


```

TTTTTTTTTT  TTTTTTTTTT  YY      YY  SSSSSSSS  UU      UU  BBBB8888
TTTTTTTTTT  TTTTTTTTTT  YY      YY  SSSSSSSS  UU      UU  BBBB8888
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SSSSSS  UU      UU  BBBB8888
      TT      TT      YY      YY  SSSSSS  UU      UU  BBP88888
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SS      SS  UU      UU  BB      BB
      TT      TT      YY      YY  SSSSSSSS  UUUUUUUUU  BBBB8888
      TT      TT      YY      YY  SSSSSSSS  UUUUUUUUU  BBBB8888

```

```

LL      LL      SSSSSSSS
LL      LL      SSSSSSSS
LL      LL      SS
LL      LL      SS
LL      LL      SS
LL      LL      SS
LL      LL      SSSSSS
LL      LL      SSSSSS
LL      LL      SS
LL      LL      SS
LL      LL      SS
LL      LL      SS
LLLLLLLLLL  LLLLLLLL  SSSSSSSS
LLLLLLLLLL  LLLLLLLL  SSSSSSSS

```

T1
V0

(2)	236	Declarations
(3)	270	TTY\$ABORT IO - ABORT I/O
(4)	330	TTY\$AUTOBAUD - SENSE AND SET BAUD RATE
(5)	414	TTY\$CANCELIO - CANCEL I/O ON TERMINAL
(14)	733	TTY\$SETUP READ - Set up UCB and states for a read
(15)	820	TTY\$NOTIFY - NOTIFY JOB CONTROLLER OF LOGIN IF NECESSARY
(16)	865	TTY\$PURGE AHEAD - PURGE TYPEAHEAD BUFFER
(17)	897	TTY\$READERROR - Line error occurred.
(18)	968	TTY\$FRAMERROR - FRAME ERROR ON A READ
(19)	1051	TTY\$RESTARTIO - RESTART CURRENT I/O
(20)	1120	TTY\$RTIMOU - READ TIMED OUT ENTRY
(21)	1161	TTY\$VT TIMEOUT - DETACHED TERMINAL TIMEOUT
(22)	1204	TTY\$SETUP UCB - SETUP TERMINAL UCB
(24)	1298	TTY\$CLASS_FORK - PORT DRIVER FORK ROUTINE
(25)	1321	TTY\$CRE_FORK - General purpose driver fork routine
(27)	1358	FORK DISPATCHER
(28)	1391	FORK TO DISCONNECT THE TERMINAL AND DELIVER THE HANGUP AST
(32)	1528	FORK TO CHECK FOR TYPEAHEAD BUFFER FETCH
(33)	1577	LINKFORK - RECONNECT LUCB/PUCB
(34)	1627	UNLINKFORK - DISCONNECT LUCB/PUCB
(35)	1724	FORK TO SEND UNSOLICITED DATA MESSAGE
(35)	1778	CLONE NEW LOGICAL UCB
(36)	1833	ATTENTION - UNIT TIMEOUT, POWERFAIL ATTENTION ROUTINE
(37)	1902	RESTART - RESTART EVERYTHING ON POWERFAIL
(38)	1930	TTY\$POWERACTION - PORT ENTRY FOR POWERFAIL RECOVERY ACTION
(39)	1959	CLASS DRIVER JACKET INTERFACE TO PORT DRIVERS
(50)	2119	MODEM ROUTINES
(50)	2500	MODEM STATE TABLES
(50)	2665	MODEM ACTION ROUTINES
(51)	2714	CLASS MODEM DIS - CLASS SERVICE TO FORCE MODEM SHUTDOWN
(51)	2715	TTY\$CLASS_DISCONNECT - CLASS SERVICE TO SIGNAL HANGUP CONDITION
(52)	2751	End of module

:MIR0001
-1

```

0000 1 .TITLE TTYSUB - Terminal driver miscellaneous subroutines
0000 .1 .IDENT 'V04-002'
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
0000 9 * ALL RIGHTS RESERVED. *
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
0000 16 * TRANSFERRED. *
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
0000 20 * CORPORATION. *
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 28 **
0000 29
0000 30 FACILITY:
0000 31
0000 32 VAX/VMS TERMINAL DRIVER
0000 33
0000 34 ABSTRACT:
0000 35
0000 36 THIS MODULE CONTAINS ROUTINES COMMON TO ALL TERMINAL DRIVER FUNCTIONS.
0000 37
0000 38 AUTHOR:
0000 39
0000 40 R.HEINEN 11-AUG-1976
0000 41
0000 42 Revision history:
0000 43
0000 .1 V04-002 MIR0001 Michael I. Rosenblum 5-Nov-1984
0000 .2 Remove code added in V04-001 it is no longer necessary
0000 .3 Fix problem with disconnect where disconnection could
0000 .4 happen when a read completion was waiting on the queue.
0000 .5 This required a fork queue length check.
0000 44
0000 45 V04-001 MIR1100 Michael I. Rosenblum 7-Sep-1984
0000 46 Add code to catch fired hangup ast's.
0000 47
0000 48 V03-057 JLV0394 Jake VanNoy 13-AUG-1984
0000 49 Change exit path in read error such that condition code is
0000 50 is correctly on return to port driver.
0000 51
0000 52 V03-056 MIR0460 Michael I. Rosenblum 07-Aug-1984

```

:MIR0001
:MIR00C1
:MIR0001
:MIR0001
:MIR0001

```

0000 53 : Fix bug described in QAR 915 that points out a condition
0000 54 : Where the DMF hardware XON/XOFF handling get's out of synch
0000 55 : With the terminal driver's reconing.
0000 56 :
0000 57 : V03-055 M1R0455 MICHAEL I. ROSENBLUM 20-JUN-1984
0000 58 : Add check to catch late rtimeout ast's.
0000 59 : make detach cause a hangup if the line is set as such.
0000 60 :
0000 61 : V03-054 RKS0054 RICK SPITZ 10-APR-1984
0000 62 : Show Physical terminal UCB redirected when associated
0000 63 : with logical TT UCB. Remove characteristics logic on rebind
0000 64 : as it is done in the connect startio I/O action code.
0000 65 :
0000 66 : V03-053 M1R0390 Michael I. Rosenblum 04-Apr-1984
0000 67 : Return the offset when a readverify read terminates
0000 68 : Due to timeout. Terminate a Read verify read if
0000 69 : it is restarted by another io also return the correct
0000 70 : offset.
0000 71 :
0000 72 : V03-052 M1R0370 Michael I. Rosenblum 20-Mar-1984
0000 73 : Remove local definition of TTY$V_ST_CTSLOW.
0000 74 :
0000 75 : V03-051 RKS0051 RICK SPITZ 15-MAR-1984
0000 76 :
0000 77 : Cleanup modem disconnect routines and
0000 78 : add login timeout logic to modem processing.
0000 79 : Cleanup routine headers.
0000 80 :
0000 81 : V03-050 RKS0050 RICK SPITZ 05-MAR-1984
0000 82 : Enhance virtual terminal timeout code to delete
0000 83 : a virtual terminal when specified time period expires.
0000 84 :
0000 85 : V03-048 M1R0310 Michael Rosenblum 09-Feb-1984
0000 86 : Fix bugs.
0000 87 : Clear breakthru classes on last deassign.
0000 88 : Maintnace functions shold not always return abort.
0000 89 : Add entry point to Transition to not check modem in
0000 90 : Devdepend so that the modem can be shutdown when
0000 91 : modem is turned off.
0000 92 :
0000 93 : V03-047 MHB0100 Mark H. Bramhall 1-Feb-1984
0000 94 : Add the setting/resetting of DEV$M_DET in DEVCHAR2 during
0000 95 : LUCB unlinks/(re-)links.
0000 96 :
0000 97 : V03-046 M1R0200 Michael i. Rosenblum 15-Oct-1983
0000 98 : make calls to EXESFORK call TTY$SYNCH to allow the terminal
0000 99 : driver to provide different synchronization methods.
0000 100 : Change input IPL to the RTIMOU routine to allow the
0000 101 : class driver to provide it's own synchronization.
0000 102 :
0000 103 : V03-045 M1R0082 Michael I. Rosenblum 19-Aug-1983
0000 104 : Add argument in R0 to the PORT DISCONNECT routine.
0000 105 :
0000 106 : V03-044 M1R1080 Michael I. Rosenblum 11-Aug-1983
0000 107 : remove restriction that modem state table routines must
0000 108 : be forward references.
0000 109 :

```

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 controller.	
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 LINK AND UNLINK LOGICAL AND PHYSICAL UCBs.	
0000	132	:			RESTRUCTURE CANCEL I/O CODE TO BE MORE MODULAR.	
0000	133	:			MOVE UCBSV_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 special 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 transmitting 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 original	
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 UCBSB_ERTCNT to UCBSW_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 service 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 interrupt for the
0000 183 : port driver
0000 184 : CLASS_DISCONNECT - Performs cleanup then delivers
0000 185 : DCL's hangup ast.
0000 186 : Add class service:
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 MIR001' 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 references 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 TYPEAHEAD 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 :
0000 225 :
0000 226 :
0000 227 :
0000 228 :
0000 229 :
0000 230 :
0000 231 :
0000 232 :
0000 233 :
0000 234 :--

OCCURS BETWEEN MODEM TRANSITION AND DELIVERY OF ANY
ATTENTION AST. IT IS NEEDED TO KEEP FROM
CALLING COM\$DELATTNAST
AT DEVICE IPL (IN THE CASE OF CONTROLLERS WHICH
INTERRUPT ON MODEM TRANSITIONS).

V03-021 KDM0002 Kathleen D. Morse 28-Jun-1982
Added \$PRDEF.


```

0000 236          .SBTTL Declarations
0000 237
0000 238
0000 239
0000 240          : EXTERNAL SYMBOLS
0000 241          :
0000 242
0000 243          $ACBDEF
0000 244          $CANDEF
0000 245          $CRBDEF
0000 246          $IDBDEF
0000 247          $DCDEF
0000 248          $DEVDEF
0000 249          $DPTDEF
0000 250          $DYNDDEF
0000 251          $IODEF
0000 252          $IPLDEF
0000 253          $IRPDEF
0000 254          $MSGDEF
0000 255          $PCBDEF
0000 256          $PRDEF
0000 257          $SSDEF
0000 258          $UCBDEF
0000 259          $TTDEF
0000 260          $TT2DEF
0000 261          $TTYDEF
0000 262          $VECDEF
0000 263          $TTYMACS
0000 264          $TTYDEFS
0000 265          $TTYMODEM
0000 266          $TQDEF
00000000 267
00000000 268          .PSECT $$$115_DRIVER,QUAD

```

```

: DEFINE CANCEL DEFINITIONS
: DEFINE CRB
: DEFINE IDB OFFSETS
: DEFINE DEVICE CODES
: DEFINE DEV STATUS AND DEVCHAR BITS
: DEFINE DPT OFFSETS
: DEFINE DYNAMIC REGION TYPES
: DEFINE I/O FUNCTION CODES
: DEFINE INTERRUPT PRIORITY LEVELS
: DEFINE IRP
: DEFINE MESSAGE TYPES
: DEFINE PCB
: DEFINE PROCESSOR REGISTERS
: DEFINE SYSTEM STATUS CODES
: DEFINE UCB
: DEFINE TERMINAL CHARACTERISTICS
: DEFINE TERMINAL CHARACTERISTICS
: DEFINE TERMINAL DRIVER SYMBOLS
: DEFINE CRB VECTOR
: DEFINE TERMINAL MACROS
: DEFINE TERMINAL DEFINITIONS
: DEFINE TERMINAL MODEM SYMBOLS
: TIMER QUEUE DEFINITIONS
: 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$ABORT_IO::
0000 282
0000 283      BSBW  TTY$PURGE_AHEAD      ; PURGE THE TYPEAHEAD
0003 284
51  00CC C5 9E 0003 285      MOVAB  UCBSL_TT_WFLINK(R5),R1 ; Get start of write queue.
53  53  51  D0 0008 286      MOVL   R1,R3                    ; Copy it to check for end.
000B 287 30$:
53  63  D0 000B 288      MOVL   TTYSL_WB_FLINK(R3),R3 ; Get next entry.
51  53  D1 000E 289      Cmpl  R3,R1                    ; At end of queue?
13  13  0011 290      BEQL  50$                      ; Yes. Branch forward.
54  24  A3  D0 0013 291      MOVL   TTYSL_WB_IRP(R3),R4     ; Get associated IRP address.
53  04  A3  D0 0019 292      BEQL  30$                      ; If no IRP, it's a broadcast,
54  00  B3  OF 001D 293      MOVL   TTYSL_WB_BLINK(R3),R3 ; leave it in the queue.
FFDC' 30 0021 294      REMQUE @TTYSL_WB_FLINK(R3),R4 ; Get entry's backward link.
E5  11  0024 295      BSBW  TTY$WRITEPOST          ; Remove the link.
0026 296      BRB  30$                  ; Queue it for completion
0026 297      ; Try for next link.
0026 298
0026 299 50$:
0026 300      IF_NOT_STATE -            ; Stop the current I/Os.
0026 301      READ,60$                ; Branch forward if no read is
002A 302      CLR_STATE -            ; currently in progress.
002A 303      <ESC,BADESC>          ; Clear escape bits.
53  78  A5  D0 0030 304      MOVL   UCBSL_SVAPE(R5),R3     ; GET THE PACKET ADDRESS
54  58  A5  D0 0034 305      MOVL   UCBSL_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 IRPSL_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  IRPSL_MEDIA(R4)        ; Clear terminators.
FFB6' 30 0047 311      BSBW  TTY$READONE           ; Complete the read.
004A 312
004A 313 60$:
00DC C5 B5 004A 314      TSTW  UCBSW_TT_MULTILEN(R5) ; Check for a write to complete.
04  13  004E 315      BEQL  65$                      ; Test for non-zero
0050 316      SET_STATE MULTI        ; Skip if zero
0054 317      ; Set multiecho here because
0054 318 65$: IF_NOT_STATE -    ; READONE will clear it if called
0054 319      WRITE,70$            ; Branch forward if no write is
0058 320      BSBW  TTY$ABORT          ; in progress.
0058 321      BSBW  TTY$RESUME         ; Abort any current output activity
00D4 C5 D0 005E 322      MOVL   UCBSL_TT_WRTBUF(R5),R3 ; Restart output
7C  A5  B0 0063 323      MOVW  UCBSW_BOFF(R5),-       ; Get address of write packet.
28  A3  0066 324      TTY$W_WB STATUS(R3)        ; Put status code in the
FF95' 30 0068 325      BSBW  TTY$WRITEDONE        ; packet.
006B 326      ; Complete the I/O.

```

TTYSUB
V04-002

- Terminal driver miscellaneous subrou^{1 7}ti
TTY\$ABORT_10 - ABORT I/O

8-JAN-1985 17:22:31 VAX/VMS Macro V04-00
7-SEP-1984 17:57:12 [TTDRVR.BUGSRC]TTYSUB.MAR;1

Page 8
(3)

05 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$AUTOBAUD::
006C 361
006C 362      CMPL   UCBSL TT_RDUE(R5),-
0070 363      G^EXE$GL_ABSTIM          ; compare to current
0075 364      BEQL   50$                ; branch if too soon
0077 365
0077 366      BBCC   #7,R3,10$          ; Clear high order bit
007B 367
007B 368      ; check baud rate
007B 369
007B 370 10$:  MOVAB  TTY$AB_9600,R4    ; set 9600 baud table
0081 371
0082 371      CMPB   #TT$C_BAUD_9600,UCBSW_TT_SPEED(R5) ; speed at 9600 ?
0087 372      BEQL   20$                ; branch if yes
0089 373
0089 374      MOVAB  TTY$AB_600,R4       ; set 600 baud table
008F 375
0090 375      CMPB   #TT$C_BAUD_600,UCBSW_TT_SPEED(R5) ; speed at 600 ?
0095 376      BEQL   20$                ; branch if no
0097 377
0097 378      MOVZBL #TT$C_BAUD_9600,R4    ; Wrong baud baud rate for sampling
009A 379      BSBW   SET_SPEED           ; Set to 9600
009D 380      BRB    50$                ; Exit
009F 381
009F 382      ; now test character
009F 383
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 98 00AC 393 30$: MOVZBW (R4),R4       ; set baud rate
      18 10 00AF 394  BSBW SET_SPEED      ; Set line speed
      00B1 395
64 A5 01 AA 00B1 396      BICW #UCBSM_TIM,UCBSW_STS(R5) ; Clear timeout expeted
53 0D 9A 00B5 397      CLR STATE <AUTOP>      ; Clear AUTOBAUD pending
00000000 GF D0 00B9 398      MOVZBL #TTY$C_CR,R3       ; Set carriage return as input
00B0 C5 00BC 399      MOVL G*EXESGL_ABSTIM,-   ; Set time this path taken
      00C2 400
      00C5 401
      00C5 402 40$:
      00C5 403 ;
      00C5 404 ; No appropriate mapping was found, make sure that character isn't a terminator
      00C5 405 ;
>3 20 9A 00C5 406 50$: MOVZBL #TTY$C_BLANK,R3 ; Invalidate character
      05 00C8 407      RSB
      00C9 408
00F4 C5 54 98 00C9 409 SET_SPEED:
      09A4 30 00CE 410      MOVZBW R4,UCBSW_IT_SPEED(R5) ; set speed, zero out speed+1
      05 00D1 411      BSBW TTY$SET_LINE ; Set line speed
      412      RSB ; Return
  
```

