; RESET BSY BIT IN LUCB
       64 A5            0840
                        0842  1622                                                          ; TO ALLOW NEW OUTPUT
55    00A0 C5   D0    0842  1623 20$:     MOVL    UCB$L_TL_PHYUCB(R5),R5                  ; SWITCH BACK TO PUCB
                  05    0847  1624          RSB                                             ; DONE
                        0848  1625
```

```
                                0848  1627          .SBTTL UNLINKFORK - DISCONNECT LUCB/PUCB
                                0848  1628
                                0848  1629 ;++
                                0848  1630 ; UNLINKFORK - UNLINK LOGICAL AND PHYSICAL UCB
                                0848  1631 ;
                                0848  1632 ; FUNCTIONAL DESCRIPTION:
                                0848  1633 ;
                                0848  1634 ;       THIS FORK ROUTINE IS QUEUED TO ALLOW ANY READS AND WRITE FORKS
                                0848  1635 ;       TO FINISH PRIOR TO UNLINKING LOGICAL AND PHYSICAL UCBS.
                                0848  1636 ;       CURRENTLY QUEUED WRITES ARE COMPLETED WITH SS$_NORMAL. ANY
                                0848  1637 ;       CURRENT READ, IS INSERTED AT THE HEAD OF THE I/O QUEUE. THE
                                0848  1638 ;       UCB "BUSY" BIT IS SET TO PREVENT ANY NEW I/O FROM STARTING.
                                0848  1639 ;
                                0848  1640 ; INPUTS:
                                0848  1641 ;
                                0848  1642 ;       R5 = PUCB ADDRESS
                                0848  1643 ;
                                0848  1644 ;--
                                0848  1645 UNLINKFORK:
          54   00C0 C5   D0     0848  1646          MOVL    UCB$L_TT_LOGUCB(R5),R4          ; GET LOGICAL UCB ADDRESS
          52   00B8 C5   9E     084D  1647          MOVAB   UCB$Q_TT_STATE(R5),R2           ; GET STATE ADDRESS
                                0852  1648
                                0852  1649 ;       CHECK FOR ANY ACTIVE I/O
                                0852  1650          IF_STATE READ,40$
                                0856  1651          IF_STATE WRITE,30$
;MIR0001      00000000'EF   B5   085A    .1         TSTW    QLEN                            ; CHECK TO SEE IF THE FORK QUEUE
;MIR0001           3E   12   0860    .2         BNEQ    30$                             ; IS EMPTY NO THEN WAIT SOME MORE
-1
                                0862  1653 ;       THE LUCB IS IDLE, SO WE FORCE BSY TO PREVENT ANY NEW I/O FROM
                                0862  1654 ;       BEING STARTED.  THIS WILL ALLOW THE LUCB TO BE DETACHED, AND THE
                                0862  1655 ;       PUCB TO BE FREE FOR FUTURE CONNECTS
                                0862  1656
                 0100 8F   A8   0862  1657          BISW    #UCB$M_BSY,UCB$W_STS(R4)        ; FORCE BSY
                   64 A4        0866
                   58 A4   D4   0868  1658          CLRL    UCB$L_IRP(R4)                   ; SHOW NO I/O ACTIVE
                00A0 C4   D4   086B  1659          CLRL    UCB$L_TL_PHYUCB(R4)             ; AND BREAK LINK TO PUCB
          00C0 C5    55   D0   086F  1660          MOVL    R5,UCB$L_TT_LOGUCB(R5)          ; AND MAKE PUCB=LUCB
           3C A4     02   C8   0874  1661          BISL    #DEV$M_DET,UCB$L_DEVCHAR2(R4)   ; LUCB IS NOW DETACHED
                                0878  1662
                                0878  1663 ;       INIT TIMER ON LUCB TO INSURE IT DISCONNECTS
                                0878  1664 ;       IF NOONE RECLAIMS IT WITHIN 30 MINUTES
                                0878  1665
                FCB9 CF   9E    0878  1666          MOVAB   TTY$VT_TIMEOUT,UCB$L_FPC(R4)    ; LOAD VECTOR FOR TIMEOUT ENTRY
                  0C A4         087C
          00000000'GF   C1      087E  1667          ADDL3   G^TTY$GL_TIMEOUT,G^EXE$GL_ABSTIM,-; AND SET TIMER
          00000000'GF           0884
                  6C A4         0889  1668                  UCB$L_DUETIM(R4)
           64 A4    01   88    088B  1669          BISB    #UCB$M_TIM,UCB$W_STS(R4)        ; AND TURN ON TIMER BIT IN LUCB
                                088F  1670
                                088F  1671
                                088F  1672 ;       IF REBINDING STATE, THEN GOTO LINKFORK
                     3C   E4    088F  1673          BBSC    #TTY$V_SX_RECONNECT,-
                08 00B8 C5      0891  1674                  UCB$Q_TT_STATE(R5),20$
                                0895  1675
                      F997 30   0895  1676          BSBW    CANCEL_MODEM                    ; CANCEL THE MODEM (HANGUP IF
                                0898  1677                                                  ; NECESSARY)
                   F9BB CF 16   0898  1678          JSB     CANCEL_RESET                    ; GO RESET PUCB. THIS WILL ALSO
                                089C  1679                                                  ; MAKE IT AVAILABLE
```

```
                 05   089C  1680          RSB
                      089D  1681
          FF37   31   089D  1682 20$:     BRW     LINKFORK                        ; GO LINK PUCB TO REQUESTED LUCB
                      08A0  1683
                      08A0  1684
                      08A0  1685 ;
                      08A0  1686 ;       STOP IO ON PUCB PRIOR TO DISCONNECT
                      08A0  1687 30$:
    7C A5   01   3C   08A0  1688          MOVZWL  #SS$_NORMAL,UCB$W_BOFF(R5) ; SET THE RETURN STATUS
          FE73   30   08A4  1689          BSBW    DISC_STOPIO                ; STOP CURRENT IO
            06   E5   08A7  1690          BBCC    #TTY$V_FD_BUSY,-           ; FORCE NEW FORK TO QUEUE BEHIND ALL OTHERS
      00 14 A5         08A9  1691                  UCB$L_FR4(R5),35$
       54   04   D0   08AC  1692 35$:     MOVL    #TTY$V_FD_UNLINK,R4        ; AND QUEUE THE FORK
          FD79   30   08AF  1693          BSBW    TTY$CRE_FORK
            54 8ED0   08B2  1694          POPL    R4                         ; POP OFF TERMINAL FORK CONTROL RETURN PC
                 05   08B5  1695          RSB                                ; AND EXIT
                      08B6  1696
                      08B6  1697 ;       REQUEUE ANY ACTIVE READ TO THE HEAD OF THE QUEUE
                      08B6  1698 40$:
                      08B6  1699          DSBINT  UCB$B_DIPL(R5)             ; INTERLOCK WITH DEVICE INTERRUPTS
    53   58 A5   D0   08BD  1700          MOVL    UCB$L_IRP(R5),R3           ; GET IRP OF ACTIVE READ
                      08C1  1701
                      08C1  1702 ;       INIT READ STATUS FIELDS
                      08C1  1703
16 68 A5    01   E4   08C1  1704          BBSC    #UCB$V_TT_TIMO,UCB$W_DEVSTS(R5),50$; RESET ANY TIMED READ STATE
                      08C6  1705
    4C A4   63   0E   08C6  1706          INSQUE  (R3),UCB$L_IOQFL(R4)       ; AND PUT AT HEAD OF THE QUEUE
                      08CA  1707
                      08CA  1708          CLR_STATE -
                      08CA  1709                  <READ,DEL,EOL,-            ; RESET READ STATE BITS, BUT LEAVE "BSY" SET
                      08CA  1710                  PROMPT,CTRLR,NOFLTR,-
                      08CA  1711                  ESC,ESC_O,BADESC,PRE,-
                      08CA  1712                  REFRSH,EDITREAD,SKIPCRLF,RDVERIFY,-
                      08CA  1713                  MULTI,RECALL,OVERSTRIKE,EDITING,-
                      08CA  1714                  QUOTING,BACKSPACE,TERMNORM>
                      08D7  1715
                      08D7  1716 45$:     ENBINT
          C4   11     08DA  1717          BRB     30$                        ; AND FULSH COMPLETE ANY WRITES
                      08DC  1718
02CC 8F     7C   3C   08DC  1719 50$:     MOVZWL  #SS$_HANGUP,UCB$W_BOFF(R5)         ; SET THE RETURN STATUS
      7C A5         08E0  1720
       F71B'   30   08E2  1721          BSBW    TTY$READONE                ; AND COMPLETE THE READ
          F0   11   08E5  1722          BRB     45$
                      08E7  1723
                      08E7  1724
```

```
                        08E7  1724          .SBTTL  FORK TO SEND UNSOLICITED DATA MESSAGE
                        08E7  1725 ;++
                        08E7  1726 ; UNSOL
                        08E7  1727 ;
                        08E7  1728 ; Description:
                        08E7  1729 ;       Will send a message to the job controler indicating that
                        08E7  1730 ; unsoliced input has been recieved
                        08E7  1731 ;
                        08E7  1732 ; Note: This routine runs in context of the logical UCB.
                        08E7  1733 ;
                        08E7  1734 ; INPUTS:
                        08E7  1735 ;       R2 = UNIT STATE VECTOR
                        08E7  1736 ;       R5 = UCB ADDRESS
                        08E7  1737 ;
                        08E7  1738 ; OUTPUTS:
                        08E7  1739 ;       R3 IS DESTROYED
                        08E7  1740 ;       R2,R5 ARE PRESERVED
                        08E7  1741 ;
                        08E7  1742 ; REGISTER USAGE:
                        08E7  1743 ;       R5 = LOGICAL UCB
                        08E7  1744 ;       R6 = PHYSICAL UCB
                        08E7  1745 ;--
                        08E7  1746 UNSOL:
     0060 8F    BB      08E7  1747          PUSHR   #^M<R5,R6>                      ; SAVE REGISTERS
55   00C0 C5    D0      08EB  1748          MOVL    UCB$L_TT_LOGUCB(R5),R5          ; LOAD LUCB ADDRESS
56   00A0 C5    D0      08F0  1749          MOVL    UCB$L_TL_PHYUCB(R5),R6          ; LOAD PUCB ADDRESS
                        08F5  1750
       54  01    9A      08F5  1751          MOVZBL  #MSG$_TRMUNSOLIC,R4             ; SET MESSAGE TYPE
         5C A5    B5      08F8  1752          TSTW    UCB$W_REFC(R5)                  ; IF REF COUNT 0 THEN
            0D    13      08FB  1753          BEQL    50$                             ; JOB CONTROLLER
   53   60 A5    D0      08FD  1754          MOVL    UCB$L_AMB(R5),R3                ; GET USER'S MAILBOX
            33    13      0901  1755          BEQL    65$                             ; IF EQL THEN NONE
2E 68 A6    02    E0      0903  1756          BBS     #UCB$V_TT_NOTIF,UCB$W_DEVSTS(R6),65$; BR IF ALREADY NOTIFIED
            14    11      0908  1757          BRB     55$                             ; CONTINUE
                        090A  1758 50$:
  00000000'GF   D0      090A  1759          MOVL    G^TTY$GL_JOBCTLMB,R3            ; ADDRESS JOB CONTROLLER MAILBOX
            53          0910
20 68 A6    00    E0      0911  1760          BBS     #UCB$V_JOB,UCB$W_DEVSTS(R6),65$; BR IF NOTIFIED ALREADY
                        0916  1761
            11    E1      0916  1762          BBC     #TT2$V_DISCONNECT,-             ; SKIP IF UNIT NOT ABLE TO DISCONNECT
      03 48 A6            0918  1763                  UCB$L_DEVDEPND2(R6),55$
                        091B  1764
          001D    30      091B  1765          BSBW    CLONE_UCB                       ; GO CREATE LOGICAL UCB
                        091E  1766
  00000000'GF   16      091E  1767 55$:      JSB     G^EXE$SNDEVMSG                  ; SEND THE MESSAGE
          0F 50   E9      0924  1768          BLBC    R0,65$                          ; IF FAILED THEN LEAVE DATA BASE UNCHANGED
          5C A5   B5      0927  1769          TSTW    UCB$W_REFC(R5)                  ; REF COUNT 0?
             06   12      092A  1770          BNEQ    60$                             ; IF NEQ THEN USER
    68 A6    01   A8      092C  1771          BISW    #UCB$M_JOB,UCB$W_DEVSTS(R6); SET NOTIFIED
             04   11      0930  1772          BRB     65$                             ; LOOK TO UNSOLICITED DATA
    68 A6    04   A8      0932  1773 60$:      BISW    #UCB$M_TT_NOTIF,UCB$W_DEVSTS(R6); SET NOTIFIED
       0060 8F    BA      0936  1774 65$:      POPR    #^M<R5,R6>                      ; RESTORE REGISTERS AND SWITCH BACK TO PUCB
             05    05      093A  1775          RSB                                     ; RETURN TO THE DISPATCHER
                        093B  1776
```