```

.OOD2 414 .SBTTL TTYSCANCELIO - CANCEL I/O ON TERMINAL
.OOD2 415
.OOD2 416 :
.OOD2 417 : ** TTYSCANCELIO - CANCEL I/O, DEASSIGN, AND DEALLOCATE
.OOD2 418 :
.OOD2 419 : FUNCTIONAL DESCRIPTION:
.OOD2 420 :
.OOD2 421 : THIS ROUTINE IS ENTERED TO STOP CURRENT I/O ON A TERMINAL UNIT.
.OOD2 422 :
.OOD2 423 : THE ASSOCIATED CHANNEL'S READ OR WRITE I/O IS CANCELED ALONG WITH
.OOD2 424 : A FLUSH OF CONTROL-C AST REQUESTS.
.OOD2 425 : IF THIS OPERATION IS DONE ON A UNIT WITH A ZERO REFERENCE COUNT, THE CONTROL
.OOD2 426 : Y ASTS ARE FLUSHED. THE TYPEAHEAD BUFFER IS DEALLOCATED AND THE UNIT
.OOD2 427 : IS INITIALIZED FOR THE NEXT USER.
.OOD2 428 :
.OOD2 429 : This routine always clears the control-0 state bit, and also clears
.OOD2 430 : the control-S state bit if the UCB reference count is zero.
.OOD2 431 :
.OOD2 432 : INPUTS:
.OOD2 433 :
.OOD2 434 : R2 = NEGATIVE OF THE CHANNEL NUMBER
.OOD2 435 : R3 = CURRENT I/O PACKET ADDRESS
.OOD2 436 : R4 = PCB OF CANCELING PROCESS
.OOD2 437 : R5 = UCB ADDRESS
.OOD2 438 :
.OOD2 439 : OUTPUTS:
.OOD2 440 :
.OOD2 441 : R4,R5 ARE PRESERVED
.OOD2 442 : --
.OOD2 443 :
.OOD2 444 TTYSCANCELIO::
.OOD2 445 : CANCELIO TTY USAGE
.OOD2 445 MOVL R2,R3 : SAVE THE CHANNEL NUMBER
.OOD2 446 BSBW TTY$LOCK : SETUP IPL AND REGISTERS
.OOD2 447 :
.OOD2 448 PUSHR #*M<R4,R6,R7,R9>
.OOD2 449 MOVL R3,R6 : COPY CHANNEL NUMBER
.OOD2 450 MOVL UCBSL_IRP(R5),R3 : GET THE CURRENT IRP
.OOD2 451 MOVL R5,R9 : SAVE LOGICAL UCB ADDRESS
.OOD2 452 MOVL UCBSL_TL_PHYUCB(R5),R5 : GET THE PHYSICAL UCB
.OOD2 453 BNEQ 5$ : CHECK AGAIN AFTER LOCKING THE DAT BASE
.OOD2 454 BRW CANCEL_DETACH
.OOD2 455 5$: PUSHAB CANCEL_DONE : EXIT TO HERE WHEN FINISHED
.OOD2 456 :
.OOD2 457 : Empty the write queue.
.OOD2 458 :
.OOD2 459 :
.OOD2 460 MOVAB UCBSL_TT_WFLINK(R5),R1 : Get first buffer in queue.
.OOD2 461 MOVL R1,R7 : Copy buffer address.
.OOD2 462 :
.OOD2 463 10$:
.OOD2 464 MOVL TTY$SL_WB_FLINK(R7),R7 : Empty write queue.
.OOD2 465 CMPL R7,R1 : Get next buffer.
.OOD2 466 BEQL 40$ : Is this the end of queue?
.OOD2 467 MOVL TTY$SL_WB_IRP(R7),R0 : Yes. Branch past loop.
.OOD2 468 BEQL 10$ : Get address of associated IRP.
.OOD2 469 : Branch if broadcast buffer
.OOD2 470 BBS #IRPSV_VIRTUAL,- : (leave these in queue).
.OOD2 470 : Don't dequeue virtual I/O

```

```

53 52 DO
FF28' 30
02D0 8F BB
56 53 DO
53 58 A5 DO
59 55 DO
55 00A0 C5 DO
03 12
0115 31
00000227'EF 9F
51 00CC C5 9E
57 51 DO
57 67 DO
51 57 D1
36 13
50 24 A7 DO
F2 13
04 E0

```

ED	2A	A0		010E	471			IRPSW_STS(R0),10\$:	packet.
	0C	A0	D1	0111	472	CMP		IRPSL_PID(R0),-	:	Compare packet's PID with
	60	A4		0114	473			PCBSL_PID(R4)	:	current process' PID.
		E6	12	0116	474	BNEQ		10\$:	No match. Go to next packet.
28	A0	56	B1	0118	475	CMP		R6,IRPSW_CHAN(R0)	:	Does the channel match?
		E0	12	011C	476	BNEQ		10\$:	No. Go to next packet.
		00	E1	011E	477	BBC		#IRPSV_BUFIO,-	:	Branch on direct I/O.
04	2A	A0		0120	478			IRPSW_STS(R0),20\$		
		02	AA	0123	479	BICW		#IRPSM_FUNC,-	:	Otherwise, clear function of
	2A	A0		0125	480			IRPSW_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			UCBSQ_BOFF(R5)	:	code.
		54	DD	012D	485	PUSHL		R4	:	Save R4
57	04	A7	DD	012F	486	MOVL		TTYSL_WB_BLINK(R7),R7	:	Get backward link.
54	00	B7	OF	0133	487	REMQUE		@TTY\$C_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				:	canceled process.
				013F	495					
				013F	496					
				013F	497		40\$:			
				013F	498					
				013F	499					
		08	E1	013F	500	BBC		#UCBSV_BSY,-	:	Check for read in progress.
	29	64	A9	0141	501			UCBSW_STS(R9),50\$:	Branch if no read in progress.
0C	A3	60	A4	D1	0144	CMP		PCBSL_PID(R4),IRPSL_PID(R3);	:	PID MATCH?
			12	0149	503	BNEQ		50\$:	Branch if no match.
28	A3	56	B1	014B	504	CMP		R6,IRPSW_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	DD	015D	511	MOVL		UCBSL_IRP(R5),R4	:	Get address of IRP.
	38	A4	D4	0161	512	CLRL		IRPSL_MEDIA(R4)	:	Clear terminators.
		2C	3C	0164	513	MOVZWL		#SS\$_ABORT,-	:	Set up abort status code.
	7C	A5		0166	514			UCBSQ_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 no write request is in
				016D	522				:	progress, branch.
57	00D4	C5	DD	0171	523	MOVL		UCBSL_TT_WRTBUF(R5),R7	:	Get address of write buffer.
50	24	A7	DD	0176	524	MOVL		TTYSL_WB_IRP(R7),R0	:	Get associated IRP.
		25	13	017A	525	BEQL		60\$:	no irp then don't cancel
	0C	A0	D1	017C	526	CMP		IRPSL_PID(R0),-	:	See if packet's PID matches
	60	A4		017F	527			PCBSL_PID(R4)	:	process' PID.

			0181	528	BNEQ	60\$:	Branch if no match.
28	A0	1E	B1	529	CMPW	R6,IRPSW_CHAN(R0)	:	See if channels match.
		56	12	530	BNEQ	60\$:	No. Don't stop this write.
		18	30	531	BSBW	TTY\$ABORT	:	Abort any current output activity
		0896	3C	532	MOVZWL	#SS\$ ABORT,-	:	Set abort status in the
		2C		533		UCBSQ_BOFF(R5)	:	UCB status word.
		7C	A5	534	PUSHR	#*M<R4>	:	Save the PCB address.
53		10	BB	535	MOVL	UCBSL_TT_WRTBUF(R5),R3	:	Get address of write packet.
		00D4	C5	536	MOVW	UCBSW_BOFF(R5),-	:	Put status code in the
		7C	A5	537		TTY\$WB_STATUS(R3)	:	packet.
		28	A3	538	BSBW	TTY\$WRITEDONE	:	Complete the I/O.
		FE61'	30	539	POPR	#*M<R4>	:	Restore PCB address.
		10	BA	540				
			01A1	541				
			01A1	542				
			01A1	543				
			01A1	544				
		002D	30	545	60\$: BSBW	CANCEL_AST	:	CANCEL OUTSTANDING ASTS
			01A4	546				
			01A4	547				
			01A4	548				
			01A4	549				
		5C	A9	550	TSTW	UCBSW_REFC(R9)	:	REF COUNT 0?
		10	12	551	BNEQ	65\$		
			01A9	552				
			01A9	553				
64	A9	01	8A	554	BICB	#UCBSM_TIM,UCBSW_STS(R9);	:	RESET HANGUP TIMER IN LUCB
			01AD	555				
		3C	E4	556	BBSC	#TTY\$V_SX_RECONNECT,-		
12	00B8	C5	01AF	557		UCBSQ_TT_STATE(R5),70\$		
			01B3	558				
		0079	30	559	BSBW	CANCEL_MODEM	:	SIGNAL HANGUP IF NEEDED
		009E	30	560	BSBW	CANCEL_RESET	:	RESET PHYSICAL UCB
			01B9	561	65\$:			
52	00B8	C5	DE	562	MOVAL	UCBSQ_TT_STATE(R5),R2	:	RESTORE STATE ADDRESS
			01BE	563	CLR_STATE	ZCTRL0>	:	Clear control-0 state always.
		FE3B'	31	564	BRW	TTY\$STARTOUTPUT	:	RESTART ANY STOPPED OUTPUT
			01C5	565				
			01C5	566				
			01C5	567	70\$:			
		54	DD	568	PUSHL	R4	:	PRESERVE R4
54		05	DD	569	MOVL	#TTY\$V_FD_LINK,R4	:	SCHEDULE LINK FORK FUNCTION
		045E	30	570	BSBW	TTY\$CRE_FORK		
		54	8ED0	571	POPL	R4		
			05	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 UCBSL_TL_CTRLC(R9),R7 : ADDRESS THE CONTROL C ENABLE LIST
52 0098 C9 D4 01D6 586 CLRL R2 : SPECIFY NULL MASK
00000000'GF 16 01D8 587 JSB G^COM$FLUSHATTNS : FLUSH THE LIST
01DE 588 :
01DE 589 : FLUSH OUT OF BAND ASTS
01DE 590 :
57 009C C9 9E 01DE 591 MOVAB UCBSL_TL_BANDQUE(R9),R7 : ADDRESS THE OUT OF BAND LIST
52 0098 C9 DE 01E3 592 MOVAL UCBSL_TL_OUTBAND(R9),R2 : GET CURRENT SUMMARY MASK
0000000C'GF 16 01E8 593 JSB G^COM$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 UCBSL_TL_CTRLY(R9),R7 : ADDRESS LIST HEAD FOR CONTROL Y ENABLF
52 0098 C9 C4 01F8 600 CLRL R2 : SPECIFY NULL MASK
00000000'GF 16 01FA 601 JSB G^COM$FLUSHATTNS : FLUSH LIST
00A4 C5 D4 0200 602 CLRL UCBSI_TL_CTLPID(R5) : RESET CONTROLLING PID
05 0204 603 10$:
0204 604 RSB
0205 605
0205 606
  
```

```

0205 608
0205 609
0205 610
0205 611
0205 612
0205 613
0205 614
0205 615
0205 616
0205 617
0205 618
0208 619
0208 620
020E 621
0210 622
0214 623
0217 624
021C 625
021E 626
0222 627
0226 628
0227 629
0227 630

PROCESS CANCEL FOR DETACHED LOGICAL UCB.
THIS ROUTINE CANCELS ANY INTERNAL STRUCTURES FOR
A DETACHED LUCB.

INPUTS: R9 - LUCB

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 0208 620 TSTW UCBSW_REFC(R5) ; CHECK FOR LAST DEASSIGN
12 12 020E 621 BNEQ 10$
64 A9 01 8A 0210 622 BICB #UCBSM_TIM,UCBSW_STS(R9); RESET HANGUP TIMER IN LUCB
6C A9 D4 0214 623 CLRL UCBSL_DUETIME(R9) ; RESET HANGUP TIMEOUT VALUE IN LUCB
50 0084 C5 D0 0217 624 MOVL UCBSL_PDT(R5),R0 ; CHECK FOR PENDING REBIND FUNCTION
04 13 021C 625 BEQL 10$ ; NONE
0084 C0 D4 021E 626 CLRL UCBSL_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 DO 0227 635      MOVL R9,R5      ; RESTORE LOGICAL UCB ADDRESS
02D0 8F BA 022A 636      POPR #^M<R4,R6,R7,R9> ; Restore registers.
                    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 :
03 48 02 E1 0231 653 BBC #TT2$V HANGUP,-
0C92 30 0234 654 UCBSL_DEVDEPND2(R5),15$ ; BRANCH IF LOGOUT/NOHANGUP
0237 655 BSBW CLASS_MODEM_DIS ; RESET SPEED AND UCB
0237 656 15$:
0237 657 :
0237 658 :
0237 659 : CHECK FOR NEED TO REACTIVATE MODEM PROCESSING
0237 660 :
0237 661 :
OE 44 A5 15 E0 0237 662 BBS #TT$V_MODEM,UCBSL_DEVDEPEND(R5),20$ ; CURRENT STATE IS MODEM
00C4 C5 15 E1 023C 663 BBC #TT$V_MODEM,UCBSL_IT_DECHAR(R5),25$ ; NO CHANGE
14 0241 :
51 00 9A 0242 664 MOVZBL #MODEM$C_INIT,R1 ; NEED TO RESTART MODEM
08F4 30 0245 665 BSBW TRANSITION
OC 11 0248 666 BRB 25$
024A 667 :
00C4 C5 15 E0 024A 668 20$: BBS #TT$V_MODEM,UCBSL_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 #UCBSM_TT_HANGUP,UCBSW_DEVSTS(R5); RESET HANGUP FLAG
0258 685 : ; NOTE THIS IS CORRECTLY REFERENCED
0258 686 : ; BUT IS NORMALLY MANIPULATED IN THE
0258 687 :
0258 688 : DELETE TYPEAHEAD BUFFER
0258 689 :
50 00E4 C5 D0 0258 690 MOVL UCBSL_TT_TYPAHD(R5),R0 ; GET ADDRESS OF TYPE AHEAD
00E4 0A 13 0260 691 BEQL 10$ ; NONE
00E4 C5 D4 0262 692 CLRL UCBSL_TT_TYPAHD(R5)
00000000'GF 16 0266 693 JSB G*COM$DRVDALMEM ; DEALLOCATE TYPE AHEAD BUFFER
00FC 30 026C 694 10$: BSBW TTY$PURGE_AHEAD ; PURGE TYPE AHEAD BUFFER
026F 695 :
026F 696 :
026F 697 : SET UP DISCONNECT SIGNAL TYPE FOR PORT
026F 698 :
7E 01 D0 026F 699 MOVL #1,-(SP) ; INDICATE NOHANGUP
02 48 A5 E1 0272 700 BBC #TT2$V_HANGUP,-
02 6E D4 0274 701 UCBSL_DEVDEPND2(R5),15$ ; BRANCH IF LOGOUT/NOHANGUP
0277 702 CLRL (SP) ; ASSUME HANGUP
0279 703 15$:
0279 704 :
0279 705 : OF LAST DEASSIGN
0279 706 : RESTORE PERM CHARACTERISTICS
0279 707 :
0279 708 :
00A8 C5 7C 0279 709 CLRQ UCBSQ_TL_BRKTHRU(R5) ; RESET ALL THE BREAKTHRU CLASSES
02E8 30 027D 710 BSBW TTY$SETUP_UCB ; SETUP UCB FIELDS, AND RESET PUCB
0280 711 :
00000000'GF D0 0280 712 MOVL G*EXE$GL_ABSTIM,-
0080 C5 0286 713 UCBSL_TT_RDUE(R5) ; SET TIME TO 'TURN OFF' AUTOBAUD
07E9 30 0289 714 BSBW TTY$SET_CTIME ; RESET SPEED AND UCB
028C 715 :
00 68 A5 00 E5 028C 716 BBCC #UCBSV_JOB,UCBSW_DEVSTS(R5),30$; RESET JOB CONTROLLER OWNERSHIP BIT
0291 717 30$:
7E 0122 C5 BC 0291 718 MOVW UCBSW_TT_PRTCTL(R5),-(SP); SAVE THE PORT CONTROL WORD
0040 8F AA 0296 719 BICW #TTY$M_PC_XOFENA,UCBSW_TT_PRTCTL(R5); MAKE SURE THAT THE CONTROL
0122 C5 029A :
029D 720 : AND THE DRIVER AGREE ON XOFF SETTING
0122 C5 07BC 30 029D 721 BSBW TTY$RESUME ; CONTINUE ANY PORT OUTPUT
02A0 722 MOVW (SP)+,UCBSW_TT_PRTCTL(R5)
02A5 723 :
52 00B8 C5 DE 02A5 724 MOVAL UCBSQ_TT_STATE(R5),R2 ; RESTORE STATE ADDRESS
02AA 725 CLR_STATE <CTRL0> ; Clear control-0 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
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$SETUP_READ::
54  2C A3  D0 02B5 763      MOVL  IRP$S_SVAPE(R3),R4      ; Get buffered block address.
      00FC C5  B0 02B9 764      MOVW  UCBSW_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  $$
      3A A4  B4 02C5 773      CLRW  TTY$W_RB_CPZORG(R4)      ; NO THEN LEAVE CPZORG HERE
      0A E1 02C8 774      BBC   #IOSV_DSABLMBX,-    ; ELSE ZERO THE CURSOR POSITION
      15 20 A3 02CA 775      IRP$W_FUNC(R3),10$      ; Branch forward if disable
      ; mailbox not requested.
02CD 776
51  00C0 C5  D0 02CD 777      MOVL  UCBSL_TT_LOGUCB(R5),R1      ; GET LUCB ADDRESS
      00010000 8F  C8 02D2 778      BISL  #TTSM_MBXDSABL,UCBSL_DEVDEPEND(R5)
      44 A5 02D8
      00010000 8F  C8 02DA 779      BISL  #TTSM_MBXDSABL,UCBSL_DEVDEPEND(R1)
      44 A1 02E0
02E2 780      :
02E2 781      : Set up read with time out if necessary.
02E2 782
02E2 783      :
02E2 784      :
02E2 785      :
02E2 786      :
02E2 787      :
      07 E1 02E2 784      BBC   #IOSV_TIMEC,-    ; 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      UCBSL_TT_RDUE(R5)      ; (meaning just empty typeahead
  
```

```

54 36 A4 3C 02F0 788
02F0 789 MOVZWL TTY$W_RB_TIMOS(R4),R4 ; buffer into read buffer).
02F4 790 ; Get number of seconds to wait
02F4 791 BEQL 20$ ; between each character.
02F6 792 ADDL3 R4,G*EXESGL_ABSTIM,- ; Branch if none.
00000000'GF 02F8 ; Calculate and store the due
00B0 C5 02FD 793 UCBSL_TT_RDUE(R5) ; time.
0300 794
0300 795 20$: BISW #UCBSM_TT_TIMO,- ; Set the 'this is a read with
02 A8 0300 796 UCBSW_DEVSTS(R5) ; timeout' bit.
68 A5 0302 797
0304 798
0304 799 ;
0304 800 ; Always turn off control-0 state.
0304 801 ;
0304 802
0304 803 30$: CLR_STATE - ; Always clear control-0 state.
0304 804 <CTRL0>
0304 805
0308 806
0308 807 ;
0308 808 ; Handle a readsync terminal.
0308 809 ;
0308 810
12 E1 0308 811 BBC #TT$V_READSYNC,- ; Branch forward if terminal is
OF 44 A5 030A 812 UCBSL_DEVDEPEND(R5),40$ ; not in readsync mode.
030D 813 IF_STATE - ; If in EOL state, don't issue
030D 814 <TYPFUL,EOL>,40$ ; an XON code.
07B6 30 0319 815 BSBW TTY$XON ; Send XON
031C 816
031C 817 40$:
05 031C 818 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 CONTROLLER
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$NOTIFY::
43 44 A5 50 DD 031D 840 PUSHL R0 ; SAVE A SCRATCH REGISTER
50 00C0 C5 D0 031F 841 BBS #TT$V_NOTYPEAHD,UCB$L_DEVDEPEND(R5),100$; IF SLAVED TERMINAL, IGNORE
5C A0 B5 0324 842 MOVL UCB$L_TT_LOGUCB(R5),R0 ; GET LOGICAL UCB ADDRESS
60 A0 D5 0329 843 TSTW UCB$W_REFC(R0) ; UNIT REF COUNT 0?
11 13 032C 844 BEQL 10$ ; IF EQL THEN JOB CONTROLLER POSSIBILITY
2F 44 A5 1C E0 032E 845 TSTL UCB$L_AMB(R0) ; USER ASSOCIATED MAILBOX?
2A 68 A5 02 E0 0331 846 BEQL 100$ ; IF EQL THEN NO
1E 38 A5 00 E0 0333 847 BBS #TT$V_MBXDSABL,UCB$L_DEVDEPEND(R5),100$; BR IF NOT ENABLED
00000000'GF 9E 0338 848 BBS #UCB$V_TT_NOTIF,UCB$W_DEVSTS(R5),100$; BR IF ALREADY NOTIFIED
55 50 D1 033D 849 BRB 15$ ; CONTINUE IN COMMON
08 13 033F 850 10$: BBS #UCB$V_JOB,UCB$W_DEVSTS(R5),100$; IF ALREADY NOTIFIED THEN GO
0000068'GF B5 0344 851 BBS #DEV$V_SPL,UCB$L_DEVCHAR(R5),100$; IGNORE IF DEVICE IS SPOOLED
18 15 0349 852 MOVAB G^OPAS$OCB0,R0 ; ADDRESS CONSOLE UCB
54 00 D0 034F 853 CMPL R0,R5 ; IS THIS THE CONSOLE
02C6 30 0353 854 BEQLU 15$ ; IF EQL THEN SKIP TERMINAL ENABLED TEST
18 BA 0355 855 TSTW G^SYSS$GL_JOBCTLMB+UCB$W_DEVSTS; TERMINALS ENABLED FOR JOBCTLR?
50 8ED0 035B 856 BLEQ 100$ ; IF LEQ THEN DISMISS
05 035D 857 15$: PUSHR #*M<R3,R4> ; SAVE THESE REGISTERS OVER THE FORK
035F 858 MOVL #TTY$V_FD_UNSQL,R4 ; ASK FOR UNSOLICITED DATA FORK
0362 859 BSBW TTY$CRE_FORK ; FORK TO SEND MESSAGE
0365 860 20$: POPR #*M<R3,R4>
0367 861 POPL R0
0367 862 100$: RSB
036A 863

```



```

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$PURGE_AHEAD::
54 00E4 C5 D0 036B 887 MOVL UCBSL_TT_TYPAHD(R5),R4 ; PURGE TYPEAHEAD
; GET TYPE AHEAD BUFFER ADDRESS
; IF EQL THEN NONE
036B 888 BEQL 55$
64 0118 C4 B4 0372 889 CLRW TTY$W_TA_INAHD(R4) ; SET NO DATA IN TYPEAHEAD BUFFER
0118 C4 9E 0375 890 MOVAB TTY$SL_TA_DATA(R4),TTY$SL_TA_PUT(R4); RESET PUT POINTER
0118 C4 9E 037A 891 MOVAB TTY$SL_TA_DATA(R4),TTY$SL_TA_GET(R4); RESET GET POINTER
037E
07 04 A2 0C E5 0380 892 50$: BBCC #TTY$V_ST_TYFUL,4(R2),55$; TYPE AHEAD FULL AND INPUT STOPPED?
0385 893 CLR STATE <OVRFL05 ; RESET OVERFLOW CONDITION
0746 30 0389 894 BSQB TTY$XON ; SEND XON
05 038C 895 55$: RSB
  
```

```

038D 897 .SBTTL TTY$READERROR - Line error occurred.
038D 898 :++
038D 899 : TTY$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$READERROR::
038D 920     PUSHR    #*M<R0,R4>                : SAVE REGISTERS
038D 921     MOVZWL   #SS$ PARITY,R4             : ASSUME PARITY
038D 922     MOVAB    UCBS$ TT STATE(R5),R2     : ADDRESS STATE DATA
038D 923     BBS      #12,R3,10$                : BRANCH IF PARITY ERROR
038D 924     BBS      #13,R3,TTY$FRAMERROR     : BRANCH IF FRAME ERROR
038D 925     MOVZWL   #SS$ DATAOVERUN,R4      : OVERUN ERROR
038D 926 10$:   MOVL    UCBS$ TT LOGUCB(R5),R1 : Get logical UCB address
038D 927     TSTW    UCBSW_REF$C(R1)           : Terminal in use ?
038D 928     BNEQ    12$                          : Branch if yes
038D 929     BBC     #TT2$V_AUTOBAUD,UCBS$L_DEVDEPND2(R5),12$ : Check AUTOBAUD
038D 930     CMPB    #TTY$C_CR,R3              : IS THIS CHARACTER A <CR>
038D 931     BEQL   11$                          : YES THEN DO AUTOPARITY
038D 932     BSBW    TTY$AUTOBAUD              : YES THEN JUMP THERE
038D 933     BRW    15$                          : RETURN TO THE USER
038D 934
038D 935 11$:   BBCC    #UCBSV_TT_PARTY,UCBSB_TT_PARITY(R5),70$ : CLEAR PARITY
038D 936     BBS    #UCBSV_TT_USERFRAME,UCBSB_TT_PARITY(R5),70$ : DON'T CHANGE THE FRAME
038D 937
038D 938     BISB    #UCBSM_TT_LEN,UCBSB_TT_PARITY(R5) : IF THE USER SPECIFIED IT.
038D 939 70$:   BSBW    TTY$SET_LINE           : SET EIGHT BIT FRAME SIZE
038D 940     BRW    15$                          : THEN TELL THE PORT
038D 941
038D 942 12$:   IF NOT STATE READ,90$          : SKIP IF NO READ ACTIVE
038D 943     BBC    #UCBSV_TT_DISPERR,UCBSB_TT_PARITY(R5),13$
038D 944     CMPL   R4,#SS$ PARITY            : IS THIS A PARITY ERROR
038D 945     BEQL   15$                          : YES THEN IGNORE IT
038D 946 13$:   MOVW    R4,UCBSW_BOFF(R5)     : SET ERROR CODE TO RETURN
038D 947     BSBW    TTY$READONE              : COMPLETE THE REQUEST
038D 948     BRB    15$
038D 949

```

```

11 BB
54 01F4 8F 3C
52 00B8 C5 9E
09 53 0C E0
7E 53 0D E0
54 0838 8F 3C
51 00C0 C5 D0
   5C A1 B5
   27 12
22 48 A5 01 E1
   53 0D 91
   06 13
   FCAF 30
   0042 31
   03C0 934
00F8 C5 06 E5
   08 03C5 935
00F8 C5 02 E0
   05 03C6 936
   03CB 937
00F8 C5 18 88
   06A1 30
   002B 31
   03D1 938
   03D4 939
   03D7 940
   03D7 941
00F8 C5 01 E1
   09 03D8 942
   D1 03E0 943
   54 03E1 944
000001F4 8F 03E3
   18 13 03E8 945
7C A5 54 80 03EA 946
   FCOF 30 03EE 947
   OF 11 03F1 948
   03F3 949

```

00000838	54	D1	03F3	950	90\$:	CMP	R4,#SS\$_DATAOVERUN	:	IS THIS A DATA OVERRUN
	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 SIGNALLED
			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\$ 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 TTYSFRAMERROR - FRAME ERROR ON A READ
041F 969 :++
041F 970 : TTYSFRAMERROR - FRAME READ ERROR OCCURED ON LINE
041F 971 :
041F 972 : Functional description:
041F 973 :
041F 974 : This routine is branched to from a TTYSREADERROR 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 TTYSFRAMERROR:
041F 992 :
041F 993 : Check if we should attempt autobaud on this line
041F 994 :
54 00C0 C5 D0 041F 995 MOVL UCBSL_TT_LOGUCB(R5),R4 ; Get logical UCB address
   SC A4 B5 0424 996 TSTW UCBSW_REFC(R4) ; Terminal in use ?
   OF 13 0427 997 BEQL 408 ; Branch if yes
E4 48 A5 10 F1 0429 998
   53 95 042E 1000 BBC #TT2$V_SECURE,UCBSL_DEVDEPND2(R5),18$ ; NOT SECURE THEN
   E0 12 0430 1001 ; GIVE ERROR
   0A9F 30 0432 1002 TSTB R3 ; NOT NULL DATA THEN NOT A BREAK
   FFCA 31 0435 1003 BNEQ 18$ ; SO GIVE AN ERROR
   0A 48 A5 01 E0 0438 1004 BSBW TTYSCLASS_DISCONNECT ; ELSE SIGNAL HANGUP STATUS
   53 95 0438 1005 BRW 15$ ; THEN RETURN NOTHING
   D1 12 043D 1006 408: BBS #TT2$V_AUTOBAUD,UCBSL_DEVDEPND2(R5),43$ ; Check AUTOBAUD
   FED9 30 043D 1007 TSTB R3 ; BREAK?
   FFBB 31 043F 1008 BNEQ 18$ ; NO THEN GIVE ERROR
   0441 1009 BSBW TTYSNOTIFY ; LET THE JOB CONTROLER KNOW ABOUT
   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$:
   00B0 C5 D1 0447 1017 CML UCBSL_TT_RDUE(R5),-
   00000000 GF 0448 1018 G^EXE$GL_ABSTIM ; check against current
   CO 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 TTYSAUTOBAUD and
   0452 1022 ; don't toggle baud rate
   0452 1023
   010B C5 01 90 0452 1024 MOVB #1,UCBSB_TT_OUTYPE(R5) ; ASSUME GOOD DATA
  
```

```

      53 95 0457 1025      TSTB  R3      ; test
      34 12 0459 1026      BNEQ  30$      ; Branch if data there
010R C5 94 045B 1027      CLRB  UCBSB_TT_OUTYPE(R5) ; BUT WE FOUND NO GOOD DATA
      045F 1028
      045F 1029 ; time is OK - continue by checking baud rate
      045F 1030
54  OF 9A 045F 1031      MOVZBL #TTSC_BAUD_9600,R4 ; Assume set to 9600 baud
  OF 91 0462 1032      CMPB  #TTSC_BAUD_9600,-
00F4 C5 0464 1033      UCBSW_TT_SPEED(R5) ; At 9600 ?
  1A 12 0467 1034      BNEQ  20$      ; no, set 9600
      0469 1035 ;
      0469 1036 ; Toggle to 600 baud for 2 t, 3 seconds
      0469 1037 ;
54  07 9A 0469 1038      MOVZBL #TTSC_BAUD_600,R4 ; otherwise, set 600 baud
      046C 1039 80$: SET_STATE -
      046C 1040      <AUTOP> ; Set state autobaud pending
      0470 1041      TIMSET 2
      0483 1042
00000000 FC43 30 0483 1043 20$: BSBW  SET SPEED ; Set line speed
0080 GF D0 0486 1044      MOVL  G^EXESGL_ABSTIM,-
0080 C5 048C 1045      UCBSL_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$RESTARTIO::
0492 1074 IF NOT STATE WRITE,10$ : RESTART THE I/O
0496 1075 IF STATE WRTALL,30$ : IF NOT WRITE THEN CONTINUE
049A 1076 : If passall mode, no special
: restart is required.
54 00D4 C5 D0 049A 1077 MOVL UCBSL TT WRTBUF(R5),R4 : Get address of write buffer.
: BSBW TTY$ABORT : Abort any output activity
: 2A A4 B4 04A2 1078 CLRW TTY$WB_BCNT(R4) : No. Clear output byte count.
: 30 A4 9E 04A5 1080 MOVAB TTY$WB_DATA(R4),- : And reset the start of data
: 1C A4 05 04A8 1081 TTY$WB_NEXT(R4) : pointer.
: 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 UCBSL IRP(R5),R3 : Get address of IRP.
: FDFB 30 04B3 1087 IF STATE RDVERIFY,40$ : NOTHING SPECIAL FOR READ VERIFY
: 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$V LOCALECHO, - : without local echo.
: 04C7 1094 UCBSL DEVDEPND2(R5),20$
: 04C7 1095 15$: IF NOT STATE = : Also no special handling if
: 04C7 1096 REFRESH,30$ : REFRESH state is not set.
: 0001 CF 9E 04CB 1097 MOVAB W*TTY$A_CTRLU+1,- : GET A <CR> TO MULTIECHO
: 00D8 C5 04CF 1098 UCBSL TT_MULTI(R5)
: 00DC C5 01 B0 04D2 1099 MOVW #1,UCBSW_TT_MULTILEN(R5) : AND ONLY 1 CHARACTER HERE.
: 04D7 1100 CLR_STATE -
: 04D7 1101 <CTRLR,SKIPCRLF>
: 04DB 1102 SET_STATE - : Set the control-R and
: 04DB 1103 <EDITREAD,MULTI,SKIPLF>; send a linefeed and then control-r the read
: 04E4 1104 20$:
: 04E4 1105 BBS #IOSV_REFRESH,- : If the read specified a
: 04E6 1106 IRPSW_FUNC(R3),30$ : refresh, don't cancel it.
: 04E9 1107 CLR_STATE - : Otherwise, cancel refresh

```

```

04E9 1108
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
54 7C A5 D0 04F4 1113 MOVL UCBSL_SVAPTE(R5),R4 ; GET THE READ BUFFER ADDRESS
78 A5 D4 04F8 1114 CLPL IRPSL_MEDIA(R3) ; ZERO ALL OTHER READ VERIFY FIELDS
38 A3 90 04F3 1115 MOVB TTY$W_RB_LINOFF(R4),- ; PUT THE INDEX INTO THE IOSB
30 A4 04FE 1116 IRPSL_MEDIA+3(R3)
3B A3 31 0500 1117 BRW TTY$READONE ; Go complete the read
FAFD' 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 scheduler)
0503 1124      :
0503 1125      : Functional description:
0503 1126      :
0503 1127      : This routine is called from the time dependent scheduler via the UCB
0503 1128      : offset UCBSL TT_RTIMOU whenever a timed read times out. It completes
0503 1129      : the read with time SSS_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$RTIMOU:
0503 1144     SETIPL #IPL$ TIMER          ; Read timed out entry
0506 1145     BSBW  TTY$LOCK        ; REDUCE IPL TO BEFORE LOCKING THE DATA BASE
0509 1146     MOVAB UCBSQ TT STATE(R5),R2 ; MAKE SURE THE UCB IS LOCKED
050E 1147     IF NOT_STATE READ,20$    ; Address state vector
0512 1148     MOVW  #SS$_TIMEOUT,UCBSW_BOFF(R5) ; IF THE AST IS LATE (I.E. THE READ IS
0516 1149     ; NO LONGER ACTIVE) THEN EXIT.
0518 1150     IF NOT_STATE RDVERIFY,5$   ; Set I/O completion status
051C 1151     MOVL  UCBSL_IRP(R5),R3    ; NOTHING SPECIAL FOR OTHER THAN READ VERIFY
0520 1152     MOVL  UCBSL_SVAPTE(R5),R4 ; GET THE IRP ADDRESS IN R3
0524 1153     CLRL  IRP$L_MEDIA(R3)    ; GET THE READ BUFFER ADDRESS
0527 1154     MOVB  TTY$W_RB_LINOFF(R4),- ; ZERO ALL OTHER READ VERIFY FIELDS
052A 1155     IRP$L_MEDIA+3(R3)          ; PUT THE INDEX INTO THE IOSB
052C 1156 5$:  BSBW  TTY$READONE     ; Go complete the read
052F 1157 10$: BRW  TTY$STARTOUTPUT ; then startup any waiting activity
0532 1158 20$: RSB
0533 1159

```

```

52  FAF7' 30
    00BB C5 9E
    022C BF B0
    7C A5
    58 A5 D0
    78 A5 D0
    38 A3 D4
    30 A4 40
    3B A3
    FAD1' 30
    FACE' 31
    05

```