```
                     093B  1778          .SBTTL  CLONE NEW LOGICAL UCB
                     093B  1779 ;++
                     093B  1780 ; CLONE_UCB
                     093B  1781 ;
                     093B  1782 ; Description:
                     093B  1783 ;       This routine will clone a logical ucb and cross link it to the
                     093B  1784 ;       specified physical ucb. The logical ucb fields are initialized
                     093B  1785 ;       as appropriate.
                     093B  1786 ;
                     093B  1787 ; Inputs:
                     093B  1788 ;
                     093B  1789 ;       R6 = PHYSICAL UCB ADDRESS
                     093B  1790 ;
                     093B  1791 ; Outputs:
                     093B  1792 ;
                     093B  1793 ;       R5 = LOGICAL UCB ADDRESS
                     093B  1794 ;       R6 = PHYSICAL UCB ADDRESS
                     093B  1795 ;
                     093B  1796 ;       R0-R4 DESTROYED
                     093B  1797 ;--
                     093B  1798 CLONE_UCB:
           18   BB   093B  1799          PUSHR   #^M<R3,R4>
00000000'EF   D0   093D  1800          MOVL    VT$UCB,R5                       ; GET ADDRESS OF TEMPLATE UCB ADDRESS
           55        0943
           56   13   0944  1801          BEQL    50$                             ; NONE ESTABLISHED
00000000'GF   16   0946  1802          JSB     G^IOC$CLONE_UCB                 ; GO CREATE A DETACHED UCB
         4D 50   E9   094C  1803          BLBC    R0,50$                          ; ERROR, SO SKIP ITS USE
      55    52   D0   094F  1804          MOVL    R2,R5                           ; GET NEW LUCB ADDRESS
        5C A5   B4   0952  1805          CLRW    UCB$W_REFC(R5)                  ; INIT REFERENCE COUNT
                     0955  1806
  00010000 8F   C8   0955  1807          BISL    #UCB$M_DELETEUCB,-              ; MARK THIS LUCB TO BE DELETED ON LAST DEASS
           64 A5        095B  1808                  UCB$L_STS(R5)
  00040000 8F   CA   095D  1809          BICL    #DEV$M_AVL,-                    ; SHOW PUCB NOT AVALABLE FOR USE
           38 A6        0963  1810                  UCB$L_DEVCHAR(R6)
  00000100 8F   C8   0965  1811          BISL    #DEV$M_RED,UCB$L_DEVCHAR2(R6) ; SHO PUCB REDIRECTED
           3C A6        096B
                     096D  1812
                     096D  1813
                     096D  1814 ;       INIT LUCB FIELDS
                     096D  1815
41 A5    41 A6   90   096D  1816          MOVB    UCB$B_DEVTYPE(R6),UCB$B_DEVTYPE(R5)     ; TERMINAL TYPE
0B A5    0B A6   90   0972  1817          MOVB    UCB$B_FIPL(R6),UCB$B_FIPL(R5)           ; FORK IPL
5E A5    5E A6   90   0977  1818          MOVB    UCB$B_DIPL(R6),UCB$B_DIPL(R5)           ; DEVICE IPL
42 A5    42 A6   B0   097C  1819          MOVW    UCB$W_DEVBUFSIZ(R6),UCB$W_DEVBUFSIZ(R5) ; BUFFER SIZE
44 A5    44 A6   D0   0981  1820          MOVL    UCB$L_DEVDEPEND(R6),UCB$L_DEVDEPEND(R5) ; DEVICE CHARACTERISTICS
48 A5    48 A6   D0   0986  1821          MOVL    UCB$L_DEVDEPND2(R6),UCB$L_DEVDEPND2(R5) ;
5F A5    5F A6   90   098B  1822          MOVB    UCB$B_AMOD(R6),UCB$B_AMOD(R5)           ; DEVICE PROTECTION
 00A0 C5    56   D0   0990  1823          MOVL    R6,UCB$L_TL_PHYUCB(R5)                  ; LINK LUCB TO PUCB
 00C0 C6    55   D0   0995  1824          MOVL    R5,UCB$L_TT_LOGUCB(R6)                  ; LINK PUCB TO LUCB
           03   11   099A  1825          BRB     55$
                     099C  1826
      55   56   D0   099C  1827 50$:     MOVL    R6,R5                           ; ERROR, NO LOGICAL UCB CREATED
           18   BA   099F  1828 55$:     POPR    #^M<R3,R4>
                05   09A1  1829          RSB
                     09A2  1830
                     09A2  1831
```

```
                        09A2  1833          .SBTTL  ATTENTION - UNIT TIMEOUT, POWERFAIL ATTENTION ROUTINE
                        09A2  1834
                        09A2  1835 ;++
                        09A2  1836 ; ATTENTION - UNIT TIMEOUT, POWERFAIL ATTENTION ROUTINE
                        09A2  1837 ;
                        09A2  1838 ; THIS ROUTINE IS ENTERED VIA THE UCB$L_FPC ON POWERFAIL OR TIMEOUT.
                        09A2  1839 ; THE ACTION IS TO RESTART THE I/O IF THE POWER FAILED AND TO CANCEL IT
                        09A2  1840 ; IF A TIMEOUT OCCURRED.
                        09A2  1841 ;
                        09A2  1842 ; INPUTS:
                        09A2  1843 ;
                        09A2  1844 ;       R5 = UCB ADDRESS
                        09A2  1845 ;
                        09A2  1846 ; OUTPUTS:
                        09A2  1847 ;
                        09A2  1848 ;       R5 = UCB ADDRESS
                        09A2  1849 ;--
                        09A2  1850
                        09A2  1851 ATTENTION:                                   ; ATTENTION INTERRUPT
              F65B'  30 09A2  1852          BSBW    TTY$LOCK                  ; SET IPL AND REGISTERS
06 64 A5        05  E1 09A5  1853          BBC     #UCB$V_POWER,UCB$W_STS(R5),20$; CANCEL IF NOT POWERFAIL
                        09AA  1854
                        09AA  1855 ;
                        09AA  1856 ; POWER FAILED WHILE THIS UNIT WAS WAITING FOR AN INTERRUPT.
                        09AA  1857 ;
                        09AA  1858
              004A  30  09AA  1859          BSBW    RESTART                   ; Restart in progress I/O.
              F650'  31 09AD  1860          BRW     TTY$STARTOUTPUT           ; BEGIN THE I/O AGAIN
                        09B0  1861 20$:
                        09B0  1862 ;
                        09B0  1863 ; CHECK FOR AUTOBAUD AT 600 BAUD TIMEOUT
                        09B0  1864 ;
                        09B0  1865          IF_NOT_STATE    AUTOP,50$         ; SKIP IF NOT
                  0F  9B 09B4  1866          MOVZBW  #TT$C_BAUD_9600,-
                00F4 C5      09B6  1867                  UCB$W_TT_SPEED(R5)        ; SET 9600 BAUD
                00EC C5  90 09B9  1868          MOVB    UCB$B_TT_DEPARI(R5),UCB$B_TT_PARITY(R5); RESET THE PARITY
                00F8 C5      09BD
                   00B2  30 09C0  1869          BSBW    TTY$SET_LINE              ; RESET SPEED ON LINE,
                        09C3  1870                                            ;  SINCE TIMER EXPIRED
                        05  09C3  1871          RSB
                        09C4  1872
                        09C4  1873 ;
                        09C4  1874 ; ACTUAL HARDWARE TIMEOUT
                        09C4  1875 ;
                        09C4  1876
                        09C4  1877 50$:                                         ; Handle timeout.
                0082 C5  B6 09C4  1878          INCW    UCB$W_ERRCNT(R5)          ; BUMP ERROR COUNTER
                022C 8F  3C 09C8  1879          MOVZWL  #SS$_TIMEOUT,-            ; Load status code into the
                  7C A5     09CC  1880                  UCB$W_BOFF(R5)            ; UCB status word.
                        09CE  1881
                        09CE  1882 60$:
                        09CE  1883          IF_NOT_STATE -                    ; Branch forward if not in a
                        09CE  1884                  WRITE,70$                 ; write state.
                   004D  30 09D2  1885          BSBW    TTY$ABORT                 ; Abort any output activity
             53 00D4 C5  D0 09D5  1886          MOVL    UCB$L_TT_WRTBUF(R5),R3    ; Get address of write buffer.
                  7C A5  B0 09DA  1887          MOVW    UCB$W_BOFF(R5),-          ; Save status in the write
                  28 A3     09DD  1888                  TTY$W_WB_STATUS(R3)       ; packet.
```

```
                F61E'  30   09DF  1889          BSBW    TTY$WRITEDONE           ; Complete this write.
                       09E2  1890
                       09E2  1891  70$:                                         ; Check for a read.
                       09E2  1892          IF_NOT_STATE -                      ; Branch forward if not reading.
                       09E2  1893                READ,80$
                       09E6  1894          CLR_STATE -                         ; Clear escape states.
                       09E6  1895                <ESC,BADESC>
54   58 A5     D0     09EC  1896          MOVL    UCB$L_IRP(R5),R4            ; GET CURRENT PACKET
     38 A4     D4     09F0  1897          CLRL    IRP$L_MEDIA(R4)             ; SET NO TERMINATORS
      F60A'    30     09F3  1898          BSBW    TTY$READONE                 ; COMPLETE READ
                       09F6  1899  80$:
                05     09F6  1900          RSB
```

```
                     09F7  1902          .SBTTL  RESTART - RESTART EVERYTHING ON POWERFAIL
                     09F7  1903 ;++
                     09F7  1904 ; RESTART
                     09F7  1905 ;
                     09F7  1906 ; Description:
                     09F7  1907 ;
                     09F7  1908 ;       Power fail recovery code.  Called by the state dispatcher when
                     09F7  1909 ; a powerfail is detected by the port unit-init routines.
                     09F7  1910 ;
                     09F7  1911 ; Inputs:
                     09F7  1912 ;
                     09F7  1913 ;       R5 = UCB address
                     09F7  1914 ;
                     09F7  1915 ; Outputs:
                     09F7  1916 ;
                     09F7  1917 ;       R2 = UNIT state vector
                     09F7  1918 ;       R5 = UCB address
                     09F7  1919 ;
                     09F7  1920 RESTART::
  64 A5    20   AA   09F7  1921          BICW    #UCB$M_POWER,UCB$W_STS(R5); RESET POWERFAIL
52   00B8 C5   9E   09FB  1922          MOVAB   UCB$Q_TT_STATE(R5),R2   ; ADDRESS STATE VECTOR
        001F   30   0A00  1923          BSBW    TTY$ABORT               ; ABORT PORT ACTIVITY
        0056   30   0A03  1924          BSBW    TTY$RESUME              ; AND THEN RESUME PORT ACTIVITY
               0A06  1925          CLR_STATE POWER                 ; RESET POWER FAIL STATE
               0A09  1926          SET_STATE REFRSH                ; FORCE READS TO BE REFRESHED
     FA81 CF   17   0A0D  1927          JMP     TTY$RESTARTIO           ; SET UP THE CURRENT I/O
               0A11  1928
```

```
                     0A11  1930          .SBTTL  TTY$POWERACTION - PORT ENTRY FOR POWERFAIL RECOVERY ACTION
                     0A11  1931 ;++
                     0A11  1932 ; TTY$POWERACTION
                     0A11  1933 ;
                     0A11  1934 ; Description:
                     0A11  1935 ;
                     0A11  1936 ;       Called by the port driver's unit-init routine when a powerfail is
                     0A11  1937 ; detected.  This routine will clear the interupt expected bit and setup
                     0A11  1938 ; a timer to call us back during the first pass of the timer survice routine.
                     0A11  1939 ; This routine also sets the POWER bit in the unit state vector to perform
                     0A11  1940 ; powerfail recovery if the timer is reset by a burst or character output.
                     0A11  1941 ;
                     0A11  1942 ; Inputs:
                     0A11  1943 ;
                     0A11  1944 ;       R5 = UCB address
                     0A11  1945 ;
                     0A11  1946 ; Outputs:
                     0A11  1947 ;
                     0A11  1948 ;       All registers are preserved
                     0A11  1949 ;
                     0A11  1950 ;--
                     0A11  1951 TTY$POWERACTION::
  64 A5    02   AA   0A11  1952          BICW    #UCB$M_INT,UCB$W_STS(R5)
  64 A5    01   A8   0A15  1953          BISW    #UCB$M_TIM,UCB$W_STS(R5)
     6C A5      D4   0A19  1954          CLRL    UCB$L_DUETIM(R5)
00B8 C5    01   C8   0A1C  1955          BISL    #TTY$M_ST_POWER,UCB$Q_TT_STATE(R5)
           05   05   0A21  1956          RSB
                     0A22  1957
```

```
                     0A22  1959          .SBTTL  CLASS DRIVER JACKET INTERFACE TO PORT DRIVERS
                     0A22  1960 ;++
                     0A22  1961 ;
                     0A22  1962 ; FUNCTIONAL DESCRIPTION:
                     0A22  1963 ;
                     0A22  1964 ;       These routines are the only place in the driver that the terminal
                     0A22  1965 ; port driver is actually called.  These routines allow the terminal class
                     0A22  1966 ; driver to add code that would have been duplicated in every port driver.
                     0A22  1967 ;
                     0A22  1968 ; Inputs to all routines:
                     0A22  1969 ;
                     0A22  1970 ;       R5 - ucb address
                     0A22  1971 ;
                     0A22  1972 ;--
                     0A22  1973 ;
                     0A22  1974 ; TTY$ABORT - Abort current output activity
                     0A22  1975 ;
                     0A22  1976
                     0A22  1977 TTY$ABORT::
        01   BB     0A22  1978          PUSHR   #^M<R0>                         ; SAVE A REGISTER
00B8 C5  01   E5     0A24  1979          BBCC    #TTY$V_ST_CTRLS,UCB$Q_TT_STATE(R5),-    ; ALLOW CLASS OUTPUT
             00      0A29  1980                  5$
                     0A2A  1981 5$:
50   0118 C5  D0     0A2A  1982          MOVL    UCB$L_TT_PORT(R5),R0            ; GET THE ADDRESS OF THE POR
          20 B0  16  0A2F  1983          JSB     @PORT_ABORT(R0)                 ; GOTO THE ROUTINE
             01   BA  0A32  1984          POPR    #^M<R0>
             05   0A34  1985          RSB                                     ; and return
                     0A35  1986
```