```

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 $$$ HANGUP STATUS. IN THIS STATE THERE SHOULD BE
0533 1173 : NO CURRENT READ OR WRITE. LAST DEASSIGN CODE IN CANCEL
0533 1174 : WILL CLEAN UP THE LUCB, AND DEASSIGN WILL DELETE IT BACK TO POOL.
0533 1175 :
0533 1176 : INPUTS:
0533 1177 : INTERLOCKED AT DIPL
0533 1178 : R5 = LUCB ADDRESS
0533 1179 :
0533 1180 :--
0002' 0533 1181          .WORD TTY$VT_TIMEOUT-.
0533 1182 TTY$VT_TIMEOUT:
53    56 DD 0535 1183          PUSHL R6          ; SAVE SCRATCH REGISTER
      55 DO 0537 1184          MOVL R5,R6          ; LUCB REQUIRED IN R6
0084 C6 D0 053A 1185          MOVL UCBSL_PDT(R6),R3 ; GET ADDRESS OF ANY REBINDING PUCB
      04 13 053F 1186          BEQL 5$          ; NONE
0084 C3 D4 0541 1187          CLRL UCBSL_PDT(R3) ; CANCEL ANY PENDING REBIND OPERATION
      10 AA 0545 1188 5$:    BICW #UCBSM_ONLINE,- ; SET LUCB OFFLINE TO PREVENT ANY MORE IO
      64 A6 0547 1189          UCBSW_STS(R6)
0198 30 0549 1190          BSBW DISC FIREAST ; SIGNAL HANGUP AST TO OWNER
53    4C B6 OF 054C 1191 10$:  REMQUE @UCBSL_IOQFL(R6),R3 ; DRAIN IO QUEUE
      12 1D 0550 1192          BVS 20$          ; NO MORE PACKETS QUEUED
02CC 8F 3C 0552 1193          MOVZWL #$$$ HANGUP,- ; FILL IN COMPLETION STATUS
      38 A3 0556 1194          IRPSC MEDIA(R3)
02A 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$POST ; AND COMPLETE THE IO
      EB 11 0562 1197          BRB 10$
      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 (UCBSL_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$SETUP_UCB:: ; SET UP THE UCB (PORT INTERFACE)
50 00C0 C5 DD 0568 1233 POSHL R0 ; GET THE LOGICAL UCB ADDRESS
      09 13 056A 1234 MOVL UCBSL_TT_LOGUCB(R5),R0 ; NONE THEN INIT THE UCB
      5C A0 B5 056F 1235 BEQL 3$ ; IS THIS UCB ACTIVE?
      04 13 0571 1236 TSTW UCBSW_REFC(R0) ; NO THEN INIT IT
      50 BED0 0574 1237 BEPL 3$ ; IF ACTIVE THEN RETURN NORMALLY
      05 0576 1238 POPL R0
      05 0579 1239 RSB
00C0 C5 55 DD 057A 1240 3$: MOVL R5,UCBSL_TT_LOGUCB(R5) ; INIT LOGICAL UCB ADDRESS
00A0 C5 55 DD 057F 1241 MOVL R5,UCBSL_TL_PHYUCB(R5) ; INIT PHYSICAL UCB
00040000 8F C8 0584 1242 BISL #DEVSM_AVL,- ; SHOW PUCB AVAILABLE FOR USE
      38 A5 058A 1243 UCBSL_DEVCHAR(R5)
00000100 8F CA 058C 1244 BICL #DEVSM_RED,UCBSL_DEVCHAR2(R5) ; SHO PUCB NOT REDIRECTED
      3C A5 0592
00F4 C5 D4 0594 1245 CLRL UCBSL_TL_CTLPID(R5) ; CLEAN OUT A LEFT OVER PID
00F8 C5 01 B0 0594 1246 MOVW #1,UCBSW_TT_CURSOR(R5) ; SET BAD CURSOR TO FORCE CRLF
0108 C5 B4 0598 1247 CLRW UCBSW_TT_HOLD(R5) ; INIT HOLDING TANK
064D CF 9E 059D 1248 MOVAB W^TTY$FORK_ADDR,UCBSL_FPC(R5); SET UP FORK ADDRESS
      0C A5 05A5
00E8 C5 D0 05A7 1249 MOVL UCBSW_TT_DESPEE(R5),UCBSW_TT_SPEED(R5); SET UP TEMP CHARS
00F4 C5 05AB
00EC C5 90 05AE 1250 MOVB UCBSB_TT_DEPARI(R5),UCBSB_TT_PARITY(R5);SET UP DEFAULT PARITY
00F8 C5 05B2
00F0 C5 D0 05B5 1251 MOVL UCBSB_TT_DETYPE(R5),UCBSB_DEVTYPE(R5);
      41 A5 05B9
FFFFDFFF 8F CB 05BB 1252 BICL3 #^C<TTSM_REMOTE>,UCBSL_DEVDEPEND(R5),R0; SAVE CURRENT REMOTE STATE
      05BB 1255

```

-1

```

50 44 A5 05C1
00002000 8F CB 05C4 1257 BICL3 #TTSM_REMOTE,UCBSL_TT_DECHAR(R5),UCBSL_DEVDEPEND(R5);
00C4 C5 05CA
44 44 A5 05CD
44 A5 50 C8 05CF 1258 BICL RO,UCBSL_DEVDEPEND(R5) ; KEEP PREVIOUS SETTING
44 A5 FC 8F 8B 05D3 1259 BICB3 #^C<TTSM_PASSALL!TTSM_NOECHO>,UCBSL_DEVDEPEND(R5),RO; GET PASSALL AN
50 05D8
05D9 1260
05D9 1261 ;
05D9 1262 ; Set up default DMA characteristics
00C8 C5 D0 05D9 1263 MOVL UCBSL_TT_DECHA1(R5),UCBSL_DEVDEPND2(R5) ; SET UP DEFAULT CHAR WORD 2
48 A5 05DD
1E 48 A5 06 E1 05DF 1264 BBC #TT2$V_DMA,UCBSL_DEVDEPND2(R5),8$ ; SKIP IF NOT SET
0122 C5 02 E0 05E4 1265 BBS #TTY$V_PC_DMAAVL,UCBSW_TT_PRTCTL(R5),5$ ; DMA FEATURE AVAILABLE
00000040 8F CA 05EA 1266 BICL #TT2$M_DMA,UCBSL_TT_DECHA1(R5) ; SHOW DMA NOT AVAILABLE
00C8 C5 05F0
00000040 8F CA 05F3 1267 BICL #TT2$M_DMA,UCBSL_DEVDEPND2(R5) ; IF DMA IS NOT AVAILABLE
48 A5 05F9
05 11 05FB 1268 BRB 8$
0122 C5 02 AB 05FD 1269 5$:
0602 1270 BISW #TTY$M_PC_DMAENA,UCBSW_TT_PRTCTL(R5) ; ENABLE DMA USAGE
0602 1271 8$:
02 02 50 F0 0602 1272 INSV RO,#TTY$V_ST_PASALL,#2,UCBSQ_TT_STATE+4(R5); RESET CURRENT STATE
00BC C5 0606
50 8ED0 0609 1274 POPL RO
060C 1275
060C 1276 ;
060C 1277 ; Initialize the write queue.
060C 1278 ;
060C 1279 ;
00CC C5 D5 060C 1280 TSTL UCBSL_TT_WFLINK(R5) ; Is the queue initialized?
0E 12 0610 1281 BNEQ 10$ ; Yes. Don't redo it.
00CC C5 9E 0612 1282 MOVAB UCBSL_TT_WFLINK(R5),- ; Initialize queue to point to
00CC C5 0616 1283 UCBSL_TT_WFLINK(P5) ; itself in both directions.
00CC C5 9E 0619 1284 MOVAB UCBSL_TT_WFLINK(R5),- ; Now the backward link.
00D0 C5 061D 1285 UCBSL_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,UCBSL_TT_RTIMOU(R5) ; Set read timed out d'spatch
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 :--
54 03 D0 0628 1317 TTY$CLASS_FORK:: : CREATE A FORK FOR THE TERMINAL PORT
0628 1318 MOVL #TTY$V_FD_PORTFORK,R4 : SET THE PORT FORK BIT.
0628 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$CRE_FORK::
062B 1345 BBSS #TTY$V_FD_BUSY,UCB$L_FR4(R5),20$ : CREATE FORK
0630 1346 BBSS R4,UCB$L_FR4(R5),3$ : ARE WE BUSY
0635 1347 3$: MOVL UCB$L_FR4(R5),R4 : SET THE APROPRIATE BIT
0639 1348 5$: PUSHL UCB$L_PID(R5) : AND GET THE BIT INTO R4
063C 1349 PUSHAB B^TTY$FORK_ADDR : SAVE THE PID ON THE STACK
063F 1350 JMP TTY$SYNCH : SET FORK ROUTINE ADDRESS
0645 1351 : CREATE FORK PROCESS
0645 1352 : FORK ALREADY ACTIVE
0645 1353 :
0645 1354 20$: BBSS R4,UCB$L_FR4(R5),25$ : SET THE APROPRIATE BIT
05 064A 1355 25$: RSB : AND RETURN TO THE CALLER

```

```

15 14 A5 06 E2
00 14 A5 54 E2
54 14 A5 D0
2C A5 DD
4D AF 9F
00000000 EF 17
00 14 A5 54 E2
05 064A 1355 25$ RSB

```

```

0357' 064B 1357
      064B 1358          .SBTTL FORK DISPATCHER
      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 UCBSu_TT_STATE(R5),R2 ; ADDRESS UNIT STATE
      0652 1365
      0652 1366          DSBINT UCBSB_DIPL(R5)          ; DISABLE INTERUPTS FOR THE CHECK
      0659 1367          FFS #0,#TTY$V_FD_BUSY+1,UCBSL_FR4(R5),R4; FIND THE FIRST BIT SET
      065C
00 14 A5 54 E4 065F 1368 15$: BBSC R4,UCBSL_FR4(R5),15$ ; CLEAR THAT BIT
      0664 1369          ENBINT ; ENABLE INTERUPTS
      0667 1370
      0667 1371          PUSHAL 10$ ; SAVE A RETURN ADDRESS
      066A 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:
      067C 1382          POPL R4 ; POP OFF THE ADDRESS OF THE DISPATCHER
      067F 1383          RSB ; THEN RETURN
      0680 1384
      0680 1385          PORTFORK:
51 0118 C5 D0 0680 1386          MOVL UCBSL_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      : Description:
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      : R3 = PUCb address
0688 1402      :
0688 1403      : Outputs:
0688 1404      :
0688 1405      : R5 = PRESERVED
0688 1406      :--
0688 1407      DISCONNECT:
56 0040 8F BB 0688 1408      PUSHR  #*M<R6>
00C0 C5 D0 068C 1409      MOVL  UCBSL_TT_LOGUCB(R5),R6 ; GET LOGICAL UCB ADDRESS
0691 1410
0691 1411      DSBINT UCBSB_DIPL(R5)
0387 30 0698 1412      BSBW  TTYSABORT ; ABORT ANY OUTPUT,
03BE 30 0698 1413      BSBW  TTYSRESUME ; AND RESET CONTROL S
069E 1414      ENBINT
00 44 A5 OD E5 06A1 1415
44 A6 44 A5 D0 06A1 1416      BBCC  #TT$V_REMOTE,UCBSL_DEVDEPEND(R5),10$ ;RESET REMOTE TERMINAL
06A6 1417 10$: MOVL  UCBSL_DEVDEPEND(R5),UCBSL_DEVDEPEND(R6) ;UPDATE MODIFIED FIELD
06AB 1418
08 48 A5 11 E1 06AB 1419      BBC  #TT2$V_DISCONNECT,UCBSL_DEVDEPN2(R5),15$;SKIP IF DISCONNECT NOT AVA
56 55 D1 06B0 1420      CMPL  R5,R6 ; MAKE SURE ITS REASONABLE
03 13 06B2 1421      BEQL  15$ ; NO, LUCB/PUCB THE SAME
0028 31 06B5 1422      BRW  50$
0688 1423
0688 1424 15$:
002C 30 0688 1425      BSBW  DISC_FIREAST ; FIRE HANGUP AST
02CC 8F 3C 0688 1426      MOVZWL #SS$HANGUP,UCBSW_BOFF(R5) ; SET THE RETURN STATUS
7C A5
0056 30 06BF 1427      BSBW  DISC_STOPIO ; STOP CURRENT I/O
06C4 1428
06C4 1429
54 06 D0 06C4 1430      MOVL  #MSG$TRMHANGUP,R4 ; SET MESSAGE TYPE
55 56 D0 06C7 1431      MOVL  R6,R5 ; SWITCH TO LUCB CONTEXT
53 60 A5 D0 06CA 1432      MOVL  UCBSL_AMB(R5),R3 ; GET ASSOC MAILBOX
06 13 06CE 1433      BEQL  20$ ; 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  UCBSL_TL_PHYUCB(R5),R5 ; SWITCH BACK TO PUCB CONTEXT
0040 8F BA 06DB 1437      POPR  #*M<R6>
05 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 :
  
```

```

54 0090 C6 9E 06E7 1460 DISC_FIREAST:
50 54 D0 06EE 1461 DSBINT UCBSB_DIPL(R6) : INTERLOCK TO DEVICE IPL
60 D5 06F3 1462 MOVAB UCBSL_TL_CTRLY(R6),R4 : ADDRESS CONTROL Y AST LIST
09 12 06F6 1464 MOVL R4,R0 : COPY LIST ADDRESS
5C A6 B5 06F8 1465 TSTL (R0) : LIST HEAD EMPTY?
04 13 06FA 1466 BNEQ 5$ : NO,PROCESS HANGUP ASTS
08 A8 06FD 1467 TSTW UCBSW_REFC(R6) : IS THIS UNIT OWNED OR ACTIVE?
68 A6 06FD 1468 BEQL 5$ : (OR IS IT LOGGED OUT)
50 60 D0 06FF 1469 BISW #UCBSM_TT_HANGUP,- : NOT ACTIVE,SO SKIP FLAG SETTING
02CC 8F 3C 0701 1470 UCBSW_DEVSTS(R6) : SET HANGUP FLAG TO NOTIFY NEXT CTRL-Y
1C A0 0703 1471 : ATTENTION AST WHEN IT IS QUEUED.
00000000'GF 11 0703 1472 5$: MOVL (R0),R0 : GET NEXT ENTRY
16 0706 1473 BEQL 10$ : IF EQL THEN DONE
0710 1475 MOVZWL #SS$ _HANGUP,ACBSL_KAST+4(R0); INSERT NEW PARAM FOR AST
0716 1476 BRB 5$ : CONTINUE UNTIL DONE
0719 1477 10$: JSB G^COM$DELATTNAST : DELIVER THE AST'S
071A 1478 ENBINT
071A 1481 RSB
  
```

-1

-1


```

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 UCBSB_DIPL(R5) : INTERLOCK TO DEVICE IPL
52 00B8 C5 9E 0721 1494 MOVAB UCBSQ_TT_STATE(R5),R2 : SET UP LUCB STATE POINTER
51 00CC C5 9E 0726 1495 MOVAB UCBSL_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$:
53 63 D0 072E 1499 MOVL TTYSL_WB_FLINK(R3),R3 : Get next entry.
51 53 D1 0731 1500 CMPL R3,R1 : At end of queue?
OD 13 0734 1501 BEQL 15$ : Yes. Branch forward.
0736 1502 : Leave it in the queue.
53 04 A3 D0 0736 1503 MOVL TTYSL_WB_BLINK(R3),R3 : Get entry's backward link.
54 00 B3 OF 073A 1504 REMQUE @TTYSL_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$:
0743 1509 IF_NOT_STATE - : Stop the current I/Os.
0743 1510 READ,20$ : Branch forward if no read is
0747 1511 CLR_STATE - : currently in progress.
0747 1512 <ESC,BADESC> : Clear escape bits.
54 58 A5 D0 074D 1513 MOVL UCBSL_IRP(R5),R4 : Get address of IRP.
38 A4 D4 0751 1514 CLRL IRPSL_MEDIA(R4) : Clear terminators.
F8A9' 30 0754 1515 BSBW TTY$READONE : Complete the read.
0757 1516 20$:
0757 1517 IF_NOT_STATE - : Check for a write to complete.
0757 1518 WRITE,30$ : Branch forward if no write is
075B 1519 MOVL UCBSL_TT_WRTBUF(R5),R3 : in progress.
0760 1520 MOVW UCBSW_BOFF(R5) - : Get address of write packet.
0763 1521 TTY$WB_STATUS(R3) - : Put status code in the
0765 1522 BSBW TTY$WRITEDONE : packet.
0768 1523 30$:
0768 1524 ENRINT : Complete the I/O.
05 076B 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 : UCBSL_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 UCBSL_TT_TYPAHD(R5) : ALREADY ALLOCATED?
64 12 0770 1550 BNEQ 100$ : IF NEQ THEN YES
07 E1 0772 1551 BBC #TT2$V_ALTYPAHD,- : SKIP IF NORMAL SIZE
09 4E A5 0774 1552 UCBSL_DEVDEPND2(R5),72$
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^EXESALONONPAGED : ALLOCATE THE TYPEAHEAD BUFFER
54 52 D0 0794 1558 MOVL R2,R4 : COPY BLOCK ADDRESS
3C 50 E9 0797 1559 BLBC R0,100$ : IF FAILURE THEN TOUGH
08 A4 51 B0 079A 1560 MOVW R1,TTY$W_TA_SIZE(R4) : SET UP SIZE
10 A4 6441 9E 079E 1561 MOVAB (R4)[R1],TTY$L_TA_END(R4) : SET UP END POINTER
0A A4 : 90 07A3 1562 MOVB #DYN$C_TYPAHD,TTY$B_TA_TYPE(R4);
14 A4 B4 07A7 1563 CLRW TTY$W_TA_RCLSIZ(R4) : CLEAR OUT THE RECALL SIZE
07AA 1564 DSBINT UCBSB-DIPL(R5) : LOCK TERMINAL DATA BASE
00E4 C5 54 D0 07B1 1565 MOVL R4,UCBSL_TT_TYPAHD(R5) : STORE ADDRESS
52 00B8 C5 9E 07B6 1566 MOVAB UCBSQ_TT_STATE(R5),R2 : RESTORE THE UNIT STATE VECTOR ADDRESS
FBAD 30 07BB 1567 BSBW TTY$PORGE_AHEAD : SET UP THE BLOCK
10 A5 90 07BE 1568 MOVB UCBSL_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
FB32' 30 07CB 1572 BSBW TTY$STARTOUTPUT : BEGIN NEXT STATE
00 14 A5 01 E4 07CE 1573 80$: BBSC #TTY$V_FD_GETAHD,UCBSL_FR4(R5),90$: 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 UCBSL_PDT. THIS ROUTINE RESETS BSY IN UCBSW_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      UCBSL_PDT(R5),R4          ; GET SPECIFIED LUCB
0B A4 0B A5 90 07DC 1596      MOVVB     UCBSB_FIPL(R5),UCBSB_FIPL(R4)      ; FORK IPL
5E A4 5E A5 90 07E1 1597      MOVVB     UCBSB_DIPL(R5),UCBSB_DIPL(R4)    ; DEVICE IPL
64 A4 01 8A 07E6 1598      BICB      #UCBSM_TIM,UCBSW_STSTR4)    ; AND TURN OFF TIMER BIT IN
00020000 8F C8 07EA 1599      BISL      #TT2SM_DISCONNECT,UCBSL_DEVDEPND2(R5) ; SET DISCONNECT BIT
48 A5 07F0 1601
44 A4 44 A5 D0 07F2 1602      MOVL      UCBSL_DEVDEPEND(R5),UCBSL_DEVDEPEND(R4) ; BECAUSE WE ARE DISCONNECTA
48 A4 48 A5 D0 07F7 1603      MOVL      UCBSL_DEVDEPND2(R5),UCBSL_DEVDEPND2(R4) ; MAKE SURE LUCB IS
00A0 C4 55 D0 07FC 1604      MOVL      R5,UCBSL_TL_PHYUCB(R4) ; IMAGE COPY OF PUCB
00C0 C5 54 D0 0801 1605      MOVL      R4,UCBSL_TT_LOGUCB(R5) ; LINK LUCB TO PUCB
00A0 C5 55 D0 0806 1606      MOVL      R5,UCBSL_TL_PHYUCB(R5) ; LINK PUCB TO LUCB
0084 C4 D4 080B 1607      CLRL     UCBSL_PDT(R4) ; SHOW PUCB=PUCB
0084 C5 D4 080F 1608      CLRL     UCBSL_PDT(R5) ; RESET REBIND FIELDS
0813 1609      CLRL     UCBSL_PDT(R5) ; RESET REBIND FIELDS
3C A4 02 CA 0813 1610      BICL     #DEVSM_DET,UCBSL_DEVCHAR2(R4) ; LUCB NOW NOT DETACHED
00040000 8F CA 0817 1611      BICL     #DEVSM_AVL,- ; SHOW PUCB NOT AVAlABLE FOR
38 A5 081D 1612
00000100 8F C8 081F 1613      BISL     #DEVSM_RED,UCBSL_DEVCHAR2(R5) ; SHO PUCB REDIRECTED
3C A5 0825 1614
55 54 D0 0827 1615      MOVL     R4,R5 ; SWITCH TO LUCB CONTEXT
53 4C B5 OF 082A 1616      REMQUE  @UCBSL_IOQFL(R5),R3 ; GET A PACKET OFF QUEUE
0C 1D 082E 1617      BVS     10$ ; QUEUE EMPTY
00000842'EF DF 0830 1618      PUSHAL 20$ ; SAVE RETURN ADDRESS
00000000'GF 17 0836 1619      JMP     C^IOC$INITIATE ; START THE PACKET
0100 8F AA 083C 1620      BICW    #UCBSM_BSY,UCBSW_STS(R5) ; RESET BSY BIT IN LUCB
64 A5 0840 1621
55 00A0 C5 D0 0842 1622      MOVL     UCBSL_TL_PHYUCB(R5),R5 ; TO ALLOW NEW OUTPUT
05 0842 1623      RSB ; SWITCH BACK TO PUCB
0847 1624      ; DONE
0848 1625

```

:MIR000,
:MIR0001
-1