```
                        0A35  1988 ;
                        0A35  1989 ; TTY$DISCONNECT - Indicates last deassign.
                        0A35  1990 ;
                        0A35  1991 ; INPUTS:
                        0A35  1992 ;
                        0A35  1993 ;       R5 - UCB address
                        0A35  1994 ;
                        0A35  1995 TTY$DISCONNECT::
          02    BB      0A35  1996         PUSHR   #^M<R1>                         ; SAVE A REGISTER
51  0118  C5    D0      0A37  1997         MOVL    UCB$L_TT_PORT(R5),R1            ; GET THE ADDRESS OF THE PORT
    04    B1    16      0A3C  1998         JSB     @PORT_DISCONNECT(R1)            ; GOTO THE ROUTINE
          02    BA      0A3F  1999         POPR    #^M<R1>
          05    05      0A41  2000         RSB                                     ; and return
                        0A42  2001
```

```
                        0A42  2003
                        0A42  2004 ; TTY$DS_SET - SET OUTPUT MODEM SIGNALS
                        0A42  2005 ;
                        0A42  2006 ; INPUTS:
                        0A42  2007 ;
                        0A42  2008 ;       R2 - LOW BYTE - SIGNALS TO ACTIVATE
                        0A42  2009 ;            HIGH BYTE - SIGNALS TO DEACTIVATE
                        0A42  2010 ;       R5 - UCB ADDRESS
                        0A42  2011 ;
                        0A42  2012 ; OUTPUTS:
                        0A42  2013 ;       R1 - R3 ARE USED
                        0A42  2014 ;--
                        0A42  2015 TTY$DS_SET::
          01    BB      0A42  2016         PUSHR   #^M<R0>                 ; SAVE A REGISTER
50  0118  C5    D0      0A44  2017         MOVL    UCB$L_TT_PORT(R5),R0    ; GET THE ADDRESS OF THE PORT
    OC    B0    16      0A49  2018         JSB     @PORT_DS_SET(R0)        ; GOTO THE ROUTINE
          01    BA      0A4C  2019         POPR    #^M<R0>
                05      0A4E  2020         RSB                             ; and return
```

```
                     0A4F  2022 ;
                     0A4F  2023 ; TTY$MAINT - MAINTNENCE ROUTINES
                     0A4F  2024 ;
                     0A4F  2025 TTY$MAINT::
          02   BB    0A4F  2026          PUSHR   #^M<R1>                         ; SAVE A REGISTER
51  0118 C5   D0    0A51  2027          MOVL    UCB$L_TT_PORT(R5),R1            ; GET THE ADDRESS OF THE PORT
      30 B1   16    0A56  2028          JSB     @PORT_MAINT(R1)                 ; GOTO THE ROUTINE
          02   BA    0A59  2029          POPR    #^M<R1>
               05    0A5B  2030          RSB                                     ; and return
```

```
                        0A5C  2032 ;
                        0A5C  2033 ; TTY$RESUME - CONTINUE OUTPUT ON A LINE
                        0A5C  2034 ;
                        0A5C  2035 TTY$RESUME::
                 01 BB  0A5C  2036          PUSHR   #^M<R0>                         ; SAVE A REGISTER
00BC C5          1D E0  0A5E  2037          BBS     #TTY$V_ST_CTSLOW,UCB$Q_TT_STATE+4(R5),10$;IF CLEAR TO SEND IS LOW
                    0E  0A63
                        0A64  2038                                                  ; THEN DON'T RESUME YET
00B8 C5          01 E5  0A64  2039          BBCC    #TTY$V_ST_CTRLS,UCB$Q_TT_STATE(R5),-    ; ALLOW CLASS OUTPUT
                    00  0A69  2040                  5$
                        0A6A  2041 5$:
50   0118 C5     D0     0A6A  2042          MOVL    UCB$L_TT_PORT(R5),R0            ; GET THE ADDRESS OF THE PORT
       24 B0     16     0A6F  2043          JSB     @PORT_RESUME(R0)                ; GOTO THE ROUTINE
                 01 BA  0A72  2044 10$:     POPR    #^M<R0>
                    05  0A74  2045          RSB                                     ; and return
```

```
                     0A75  2047 ; TTY$SET_LINE - SET SPEED AND PARITY
                     0A75  2048 ;
                     0A75  2049 ; IMPLICIT INPUTS:
                     0A75  2050 ;       UCB$B_TT_PARITY - CONTAINS PARITY, STOP BIT AND FRAME SIZE INFO
                     0A75  2051 ;       UCB$W_TT_SPEED - LOW BYTE XMIT SPEED
                     0A75  2052 ;                        HIGH BYTE ZERO OR RECIEVE SPEED
                     0A75  2053 ;       UCB$B_TT_PRTCTL - DMA AND AUTOXON ENABLE FLAGS
                     0A75  2054 ;
                     0A75  2055 TTY$SET_LINE::
          01   BB   0A75  2056         PUSHR   #^M<R0>                         ; SAVE A REGISTER
50   0118 C5   D0   0A77  2057         MOVL    UCB$L_TT_PORT(R5),R0            ; GET THE ADDRESS OF THE PORT
       08 B0   16   0A7C  2058         JSB     @PORT_SET_LINE(R0)              ; GOTO THE ROUTINE
          01   BA   0A7F  2059         POPR    #^M<R0>
               05   0A81  2060         RSB                                     ; and return
```

```
                        0A82  2062 ; TTY$SET_MODEM - START MODEM POLLING
                        0A82  2063 ;
                        0A82  2064 ; R0 - R4 USED
                        0A82  2065 ;
                        0A82  2066 TTY$SET_MODEM:
          01    BB      0A82  2067          PUSHR   #^M<R0>                         ; SAVE A REGISTER
50  0118  C5    D0      0A84  2068          MOVL    UCB$L_TT_PORT(R5),R0            ; GET THE ADDRESS OF THE PORT
    28    B0    16      0A89  2069          JSB     @PORT_SET_MODEM(R0)             ; GOTO THE ROUTINE
          01    BA      0A8C  2070          POPR    #^M<R0>
                05      0A8E  2071          RSB                                     ; and return
```

```
                     0A8F  2073 ; TTY$STOP - STOP OUTPUT TO THE TERMINAL LINE
                     0A8F  2074 ;
                     0A8F  2075 TTY$STOP::
            01   BB  0A8F  2076         PUSHR   #^M<R0>                         ; SAVE A REGISTER
            01   E2  0A91  2077         BBSS    #TTY$V_ST_CTRLS,-
 00 00B8    C5       0A93  2078                 UCB$Q_TT_STATE(R5),5$           ; BLOCK NEW OUTPUT
                     0A97  2079 5$:
50  0118    C5   D0  0A97  2080         MOVL    UCB$L_TT_PORT(R5),R0            ; GET THE ADDRESS OF THE PORT
      18    B0   16  0A9C  2081         JSB     @PORT_STOP(R0)                  ; GOTO THE ROUTINE
            01   BA  0A9F  2082         POPR    #^M<R0>
                 05  0AA1  2083         RSB                                     ; and return
```

```
                        0AA2  2085
                        0AA2  2086 ; TTY$XOFF - CALCULATE CHARACTER TO SEND FOR XOFF AND SEND IT TO THE PORT
                        0AA2  2087 ; TTY$XON  - SAME AS XOFF FOR XON FLOW CONTROL.
                        0AA2  2088 .ENABLE LSB
                        0AA2  2089 TTY$XOFF::
            0A   BB     0AA2  2090          PUSHR   #^M<R1,R3>                  ; SAVE REGISTERS
      53    13   9A     0AA4  2091          MOVZBL  #TTY$C_XOFF,R3              ; USE XOFF
           2C    E1     0AA7  2092          BBC     #TTY$V_SX_TYPFUL,-
   19 00B8  C5          0AA9  2093                  UCB$Q_TT_STATE(R5),30$      ; SKIP IF NOT TYPEAHEAD FULL
09 44 A5    04   E0     0AAD  2094          BBS     #TT$V_HOSTSYNC,UCB$L_DEVDEPEND(R5),10$ ; XOFF CORRECT
           22    E0     0AB2  2095          BBS     #TTY$V_SX_PASALL,-
   0B 00B8  C5          0AB4  2096                  UCB$Q_TT_STATE(R5),20$      ; SKIP IF PASSALL MODE
      53    07   9A     0AB8  2097          MOVZBL  #TTY$C_BELL,R3              ; USE BELL CHARACTER
51    0118  C5   D0     0ABB  2098 10$:     MOVL    UCB$L_TT_PORT(R5),R1        ; GET THE ADDRESS OF THE PORT VECTOR
        14  B1   16     0AC0  2099          JSB     @PORT_XOFF(R1)              ;
            0A   BA     0AC3  2100 20$:     POPR    #^M<R1,R3>
            05   0AC5  2101          RSB
                        0AC6  2102
 00040010   8F   D3     0AC6  2103 30$:     BITL    #TT$M_HOSTSYNC!TT$M_READSYNC,-
        44  A5          0ACC  2104                  UCB$L_DEVDEPEND(R5)         ; HOST OR READ SYNC TERMINAL?
            EB   12     0ACE  2105          BNEQ    10$                         ; YES THEN OUTPUT THE CHARACTER
            F1   11     0AD0  2106          BRB     20$                         ; NO, SKIP OUTPUT
                        0AD2  2107
                        0AD2  2108 TTY$XON::
            0A   BB     0AD2  2109          PUSHR   #^M<R1,R3>
      53    11   9A     0AD4  2110          MOVZBL  #TTY$C_XON,R3               ; SEND XON
 00040010   8F   D3     0AD7  2111          BITL    #TT$M_HOSTSYNC!TT$M_READSYNC,-
        44  A5          0ADD  2112                  UCB$L_DEVDEPEND(R5)         ; HOST OR READ SYNC TERMINAL?
            E2   13     0ADF  2113          BEQL    20$                         ; NO, SKIP OUTPUT
51    0118  C5   D0     0AE1  2114          MOVL    UCB$L_TT_PORT(R5),R1        ; GET THE ADDRESS OF THE PORT VECTOR
        10  B1   16     0AE6  2115          JSB     @PORT_XON(R1)               ;
            D8   11     0AE9  2116          BRB     20$                         ; GO BACK AND RETURN
                        0AEB  2117 .DISABLE LSB
```

```
                        0AEB  2119          .SBTTL  MODEM ROUTINES
                        0AEB  2120
                        0AEB  2121 ;       MODEM CONTROL DATA AREA
                        0AEB  2122
                        0AEB  2123
                        0AEB  2124
                        0AEB  2125 TT$L_DIALUP::                                ;ROOT OF MODEM TIMER CRB LIST
00000000                0AEB  2126          .LONG   0
                        0AEF  2127 TT$W_REFCNT:
                0000    0AEF  2128          .WORD   0                           ;COUNT OF TIMERS IN USE
                        0AF1  2129 TT$W_TIMCTRL:
                0000    0AF1  2130          .WORD   0                           ;FLAG WORD
                        0AF3  2131
                        0AF3  2132
                        0AF3  2133          .ALIGN  QUAD
                        0AF8  2134 TT$TIMQUENT:                                 ;TIMER QUE ENTRY FOR MODEM TIMER
00000B28                0AF8  2135          .BLKB   TQE$C_LENGTH                ;ALLOCATE BLOCK HERE
                        0B28  2136          STO_TQE TQE$W_SIZE,WORD,TQE$C_LENGTH,TT$TIMQUENT
                        0B28  2137          STO_TQE TQE$B_TYPE,BYTE,DYN$C_TQE,TT$TIMQUENT
                        0B28  2138          STO_TQE TQE$B_RQTYPE,BYTE,TQE$C_SSREPT,TT$TIMQUENT
                        0B28  2139
                        0B28  2140
```