```

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 UCBSL_TT_LOGUCB(R5),R4 ; GET LOGICAL UCB ADDRESS
52 00B8 C5 9E 084D 1647 MOVAB UCBSQ_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$
085A .1 TSTW QLEN ; CHECK TO SEE IF THE FORK QUEUE
0860 .2 BNEQ 30$ ; IS EMPTY NO THEN WAIT SOME MORE
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 :
0862 1657 BISW #UCBSM_BSY,UCBSW_STS(R4) ; FORCE BSY
0866
0868 1658 CLRL UCBSL_IRP(R4) ; SHOW NO I/O ACTIVE
0868 1659 CLRL UCBSL_TL_PHYUCB(R4) ; AND BREAK LINK TO PUCB
00C0 C5 55 D0 086F 1660 MOVL R5,UCBSL_TT_LOGUCB(R5) ; AND MAKE PUCB=LUCB
3C A4 02 C8 0874 1661 BISL #DEVSM_DET,UCBSL_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 TTYSVT_TIMEOUT,UCBSL_FPC(R4) ; LOAD VECTOR FOR TIMEOUT ENTRY
0C A4 087C
00000000'GF C1 087E 1667 ADDL3 G^TTYSGL_TIMEOUT,G^EXESGL_ABSTIM,-; AND SET TIMER
00000000'GF 0884
6C A4 01 88 0889 1668 UCBSL_DUETIME(R4)
088B 1669 BISB #UCBSM_TIM,UCBSW_STS(R4) ; AND TURN ON TIMER BIT IN LUCB
088F 1670
088F 1671
088F 1672 :
088F 1673 : IF REBINDING STATE, THEN GOTO LINKFORK
08 00B8 C5 E4 088F 1673 BBSC #TTYSV_SX_RECONNECT -
0891 1674 UCBSQ_TT_STATE(R5),20$
0895 1675
0895 1676 BSBW CANCEL_MODEM ; CANCEL THE MODEM (HAI.SUP IF
0898 1677 ; NECESSARY)
F997 30 0895 1676 ; GO RESET PUCB. THIS WILL ALSO
0898 1677 ; MAKE IT AVAILABLE
F9BB CF 16 0898 1678 JSB CANCEL_RESET
089C 1679

```

```

05 089C 1680 RSB
FF37 31 089D 1681
089D 1682 20$: BRW LINKFORK ; GO LINK PUCB TO REQUESTED LUCB
08A0 1683
08A0 1684
08A0 1685 ;
08A0 1686 30$: STOP IO ON PUCB PRIOR TO DISCONNECT
7C A5 01 3C 08A0 1687 MOVZWL #SS$ _NORMAL,UCB$W_BOFF(R5) ; SET THE RETURN STATUS
FE73 30 08A4 1688 BSBW DISC_STOPIO ; STOP CURRENT IO
06 E5 08A7 1689 BBCC #TTY$V_FD_BUSY,- ; FORCE NEW FORK TO QUEUE BEHIND ALL OTHERS
GO 14 A5 08A9 1690 UCBSL_FR4(R5),35$
54 04 DO 08AC 1691 35$: MOVL #TTY$V_FD_UNLINK,R4 ; AND QUEUE THE FORK
FD79 30 08AF 1692 BSBW TTY$CRE_FORK
54 8ED0 08B2 1693 POPL R4 ; POP OFF TERMINAL FORK CONTROL RETURN PC
05 08B5 1694 RSB ; AND EXIT
08B6 1695
08B6 1696 ;
08B6 1697 40$: REQUEUE ANY ACTIVE READ TO THE HEAD OF THE QUEUE
53 58 A5 DO 08BD 1699 DSBINT UCBSB_DIPL(R5) ; INTERLOCK WITH DEVICE INTERRUPTS
08C1 1700 MOVL UCBSL_IRP(R5),R3 ; GET IRP OF ACTIVE READ
08C1 1701 ;
08C1 1702 INIT READ STATUS FIELDS
16 68 A5 01 E4 08C1 1703 BBSC #UCBSV_TT_TIMC,UCB$W_DEVSTS(R5),50$; RESET ANY TIMED READ STATE
08C6 1704
4C A4 63 OE 08C6 1705 INSQUE (R3),UCBSL_IOQFL(R4) ; AND PUT AT HEAD OF THE QUEUE
08CA 1706
08CA 1707 CLR_STATE -
08CA 1708 <READ,DEL,EOL,- ; RESET READ STATE BITS, BUT LEAVE 'BSY' SET
08CA 1709 PROMPT,CTRLR,NOFLTR,-
08CA 1710 ESC,ESC_O,BADESC,PRE,-
08CA 1711 REFRSH,EDITREAD,SKIPCRLF,RDVERIFY,-
08CA 1712 MULTI,RECALL,OVERSTRIKE,EDITING,-
08CA 1713 QUOTING,BACKSPACE,TERMNORM>
08D7 1714
08D7 1715 45$: ENBINT
C4 11 08DA 1716 BRB 30$ ; AND FULSH COMPLETE ANY WRITES
08DC 1717
02CC 8F 3C 08DC 1718 50$: MOVZWL #SS$ _HANGUP,UCB$W_BOFF(R5) ; SET THE RETURN STATUS
7C A5 08E0
F71B' 30 08E2 1719 BSBW TTY$READONE ; AND COMPLETE THE READ
FO 11 08E5 1720 BRB 45$
08E7 1721
08E7 1722

```

```

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 controller indicating that
08E7 1730 : unsolicited 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:
55 0060 8F BB 08E7 1747 : PUSHR #^M<R5,R6> : SAVE REGISTERS
56 00C0 C5 D0 08EB 1748 : MOVL UCBSL_IT_LOGUCB(R5),R5 : LOAD LUCB ADDRESS
56 00A0 C5 D0 08F0 1749 : MOVL UCBSL_TL_PHYUCB(R5),R6 : LOAD PUCB ADDRESS
54 01 9A 08F5 1750 :
54 01 9A 08F5 1751 : MOVZBL #MSG$ TRMUNSOLIC,R4 : SET MESSAGE TYPE
5C A5 B5 08F8 1752 : TSTW UCBSW_REFC(R5) : IF REF COUNT 0 THEN
5C A5 B5 08FB 1753 : BEQL 50$ : JOB CONTROLLER
5C A5 B5 08FD 1754 : MOVL UCBSL_AMB(R5),R3 : GET USER'S MAILBOX
5C A5 B5 0901 1755 : BEQL 65$ : IF EQL THEN NONE
2E 68 A6 02 E0 0903 1756 : BBS #UCBSV_TT_NOTIF,UCBSW_DEVSTS(R6),65$; BR IF ALREADY NOTIFIED
2E 68 A6 14 11 0908 1757 : BRB 55$ : CONTINUE
090A 1758 50$:
00000000'GF D0 090A 1759 : MOVL G^TTY$GL_JOBCTLMB,R3 : ADDRESS JOB CONTROLLER MAILBOX
20 68 A6 00 E0 0911 1760 : BBS #UCBSV_JOB,UCBSW_DEVSTS(R6),65$; BR IF NOTIFIED ALREADY
0916 1761 :
0916 1762 : BBC #TT2$V_DISCONNECT,- : SKIP IF UNIT NOT ABLE TO DISCONNECT
03 48 A6 E1 0918 1763 : UCBSL_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
OF 50 E9 0924 1768 : BLBC R0,65$ : IF FAILED THEN LEAVE DATA BASE UNCHANGED
5C A5 B5 0927 1769 : TSTW UCBSW_REFC(R5) : REF COUNT 0?
5C A5 B5 092A 1770 : BNEQ 60$ : IF NEQ THEN USER
68 A6 01 A8 092C 1771 : BISW #UCBSM_JOB,UCBSW_DEVSTS(R6); SET NOTIFIED
68 A6 04 11 0930 1772 : BRB 65$ : LOOK TO UNSOLICITED DATA
68 A6 04 A8 0932 1773 60$: BISW #UCBSM_TT_NOTIF,UCBSW_DEVSTS(R6); SET NOTIFIED
0060 8F BA 0936 1774 65$: POPR #^M<R5,R6> : RESTORE REGISTERS AND SWITCH BACK TO PUCB
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:
00000000'EF 18 BB 093B 1799 PUSH  #^M<R3,R4>
55 55 DO 093D 1800 MOVL  VTSUCB,R5 ; GET ADDRESS OF TEMPLATE UCB ADDRESS
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 DO 094F 1804 MOVL  R2,R5 ; GET NEW LUCB ADDRESS
5C A5 B4 0952 1805 CLRW  UCBSW_REF C(R5) ; INIT REFERENCE COUNT
0955 1806
00010000 8F C8 0955 1807 BISL  #UCBSM_DELETEUCB,- ; MARK THIS LUCB TO BE DELETED ON LAST DEASS
64 A5 095B 1808 UCBSL STS(R5)
00040000 8F CA 095D 1809 BICL  #DEVSM_AVL,- ; SHOW PUCB NOT AVAILABLE FOR USE
38 A6 0963 1810 UCBSL DEVCHAR(R6)
00000100 8F C8 0965 1811 BISL  #DEVSM_RED,UCBSL_DEVCHAR2(R6) ; SHO PUCB REDIRECTED
3C A6 096B
096D 1812
096D 1813
096D 1814 :
096D 1815 : INIT LUCB FIELDS
41 A5 41 A6 90 096D 1816 MOVB  UCBSB_DEVTYPE(R6),UCBSB_DEVTYPE(R5) ; TERMINAL TYPE
0B A5 0B A6 90 0972 1817 MOVB  UCBSB_FIPL(R6),UCBSB_FIPL(R5) ; FORK IPL
5E A5 5E A6 90 0977 1818 MOVB  UCBSB_DIPL(R6),UCBSB_DIPL(R5) ; DEVICE IPL
42 A5 42 A6 B0 097C 1819 MOVW  UCBSW_DEVBUFSIZ(R6),UCBSW_DEVBUFSIZ(R5) ; BUFFER SIZE
44 A5 44 A6 D0 0981 1820 MOVL  UCBSL_DEVDEPEND(R6),UCBSL_DEVDEPEND(R5) ; DEVICE CHARACTERISTICS
48 A5 48 A6 D0 0986 1821 MOVL  UCBSL_DEVDEPN2(R6),UCBSL_DEVDEPN2(R5) ;
5F A5 5F A6 90 098B 1822 MOVB  UCBSB_AMOD(R6),UCBSB_AMOD(R5) ; DEVICE PROTECTION
00A0 C5 56 D0 0990 1823 MOVL  R6,UCBSL_TL_PHYUCB(R5) ; LINK LUCB TO PUCB
00C0 C6 55 D0 0995 1824 MOVL  R5,UCBSL_TT_LOGUCB(R6) ; LINK PUCB TO LUCB
03 11 099A 1825 BRB   55$
55 56 D0 099C 1826 50$: MOVL  R6,R5 ; ERROR, NO LOGICAL UCB CREATED
18 BA 099F 1827 55$: POPR
05 09A1 1828 RSB
09A2 1829
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 UCBSL FPC ON POWERFAIL OR TIMEOUT.
09A2 1839 : THE ACTION IS TO RESTART THE I/O IF TRE 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:
09A2 1852         BSBW  TTY$LOCK          ; ATTENTION INTERRUPT
09A2 1853         BBC   #UCBSV_POWER,UCBSW_STS(R5),20$; CANCEL IF NOT POWERFAIL
09AA 1854
09AA 1855 :
09AA 1856 : POWER FAILED WHILE THIS UNIT WAS WAITING FOR AN INTERRUPT.
09AA 1857 :
09AA 1858
09AA 1859         BSBW  RESTART          ; Restart in progress I/O.
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
09B4 1866         MOVZBW  #TTSC_BAUD_9600,-
09B6 1867         UCB$W_TT_SPEED(R5)          ; SET 9600 BAUD
09B9 1868         MOVB   UCB$B_TT_DEPARI(R5),UCB$B_TT_PARITY(R5); RESET THE PARITY
09BD 1869
09C0 1869         BSBW  TTY$SET_LINE        ; RESET SPEED ON LINE,
09C3 1870         RSB   ; SINCE TIMER EXPIRED
09C3 1871
09C4 1872 :
09C4 1873 :
09C4 1874 : ACTUAL HARDWARE TIMEOUT
09C4 1875 :
09C4 1876 :
09C4 1877 50$:
09C4 1878         INCW   UCB$W_ERRCNT(R5)      ; Handle timeout.
09C8 1879         MOVZWL #SS$ TIMEOUT,-      ; BUMP ERROR COUNTER
09CC 1880         UCB$W_BOFF(R5)              ; Load status code into the
09CE 1881         ; UCB status word.
09CE 1882 60$:
09CE 1883         IF_NOT_STATE -                ; Branch forward if not in a
09CE 1884         WRITE,70$                      ; write state.
09D2 1885         BSBW  TTY$ABORT            ; Abort any output activity
09D5 1886         MOVL  UCBSL_TT_WRTBUF(R5),R3 ; Get address of write buffer.
09DA 1887         MOVW  UCB$W_BOFF(R5),-      ; Save status in the write
09DD 1888         TTY$W_WB_STATUS(R3)          ; packet.

```

06 64 A5 F65B' 30
05 E1

004A 30
F650' 31

OF 98
00F4 C5
00EC C5 90
00F8 C5
00B2 30

00B2 C5 86
022C 8F 3C
7C A5

53 004D 30
00D4 C5 D0
7C A5 B0
28 A3 09DD 1888

	F61E'	30	09DF	1889	BSBW	TTY\$WRITEDONE	; Complete this write.
			09E2	1890			
			09E2	1891			
			09E2	1892			; Check for a read.
			09E2	1893	IF_NOT_STATE -		; Branch forward if not reading.
			09E6	1894	READ,80\$		
			09E6	1895	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\$M-MEDIA(R4)	; SET NO TERMINATORS
	F60A'	30	09F3	1898	BSBW	TTY\$READONE	; COMPLETE READ
			09F6	1899			
		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  #UCBSM POWER,UCBSW_STS(R5); RESET POWERFAIL
52  00B8 C5 9E 09FB 1922          MOVAB UCBSQ 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
    FAB1 CF 17 0A06 1925          CLR_STATE POWER ; RESET POWER FAIL STATE
    0A09 1926          SET_STATE REFRSH ; FORCE READS TO BE REFRESHED
    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 interrupt expected bit and setup
0A11 1938      : a timer to call us back during the first pass of the timer service 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$POWERACTION::
0A11 1952      BICW  #UCBSM_INT,UCBSW_STS(R5)
0A11 1953      BISW  #UCBSM_TIM,UCBSW_STS(R5)
0A11 1954      CLRL  UCBSL_DUETIME(R5)
0A11 1955      BISL  #TTY$M_ST_POWER,UCBSQ_TT_STATE(R5)
0A11 1956      RSB
0A11 1957

```

```

64 A5 02
64 A5 01
   6C A5
00BB C5 01

```

```

AA
AB
D4
C8
05

```

```

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::
00BB C5 01 BB 0A22 1978          PUSHR  #^M<R0>          ; SAVE A REGISTER
0A22 1979          BBCC   #TTY$V_ST_CTRL$UCB$Q_TT_STATE(R5),- ; ALLOW CLASS OUTPUT
0A29 1980          S$
0A2A 1981          S$:
50 0118 C5 D0 0A2A 1982          MOVL  UCB$L_TT_PORT(R5),R0      ; GET THE ADDRESS OF THE PORT
      20 B0 16 0A2F 1983          JSB   @PORT_ABORT(R0)      ; GOTO THE ROUTINE
      01 BA 0A32 1984          POPR  #^M<R0>
0A34 1985          RSB
0A35 1986          ; and return
  
```

```

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

```

```

0A42 2003
0A42 2004 : TTYSDS_SET - SFT 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 TTYSDS_SET::
50 0118 01 BB 0A42 2016 PUSHR #*M<R0> ; SAVE A REGISTER
OC 0C B0 D0 0A44 2017 MOVL UCBSL_TT_PORT(R5),R0 ; GET THE ADDRESS OF THE PORT
01 01 BA 0A49 2018 JSB @PORT_DS_SET(R0) ; GOTO THE ROUTINE
05 0A4E 2019 POPR #*M<R0>
0A4E 2020 RSB ; and return

```

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

```

0A5C 2032 :
0A5C 2033 : TTY$RESUME - CONTINUE OUTPUT ON A LINE
0A5C 2034 :
0A5C 2035 TTY$RESUME::
00BC CS 01 BB 0A5C 2036 PUSHR #^M<R0> ; SAVE A REGISTER
          1D EO 0A5E 2037 BBS #TTY$V_ST_CTSLOW,UCB$Q_TT_STATE+4(R5),10$; IF CLEAR TO SEND IS LOW
          OE 0A63
00BB CS 01 E5 0A64 2038 ; THEN DON'T RESUME YET
          00 0A64 2039 BBCC #TTY$V_ST_CTRL$UCB$Q_TT_STATE(R5),- ; ALLOW CLASS OUTPUT
          0A69 2040 5$
          0A6A 2041 5$:
50 0118 CS DJ 0A6A 2042 MOVL UCB$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$SET_LINE - SET SPEED AND PARITY
0A75 2048 :
0A75 2049 : IMPLICIT INPUTS:
0A75 2050 : UCBSB_TT_PARITY - CONTAINS PARITY, STOP BIT AND FRAME SIZE INFO
0A75 2051 : UCBSW_TT_SPEED - LOW BYTE XMI SPEED
0A75 2052 : HIGH BYTE ZERO OR RECIEVE SPEED
0A75 2053 : UCBSB_TT_PRTCTL - DMA AND AUTOXON ENABLE FLAGS
0A75 2054 :
0A75 2055 TTY$SET_LINE::
50 0118 01 BB 0A75 2056 PUSHR #^M<R0> ; SAVE A REGISTER
08 08 B0 D0 0A77 2057 MOVL UCBSL_TT_PORT(R5),R0 ; GET THE ADDRESS OF THE PORT
01 01 BA 16 0A7C 2058 JSB @PORT_SET_LINE(R0) ; GOTO THE ROUTINE
05 05 BA 0A7F 2059 POPR #^M<R0>
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:
50 0118 01 BB 0A82 2067 PUSHR #*M<R0> ; SAVE A REGISTER
28 B0 16 D0 0A84 2068 MOVL UCBSL TT_PORT(R5),R0 ; GET THE ADDRESS OF THE PORT
01 01 BA 0A89 2069 JSB @PORT_SET_MODEM(R0) ; GOTO THE ROUTINE
05 0A8E 2070 POPR #*M<R0>
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$V ST_CTRL$ =
00 00B8 C5 0A93 2078 UCBSQ_TT_STATE(R5),5$ ; BLOCK NEW OUTPUT
50 0118 C5 0A97 2079 5$:
18 B0 16 0A97 2080 MOVL UCBSL_TT_PORT(R5),R0 ; GET THE ADDRESS OF THE PORT
01 BA 0A9C 2081 JSB @PORT_STOP(R0) ; GOTO THE ROUTINE
05 OAA1 2082 POPR #^M<R0>
RSB ; and return

```

```

00040010 8F D3 OAA2 2085
          44 A5 E0 OAA2 2086 : TTY$XOFF - CALCULATE CHARACTER TO SEND FOR XOFF AND SEND IT TO THE PORT
          EB 12 E0 OAA2 2087 : TTY$XON - SAME AS XOFF FOR XON FLOW CONTROL.
          F1 11 E0 OAA2 2088 .ENABLE LSB
          OA BB OAA2 2089 TTY$XOFF::
          53 13 9A OAA2 2090         PUSHR #^M<R1,R3>           ; SAVE REGISTERS
          19 00B8 C5 E1 OAA4 2091         MOVZBL #TTY$C_XOFF,R3           ; USE XOFF
          09 44 A5 2C OAA7 2092         BBC #TTY$V_SX_TYPFUL -
          08 00B8 C5 E0 OAA9 2093         UCBSQ_TT STATE(R5),30$ ; SKIP IF NOT TYPEAHEAD FULL
          53 07 9A OAAD 2094         BBS #TTY$V_HOSTSYNC,UCB$L_DEVDEPEND(R5),10$ ; XOFF CORRECT
          51 0118 C5 E0 OAB2 2095         BBS #TTY$V_SX_PASALL -
          14 B1 16 OAB4 2096         UCBSQ_TT STATE(R5),20$ ; SKIP IF PASSALL MODE
          OA BA OAB8 2097         MOVZBL #TTY$C_BELL,R3           ; USE BELL CHARACTER
          10 0118 C5 D0 OABB 2098 10$:  MOVL  UCB$L_TT_PORT(R5),R1 ; GET THE ADDRESS OF THE PORT VECTOR
          14 B1 16 OAC0 2099         JSB @PORT_XOFF(R1)
          OA BA OAC3 2100 20$:  POPR  #^M<RT,R3>
          05 OAC5 2101         RSB
          00040010 8F D3 OAC6 2102
          44 A5 12 OAC6 2103 30$:  BITL  #TTY$M_HOSTSYNC!TTY$M_READSYNC,-
          EB 12 OACC 2104         UCBSL_DEVDEPEND(R5) ; HOST OR READ SYNC TERMINAL?
          F1 11 OACE 2105         BNEQ 10$ ; YES THEN OUTPUT THE CHARACTER
          OA BB OAD0 2106         BRB 20$ ; NO, SKIP OUTPUT
          53 11 9A OAD2 2107
          00040010 8F D3 OAD2 2108 TTY$XON::
          44 A5 13 OAD2 2109         PUSHR #^M<R1,R3>           ; SEND XON
          51 0118 C5 D3 OAD4 2110         MOVZBL #TTY$C_XON,R3           ; SEND XON
          10 B1 16 OAD7 2111         BITL  #TTY$M_HOSTSYNC!TTY$M_READSYNC,-
          OA BB OADD 2112         UCBSL_DEVDEPEND(R5) ; HOST OR READ SYNC TERMINAL?
          14 B1 13 OADF 2113         BEQL 20$ ; NO, SKIP OUTPUT
          10 B1 16 OAE1 2114         MOVL  UCB$L_TT_PORT(R5),R1 ; GET THE ADDRESS OF THE PORT VECTOR
          10 B1 16 OAE6 2115         JSB @PORT_XON(R1)
          10 B1 11 OAE9 2116         BRB 20$ ; GO BACK AND RETURN
          OA EB OAE8 2117 .DISABLE LSB

```

```

      0AEB 2119      .SBTTL  MODEM ROUTINES
      0AEB 2120
      0AEB 2121      ;      MODEM CONTROL DATA AREA
      0AEB 2122
      0AEB 2123
      0AEB 2124
00000000 0AEB 2125 TTSL_DIALUP::      ;ROOT OF MODEM TIMER CRB LIST
      0AEB 2126      .LONG  0
      0AEF 2127 TTSW_REFcnt:
0000      0AEF 2128      .WORD  0      ;COUNT OF TIMERS IN USE
      0AF1 2129 TTSW_TIMctrl:
0000      0AF1 2130      .WORD  0      ;FLAG WORD
      0AF3 2131
      0AF3 2132
      0AF3 2133      .ALIGN  QUAD
00000B28 0AF8 2134 TT$TIMQUENT:      ;TIMER QUE ENTRY FOR MODEM TIMER
      0AF8 2135      .BLKB  TQESC_LENGTH      ;ALLOCATE BLOCK HERE
      0B28 2136      STO_TQE TQESC_SIZE,WORD,TQESC_LENGTH,TT$TIMQUENT
      0B28 2137      STO_TQE TQESC_TYPE,BYTE,DYN$C-TQE,TT$TIMQUENT
      0B28 2138      STO_TQE TQESC_RQTYPE,BYTE,TQESC_SSREPT,TT$TIMQUENT
      0B28 2139
      0B28 2140

```

```

OB28 2142 :++
OB28 2143 : TRANSITION_NOCHECK - process transition element without the devdepend bit cc heck
OB28 2144 : TRANSITION - PROCESS TRANSITION ELEMENTS ON MODEM/TIMER STATUS EVENT
OB28 2145 : PORT_TRANSITION - PROCESSES PORT CALLS THEN FALLS INTO TRANSITION
OB28 2146 :
OB28 2147 : FUNCTIONAL DESCRIPTION:
OB28 2148 :
OB28 2149 : THIS ROUTINE IS CALLED ON ALL MODEM TRANSITIONS SUCH AS MODEM SIGNAL CHANGES,
OB28 2150 : TIMER EXPIRATIONS AND CONTROL. THE TRANSITION ELEMENTS CONTAINED IN THE
OB28 2151 : STATE TABLE ASSOCIATED WITH THE CURRENT STATE ARE SCANNED. IF A MATCH IS
OB28 2152 : FOUND, THEN A NEW STATE IS DECLARED. UPON DECLARING A NEW STATE,
OB28 2153 : OUTPUT MODEM SIGNALS MAY BE SET OR RESET, TIMERS MAY BE STARTED OR
OB28 2154 : CANCELLED AND ACTION ROUTINES MAY BE INVOKED.
OB28 2155 :
OB28 2156 : PORT_TRANSITION
OB28 2157 : THIS ROUTINE PRE-PROCESSES THE CALLS FROM THE PORT DRIVERS
OB28 2158 : TO SCREEN OUT CALLS TO MODEM INIT WITH REF-COUNT NON-ZERO AND CHANGE
OB28 2159 : THOSE CALLS INTO SHUTDOWN CALLS. ALL INPUTS AND OUTPUTS ARE THE SAME
OB28 2160 : AS TRANSITION.
OB28 2161 :
OB28 2162 : INPUTS:
OB28 2163 :
OB28 2164 : R1 = TRANSITION TYPE
OB28 2165 : R2 = TYPE SPECIFIC ARGUMENT
OB28 2166 : DATASET - NEW RCV MODEM MASK
OB28 2167 : R5 = UCB ADDRESS
OB28 2168 :
OB28 2169 : OUTPUTS:
OB28 2170 : R0-R4 DESTROYED
OB28 2171 :--
OB28 2172 PORT_TRANSITION::
51 00 D1 OB28 2173 Cmpl #MODEMSC_INIT,R1 ; IS THIS AN INIT?
53 00C0 C5 D0 OB28 2174 BNEQ TRANSITION ; NO THEN DON'T INTERPRET
08 13 OB2D 2175 MOVL UCBSL TT LOGUCB(R5),R3 ; GET THE LOGICAL UCB ADDRESS
5C A3 B5 OB32 2176 BEQL TRANSITION ; NO LOGICAL THEN EXIT
03 13 OB34 2177 TSTW UCBSW REFC(R3) ; ANY REFCOUNT IN LOGICAL
51 01 9A OB37 2178 BEQL TRANSITION ; NO THEN INIT THE LINE
0126 C5 B5 OB39 2179 MOVZBL #MODEMSC_SHUTDWN,R1 ; ELSE SHUT IT DOWN
05 12 OB3C 2180 TRANSITION::
47 44 A5 15 E1 OB3C 2181 TSTW UCBSW TT DS ST(R5) ; IS MODEM STATE IDLE?
00000D7B'EF DE OB40 2182 BNEQ TRANSITION_NOCHECK ; NO, SO ALLOW CLEANUP TO FINISH
53 0842 2183 BBC #TTSV_MODEM,UCBSL_DEVDEPEND(R5),END;DON'T BOTHER IF NOT MODEM
54 0126 C5 3C OB47 2184 TRANSITION_NOCHECK::
54 53 CO OB47 2185 MOVAL STATE_INIT+MODEMSC_ST_LENGTH,R3 ;ROOT OF STATE TABLE + OFFSET
01EA 30 OB4D 2186 MOVZWL UCBSW TT_DS_ST(R5),R4
00000007'8F D0 OB53 2187 ADDL R3,R4
52 0864 2188 CASE R1,TYPE=B,<INIT,SHUTDWN,TRAN_LOOP,TRAN_LOOP,TRAN_LOOP>
008C 31 OB64 2189 INIT:
000000BE'8F D0 OB64 2190 BSBW MODEMSLINK CRB ;LINK INTO CRB CHAIN
52 0867 2191 MOVL #IDLE-STATE_INIT,R2
0082 31 OB6E 2192 BRW DECLARE_STATE ;INIT MODEM CONTROL
0871 2193 SHUTDWN:
0871 2194 MOVL #SHUTDOWN-STATE_INIT,R2 ;DECLARE SHUTDOWN STATE
0877 2195 BRW DECLARE_STATE ;TRANSITION TO NEW STATE
  
```

```

53 64 9A 0B7B 2196 TRAN_LOOP:
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 #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$B_TT_DS_RCV(R5) ;UPDATE STATUS
0B9A 2215 10$:
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$B_TRAN_ONMASK(R4),R3 ;GET COMPLEMENT OF MASK
52 52 53 8A 0BAF 2220 BICB R3,R2 ;ISOLATE BITS OF INTEREST
05 A4 8C 0BB2 2221 XORB MODEM$B_TRAN_ONMASK(R4),R2 ;TEST IF BOTH ON
13 13 0BB6 2222 BEQL 20$ ;YES
0B88 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
0B88 2230 20$:
002B 31 0BCB 2231 BRW NEW_STATE ;ENTER NEW STATE
0BCE 2232
0BCE 2233
0BCE 2234 DIALTYPE:
05 A4 93 0BCE 2235 BITB MODEM$B_TRAN_ONMASK(R4),- ;CHECK FOR MATCH
00000000 GF 1B 13 0BD1 2236 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$B_CRB(R5),R1 ;GET CRB ADDRESS
42 8F 0B A1 91 0BDF 2241 CMPB CRB$B_TT_TYPE(R1),#DTS_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 #TTSV_MODEM,UCB$B_DEVDEPEND(R5),-
03 03 0BED 2246 10$ ;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

```



```

OC36 2305 :++
OC36 2306 : MODEM_TIMER - SET/RESET MASTER MODEM TIMER ROUTINE
OC36 2307 :
OC36 2308 : FUNCTIONAL DESCRIPTION:
OC36 2309 :
OC36 2310 : THIS ROUTINE STARTS OR CANCELS A MODEM PROTECOL TIMER FOR A LINE.
OC36 2311 : ONE TQE IS USED TO TIME ALL ACTIVE LINES. THE TQE ONLY REMAINS ACTIVE
OC36 2312 : SO LONG AS AT LEAST ONE TIMER IS NEEDED. SUBSEQUENT TIMERS USE THE
OC36 2313 : SAME TQE. WHEN ALL CURRENT TIMERS EXPIRE THE TQE IS REMOVED FROM
OC36 2314 : THE SYSTEM TIMER QUEUE UNTIL NEEDED AGAIN.
OC36 2315 : SINCE TIMERS ARE ONLY NEEDED DURING TRANSITIONS, THIS IS A LOW OVERHEAD
OC36 2316 : ACTIVITY.
OC36 2317 :
OC36 2318 : INPUTS:
OC36 2319 :
OC36 2320 : R2 = NEW TIMEOUT VALUE OR 0 TO CANCEL ANY OUTSTANDING
OC36 2321 : R5 = UCB ADDRESS
OC36 2322 :
OC36 2323 : OUTPUTS:
OC36 2324 : R0-R3 DESTROYED
OC36 2325 :
OC36 2326 : MODEM_TIMER:
54 24 54 DD OC36 2327 PUSHL R4
OC36 2328 MOVL UCBSL_CRB(R5),R4 ;CRB ADDRESS
OC36 2329 TSTL R2 ;CANCEL?
OC36 2330 BEQL TIMER_CANCEL ;YES
OC40 2331
OC40 2332 :NEED TO INIT TIMER FOR LINE
OC40 2333 TSTW UCBSW_TT_DS_TIM(R5) ; ANY CURRENT TIMER ACTIVE
OC44 2334 BNEQ 58 ; YES THEN JUST CHANGE THE TIME
OC46 2335 INCB CRBSB_TT_TIMREFC(R4) ; BUMP THE REFCOUNT IN THE CRB
OC49 2336 INCW TTSW_REFCNT ; BUMP IN USE COUNTER
OC4D 2337 58: MOVW R2,UCBSW_TT_DS_TIM(R5) ;SET TIME TO WAIT
OC52 2338
OC52 2339 BICW #TIMCTRLSM_CANCEL,TTSW_TIMCTRL ;RESET ANY CANCEL REQUEST
OC57 2340 BBC #TIMCTRLSV_ACTIVE,TTSW_TIMCTRL,108 ;TIMER NOT CURRENTLY ACTIVE
OC5C 2341
OC5D 2341 BRB TIMER_END
OC5F 2342 108:
OC5F 2343 PUSHL R5 ;NEED TO RESTART TIMER
OC61 2344 MOVAL TTSTIMQUENT,R5 ;SAVE UCB ADDRESS
OC66 2345 MOVW #IPLS_QUEUEAST,TQESB_RQTYPE(R5) ;SET UP FORK IPL
OC6A 2346 PUSHAB 208 ;RETURN ADDRESS
OC70 2347 BRB 308 ;QUEUE FORK
OC72 2348
OC72 2349 208: POPL R5 ;RESTORE UCB ADDRESS
OC75 2350 EISW #TIMCTRLSM_ACTIVE,TTSW_TIMCTRL ;SHOW TIMER ACTIVE
OC7A 2351 BRB
OC7C 2352 308:
OC7C 2353 JSB G^EXESFORK
OC82 2354 DSB:NT #IPLS_SYNCH ;INTERLOCK TO ENTER TIMER ENTRY
OC88 2355 MOVAB W^,TSTIMER,TQESL_FPC(R5) ;SET TIMEOUT ROUTINE ADDRESS
OC8C 2356
OC8E 2356 MOVL #500000,TQESQ_DELTA(R5) ;SET REPEAT AMOUNT (500MS)
OC94 2357
OC96 2357 MOVW #TQESC_SSREPT,TQESB_RQTYPE(R5) ;SET REPEATING TYPE
OC9A 2358 MOVW G^EXESGQ_SYSIME,R0 ;CURRENT SYSTEM TIME
  