```
                                0B28  2142 ;++
                                0B28  2143 ; TRANSITION_NOCHECK -  process transition element without the devdepend bit cc heck
                                0B28  2144 ; TRANSITION    -      PROCESS TRANSITION ELEMENTS ON MODEM/TIMER STATUS EVENT
                                0B28  2145 ; PORT_TRANSITION -    PROCESSES PORT CALLS THEN FALLS INTO TRANSITION
                                0B28  2146 ;
                                0B28  2147 ; FUNCTIONAL DESCRIPTION:
                                0B28  2148 ;
                                0B28  2149 ; THIS ROUTINE IS CALLED ON ALL MODEM TRANSITIONS SUCH AS MODEM SIGNAL CHANGES,
                                0B28  2150 ; TIMER EXPIRATIONS AND CONTROL. THE TRANSITION ELEMENTS CONTAINED IN THE
                                0B28  2151 ; STATE TABLE ASSOCIATED WITH THE CURRENT STATE ARE SCANNED. IF A MATCH IS
                                0B28  2152 ; FOUND, THEN A NEW STATE IS DECLARED. UPON DECLARING A NEW STATE,
                                0B28  2153 ; OUTPUT MODEM SIGNALS MAY BE SET OR RESET, TIMERS MAY BE STARTED OR
                                0B28  2154 ; CANCELLED AND ACTION ROUTINES MAY BE INVOKED.
                                0B28  2155 ;
                                0B28  2156 ; PORT_TRANSITION
                                0B28  2157 ;       THIS ROUTINE PRE-PROCESSES THE CALLS FROM THE PORT DRIVERS
                                0B28  2158 ; TO SCREEN OUT CALLS TO MODEM INIT WITH REF-COUNT NON-ZERO AND CHANGE
                                0B28  2159 ; THOSE CALLS INTO SHUTDOWN CALLS.  ALL INPUTS AND OUTPUTS ARE THE SAME
                                0B28  2160 ; AS TRANSITION.
                                0B28  2161 ;
                                0B28  2162 ; INPUTS:
                                0B28  2163 ;
                                0B28  2164 ;       R1 =    TRANSITION TYPE
                                0B28  2165 ;       R2 =    TYPE SPECIFIC ARGUMENT
                                0B28  2166 ;                 DATASET - NEW RCV MODEM MASK
                                0B28  2167 ;       R5 =    UCB ADDRESS
                                0B28  2168 ;
                                0B28  2169 ; OUTPUTS:
                                0B28  2170 ;       R0-R4   DESTROYED
                                0B28  2171 ;--
                                0B28  2172 PORT_TRANSITION::
      51    00    D1            0B28  2173         CMPL    #MODEM$C_INIT,R1                ; IS THIS AN INIT?
            0F    12            0B2B  2174         BNEQ    TRANSITION                      ; NO THEN DON'T INTERPRET
  53    00C0  C5    D0          0B2D  2175         MOVL    UCB$L_TT_LOGUCB(R5),R3          ; GET THE LOGICAL UCB ADDRESS
            08    13            0B32  2176         BEQL    TRANSITION                      ; NO LOGICAL THEN EXIT
        5C  A3    B5            0B34  2177         TSTW    UCB$W_REFC(R3)                  ; ANY REFCOUNT IN LOGICAL
            03    13            0B37  2178         BEQL    TRANSITION                      ; NO THEN INIT THE LINE
      51    01    9A            0B39  2179         MOVZBL  #MODEM$C_SHUTDWN,R1             ; ELSE SHUT IT DOWN
                                0B3C  2180 TRANSITION::
      0126  C5    B5            0B3C  2181         TSTW    UCB$W_TT_DS_ST(R5)              ; IS MODEM STATE IDLE?
            05    12            0B40  2182         BNEQ    TRANSITION_NOCHECK              ; NO, SO ALLOW CLEANUP TO FINISH
47 44 A5    15    E1            0B42  2183         BBC     #TT$V_MODEM,UCB$L_DEVDEPEND(R5),END;DON'T BOTHER IF NOT MODEM
                                0B47  2184 TRANSITION_NOCHECK::
  00000D7B'EF   DE              0B47  2185         MOVAL   STATE_INIT+MODEM$C_ST_LENGTH,R3 ;ROOT OF STATE TABLE + OFFSET
            53                  0B4D
  54    0126  C5    3C          0B4E  2186         MOVZWL  UCB$W_TT_DS_ST(R5),R4
        54  53    C0            0B53  2187         ADDL    R3,R4
                                0B56  2188         CASE    R1,TYPE=B,<INIT,SHUTDWN,TRAN_LOOP,TRAN_LOOP,TRAN_LOOP>
                                0B64  2189 INIT:
          01EA    30            0B64  2190         BSBW    MODEM$LINK_CRB                  ;LINK INTO CRB CHAIN
  00000007'8F   D0              0B67  2191         MOVL    #IDLE-STATE_INIT,R2
            52                  0B6D
          008C    31            0B6E  2192         BRW     DECLARE_STATE                   ;INIT MODEM CONTROL
                                0B71  2193 SHUTDWN:
  000000BE'8F   D0              0B71  2194         MOVL    #SHUTDOWN-STATE_INIT,R2         ;DECLARE SHUTDOWN STATE
            52                  0B77
          0082    31            0B78  2195         BRW     DECLARE_STATE                   ;TRANSITION TO NEW STATE
```

```
                     0B7B  2196 TRAN_LOOP:
        53   64   9A  0B7B  2197          MOVZBL  (R4),R3                         ;GET TRANSITION ELEMENT TYPE
                     0B7E  2198          CASE    R3,TYPE=B,<DATASET,TIME,END,DIALTYPE,DZ11,NOMODEM>
                     0B8E  2199                                                  ;DISPATCH TO TRANSITION SERVICE
                     0B8E  2200 ;
                     0B8E  2201 END:             ;LAST TRANSITION FOR STATE
                  05 0B8E  2202          RSB
                     0B8F  2203
                     0B8F  2204 ;
                     0B8F  2205 TIME:            ;TIMEOUT TRANSITION ELEMENT TYPE
        51   04   91  0B8F  2206          CMPB    #MODEM$C_TIMER,R1               ;TIMEOUT TRANSITION CALL
                  5F 12  0B92  2207          BNEQ    NEXT_TRAN                       ;NO PROCESS NEXT ELEMENT
                0062 31  0B94  2208          BRW     NEW_STATE                       ;DECLARE NEW STATE
                     0B97  2209
                     0B97  2210 ;
                     0B97  2211 DATASET:         ;DATASET TRANSITION ELEMENT TYPE
        51   03   91  0B97  2212          CMPB    #MODEM$C_DATASET,R1             ;MODEM INTERRUPT?
                  05 12  0B9A  2213          BNEQ    10$                             ;NO , DONT UPDATE STATUS
  0124 C5   52   90  0B9C  2214          MOVB    R2,UCB$B_TT_DS_RCV(R5)          ;UPDATE STATUS
                     0BA1  2215 10$:
  52    0124 C5   9A  0BA1  2216          MOVZBL  UCB$B_TT_DS_RCV(R5),R2          ;GET CURRENT STATUS
           05 A4   95  0BA6  2217          TSTB    MODEM$B_TRAN_ONMASK(R4)         ;MASK ACTIVE
                  0D 13  0BA9  2218          BEQL    15$                             ;NO
    53    05 A4   92  0BAB  2219          MCOMB   MODEM$B_TRAN_ONMASK(R4),R3      ;GET COMPLEMENT OF MASK
        52   53   8A  0BAF  2220          BICB    R3,R2                           ;ISOLATE BITS OF INTEREST
    52    05 A4   8C  0BB2  2221          XORB    MODEM$B_TRAN_ONMASK(R4),R2      ;TEST IF BOTH ON
                  13 13  0BB6  2222          BEQL    20$                             ;YES
                     0BB8  2223 15$:
  52    0124 C5   9A  0BB8  2224          MOVZBL  UCB$B_TT_DS_RCV(R5),R2          ;GET CURRENT STATUS
           04 A4   95  0BBD  2225          TSTB    MODEM$B_TRAN_OFFMASK(R4)        ;MASK ACTIVE
                  31 13  0BC0  2226          BEQL    NEXT_TRAN                       ;NO
    52    04 A4   93  0BC2  2227          BITB    MODEM$B_TRAN_OFFMASK(R4),R2     ;TEST SIGNALS OFF CASE
                  03 13  0BC6  2228          BEQL    20$                             ;YES OFF, DECLARE NEW STATE
                0028 31  0BC8  2229          BRW     NEXT_TRAN
                     0BCB  2230 20$:
                0028 31  0BCB  2231          BRW     NEW_STATE                       ;ENTER NEW STATE
                     0BCE  2232
                     0BCE  2233 ;
                     0BCE  2234 DIALTYPE:
           05 A4   93  0BCE  2235          BITB    MODEM$B_TRAN_ONMASK(R4),-       ;CHECK FOR MATCH
  00000000'GF           0BD1  2236                  G^TTY$GB_DIALTYP
                  1B 13  0BD6  2237          BEQL    NEXT_TRAN                       ;NO MATCH
                001E 31  0BD8  2238          BRW     NEW_STATE                       ;ENTER NEW STATE
                     0BDB  2239 DZ11:
        51   24 A5   D0  0BDB  2240          MOVL    UCB$L_CRB(R5),R1                ;GET CRB ADDRESS
  42 8F    0B A1   91  0BDF  2241          CMPB    CRB$B_TT_TYPE(R1),#DT$_DZ11     ;IS CONTROLLER A DZ11?
                  0D 12  0BE4  2242          BNEQ    NEXT_TRAN
                0010 31  0BE6  2243          BRW     NEW_STATE
                     0BE9  2244 NOMODEM:
     44 A5    15   E0  0BE9  2245          BBS     #TT$V_MODEM,UCB$L_DEVDEPEND(R5),-
                  03 0BED  2246                  10$                             ;IF MODEM LINE, CONTINUE
                0008 31  0BEE  2247          BRW     NEW_STATE                       ;LINE NO LONGER MODEM
                     0BF1  2248 10$:
                  00 11  0BF1  2249          BRB     NEXT_TRAN
                     0BF3  2250
                     0BF3  2251 ;
                     0BF3  2252 NEXT_TRAN:       ;LOOK AT NEXT TRANSITION ELEMENT
```

```
                                J 11
54   06   C0  0BF3  2253          ADDL    #MODEM$C_TRAN_LENGTH,R4          ;POINT TO NEXT ELEMENT
  FF82   31  0BF6  2254          BRW     TRAN_LOOP
             0BF9  2255
             0BF9  2256
```

```
                        0BF9  2258 ;++
                        0BF9  2259 ; DECLARE_STATE -       DECLARE NEW STATE
                        0BF9  2260 ;
                        0BF9  2261 ; FUNCTIONAL DESCRIPTION:
                        0BF9  2262 ;
                        0BF9  2263 ; THIS ROUTINE IS USED TO DECLARE A NEW MODEM STATE. IT HANDLES STARTING
                        0BF9  2264 ; OR CANCELLING TIMERS, SETTING NEW MODEM OUTPUT SIGNALS (VIA THE PORT)
                        0BF9  2265 ; AND INVOKING ACTION ROUTINES.
                        0BF9  2266 ;
                        0BF9  2267 ; INPUTS:
                        0BF9  2268 ;
                        0BF9  2269 ;       R2 =    NEW STATE BEING DECLARED
                        0BF9  2270 ;       R5 =    UCB ADDRESS
                        0BF9  2271 ;
                        0BF9  2272 ; OUTPUTS:
                        0BF9  2273 ;       R0-R4   DESTROYED
                        0BF9  2274 ;--
                        0BF9  2275 NEW_STATE:                                   ;ALTERNATE ENTRY FOR CHANGING
                        0BF9  2276                                              ;STATE FOR TRANSITION ELEMENT
    52   02 A4   3C    0BF9  2277          MOVZWL  MODEM$W_TRAN_NSTATE(R4),R2   ;GET OFFSET TO NEXT STATE
                        0BFD  2278
                        0BFD  2279 DECLARE_STATE:
0126 C5    52   B0    0BFD  2280          MOVW    R2,UCB$W_TT_DS_ST(R5)       ;SET NEW STATE
 00000D75'EF  DE    0C02  2281          MOVAL   STATE_INIT,R4               ;ROOT OF STATE TABLE
          54        0C08
     54   52   C0    0C09  2282          ADDL    R2,R4                       ;POINT TO STATE ENTRY
 52    02 A4   3C    0C0C  2283          MOVZWL  MODEM$W_ST_TIMER(R4),R2     ;GET TIMER VALUE
       0023    30    0C10  2284          BSBW    MODEM_TIMER                 ;SET NEW TIMER OR
                     0C13  2285                                              ;CANCEL PREVIOUS IF ZERO
    52   64   3C    0C13  2286          MOVZWL  MODEM$B_ST_ONMASK(R4),R2    ;GET SET SIGNAL MASK
                     0C16  2287                                              ;TURN LOW BYTE ON,HIGH BYTE OFF
       03   13    0C16  2288          BEQL    5$                          ;NULL FIELDS
     FE27   30    0C18  2289          BSBW    TTY$DS_SET                  ;SET NEW MODEM SIGNAL(S)
                     0C1B  2290 5$:
 52   04 A4   32    0C1B  2291          CVTWL   MODEM$W_ST_ROUTINE(R4),R2   ;ACTION ROUTINE PRESENT?
       0F    13    0C1F  2292          BEQL    10$                         ;NO
00000D75'EF  DE    0C21  2293          MOVAL   STATE_INIT,R0
          50        0C27
     52   50   C0    0C28  2294          ADDL    R0,R2                       ;COMPUTE ADDRESS OF ACTION ROUTINE
       62    16    0C2B  2295          JSB     (R2)                        ;INVOKE ACTION ROUTINE
    03 50   E9    0C2D  2296          BLBC    R0,15$                      ;SPECIAL TRANSITION TYPE IF ERROR
                     0C30  2297 10$:
     51   02   D0    0C30  2298          MOVL    #MODEM$C_NULL,R1
       FF11   31    0C33  2299 15$:     BRW     TRANSITION_NOCHECK          ;DECLARE NULL TRANSITION IN NEW STAT
                     0C36  2300                                              ;(TO PROCESS ANY PRESET
                     0C36  2301                                              ; MODEM SIGNALS)
                     0C36  2302
                     0C36  2303
```

```
                     0C36  2305 ;++
                     0C36  2306 ; MODEM_TIMER   -       SET/RESET MASTER MODEM TIMER ROUTINE
                     0C36  2307 ;
                     0C36  2308 ; FUNCTIONAL DESCRIPTION:
                     0C36  2309 ;
                     0C36  2310 ; THIS ROUTINE STARTS OR CANCELS A MODEM PROTECOL TIMER FOR A LINE.
                     0C36  2311 ; ONE TQE IS USED TO TIME ALL ACTIVE LINES. THE TQE ONLY REMAINS ACTIVE
                     0C36  2312 ; SO LONG AS AT LEAST ONE TIMER IS NEEDED. SUBSEQUENT TIMERS USE THE
                     0C36  2313 ; SAME TQE. WHEN ALL CURRENT TIMERS EXPIRE THE TQE IS REMOVED FROM
                     0C36  2314 ; THE SYSTEM TIMER QUEUE UNTIL NEEDED AGAIN.
                     0C36  2315 ; SINCE TIMERS ARE ONLY NEEDED DURING TRANSITIONS, THIS IS A LOW OVERHEAD
                     0C36  2316 ; ACTIVITY.
                     0C36  2317 ;
                     0C36  2318 ; INPUTS:
                     0C36  2319 ;
                     0C36  2320 ;       R2 =    NEW TIMEOUT VALUE OR 0 TO CANCEL ANY OUTSTANDING
                     0C36  2321 ;       R5 =    UCB ADDRESS
                     0C36  2322 ;
                     0C36  2323 ; OUTPUTS:
                     0C36  2324 ;       R0-R3   DESTROYED
                     0C36  2325 ;--
                     0C36  2326 MODEM_TIMER:
          54    DD   0C36  2327         PUSHL   R4
    54    24 A5 D0   0C38  2328         MOVL    UCB$L_CRB(R5),R4                ;CRB ADDRESS
          52    D5   0C3C  2329         TSTL    R2                              ;CANCEL?
          75    13   0C3E  2330         BEQL    TIMER_CANCEL                    ;YES
                     0C40  2331
                     0C40  2332         ;NEED TO INIT TIMER FOR LINE
       0128 C5  B5   0C40  2333         TSTW    UCB$W_TT_DS_TIM(R5)             ; ANY CURRENT TIMER ACTIVE
          07    12   0C44  2334         BNEQ    5$                              ; YES THEN JUST CHANGE THE TIME
         1F A4  96   0C46  2335         INCB    CRB$B_TT_TIMREFC(R4)            ; BUMP THE REFCOUNT IN THE CRB
     FEA2 CF    B6   0C49  2336         INCW    TT$W_REFCNT                     ;BUMP IN USE COUNTER
0128 C5    52   B0   0C4D  2337 5$:     MOVW    R2,UCB$W_TT_DS_TIM(R5)  ;SET TIME TO WAIT
                     0C52  2338
FE9A CF    01   AA   0C52  2339         BICW    #TIMCTRL$M_CANCEL,TT$W_TIMCTRL  ;RESET ANY CANCEL REQUEST
FE95 CF    01   E1   0C57  2340         BBC     #TIMCTRL$V_ACTIVE,TT$W_TIMCTRL,10$ ;TIMER NOT CURRENTLY ACTIVE
                02   0C5C
          72   11   0C5D  2341         BRB     TIMER_END
                     0C5F  2342 10$:                                          ;NEED TO RESTART TIMER
          55   DD   0C5F  2343         PUSHL   R5                              ;SAVE UCB ADDRESS
55    FE93 CF   DE   0C61  2344         MOVAL   TT$TIMQUENT,R5                  ;
  0B A5    06   90   0C66  2345         MOVB    #IPL$_QUEUEAST,TQE$B_RQTYPE(R5) ;SET UP FORK IPL
00000C72'EF   9F   0C6A  2346         PUSHAB  20$                             ;RETURN ADDRESS
          0A   11   0C70  2347         BRB     30$                             ;QUEUE FORK
                     0C72  2348
          55 8ED0   0C72  2349 20$:     POPL    R5                              ;RESTORE UCB ADDRESS
FE77 CF   02   A8   0C75  2350         BISW    #TIMCTRL$M_ACTIVE,TT$W_TIMCTRL  ;SHOW TIMER ACTIVE
          55   11   0C7A  2351         BRB     TIMER_END
                     0C7C  2352 30$:
00000000'GF   16   0C7C  2353         JSB     G^EXE$FORK
                     0C82  2354         DSBINT  #IPL$_SYNCH                     ;INTERLOCK TO ENTER TIMER ENTRY
     0CD5'CF   9E   0C88  2355         MOVAB   W^TT$TIMER,TQE$L_FPC(R5)        ;SET TIMEOUT ROUTINE ADDRESS
        0C A5       0C8C
004C4B40 8F   D0   0C8E  2356         MOVL    #5000000,TQE$Q_DELTA(R5)        ;SET REPEAT AMOUNT (500MS)
      20 A5       0C94
  0B A5    05   90   0C96  2357         MOVB    #TQE$C_SSREPT,TQE$B_RQTYPE(R5)  ;SET REPEATING TYPE
00000000'GF   7D   0C9A  2358         MOVQ    G^EXE$GQ_SYSTIME,R0             ;CURRENT SYSTEM TIME
```

```
                                                                M 11
                50      0CA0
004C4B40        8F      C0  0CA1  2359          ADDL    #5000000,R0                     ;TIMEOUT DUE IN 500MS
                50      0CA7
            51  00      D8  0CA8  2360          ADWC    #0,R1                           ;
00000000'       GF      16  0CAB  2361          JSB     G^EXE$INSTIMQ                   ;INSERT IN TIMER QUEUE
                        0CB1  2362              ENBINT                                  ;RESTORE IPL
                        05  0CB4  2363          RSB
                        0CB5  2364
                        0CB5  2365  TIMER_CANCEL:
            0128        C5      B5  0CB5  2366          TSTW    UCB$W_TT_DS_TIM(R5)     ; IS THE TIMER
                        0B      13  0CB9  2367          BEQL    20$                     ; ACTIVE NO THEN EXIT
            0128        C5      B4  0CBB  2368          CLRW    UCB$W_TT_DS_TIM(R5)     ;RESET TO INDICATE IDLE
              1F        A4      97  0CBF  2369          DECB    CRB$B_TT_TIMREFC(R4)    ; DECREMENT THE REFCOUNTS
            FE29        CF      B7  0CC2  2370          DECW    TT$W_REFCNT
                        0CC6  2371  20$:
            FE25        CF      B5  0CC6  2372          TSTW    TT$W_REFCNT             ;ANY ACTIVE TIMERS?
                        05      12  0CCA  2373          BNEQ    TIMER_END               ;YES
                        0CCC  2374
                        0CCC  2375          ;NO ACTIVE TIMERS  SO,
                        0CCC  2376          ;TIMER NEEDS TO BE CANCELLED
                        0CCC  2377
        FE20        CF      01      A8  0CCC  2378          BISW    #TIMCTRL$M_CANCEL,TT$W_TIMCTRL  ;REQUEST TIMER CANCEL
                        0CD1  2379
                        0CD1  2380  TIMER_END:
                    54  8ED0  0CD1  2381          POPL    R4
                        05  0CD4  2382          RSB
                        0CD5  2383
```

```
                        0CD5  2385 ;++
                        0CD5  2386 ; TT$TIMER      -       TIMER INTERRUPT ROUTINE
                        0CD5  2387 ;
                        0CD5  2388 ; FUNCTIONAL DESCRIPTION:
                        0CD5  2389 ;
                        0CD5  2390 ; THIS ROUTINE IS INVOKED UPON THE EXPIRATION OF THE MODEM PROTECOL TQE.
                        0CD5  2391 ; IT COUNTS DOWN ALL ACTIVE TIMERS AND INVOKES THE TRANSITION
                        0CD5  2392 ; ROUTINE FOR ANY LINES THAT HAVE TIMED OUT. THE ROUTINE REQUEUES THE
                        0CD5  2393 ; TQE INTO THE SYSTEM TIMER QUEUE UNLESS ALL ACTIVE TIMERS HAVE
                        0CD5  2394 ; EXPIRED.
                        0CD5  2395 ;
                        0CD5  2396 ; INPUTS:
                        0CD5  2397 ;
                        0CD5  2398 ;       R5 =    TQE ADDRESS
                        0CD5  2399 ;
                        0CD5  2400 ; OUTPUTS:
                        0CD5  2401 ;       R0-R4   DESTROYED
                        0CD5  2402 ;--
                        0CD5  2403 TT$TIMER:
            20   BB     0CD5  2404         PUSHR   #^M<R5>
    54  FE10 CF   DE     0CD7  2405         MOVAL   TT$L_DIALUP,R4                  ;GET ROOT OF CRB LIST
                        0CDC  2406 10$:
        54   64  D0     0CDC  2407         MOVL    (R4),R4                         ;GET NEXT CRB ADDRESS
             52  13     0CDF  2408         BEQL    60$                             ;PROCESS LINES FOR CRB
          0B A4  95     0CE1  2409         TSTB    CRB$B_TT_TIMREFC-CRB$L_TT_MODEM(R4); ANY TIMERS ON THIS LINE?
             F6  13     0CE4  2410         BEQL    10$                             ; NO  THEN EXIT
                        0CE6  2411 ;
                        0CE6  2412 ; TIMERS ARE ACTIVE ON THIS CONTROLER
                        0CE6  2413 ;
             54  DD     0CE6  2414         PUSHL   R4                              ;SAVE TIMER THREAD
        54   14  C2     0CE8  2415         SUBL    #CRB$L_TT_MODEM,R4              ;GET ACTUAL CRB ADDRESS
                        0CEB  2416
                        0CEB  2417         ;PROCESS LINES WITH ACTIVE TIMERS
    53   2C A4   D0     0CEB  2418         MOVL    CRB$L_INTD+VEC$L_IDB(R4),R3     ;GET ADRESS OF IDB
    52   0C A3   3C     0CEF  2419         MOVZWL  IDB$W_UNITS(R3),R2              ; GET THE NUMBER OF UNITS
             52  D7     0CF3  2420         DECL    R2                              ; ON THIS CONTROLER (ZERO BASED)
                        0CF5  2421 20$:
55   18 A342   D0     0CF5  2422         MOVL    IDB$L_UCBLST(R3)[R2],R5         ;GET UCB FOR THAT LINE
             10  13     0CFA  2423         BEQL    35$                             ; NO UNIT THEN GO ON TO THE NEXT
                        0CFC  2424         DSBINT  UCB$B_DIPL(R5)                  ;INTERLOCK WITH DEVICE INTERRUPTS
        0128 C5  B5     0D03  2425         TSTW    UCB$W_TT_DS_TIM(R5)             ; DOES THIS UNIT HAVE THE TIMER
             0C  12     0D07  2426         BNEQ    50$                             ; YES THEN HANDLE IT
                        0D09  2427 30$:    ENBINT                                  ; NO MORE THEN ENABLE THE INTERRUPTS
          E6 52   F4     0D0C  2428 35$:    SOBGEQ  R2,20$                          ; NO THEN GO ON
             54 8ED0     0D0F  2429 40$:    POPL    R4                              ; RESTORE TIMER THREAD
           FFC7  31      0D12  2430         BRW     10$
                        0D15  2431 ;
                        0D15  2432 ; DECREMENT/FIRE TIMER REQUESTS
                        0D15  2433 ;
        51   04  90     0D15  2434 50$:    MOVB    #MODEM$C_TIMER,R1               ;SET TRANSITION TYPE
        0128 C5  B7     0D18  2435         DECW    UCB$W_TT_DS_TIM(R5)             ;COUNT DOWN LINE
             EB  12     0D1C  2436         BNEQ    30$                             ;NOT TIME YET
      FDCD   CF  B7     0D1E  2437         DECW    TT$W_REFCNT                     ;
             1C  BB     0D22  2438         PUSHR   #^M<R2,R3,R4>                   ; SAVE SOME REGISTERS
           FE15  30     0D24  2439         BSBW    TRANSITION                      ;INVOKE TRANSITION ROUTINE
             1C  BA     0D27  2440         POPR    #^M<R2,R3,R4>
          1F A4  B7     0D29  2441         DECW    CRB$B_TT_TIMREFC(R4)            ; DECREMENT THE REFCOUNT
```

```
                DB   12  0D2C  2442          BNEQ    30$                                     ; CONTINUE
                         0D2E  2443          ENBINT                                          ; RE-ENABLE INTERUPTS IF
                DC   11  0D31  2444          BRB     40$                                     ; NO MORE TIMERS ON THIS
                         0D33  2445                                                          ; CONTROLER AND MOVE TO THE NEXT
                         0D33  2446                                                          ; CONTROLER
                         0D33  2447 ;
                         0D33  2448 ; CHECK TO SEE IF WE ARE TO CANCEL THE TIMER OR RE-ENABLE IT
                         0D33  2449 ;
                20   BA  0D33  2450 60$:    POPR    #^M<R5>
                         0D35  2451          DSBINT  #31                                     ;INTERLOCK
          FDB1 CF    00   E1  0D3B  2452          BBC     #TIMCTRL$V_CANCEL,TT$W_TIMCTRL,70$;DONT CANCEL TIMER
                         0C  0D40
                03   AA  0D41  2453          BICW    #<TIMCTRL$M_ACTIVE!TIMCTRL$M_CANCEL>-
                    FDAB CF  0D43  2454                  ,TT$W_TIMCTRL                      ;RESET CANCEL +TIMER ACTIVE BITS
            00000000'GF  DE  0D46  2455          MOVAL   G^EXE$AL_TQENOREPT, R5              ; Point R5 to no-repeat TQE
                55       0D4C
                         0D4D  2456                                                          ; thus ending timer thread.
                         0D4D  2457 70$:
                         0D4D  2458          ENBINT
                05       0D50  2459          RSB
                         0D51  2460
```