```



```

OCDS 2385 :++
OCDS 2386 : TT$TIMER - TIMER INTERRUPT ROUTINE
OCDS 2387 :
OCDS 2388 : FUNCTIONAL DESCRIPTION:
OCDS 2389 :
OCDS 2390 : THIS ROUTINE IS INVOKED UPON THE EXPIRATION OF THE MODEM PROTECOL TQE.
OCDS 2391 : IT COUNTS DOWN ALL ACTIVE TIMERS AND INVOKES THE TRANSITION
OCDS 2392 : ROUTINE FOR ANY LINES THAT HAVE TIMED OUT. THE ROUTINE REQUEUES THE
OCDS 2393 : TQE INTO THE SYSTEM TIMER QUEUE UNLESS ALL ACTIVE TIMERS HAVE
OCDS 2394 : EXPIRED.
OCDS 2395 :
OCDS 2396 : INPUTS:
OCDS 2397 :
OCDS 2398 : R5 = TQE ADDRESS
OCDS 2399 :
OCDS 2400 : OUTPUTS:
OCDS 2401 : R0-R4 DESTROYED
OCDS 2402 :
OCDS 2403 : TT$TIMER:
54 FE10 CF BB OCDS 2404 : PUSHR #^M<R5>
DE OGD7 2405 : MOVAL TTSL_DIALUP,R4 ;GET ROOT OF CRB LIST
OCDC 2406 10$:
54 64 DO OCDC 2407 : MOVL (R4),R4 ;GET NEXT CRB ADDRESS
52 13 OCDF 2408 : BEQL 60$ ;PROCESS LINES FOR CMB
OB A4 95 OCE1 2409 : TSTB CRBSB_TT_TIMREFC-CRBSL_TT_MODEM(R4); ANY TIMERS ON THIS LINE?
F6 13 OCE4 2410 : BEQL 10$ ; NO THEN EXIT
OCE6 2411 :
OCE6 2412 : TIMERS ARE ACTIVE ON THIS CONTROLER
OCE6 2413 :
54 54 DD OCE6 2414 : PUSHL R4 ;SAVE TIMER THREAD
14 C2 O.EB 2415 : SUBL #CRBSL_TT_MODEM,R4 ;GET ACTUAL CRB ADDRESS
O.EB 2416 :
OCEB 2417 : ;PROCESS LINES WITH ACTIVE TIMERS
53 2C A4 DO OCEB 2418 : MOVL CRBSL_INTD+VECSL_IDB(R4),R3 ;GET ADRESS OF IDB
52 OC A3 3C OCEF 2419 : MOVZWL IDBSW_UNITS(R3),R2 ; GET THE NUMBER OF UNITS
52 D7 OCF3 2420 : DECL R2 ; ON THIS CONTROLER (ZERO BASED)
OCE5 2421 20$:
55 18 A342 DO OCF5 2422 : MOVL IDBSL_UCBLST(R3)[R2],R5 ;GET UCB FOR THAT LINE
10 13 OCFA 2423 : BEQL 35$ ; NO UNIT THEN GO ON TO THE NEXT
OCFC 2424 : DSBINT UCBSB_DIPL(R5) ;INTERLOCK WITH DEVICE INTERRUPTS
0128 C5 B5 OD03 2425 : TSTW UCBSW_TT_DS_TIM(R5) ; DOES THIS UNIT HAVE THE TIMER
OC 12 OD07 2426 : BNEQ 50$ ; YES THEN HANDLE IT
OD09 2427 30$: ENBINT ; NO MORE THEN ENABLE THE INTERRUPTS
E6 52 F4 OD0C 2428 35$: SOBGEQ R2,20$ ; NO THEN GO ON
54 8ED0 OD0F 2429 40$: POPL R4 ; RESTORE TIMER THREAD
FFC7 31 OD12 2430 : BRW 10$
OD15 2431 :
OD15 2432 : DECREMENT/FIRE TIMER REQUESTS
OD15 2433 :
51 04 90 OD15 2434 50$: MOVB #MODEMSC_TIMER,R1 ;SET TRANSITION TYPE
0128 C5 B7 OD18 2435 : DECW UCBSW_TT_DS_TIM(R5) ;COUNT DOWN LINE
EB 12 OD1C 2436 : BNEQ 30$ ;NOT TIME YET
FDCD CF B7 OD1E 2437 : DECW TT$W_REFcnt ;
1C BB OD22 2438 : PUSHR #^M<R2,R3,R4> ;
FE15 30 OD24 2439 : BSBW TRANSITION ;SAVE SOME REGISTERS
1C BA OD27 2440 : POPR #^M<R2,R3,R4> ;INVOKE TRANSITION ROUTINE
1F A4 B7 OD29 2441 : DECW CRBSB_TT_TIMREFC(R4) ; DECREMENT THE REFCOUNT
  
```

TT
Psc

PSE

Pha

In
Com
Pas
Syn
Pas
Syn
Pse
Crc
Ass

The
212
The
275
65

Mac

338
The
MAC

```

DB 12 OD2C 2442      BNEQ 30$      ; CONTINUE
      OD2E 2443      ENBINT
DC 11 OD31 2444      BRB 40$      ; RE-ENABLE INTERRUPTS IF
      OD33 2445      ; NO MORE TIMERS ON THIS
      OD33 2446      ; CONTROLLER AND MOVE TO THE NEXT
      OD33 2447      ; CONTROLLER
      OD33 2448      ; CHECK TO SEE IF WE ARE TO CANCEL THE TIMER OR RE-ENABLE IT
      OD33 2449      ;
20 BA OD33 2450 60$: POPR #M<R5>
      OD35 2451      DSBINT #31      ; INTERLOCK
FDB1 CF 00 E. OD38 2452      BBC #TIMCTRLSV_CANCEL,TTSW_TIMCTRL,70$;DONT CANCEL TIMER
      OC OD40
      AA OD41 2453      BICW #<TIMCTRLSM_ACTIVE!TIMCTRLSM_CANCEL>-
      OD43 2454      .TTSW_TIMCTRL      ; RESET CANCEL +TIMER ACTIVE BITS
      OD46 2455      MOVAL G^EXESAL_TOENOREPT, R5      ; Point R5 to no-repeat TQE
      OD4C
      OD4D 2456
      OD4D 2457 70$:
      OD4D 2458      ENBINT
      OD50 2459      RSB
      OD51 2460

```

```

OD51 2462 :++
OD51 2463 : MODEMSLINK_CRB
OD51 2464 :
OD51 2465 : FUNCTIONAL DESCRIPTION:
OD51 2466 :
OD51 2467 : THIS ROUTINE IS USED TO LINK THE ASSOCIATED CRB OF A UCB
OD51 2468 : REQUIRING MODEM CONTROL INTO THE MODEM PROTECOL TIMER QUEUE.
OD51 2469 : NOTE THAT THIS QUEUE IS SEPERATE FROM THE DZ-11 SPECIFIC
OD51 2470 : MODEM TRANSITION POLLING QUEUE.
OD51 2471 :
OD51 2472 : INPUTS:
OD51 2473 :
OD51 2474 :     R5 =   UCB ADDRESS
OD51 2475 :
OD51 2476 : OUTPUTS:
OD51 2477 :     R0-R4   DESTROYED
OD51 2478 : --
OD51 2479 :
OD51 2480 :     CHECK IF CRB NEEDS INSERTION ON MODEM TIMER QUEUE
OD51 2481 :
OD51 2482 : MODEMSLINK_CRB:
OD51 2483 :
54   24  A5  D0  OD51 2484      MOVL   UCBSL_CRB(R5),R4      : GET CRB ADDRESS
51   FD92 CF  DE  OD55 2485      MOVAL  TTSL_DIALUP,R1      : ADDRESS OF LIST HEAD
53   14  A4  DE  OD5A 2486      MOVAL  CRB$C_TT_MODEM(R4),R3  : GET CRB THREAD
    52   51  D0  OD5E 2487      MOVL   R1,R2
    53   62  D1  OD61 2488 10$:  CMPL   (R2),R3      : CRB ON LIST
    52   08  13  OD64 2489      BEQL   20$          : YES
    52   62  D0  OD66 2490      MOVL   (R2),R2      : POINT TO NEXT
    63   F6  12  OD69 2491      BNEQ   10$          : IF NOT END, LOCK AGAIN
    61   61  D0  OD6B 2492      MOVL   (R1),(R3)    : LINK AT LIST HEAD
    61   53  D0  OD6E 2493      MOVL   R3,(R1)
    FDOE 30  OD71 2494      MOVL   R3,(R1)
    05   05  OD74 2495 20$:  BSBW   TTY$SET_MODEM  : INVOKE ANY PORT SPECIFIC SETUP
    OD75 2496      RSB
    OD75 2497
    OD75 2498

```

```

OD75 2500 .sbttl MODEM STATE TABLES
OD75 2501 :++
OD75 2502 : MODEM CONTROL STATE TABLES
OD75 2503 :
OD75 2504 : FUNCTIONAL DESCRIPTION:
OD75 2505 :
OD75 2506 : THIS TABLE CONTAINS STATE ENTRIES. EACH STATE CORRESPONDS TO
OD75 2507 : A MODEM PROTOCOL STATE. EACH STATE ENTRY CONSISTS OF A MODEM SIGNAL
OD75 2508 : MASK TO BE SET ON STATE ENTRY, AN INITIAL TIMER VALUE TO BE STARTED ON
OD75 2509 : STATE ENTRY AND AN OPTIONAL ACTION ROUTINE.
OD75 2510 : FOLLOWING THE STATE ENTRY ARE TRANSITION ELEMENTS. ON ANY
OD75 2511 : TRANSITION : TIMEOUT, MODEM INTERRUPT, OR PROGRAM DECLARED, EACH
OD75 2512 : TRANSITION ELEMENT IS PROCESSED SEQUENTIALLY. THE LAST TRANSITION ELEMENT
OD75 2513 : IS ALWAYS AN END TRANSITION ELEMENT WHICH SIGNALS THAT
OD75 2514 : NO TRANSITION OCCURS IF IT IS PROCESSED. OTHER TRANSITION ELEMENT TYPES
OD75 2515 : MAKE CONDITIONAL TESTS AND MAY DECLARE A NEW STATE IF THE APPROPRIATE
OD75 2516 : CONDITIONS HOLD TRUE.
OD75 2517 :--
OD75 2518 :
OD75 2519 :
OD75 2520 : MODEM STATE MACRO DEFINITIONS
OD75 2521 :
OD75 2522 :
OD75 2523 .macro state name,onmask=0,offmask=0,timer,routine
OD75 2524 name:
OD75 2525 .byte onmask
OD75 2526 .byte offmask
OD75 2527 .if nb timer
OD75 2528 .word <timer*2>
OD75 2529 .if false
OD75 2530 .word 0
OD75 2531 .endc
OD75 2532 .if nb routine
OD75 2533 .word routine-state_init
OD75 2534 .if false
OD75 2535 .word 0
OD75 2536 .endc
OD75 2537 .endm state
OD75 2538 :
OD75 2539 .macro tran type,nstate,onmask=0,offmask=0
OD75 2540 .if idn type,end
OD75 2541 .byte modem$c_tran_end
OD75 2542 .if false
OD75 2543 .byte modem$c_tran_'type'.0
OD75 2544 .word nstate-state_init
OD75 2545 .byte offmask
OD75 2546 .byte onmask
OD75 2547 .endc
OD75 2548 .endm tran
OD75 2549 :

```

```

OD75 2551
OD75 2552 STATE_INIT:
OD75 2553
OD75 2554 :
OD75 2555 : LINE NOT UNDER MODEM CONTROL
OD75 2556 :
OD75 2557 STATE OFF
OD7B 2558 TRAN END
OD7C 2559
OD7C 2560 :
OD7C 2561 : INITIAL MODEM STATE
OD7C 2562 :
OD7C 2563 STATE IDLE,OFFMASK=<TTSM_DS_DTR!TTSM_DS_RTS>,TIMER=2
OD82 2564 TRAN DIALTYPE,ONMASK=02,NSTATE=RINGWAIT
OD8B 2565 TRAN TIME,NSTATE=WAIT
OD8E 2566 TRAN END
OD8F 2567
OD8F 2568 :
OD8F 2569 : WAIT FOR RING PRIOR TO SETTING DTR
OD8F 2570 :
OD8F 2571
OD8F 2572 STATE RINGWAIT
OD95 2573 TRAN DATASET,ONMASK=<TTSM_DS_RING>,NSTATE=INIT2
OD9B 2574 TRAN END
OD9C 2575
OD9C 2576 :
OD9C 2577 : WAIT FOR DTR AND RTS
OD9C 2578 : (IF DZ-11 GO DO SUBSET SUPPORT)
OD9C 2579 :
OD9C 2580 STATE WAIT,ONMASK=<TTSM_DS_DTR!TTSM_DS_RTS>
ODA2 2581 TRAN DZ11,NSTATE=DZWAIT
ODA8 2582 TRAN DATASET,ONMASK=TTSM_DS_DSR,NSTATE=INIT1
ODAE 2583 TRAN END
ODAF 2584
ODAF 2585 :
ODAF 2586 : DZ-11 SUBSET SUPPORT
ODAF 2587 : WAIT FOR CARRIER OR RING
ODAF 2588 :
ODAF 2589 STATE DZWAIT
ODB5 2590 TRAN DATASET,ONMASK=<TTSM_DS_CARRIER>,NSTATE=TRANSMIT0
ODB8 2591 TRAN DATASET,ONMASK=<TTSM_DS_RING>,NSTATE=INIT2
ODC1 2592 TRAN END
ODC2 2593
ODC2 2594 :
ODC2 2595 : DELAY
ODC2 2596 :
ODC2 2597 STATE INIT1,TIMER=1
ODC8 2598 TRAN TIME,NSTATE=INIT2
ODCE 2599 TRAN END
ODCF 2600
ODCF 2601 :
ODCF 2602 : START TIMER AND WAIT FOR CTS AND CARRIER
ODCF 2603 : IF TIMER EXPIRES, SHUTDOWN LINE
ODCF 2604 :
ODCF 2605 STATE INIT2,ONMASK=<TTSM_DS_DTR!TTSM_DS_RTS>,TIMER=30
ODD5 2606 TRAN TIME,NSTATE=SHUTDOWN
ODDB 2607 TRAN DATASET,ONMASK=<TTSM_DS_CTS!TTSM_DS_CARRIER!TTSM_DS_DSR>,-
  
```



```

ODDB 2608          NSTATE=TRANSMIT0
ODE1 2609          TRAN      END
ODE2 2610
ODE2 2611          :
ODE2 2612          : TRANSMIT0 STATE (SIGNAL CONNECT AND ALLOW TIME TO ASSIGN CHANNELS)
ODE2 2613          :
ODE2 2614          STATE    TRANSMIT0,ROUTINE=LOGIN,TIMER=30
ODE8  2615          TRAN      DATASET,OFFMASK=TTSM_DS_DSR,NSTATE=SHUTDOWN
ODEE  2616          TRAN      DATASET,OFFMASK=TTSM_DS_CARRIER,NSTATE=TRANSMIT1
ODF4  2617          TRAN      TIME,NSTATE=TRANSMIT
ODFA  2618          TRAN      END
ODFB  2619          :
ODFB  2620          : NORMAL TRANSMIT STATE
ODFB  2621          :
ODFB  2622          STATE    TRANSMIT,ROUTINE=VERIFY ; ROUTINE VERIFY WILL SHUTDOWN IF REFC=0
OE01  2623          TRAN      DATASET,OFFMASK=TTSM_DS_DSR,NSTATE=SHUTDOWN
OE07  2624          TRAN      DATASET,OFFMASK=TTSM_DS_CARRIER,NSTATE=TRANSMIT1
OE0D  2625          TRAN      DATASET,OFFMASK=TTSM_DS_CTS,NSTATE=CTSLOW
OE13  2626          TRAN      END
OE14  2627
OE14  2628          :
OE14  2629          : LOSS OF CARRIER DETECTED
OE14  2630          :
OE14  2631          STATE    TRANSMIT1,TIMER=2,ROUTINE=CTSHIGH
OE1A  2632          TRAN      DIALTYPE,ONMASK=01,NSTATE=SHUTDOWN
OE20  2633          TRAN      TIME,NSTATE=SHUTDOWN
OE26  2634          TRAN      DATASET,OFFMASK=TTSM_DS_DSR,NSTATE=SHUTDOWN
OE2C  2635          TRAN      DATASET,ONMASK=<TTSM_DS_CARRIER!TTSM_DS_CTS>,NSTATE=TRANSMIT
OE32  2636          TRAN      END
OE33  2637
OE33  2638          :
OE33  2639          : DELAY PRIOR TO COMPLETE SHUTDOWN
OE33  2640          :
OE33  2641          STATE    SHUTDOWN,TIMER=1,OFFMASK=TTSM_DS_DTR,ROUTINE=LOGOUT
OE39  2642          TRAN      TIME,NSTATE=SHUT1
OE3F  2643          TRAN      END
OE40  2644
OE40  2645          :
OE40  2646          : COMPLETE SHUTDOWN
OE40  2647          : AND THEN REINIT
OE40  2648          :
OE40  2649          STATE    SHUT1,TIMER=2
OE46  2650          TRAN      NOMODEM,NSTATE=OFF
OE4C  2651          TRAN      TIME,NSTATE=IDLE
OE52  2652          TRAN      DATASET,OFFMASK=TTSM_DS_DSR,NSTATE=IDLE
OE58  2653          TRAN      END
OE59  2654          :
OE59  2655          : CTSLOW
OE59  2656          : STOP SENDING DATA AND WAIT FOR CTS TO GO HIGH
OE59  2657          :
OE59  2658          STATE    CTSLOW,ROUTINE=LOSTCTS
OE5F  2659          TRAN      DATASET,ONMASK=TTSM_DS_CTS,NSTATE=TRANSMIT
OE65  2660          TRAN      DATASET,OFFMASK=TTSM_DS_DSR,NSTATE=SHUTDOWN
OE6B  2661          TRAN      DATASET,OFFMASK=TTSM_DS_CARRIER,NSTATE=TRANSMIT1
OE71  2662          TRAN      END
OE72  2663