```
                        0D51  2462 ;++
                        0D51  2463 ; MODEM$LINK_CRB
                        0D51  2464 ;
                        0D51  2465 ; FUNCTIONAL DESCRIPTION:
                        0D51  2466 ;
                        0D51  2467 ; THIS ROUTINE IS USED TO LINK THE ASSOCIATED CRB OF A UCB
                        0D51  2468 ; REQUIRING MODEM CONTROL INTO THE MODEM PROTECOL TIMER QUEUE.
                        0D51  2469 ; NOTE THAT THIS QUEUE IS SEPERATE FROM THE DZ-11 SPECIFIC
                        0D51  2470 ; MODEM TRANSITION POLLING QUEUE.
                        0D51  2471 ;
                        0D51  2472 ; INPUTS:
                        0D51  2473 ;
                        0D51  2474 ;       R5 =    UCB ADDRRESS
                        0D51  2475 ;
                        0D51  2476 ; OUTPUTS:
                        0D51  2477 ;       R0-R4   DESTROYED
                        0D51  2478 ;--
                        0D51  2479 ;
                        0D51  2480 ;       CHECK IF CRB NEEDS INSERTION ON MODEM TIMER QUEUE
                        0D51  2481 ;
                        0D51  2482 MODEM$LINK_CRB:
                        0D51  2483
  54    24 A5    D0    0D51  2484          MOVL    UCB$L_CRB(R5),R4                ; GET CRB ADDRESS
51    FD92 CF    DE    0D55  2485          MOVAL   TT$L_DIALUP,R1                  ; ADDRESS OF LIST HEAD
  53    14 A4    DE    0D5A  2486          MOVAL   CRB$L_TT_MODEM(R4),R3           ; GET CRB THREAD
     52    51    D0    0D5E  2487          MOVL    R1,R2
                        0D61  2488 10$:
     53    62    D1    0D61  2489          CMPL    (R2),R3                         ; CRB ON LIST
           0B    13    0D64  2490          BEQL    20$                             ; YES
     52    62    D0    0D66  2491          MOVL    (R2),R2                         ; POINT TO NEXT
           F6    12    0D69  2492          BNEQ    10$                             ; IF NOT END, LOCK AGAIN
     63    61    D0    0D6B  2493          MOVL    (R1),(R3)                       ; LINK AT LIST HEAD
     61    53    D0    0D6E  2494          MOVL    R3,(R1)
                        0D71  2495 20$:
        FD0E    30    0D71  2496          BSBW    TTY$SET_MODEM                   ; INVOKE ANY PORT SPECIFIC SETUP
              05    0D74  2497          RSB
                        0D75  2498
```

```
0D75  2500          .sbttl  MODEM STATE TABLES
0D75  2501 ;++
0D75  2502 ; MODEM CONTROL STATE TABLES
0D75  2503 ;
0D75  2504 ; FUNCTIONAL DESCRIPTION:
0D75  2505 ;
0D75  2506 ;       THIS TABLE CONTAINS STATE ENTRIES. EACH STATE CORRESPONDS TO
0D75  2507 ; A MODEM PROTECOL STATE. EACH STATE ENTRY CONSISTS OF A MODEM SIGNAL
0D75  2508 ; MASK TO BE SET ON STATE ENTRY, AN INITIAL TIMER VALUE TO BE STARTED ON
0D75  2509 ; STATE ENTRY AND AN OPTIONAL ACTION ROUTINE.
0D75  2510 ;       FOLLOWING THE STATE ENTRY ARE TRANSITION ELEMENTS. ON ANY
0D75  2511 ; TRANSITION : TIMEOUT, MODEM INTERRUPT, OR PROGRAM DECLARED, EACH
0D75  2512 ; TRANSITION ELEMENT IS PROCESSED SEQUENTIALLY. THE LAST TRANSITION ELEMENT
0D75  2513 ; IS ALWAYS AN END TRANSITION ELEMENT WHICH SIGNALS THAT
0D75  2514 ; NO TRANSITION OCCURS IF IT IS PROCESSED. OTHER TRANSITION ELEMENT TYPES
0D75  2515 ; MAKE CONDITIONAL TESTS AND MAY DECLARE A NEW STATE IF THE APPROPRIATE
0D75  2516 ; CONDITIONS HOLD TRUE.
0D75  2517 ;--
0D75  2518
0D75  2519
0D75  2520 ;       MODEM STATE MACRO DEFINITIONS
0D75  2521 ;
0D75  2522
0D75  2523          .macro  state   name,onmask=0,offmask=0,timer,routine
0D75  2524 name:
0D75  2525          .byte   onmask
0D75  2526          .byte   offmask
0D75  2527          .if     nb      timer
0D75  2528          .word   <timer*2>
0D75  2529          .if_false
0D75  2530          .word   0
0D75  2531          .endc
0D75  2532          .if     nb      routine
0D75  2533          .word   routine-state_init
0D75  2534          .if_false
0D75  2535          .word   0
0D75  2536          .endc
0D75  2537          .endm   state
0D75  2538
0D75  2539          .macro  tran    type,nstate,onmask=0,offmask=0
0D75  2540          .if     idn type,end
0D75  2541          .byte   modem$c_tran_end
0D75  2542          .if_false
0D75  2543          .byte   modem$c_tran_'type',0
0D75  2544          .word   nstate-state_init
0D75  2545          .byte   offmask
0D75  2546          .byte   onmask
0D75  2547          .endc
0D75  2548          .endm   tran
0D75  2549
```

```
                        2551
                        2552 STATE_INIT:
                        2553
                        2554 ;
                        2555 ; LINE NOT UNDER MODEM CONTROL
                        2556 ;
00D75 2557                      STATE   OFF
00D7B 2558                      TRAN    END
00D7C 2559
                        2560 ;
                        2561 ; INITIAL MODEM STATE
                        2562 ;
00D7C 2563                      STATE   IDLE,OFFMASK=<TT$M_DS_DTR!TT$M_DS_RTS>,TIMER=2
00D82 2564                      TRAN    DIALTYPE,ONMASK=02,NSTATE=RINGWAIT
00D88 2565                      TRAN    TIME,NSTATE=WAIT
00D8E 2566                      TRAN    END
00D8F 2567
                        2568 ;
                        2569 ; WAIT FOR RING PRIOR TO SETTING DTR
                        2570 ;
00D8F 2571
00D8F 2572                      STATE   RINGWAIT
00D95 2573                      TRAN    DATASET,ONMASK=<TT$M_DS_RING>,NSTATE=INIT2
00D9B 2574                      TRAN    END
00D9C 2575
                        2576 ;
                        2577 ; WAIT FOR DTR AND RTS
                        2578 ; (IF DZ-11 GO DO SUBSET SUPPORT)
                        2579 ;
00D9C 2580                      STATE   WAIT,ONMASK=<TT$M_DS_DTR!TT$M_DS_RTS>
00DA2 2581                      TRAN    DZ11,NSTATE=DZWAIT
00DA8 2582                      TRAN    DATASET,ONMASK=TT$M_DS_DSR,NSTATE=INIT1
00DAE 2583                      TRAN    END
00DAF 2584
                        2585 ;
                        2586 ; DZ-11 SUBSET SUPPORT
                        2587 ; WAIT FOR CARRIER OR RING
                        2588 ;
00DAF 2589                      STATE   DZWAIT
00DB5 2590                      TRAN    DATASET,ONMASK=<TT$M_DS_CARRIER>,NSTATE=TRANSMIT0
00DBB 2591                      TRAN    DATASET,ONMASK=<TT$M_DS_RING>,NSTATE=INIT2
00DC1 2592                      TRAN    END
00DC2 2593
                        2594 ;
                        2595 ; DELAY
                        2596 ;
00DC2 2597                      STATE   INIT1,TIMER=1
00DC8 2598                      TRAN    TIME,NSTATE=INIT2
00DCE 2599                      TRAN    END
00DCF 2600
                        2601 ;
                        2602 ; START TIMER AND WAIT FOR CTS AND CARRIER
                        2603 ; IF TIMER EXPIRES, SHUTDOWN LINE
                        2604 ;
00DCF 2605                      STATE   INIT2,ONMASK=<TT$M_DS_DTR!TT$M_DS_RTS>,TIMER=30
00DD5 2606                      TRAN    TIME,NSTATE=SHUTDOWN
00DDB 2607                      TRAN    DATASET,ONMASK=<TT$M_DS_CTS!TT$M_DS_CARRIER!TT$M_DS_DSR>,-
```

```
0DDB  2608                      NSTATE=TRANSMIT0
0DE1  2609              TRAN    END
0DE2  2610
0DE2  2611 ;
0DE2  2612 ; TRANSMIT0 STATE (SIGNAL CONNECT AND ALLOW TIME TO ASSIGN CHANNELS)
0DE2  2613 ;
0DE2  2614              STATE   TRANSMIT0,ROUTINE=LOGIN,TIMER=30
0DE8  2615              TRAN    DATASET,OFFMASK=TT$M_DS_DSR,NSTATE=SHUTDOWN
0DEE  2616              TRAN    DATASET,OFFMASK=TT$M_DS_CARRIER,NSTATE=TRANSMIT1
0DF4  2617              TRAN    TIME,NSTATE=TRANSMIT
0DFA  2618              TRAN    END
0DFB  2619 ;
0DFB  2620 ; NORMAL TRANSMIT STATE
0DFB  2621 ;
0DFB  2622              STATE   TRANSMIT,ROUTINE=VERIFY ; ROUTINE VERIFY WILL SHUTDOWN IF REFC=0
0E01  2623              TRAN    DATASET,OFFMASK=TT$M_DS_DSR,NSTATE=SHUTDOWN
0E07  2624              TRAN    DATASET,OFFMASK=TT$M_DS_CARRIER,NSTATE=TRANSMIT1
0E0D  2625              TRAN    DATASET,OFFMASK=TT$M_DS_CTS,NSTATE=CTSLOW
0E13  2626              TRAN    END
0E14  2627
0E14  2628 ;
0E14  2629 ; LOSS OF CARRIER DETECTED
0E14  2630 ;
0E14  2631              STATE   TRANSMIT1,TIMER=2,ROUTINE=CTSHIGH
0E1A  2632              TRAN    DIALTYPE,ONMASK=01,NSTATE=SHUTDOWN
0E20  2633              TRAN    TIME,NSTATE=SHUTDOWN
0E26  2634              TRAN    DATASET,OFFMASK=TT$M_DS_DSR,NSTATE=SHUTDOWN
0E2C  2635              TRAN    DATASET,ONMASK=<TT$M_DS_CARRIER!TT$M_DS_CTS>,NSTATE=TRANSMIT
0E32  2636              TRAN    END
0E33  2637
0E33  2638 ;
0E33  2639 ; DELAY PRIOR TO COMPLETE SHUTDOWN
0E33  2640 ;
0E33  2641              STATE   SHUTDOWN,TIMER=1,OFFMASK=TT$M_DS_DTR,ROUTINE=LOGOUT
0E39  2642              TRAN    TIME,NSTATE=SHUT1
0E3F  2643              TRAN    END
0E40  2644
0E40  2645 ;
0E40  2646 ; COMPLETE SHUTDOWN
0E40  2647 ; AND THEN REINIT
0E40  2648 ;
0E40  2649              STATE   SHUT1,TIMER=2
0E46  2650              TRAN    NOMODEM,NSTATE=OFF
0E4C  2651              TRAN    TIME,NSTATE=IDLE
0E52  2652              TRAN    DATASET,OFFMASK=TT$M_DS_DSR,NSTATE=IDLE
0E58  2653              TRAN    END
0E59  2654 ;
0E59  2655 ; CTSLOW
0E59  2656 ; STOP SENDING DATA AND WAIT FOR CTS TO GO HIGH
0E59  2657 ;
0E59  2658              STATE   CTSLOW,ROUTINE=LOSTCTS
0E5F  2659              TRAN    DATASET,ONMASK=TT$M_DS_CTS,NSTATE=TRANSMIT
0E65  2660              TRAN    DATASET,OFFMASK=TT$M_DS_DSR,NSTATE=SHUTDOWN
0E6B  2661              TRAN    DATASET,OFFMASK=TT$M_DS_CARRIER,NSTATE=TRANSMIT1
0E71  2662              TRAN    END
0E72  2663
```

```
                     0E72  2665          .SBTTL  MODEM ACTION ROUTINES
                     0E72  2666 ;
                     0E72  2667
                     0E72  2668 ;       THESE ROUTINES ACT AS ACTION ROUTINES FOR MODEM CONNECTION
                     0E72  2669 ;       AND DISCONECTION. ON CONNECTION THE REMOTE BIT IS SET.
                     0E72  2670 ;       ON DISCONNECT A CONTROL Y AST IS FIRED WITH HANGUP STATUS
                     0E72  2671 ;       TO SIGNAL THE CLI. ALSO ANY OUTSTANDING ATTENTION ASTS ARE
                     0E72  2672 ;       FIRED.
                     0E72  2673 ;
                     0E72  2674 LOGIN:
51   00C0 C5   D0    0E72  2675          MOVL    UCB$L_TT_LOGUCB(R5),R1          ; GET LUCB ADDRESS
 00002000 8F   C8    0E77  2676          BISL    #TT$M_REMOTE,UCB$L_DEVDEPEND(R5); UPDATE CHARACTERISTIC
       44 A5
                     0E7D
 00002000 8F   C8    0E7F  2677          BISL    #TT$M_REMOTE,UCB$L_DEVDEPEND(R1)
       44 A1
                     0E85
         1B   10     0E87  2678          BSBB    CTSHIGH                         ; SAY CTS IS HIGH
         28   11     0E89  2679          BRB     ACTION_EXIT
                     0E8B  2680
                     0E8B  2681 LOGOUT:
         0016 30     0E8B  2682          BSBW    CTSHIGH
      54   02 D0     0E8E  2683          MOVL    #TTY$V_FD_DISCONNECT,R4         ; SIGNAL HANGUP CONDITION
         F797 30     0E91  2684          BSBW    TTY$CRE_FORK
         1D   11     0E94  2685          BRB     ACTION_EXIT
                     0E96  2686
                     0E96  2687 LOSTCTS:
 20000000 8F   C8    0E96  2688          BISL    #TTY$M_ST_CTSLOW,UCB$Q_TT_STATE+4(R5); SAY CTS IS LOW
       00BC C5
                     0E9C
         FBED 30     0E9F  2689          BSBW    TTY$STOP
         0F   11     0EA2  2690          BRB     ACTION_EXIT
                     0EA4  2691 CTSHIGH:
00BC C5  1D   E4    0EA4  2692          BBSC    #TTY$V_ST_CTSLOW,UCB$Q_TT_STATE+4(R5),10$; SAY CTS IS HIGH
                    03 0EA9
         FBAF 31     0EAA  2693          BRW     TTY$RESUME
         FBAC 30     0EAD  2694 10$:     BSBW    TTY$RESUME
         F14D' 30    0EB0  2695          BSBW    TTY$STARTOUTPUT
                     0EB3  2696
                     0EB3  2697 ACTION_EXIT:
      50   01 9A     0EB3  2698          MOVZBL  #SS$_NORMAL,R0
                     05 0EB6  2699          RSB
                     0EB7  2700
                     0EB7  2701 ;
                     0EB7  2702 ; VERIFY THAT UNIT HAS CHANNELS ASSIGNED
                     0EB7  2703 ;
                     0EB7  2704 VERIFY:
         EB   10     0EB7  2705          BSBB    CTSHIGH                         ; SAY CTS IS HIGH
51   00C0 C5   D0    0EB9  2706          MOVL    UCB$L_TT_LOGUCB(R5),R1          ; GET LUCB ADDRESS
      5C A1   B5     0EBE  2707          TSTW    UCB$W_REFC(R1)                  ; CHANNELS ASSIGNED
         F0   12     0EC1  2708          BNEQ    ACTION_EXIT                     ; YES, RETURN SUCCESS
         50   D4     0EC3  2709          CLRL    R0                              ; SIGNAL FAILURE
      51   01 9A     0EC5  2710          MOVZBL  #MODEM$C_SHUTDWN,R1             ; AND RETURN TRANSITION CODE IN R1
                     05 0EC8  2711          RSB
                     0EC9  2712
```

```
                                                H 12
                     0EC9  2714          .SBTTL  CLASS_MODEM_DIS - CLASS SERVICE TO FORCE MODEM SHUTDOWN
                     0EC9  2715          .SBTTL  TTY$CLASS_DISCONNECT - CLASS SERVICE TO SIGNAL HANGUP CONDITION
                     0EC9  2716 ;++
                     0EC9  2717 ; CLASS_MODEM_DIS
                     0EC9  2718 ;
                     0EC9  2719 ; FUNCTION:
                     0EC9  2720 ;         WILL FORCE MODEM SIGNALS TO HANGUP THE LINE AND SIGNAL HANGUP STATUS
                     0EC9  2721 ;
                     0EC9  2722 ; INPUTS:
                     0EC9  2723 ;
                     0EC9  2724 ;         R5 = UCB ADDRESS
                     0EC9  2725 ;
                     0EC9  2726 ; OUTPUTS:
                     0EC9  2727 ;
                     0EC9  2728 ;         R3,R4 DESTROYED
                     0EC9  2729 ;         R0,R1,R2 ARE PRESERVED
                     0EC9  2730 ;--
                     0EC9  2731 CLASS_MODEM_DIS::                            ; INITIATE MODEM DISCONNECT
          07   BB    0EC9  2732          PUSHR   #^M<R0,R1,R2>
  51      01   D0    0ECB  2733          MOVL    #MODEM$C_SHUTDWN,R1         ; SIGNAL RESET
        FC6B   30    0ECE  2734          BSBW    TRANSITION                  ; INVOKE MODEM TRANSITION ROUTINE
          07   BA    0ED1  2735          POPR    #^M<R0,R1,R2>
          05   05    0ED3  2736          RSB                                 ; RESTORE THE REGISTERS AND RETURN
                     0ED4  2737
                     0ED4  2738 ; GENERALIZED CLASS ROUTINE TO SIGNAL HANGUP STATUS
                     0ED4  2739
                     0ED4  2740 TTY$CLASS_DISCONNECT::
          07   BB    0ED4  2741          PUSHR   #^M<R0,R1,R2>
  54    00C0 C5    D0    0ED6  2742          MOVL    UCB$L_TT_LOGUCB(R5),R4      ; GET THE LOGICAL UCB TO CHECK
          5C A4    B5    0EDB  2743          TSTW    UCB$W_REFC(R4)              ; THE REFCOUNT.
          05   12    0EDE  2744          BNEQ    5$                          ; IF CHANNELS THEN AST'S TO FIRE OR
          50   D4    0EE0  2745          CLRL    R0                          ; ALWAYS INDICATE HANGUP IF
        FB50   30    0EE2  2746          BSBW    TTY$DISCONNECT              ; NO CHANNELS THEN NONE OF THE ABOFE
        FFA3   30    0EE5  2747 5$:      BSBW    LOGOUT                      ; AND QUEUE FORK TO SIGNAL HANGUP ST
          07   BA    0EE8  2748          POPR    #^M<R0,R1,R2>
          05   05    0EEA  2749          RSB
```

```
                        .SBTTL  End of module
                ;
TT_END::                                        ; End of Terminal Class Driver
                        .END
```

```
$$$$$$                          = 00000B28 R      02
ACB$L_KAST                      = 00000018
ACTION_EXIT                       00000EB3 R      02
ATTENTION                         000009A2 R      02
CAN$C_DASSGN                    = 00000001
CANCEL_AST                        000001D1 R      02
CANCEL_DETACH                     00000205 R      02
CANCEL_DONE                       00000227 R      02
CANCEL_MODEM                      0000022F R      02
CANCEL_RESET                      00000257 R      02
CLASS_MODEM_DIS                   00000EC9 RG     02
CLONE_UCB                         0000093B R      02
CNT                             = 00000001
COM$DELATTNAST                    ********    X   02
COM$DRVDEALMEM                    ********    X   02
COM$FLUSHATTNS                    ********    X   02
COM$FLUSHCTRLS                    ********    X   02
COM$POST                          ********    X   02
CRB$B_TT_TIMREFC                = 0000001F
CRB$B_TT_TYPE                   = 0000000B
CRB$L_INTD                      = 00000024
CRB$L_TT_MODEM                  = 00000014
CTSHIGH                           00000EA4 R      02
CTSLOW                            00000E59 R      02
DATASET                           00000B97 R      02
DECLARE_STATE                     00000BFD R      02
DEV$M_AVL                       = 00040000
DEV$M_DET                       = 00000002
DEV$M_RED                       = 00000100
DEV$V_SPL                       = 00000006
DIALTYPE                          00000BCE R      02
DISCONNECT                        00000688 R      02
DISC_FIREAST                      000006E7 R      02
DISC_STOPIO                       0000071A R      02
DT$_DZ11                        = 00000042
DYN$C_TQE                       = 0000000F
DYN$C_TYPAHD                    = 00000014
DZ11                              00000BDB R      02
DZWAIT                            00000DAF R      02
END                               00000B8E R      02
EXE$ALONONPAGED                   ********      X 02
EXE$AL_TQENOREPT                  ********      X 02
EXE$FORK                          ********      X 02
EXE$GL_ABSTIM                     ********      X 02
EXE$GQ_SYSTIME                    ********      X 02
EXE$INSTIMQ                       ********      X 02
EXE$SNDEVMSG                      ********      X 02
F                               = 00000000
FORKEXIT                          0000067C R      02
GETAHD                            0000076C R      02
IDB$L_UCBLST                    = 00000018
IDB$W_UNITS                     = 0000000C
IDLE                              00000D7C R      02
INIT                              00C00B64 R      02
INIT1                             00000DC2 R      02
INIT2                             00000DCF R      02
IO$V_DSABLMBX                   = 0000000A
IO$V_REFRESH                    = 0000000D
IO$V_TIMED                      = 00000007
IOC$CLONE_UCB                     ********    X   02
IOC$INITIATE                      ********    X   02
IPL$_QUEUEAST                   = 00000006
IPL$_SYNCH                      = 00000008
IPL$_TIMER                      = 00000008
IRP$C_MEDIA                     = 00000038
IRP$L_PID                       = 0000000C
IRP$L_SVAPTE                    = 0000002C
IRP$M_FUNC                      = 00000002
IRP$V_BUFIO                     = 00000000
IRP$V_VIRTUAL                   = 00000004
IRP$W_CHAN                      = 00000028
IRP$W_FUNC                      = 00000020
IRP$W_STS                       = 0000002A
LINKFORK                          000007D7 R      02
LOGIN                             00000E72 R      02
LOGOUT                            00000E8B R      02
LOSTCTS                           00000E96 R      02
MODEM$B_ST_ONMASK               = 00000000
MODEM$B_TRAN_OFFMASK            = 00000004
MODEM$B_TRAN_ONMASK             = 00000005
MODEM$C_DATASET                 = 00000003
MODEM$C_INIT                    = 00000000
MODEM$C_NULL                    = 00000002
MODEM$C_SHUTDWN                 = 00000001
MODEM$C_ST_LENGTH               = 00000006
MODEM$C_TIMER                   = 00000004
MODEM$C_TRAN_DATASET            = 00000000
MODEM$C_TRAN_DIALTYPE           = 00000003
MODEM$C_TRAN_DZ11               = 00000004
MODEM$C_TRAN_END                = 00000002
MODEM$C_TRAN_LENGTH             = 00000006
MODEM$C_TRAN_NOMODEM            = 00000005
MODEM$C_TRAN_TIME               = 00000001
MODEM$LINK_CRB                    00000D51 R      02
MODEM$W_ST_ROUTINE              = 00000004
MODEM$W_ST_TIMER                = 00000002
MODEM$W_TRAN_NSTATE             = 00000002
MODEM_TIMER                       00000C36 R      02
MSG$_TRMHANGUP                  = 00000006
MSG$_TRMUNSOLIC                 = 00000001
NEW_STATE                         00000BF9 R      02
NEXT_TRAN                         00000BF3 R      02
NOMODEM                           00000BE9 R      02
OFF                               00000D75 R      02
OPA$UCBO                          ********    X   02
PCB$L_PID                       = 00000060
PORTFORK                          00000680 R      02
PORT_ABORT                      = 00000020
PORT_DISCONNECT                 = 00000004
PORT_DS_SET                     = 0000000C
PORT_FORKRET                    = 00000034
PORT_MAINT                      = 00000030
PORT_RESUME                     = 00000024
PORT_SET_LINE                   = 00000008
```

```
PORT_SET_MODEM          =  00000028
PORT_STOP               =  00000018
PORT_TRANSITION            00000B28  RG    02
PORT_XOFF               =  00000014
PORT_XON                =  00000010
PR$_IPL                 =  00000012
QLEN                       ********    X   02
RESTART                    000009F7  RG    02
RINGWAIT                   00000D8F  R     02
SET_SPEED                  000000C9  R     02
SHUT1                      00000E40  R     02
SHUTDOWN                   00000E33  R     02
SHUTDWN                    00000B71  R     02
SS$_ABORT               =  0000002C
SS$_CANCEL              =  00000830
SS$_DATAOVERUN          =  00000838
SS$_HANGUP              =  000002CC
SS$_NORMAL              =  00000001
SS$_OPINCOMPL           =  000002D4
SS$_PARITY              =  000001F4
SS$_TIMEOUT             =  0000022C
STATE_INIT                 00000D75  R     02
SYS$GL_JOBCTLMB            ********    X   02
T                       =  0000002A
TIMCTRL$M_ACTIVE        =  00000002
TIMCTRL$M_CANCEL        =  00000001
TIMCTRL$V_ACTIVE        =  00000001
TIMCTRL$V_CANCEL        =  00000000
TIME                       00000B8F  R     02
TIMER_CANCEL               00000CB5  R     02
TIMER_END                  00000CD1  R     02
TQE$B_RQTYPE            =  0000000B
TQE$B_TYPE              =  0000000A
TQE$C_LENGTH            =  00000030
TQE$C_SSREPT            =  00000005
TQE$L_FPC               =  0000000C
TQE$Q_DELTA             =  00000020
TQE$W_SIZE              =  00000008
TRANSITION                 00000B3C  RG    02
TRANSITION_NOCHECK         00000B47  RG    02
TRANSMIT                   00000DFB  R     02
TRANSMIT0                  00000DE2  R     02
TRANSMIT1                  00000E14  R     02
TRAN_LOOP                  00000B7B  R     02
TT$C_BAUD_600           =  00000007
TT$C_BAUD_9600          =  0000000F
TT$L_DIALUP                00000AEB  RG    02
TT$M_DS_CARRIER         =  00000020
TT$M_DS_CTS             =  00000010
TT$M_DS_DSR             =  00000080
TT$M_DS_DTR             =  00000002
TT$M_DS_RING            =  00000040
TT$M_DS_RTS             =  00000010
TT$M_HOSTSYNC           =  00000010
TT$M_MBXDSABL           =  00010000
TT$M_NOECHO             =  00000002
TT$M_PASSALL            =  00000001
TT$M_READSYNC           =  00040000
TT$M_REMOTE             =  00002000
TT$TIMER                   00000CD5  R     02
TT$TIMQUENT                00000AF8  R     02
TT$V_HOSTSYNC           =  00000004
TT$V_MBXDSABL           =  00000010
TT$V_MODEM              =  00000015
TT$V_NOTYPEAHD          =  00000002
TT$V_PARITY             =  00000006
TT$V_READSYNC           =  00000012
TT$V_REMOTE             =  0000000D
TT$W_REFCNT                00000AEF  R     02
TT$W_TIMCTRL               00000AF1  R     02
TT2$M_DISCONNECT        =  00020000
TT2$M_DMA               =  00000040
TT2$V_ALTYPEAHD         =  00000007
TT2$V_AUTOBAUD          =  00000001
TT2$V_DISCONNECT        =  00000011
TT2$V_DMA               =  00000006
TT2$V_HANGUP            =  00000002
TT2$V_LOCALECHO         =  00000000
TT2$V_SECURE            =  00000010
TTY$ABORT                  00000A22  RG    02
TTY$ABORT_IO               00000000  RG    02
TTY$AB_600                 ********    X   02
TTY$AB_9600                ********    X   02
TTY$AUTOBAUD               0000006C  RG    02
TTY$A_CTRLU                ********    X   02
TTY$A_MAXTIME              ********    X   02
TTY$A_RB_PRM            =  0000004A
TTY$B_TA_TYPE           =  0000000A
TTY$CANCELIO               000000D2  RG    02
TTY$CLASS_DISCONNECT       00000ED4  RG    02
TTY$CLASS_FORK             00000628  RG    02
TTY$CRE_FORK               0000062B  RG    02
TTY$C_BELL              =  00000007
TTY$C_BLANK             =  00000020
TTY$C_CR                =  0000000D
TTY$C_XOFF              =  00000013
TTY$C_XON               =  00000011
TTY$DISCONNECT             00000A35  RG    02
TTY$DS_SET                 00000A42  RG    02
TTY$FORK_ADDR              0000064D  RG    02
TTY$FRAMERROR              0000041F  R     02
TTY$GB_DIALTYP             ********      X 02
TTY$GL_JOBCTLMB            ********      X 02
TTY$GL_TIMEOUT             ********      X 02
TTY$GW_ALTYPAHD            ********      X 02
TTY$GW_TYPAHDSZ            ********      X 02
TTY$LOCK                   ********      X 02
TTY$L_TA_DATA           =  00000118
TTY$L_TA_END            =  00000010
TTY$L_TA_GET            =  00000004
TTY$L_TA_PUT            =  00000000
TTY$L_WB_BLINK          =  00000004
TTY$L_WB_DATA           =  00000030
TTY$L_WB_FLINK          =  00000000
```

```
TTY$L_WB_IRP                    = 00000024
TTY$L_WB_NEXT                   = 0000001C
TTY$MAINT                         00000A4F RG    02
TTY$M_PC_DMAENA                 = 00000002
TTY$M_PC_XOFENA                 = 00000040
TTY$M_ST_AUTOP                  = 00020000
TTY$M_ST_BACKSPACE              = 00000020
TTY$M_ST_BADESC                 = 00000100
TTY$M_ST_CTRLO                  = 00000001
TTY$M_ST_CTRLR                  = 00040000
TTY$M_ST_CTSLOW                 = 20000000
TTY$M_ST_DEL                    = 00000002
TTY$M_ST_EDITING                = 00100000
TTY$M_ST_EDITREAD               = 00000200
TTY$M_ST_EOL                    = 00000100
TTY$M_ST_ESC                    = 00000080
TTY$M_ST_ESC_O                  = 00004000
TTY$M_ST_MULTI                  = 00000040
TTY$M_ST_NOFLTR                 = 00000040
TTY$M_ST_OVERSTRIKE             = 00800000
TTY$M_ST_OVRFLO                 = 00010000
TTY$M_ST_POWER                  = 00000001
TTY$M_ST_PRE                    = 04000000
TTY$M_ST_PROMPT                 = 00000020
TTY$M_ST_QUOTING                = 00400000
TTY$M_ST_RDVERIFY               = 00000400
TTY$M_ST_READ                   = 00001000
TTY$M_ST_RECALL                 = 00000800
TTY$M_ST_REFRSH                 = 00000400
TTY$M_ST_SKIPCRLF               = 00080000
TTY$M_ST_SKIPLF                 = 00002000
TTY$M_ST_TERMNORM               = 01000000
TTY$M_ST_TYPFUL                 = 00001000
TTY$NOTIFY                        0000031D RG    02
TTY$POWERACTION                   00000A11 RG    02
TTY$PURGE_AHEAD                   0000036B RG    02
TTY$READERROR                     0000038D RG    02
TTY$READONE                       ******** X     02
TTY$RESTARTIO                     00000492 RG    02
TTY$RESUME                        00000A5C RG    02
TTY$RTIMOU                        00000503 R     02
TTY$SETUP_READ                    000002B5 RG    02
TTY$SETUP_UCB                     00000568 RG    02
TTY$SET_LINE                      00000A75 RG    02
TTY$SET_MODEM                     00000A82 R     02
TTY$STARTOUTPUT                   ******** X     02
TTY$STOP                          00000A8F RG    02
TTY$SYNCH                         ******** X     02
TTY$VT_TIMEOUT                    00000535 R     02
TTY$V_FD_BUSY                   = 00000006
TTY$V_FD_DISCONNECT             = 00000002
TTY$V_FD_GETAHD                 = 00000001
TTY$V_FD_LINK                   = 00000005
TTY$V_FD_PORTFORK               = 00000003
TTY$V_FD_UNLINK                 = 00000004
TTY$V_FD_UNSOL                  = 00000000
TTY$V_PC_DMAAVL                 = 00000002
TTY$V_PC_NOTIME                 = 00000000
TTY$V_ST_CTRLS                  = 00000001
TTY$V_ST_CTSLOW                 = 0000001D
TTY$V_ST_PASALL                 = 00000002
TTY$V_ST_TYPFUL                 = 0000000C
TTY$V_SX_AUTOP                  = 00000031
TTY$V_SX_BACKSPACE              = 00000005
TTY$V_SX_BADESC                 = 00000028
TTY$V_SX_CTRLO                  = 00000020
TTY$V_SX_CTRLR                  = 00000032
TTY$V_SX_DEL                    = 00000021
TTY$V_SX_EDITING                = 00000034
TTY$V_SX_EDITREAD               = 00000009
TTY$V_SX_EOL                    = 00000008
TTY$V_SX_ESC                    = 00000027
TTY$V_SX_ESC_O                  = 0000002E
TTY$V_SX_MULTI                  = 00000006
TTY$V_SX_NOECHO                 = 00000023
TTY$V_SX_NOFLTR                 = 00000026
TTY$V_SX_OVERSTRIKE             = 00000037
TTY$V_SX_OVRFLO                 = 00000030
TTY$V_SX_PASALL                 = 00000022
TTY$V_SX_POWER                  = 00000000
TTY$V_SX_PRE                    = 0000003A
TTY$V_SX_PROMPT                 = 00000025
TTY$V_SX_QUOTING                = 00000036
TTY$V_SX_RDVERIFY               = 0000000A
TTY$V_SX_READ                   = 0000000C
TTY$V_SX_RECALL                 = 0000000B
TTY$V_SX_RECONNECT              = 0000003C
TTY$V_SX_REFRSH                 = 0000002A
TTY$V_SX_SKIPCRLF               = 00000033
TTY$V_SX_SKIPLF                 = 0000002D
TTY$V_SX_TERMNORM               = 00000038
TTY$V_SX_TYPFUL                 = 0000002C
TTY$V_SX_WRITE                  = 00000007
TTY$V_SX_WRTALL                 = 00000024
TTY$WRITEDONE                     ******** X     02
TTY$WRITEPOST                     ******** X     02
TTY$W_RB_CPZORG                 = 0000003A
TTY$W_RB_LINOFF                 = 00000030
TTY$W_RB_TIMOS                  = 00000036
TTY$W_RB_TXTOFF                 = 0000003C
TTY$W_TA_INAHD                  = 0000000C
TTY$W_TA_RCLSIZ                 = 00000014
TTY$W_TA_SIZE                   = 00000008
TTY$W_WB_BCNT                   = 0000002A
TTY$W_WB_STATUS                 = 00000028
TTY$XOFF                          00000AA2 RG    02
TTY$XON                           00000AD2 RG    02
TT_END                            00000EEB RG    02
UCB$B_AMOD                      = 0000005F
UCB$B_DEVTYPE                   = 00000041
UCB$B_DIPL                      = 0000005E
UCB$B_FIPL                      = 0000000B
UCB$B_TT_DEPARI                 = 000000EC
UCB$B_TT_DETYPE                 = 000000F0
```

```
UCB$B_TT_DS_RCV                 =  00000124
UCB$B_TT_OUTYPE                 =  0000010B
UCB$B_TT_PARITY                 =  000000F8
UCB$L_AMB                       =  00000060
UCB$L_CRB                       =  00000024
UCB$L_DEVCHAR                   =  00000038
UCB$L_DEVCHAR2                  =  0000003C
UCB$L_DEVDEPEND                 =  00000044
UCB$L_DEVDEPND2                 =  00000048
UCB$L_DUETIM                    =  0000006C
UCB$L_FPC                       =  0000000C
UCB$L_FR3                       =  00000010
UCB$L_FR4                       =  00000014
UCB$L_IOQFL                     =  0000004C
UCB$L_IRP                       =  00000058
UCB$L_PDT                       =  00000084
UCB$L_PID                       =  0000002C
UCB$L_STS                       =  00000064
UCB$L_SVAPTE                    =  00000078
UCB$L_TL_BANDQUE                =  0000009C
UCB$L_TL_CTLPID                 =  000000A4
UCB$L_TL_CTRLC                  =  00000094
UCB$L_TL_CTRLY                  =  00000090
UCB$L_TL_OUTBAND                =  00000098
UCB$L_TL_PHYUCB                 =  000000A0
UCB$L_TT_DECHAR                 =  000000C8
UCB$L_TT_DECHAR                 =  000000C4
UCB$L_TT_LOGUCB                 =  000000C0
UCB$L_TT_MULTI                  =  000000D8
UCB$L_TT_PORT                   =  00000118
UCB$L_TT_RDUE                   =  000000B0
UCB$L_TT_RTIMOU                 =  000000B4
UCB$L_TT_TYPAHD                 =  000000E4
UCB$L_TT_WBLINK                 =  000000D0
UCB$L_TT_WFLINK                 =  000000CC
UCB$L_TT_WRTBUF                 =  000000D4
UCB$M_BSY                       =  00000100
UCB$M_DELETEUCB                 =  00010000
UCB$M_INT                       =  00000002
UCB$M_JOB                       =  00000001
UCB$M_ONLINE                    =  00000010
UCB$M_POWER                     =  00000020
UCB$M_TIM                       =  00000001
UCB$M_TT_HANGUP                 =  00000008
UCB$M_TT_LEN                    =  00000018
UCB$M_TT_NOTIF                  =  00000004
UCB$M_TT_TIMO                   =  00000002
UCB$Q_TL_BRKTHRU                =  000000A8
UCB$Q_TT_STATE                  =  000000B8
UCB$V_BSY                       =  00000008
UCB$V_JOB                       =  00000000
UCB$V_POWER                     =  00000005
UCB$V_TT_DISPARERR              =  00000001
UCB$V_TT_NOTIF                  =  00000002
UCB$V_TT_PARTY                  =  00000006
UCB$V_TT_TIMO                   =  00000001
UCB$V_TT_USERFRAME              =  00000002
UCB$W_BOFF                      =  0000007C
UCB$W_DEVBUFSIZ                 =  00000042
UCB$W_DEVSTS                    =  00000068
UCB$W_ERRCNT                    =  00000082
UCB$W_REFC                      =  0000005C
UCB$W_STS                       =  00000064
UCB$W_TT_CURSOR                 =  000000FC
UCB$W_TT_DESPEE                 =  000000E8
UCB$W_TT_DS_ST                  =  00000126
UCB$W_TT_DS_TIM                 =  00000128
UCB$W_TT_HOLD                   =  00000108
UCB$W_TT_MULTILEN               =  000000DC
UCB$W_TT_PRTCTL                 =  00000122
UCB$W_TT_SPEED                  =  000000F4
UNLINKFORK                      00000848 R      02
UNSOL                           000008E7 R      02
VEC$L_IDB                       =  00000008
VERIFY                          00000EB7 R      02
VT$UCB                          ********     X  02
W0                              =  00000000
W1                              =  00000004
WAIT                            00000D9C R      02
X                               =  00000000
X0                              =  00000000
X1                              =  00000001
Z0                              =  00000003
Z1                              =  00000001
```

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	00000EEB (3819.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC QUAD

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	117	00:00:00.23	00:00:00.89
Command processing	128	00:00:00.71	00:00:03.31
Pass 1	905	00:00:43.25	00:01:11.78
Symbol table sort	0	00:00:05.02	00:00:06.41
Pass 2	418	00:00:10.09	00:00:18.06
Symbol table output	2	00:00:00.37	00:00:00.94
Psect synopsis output	1	00:00:00.02	00:00:00.13
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1574	00:00:59.71	00:01:41.54

The working set limit was 2700 pages.
212892 bytes (416 pages) of virtual memory were used to buffer the intermediate code.
There were 170 pages of symbol table space allocated to hold 3038 non-local and 147 local symbols.
2757 source lines were read in Pass 1, producing 26 object records in Pass 2.
65 pages of virtual memory were used to define 62 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA18:[SYS.OBJ]LIB.MLB;1	28
_255DUA18:[SYSLIB]STARLET.MLB;3	11
TOTALS (all libraries)	39

3384 GETS were required to define 39 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:TTYSUB/OBJ=OBJ$:TTYSUB MSRC$:TTYSUB/UPDATE=(BUG$:TTYSUB)+EXECML$/LIB
```

0449 AH-EF71A-SE
VAX/VMS V4.1 SRC LST MCRF UPD

YCDRIVER
LIS

TTYSTRSTP
LIS

TTYSUB
LIS