```

```

      OE72 2665      .SBTTL MODEM ACTION ROUTINES
      OE72 2666      ;
      OE72 2667      :
      OE72 2668      : THESE ROUTINES ACT AS ACTION ROUTINES FOR MODEM CONNECTION
      OE72 2669      : AND DISCONNECTION. ON CONNECTION THE REMOTE BIT IS SET.
      OE72 2670      : ON DISCONNECT A CONTROL Y AST IS FIRED WITH HANGUP STATUS
      OE72 2671      : TO SIGNAL THE CLI. ALSO ANY OUTSTANDING ATTENTION ASTS ARE
      OE72 2672      : FIRED.
      OE72 2673      :
      OE72 2674      LOGIN:
51 00C0 C5 D0 OE72 2675      MOVL   UCBSL_TT LOGUCB(R5),R1      ; GET LUCB ADDRESS
      00002000 8F C8 OE77 2676      BISL   #TTSM_REMOTE,UCBSL_DEVDEPEND(R5); UPDATE CHARACTERISTIC
      44 A5
      00002000 8F C8 OE7D 2677      BISL   #TTSM_REMOTE,UCBSL_DEVDEPEND(R1)
      44 A1
      1B 10 OE87 2678      BSBB   CTSHIGH      ; SAY CTS IS HIGH
      28 11 OE89 2679      BRB    ACTION_EXIT
      OE8B 2680
      OE8B 2681      LOGOUT:
      0016 30 OE8B 2682      BSBW   CTSHIGH
54 02 D0 OE8E 2683      MOVL   #TTYSV_FD_DISCONNECT,R4      ; SIGNAL HANGUP CONDITION
      F797 30 OE91 2684      BSBW   TTY$CRE_FORK
      1D 11 OE94 2685      BRB    ACTION_EXIT
      OE96 2686
      20000000 8F C8 OE96 2687      LOSTCTS:
      00BC C5 30 OE9C 2688      BISL   #TTYSM_ST_CTSLOW,UCBSQ_TT_STATE+4(R5); SAY CTS IS LOW
      FBED 30 OE9F 2689      BSBW   TTY$STOP
      OF 11 OEA2 2690      BRB    ACTION_EXIT
      OEA4 2691      CTSHIGH:
00BC C5 1D E4 OEA4 2692      BBSC   #TTYSV_ST_CTSLOW,UCBSQ_TT_STATE+4(R5),10$; SAY CTS IS HIGH
      03 OEA9
      FBAB 31 OEAA 2693      BRW    TTY$RESUME
      FBAC 30 OEAD 2694      10$:  BSBW   TTY$RESUME
      F14D 30 OEBU 2695      BSBW   TTY$STARTOUTPUT
      OEB3 2696
      50 01 9A OEB3 2697      ACTION_EXIT:
      05 05 OEB6 2699      MOVZBL #SS$_NORMAL,R0
      OEB7 2700      RSB
      OEB7 2701      :
      OEB7 2702      : VERIFY THAT UNIT HAS CHANNELS ASSIGNED
      OEB7 2703      :
      OEB7 2704      VERIFY:
51 00C0 C5 D0 OE87 2705      BSBB   CTSHIGH      ; SAY CTS IS HIGH
      5C A1 B5 OE89 2706      MOVL   UCBSL_TT LOGUCB(R5),R1      ; GET LUCB ADDRESS
      F0 12 OEC1 2707      TSTW   UCBSW_REFC(R1)      ; CHANNELS ASSIGNED
      50 D4 OEC3 2708      BNEQ   ACTION_EXIT      ; YES, RETURN SUCCESS
      51 01 9A OEC5 2709      CLRL   R0      ; SIGNAL FAILURE
      05 05 OEC5 2710      MOVZBL #MODEM$_SHUTDOWN,R1      ; AND RETURN TRANSITION CODE IN R1
      OEC8 2711      RSB
      OEC9 2712

```

```

OEC9 2714 .SBTTL CLASS_MODEM_DIS - CLASS SERVICE TO FORCE MODEM SHUTDOWN
OEC9 2715 .SBTTL TTY$CLASS_DISCONNECT - CLASS SERVICE TO SIGNAL HANGUP CONDITION
OEC9 2716 :++
OEC9 2717 : CLASS_MODEM_DIS
OEC9 2718 :
OEC9 2719 : FUNCTION:
OEC9 2720 : WILL FORCE MODEM SIGNALS TO HANGUP THE LINE AND SIGNAL HANGUP STATUS
OEC9 2721 :
OEC9 2722 : INPUTS:
OEC9 2723 :
OEC9 2724 : R5 = UCB ADDRESS
OEC9 2725 :
OEC9 2726 : OUTPUTS:
OEC9 2727 :
OEC9 2728 : R3,R4 DESTROYED
OEC9 2729 : R0,R1,R2 ARE PRESERVED
OEC9 2730 :--
OEC9 2731 CLASS_MODEM_DIS:: : INITIATE MODEM DISCONNECT
51 07 BB OEC9 2732 PUSHR #*M<R0,R1,R2>
01 D0 OECB 2733 MOVL #MODEM$C_SHUTDWN,R1 : SIGNAL RESET
FC6B 30 OECE 2734 BSBW TRANSITION : INVOKE MODEM TRANSITION ROUTINE
07 BA OED1 2735 POPR #*M<R0,R1,R2>
05 OED3 2736 RSB : RESTORE THE REGISTERS AND RETURN
OED4 2737
OED4 2738 : GENERALIZED CLASS ROUTINE TO SIGNAL HANGUP STATUS
OED4 2739
OED4 2740 TTY$CLASS_DISCONNECT::
54 07 BB OED4 2741 POSHR #*M<R0,R1,R2>
OOC0 C5 D0 OED6 2742 MOVL UCBSL_TT LOGUCB(R5),R4 : GET THE LOGICAL UCB TO CHECK
SC A4 B5 OEDB 2743 TSTW UCBSW_REFC(R4) : THE REFCOUNT.
05 12 OEDE 2744 BNEQ $$ : IF CHANNELS THEN AST'S TO FIRE OR
50 D4 OEE0 2745 CLRL R0 : ALWAYS INDICATE HANGUP IF
FB50 30 OEE2 2746 BSBW TTY$DISCONNECT : NO CHANNELS THEN NONE OF THE ABOVE
FFA3 30 OEE5 2747 5$: BSBW LOGOUT : AND QUEUE FORK TO SIGNAL HANGUP ST
07 BA OEE8 2748 POPR #*M<R0,R1,R2>
05 OEEA 2749 RSB

```

TTYSUB
V04-002

1 12
- Terminal driver miscellaneous subrouti 8-JAN-1985 17:22:31 VAX/VMS Macro V04-00 Page 73
End of module 7-SEP-1984 17:57:12 [TTDRVR.BUGSRC]TTYSUB.MAR;1 (52)

```
OEEB 2751          .SBTTL End of module
OEEB 2752
OEEB 2753 TT_END::          ; End of Terminal Class Driver
OEEB 2754          .END
```

YC
VO

SSSSSS	= 00000B28	R	02	IOSV_REFRESH	= 0000000D		
ACBSL_KAST	= 00000018			IOSV-TIMED	= 00000007		
ACTION_EXIT	00000EB3	R	02	IOCSCLONE_UCB	*****	X	02
ATTENTION	000009A2	R	02	IOCSINITIATE	*****	X	02
CANSC_DASSGN	= 00000001			IPLS_QUEUEAST	= 00000006		
CANCEL_AST	000001D1	R	02	IPLS_SYNCH	= 00000008		
CANCEL_DETACH	00000205	RR	02	IPLS-TIMER	= 00000008		
CANCEL_DONE	00000227	RR	02	IRPSC_MEDIA	= 00000038		
CANCEL_MODEM	0000022F	RR	02	IRPSL_PID	= 0000000C		
CANCEL_RESET	00000257	RR	02	IRPSL-SVAPTE	= 0000002C		
CLASS_MODEM_DIS	00000EC9	RG	02	IRPSM_FUNC	= 00000002		
CLONE_UCB	0000093B	R	02	IRPSV_BUFIO	= 00000000		
CNT	= 00000001			IRPSV-VIRTUAL	= 00000004		
COMSDELATTNAST	*****	X	02	IRPSW_CHAN	= 00000028		
COMSDRVDEALMEM	*****	X	02	IRPSW_FUNC	= 00000020		
COMSFLUSHATTNS	*****	X	02	IRPSW-STS	= 0000002A		
COMSFLUSHCTRLS	*****	X	02	LINKFORK	000007D7	R	02
COMSPOST	*****	X	02	LOGIN	00000E72	RR	02
CRBSB_TT_TIMREFC	= 0000001F			LOGOUT	00000E8B	RR	02
CRBSB_TT_TYPE	= 0000000B			LOSTCTS	00000E96	R	02
CRBSL_INTD	= 00000024			MODEMSB-ST_ONMASK	= 00000000		
CRBSL_TT_MODEM	= 00000014			MODEMSB-TRAN_OFFMASK	= 00000004		
CTSHIGH	00000EA4	R	02	MODEMSB-TRAN_ONMASK	= 00000005		
CTSLOW	00000E59	RR	02	MODEMSC_DATASET	= 00000003		
DATASET	00000B97	RR	02	MODEMSC_INIT	= 00000000		
DECLARE_STATE	00000BFD	R	02	MODEMSC_NULL	= 00000002		
DEVSM_AVL	= 00040000			MODEMSC_SHUTDOWN	= 00000001		
DEVSM_DET	= 00000002			MODEMSC-ST_LENGTH	= 00000006		
DEVSM_RED	= 00000100			MODEMSC-TIMER	= 00000004		
DEVSV-SPL	= 00000006			MODEMSC-TRAN_DATASET	= 00000000		
DIALTYPE	00000BCE	R	02	MODEMSC-TRAN-DIALTYPE	= 00000003		
DISCONNECT	00000688	RR	02	MODEMSC-TRAN-DZ11	= 00000004		
DISC_FIREAST	000006E7	RR	02	MODEMSC-TRAN-END	= 00000002		
DISC-STOPIO	0000071A	R	02	MODEMSC-TRAN_LENGTH	= 00000006		
DTS_DZ11	= 00000042			MODEMSC-TRAN-NODEM	= 00000005		
DYNSC_TQE	= 0000000F			MODEMSC-TRAN-TIME	= 00000001		
DYNSC-TYPAHD	= 00000014			MODEMSLINK_CRB	00000D51	R	02
DZ11	00000BDB	R	02	MODEMSW-ST_ROUTINE	= 00000004		
DZWAIT	00000DAF	RR	02	MODEMSW-ST-TIMER	= 00000002		
END	00000B8E	R	02	MODEMSW-TRAN_NSTATE	= 00000002		
EXESALONONPAGED	*****	X	02	MODEM_TIMER	00000C36	R	02
EXESAL_TQENOREPT	*****	X	02	MSG\$-TRMHANGUP	= 00000006		
EXESFORK	*****	X	02	MSG\$-TRMUNSOLIC	= 00000001		
EXESGL_ABSTIM	*****	X	02	NEW_STATE	00000BF9	R	02
EXESGQ-SYSTIME	*****	X	02	NEXT_TRAN	00000BF3	RR	02
EXESINSTIMQ	*****	X	02	NOMODEM	00000BE9	RR	02
EXESSNDEVMSG	*****	X	02	OFF	00000D75	R	02
F	= 00000000			OPASUCBO	*****	X	02
FORKEXIT	0000067C	R	02	PCBSL_PID	= 00000060		
GETAHD	0000076C	R	02	PORTFORK	00000680	R	02
IDBSL_UCBLST	= 00000018			PORT_ABORT	= 00000020		
IDBSW_UNITS	= 0000000C			PORT_DISCONNECT	= 00000004		
IDLE	00000D7C	R	02	PORT_DS_SET	= 0000000C		
INIT	00C00B64	RR	02	PORT_FORKRET	= 00000034		
INIT1	00000DC2	RR	02	PORT_MAINT	= 00000030		
INIT2	00000DC1	R	02	PORT_RESUME	= 00000024		
IOSV_DSABLMBX	= 0000000A			PORT_SET_LINE	= 00000008		

TTYSUB
Symbol table

K 12
- Terminal driver miscellaneous subrouti

8-JAN-1985 17:22:31 VAX/VMS Macro V04-00
7-SEP-1984 17:57:12 [TTDRVR.BUGSRC]TTYSUB.MAR;1

Page 75
(52)

PORT_SET MODEM	= 00000028			TTSM_READSYNC	= 00040000		
PORT_STOP	= 00000018			TTSM_REMOTE	= 00002000		
PORT_TRANSITION	00000B28	RG	02	TTSTIMER	00000CD5	R	02
PORT_XOFF	= 00000014			TTSTIMQUENT	00000AF8	R	02
PORT_XON	= 00000010			TTSV_HOSTSYNC	= 00000004		
PRS_TPL	= 00000012			TTSV_MBXDSABL	= 00000010		
QLEN	*****	X	02	TTSV_MODEM	= 00000015		
RESTART	000009F7	RG	02	TTSV_NOTYPEAHD	= 00000002		
RINGWAIT	00000D8F	R	02	TTSV_PARITY	= 00C00006		
SET_SPEED	000000C9	R	02	TTSV_READSYNC	= 00000012		
SHUT1	00000E40	R	02	TTSV_REMOTE	= 0000000D		
SHUTDOWN	00000E33	R	02	TTSV_REFcnt	00000AEF	R	02
SHUTDWN	00000B71	R	02	TTSV_TIMCTRL	00000AF1	R	02
SS\$ ABORT	= 0000002C			TT2SM_DISCONNECT	= 00020000		
SS\$ CANCEL	= 00000830			TT2SM_DMA	= 00000040		
SS\$ DATAOVERUN	= 00000838			TT2SV_ALTTYPEAHD	= 0C000007		
SS\$ HANGUP	= 000002CC			TT2SV_AUTOBAUD	= 00000001		
SS\$ NORMAL	= 00000001			TT2SV_DISCONNECT	= 00000011		
SS\$ OPINCOMPL	= 000002D4			TT2SV_DMA	= 00000006		
SS\$ PARITY	= 000001F4			TT2SV_HANGUP	= 00000002		
SS\$ TIMEOUT	= 0000022C			TT2SV_LOCALE.CHO	= 00000000		
STATE_INIT	00000D75	R	02	TT2SV_SECURE	= 00000010		
SYSSGL_JOBCTLMB	*****	X	02	TTY\$ABORT	00000A22	RG	02
T	= 0000002A			TTY\$ABORT_10	00000000	RG	02
TIMCTRLSM_ACTIVE	= 00C00002			TTY\$AB_600	*****	X	02
TIMCTRLSM_CANCEL	= 00000001			TTY\$AB_9600	*****	X	02
TIMCTRLSV_ACTIVE	= 00000001			TTY\$AUTOBAUD	0000006C	RG	02
TIMCTRLSV_CANCEL	= 00000000			TTY\$A_CTRLU	*****	X	02
TIME	00000B8F	R	02	TTY\$A_MAXTIME	*****	X	02
TIMER_CANCEL	00000CB5	R	02	TTY\$A_RB_PRM	= 0000004A		
TIMER_END	00000CD1	R	02	TTY\$B_TA_TYPE	= 0000000A		
TQESB_RQTYPE	= 0000000B			TTY\$C_CANCELIO	000000D2	RG	02
TQESB_TYPE	= 0000000A			TTY\$CLASS_DISCONNECT	00000ED4	RG	02
TQESC_LENGTH	= 00000030			TTY\$CLASS_FORK	00000628	RG	02
TQESC_SSREPT	= 00000005			TTY\$CRE_FORK	0000062B	RG	02
TQESL_FPC	= 0000000C			TTY\$C_BELL	= 00000007		
TQESQ_DELTA	= 00000020			TTY\$C_BLANK	= 00000020		
TQESW_SIZE	= 00000008			TTY\$C_CR	= 0000000D		
TRANSITION	00000B3C	RG	02	TTY\$C_XOFF	= 00000013		
TRANSITION_NOCHELK	00000B47	RG	02	TTY\$C_XON	= 00000011		
TRANSMIT	00000DFR	R	02	TTY\$D_DISCONNECT	00000A35	RG	02
TRANSMIT0	00000DE2	R	02	TTY\$DS_SET	00000A42	RG	02
TRANSMIT1	00000E14	R	02	TTY\$FORK_ADDR	0000064D	RG	02
TRAN_LOOP	00000B7B	R	02	TTY\$FRAMERROR	0000041F	R	02
TTSC_BAUD_600	= 00000007			TTY\$GB_DIALTYP	*****	X	02
TTSC_BAUD_9600	= 0000000F			TTY\$GL_JOBCTLMB	*****	X	02
TTSL_DIALUP	00000AEB	RG	02	TTY\$GL_TIMEOUT	*****	X	02
TTSM_DS_CARRIER	= 00000020			TTY\$GW_ALTYP AHD	*****	X	02
TTSM_DS_CTS	= 00000010			TTY\$GW_TYPAHDSZ	*****	X	02
TTSM_DS_DSR	= 00000080			TTY\$LOCK	*****	X	02
TTSM_DS_DTR	= 00000002			TTY\$SL_TA_DATA	= 00000118		
TTSM_DS_RING	= 00000040			TTY\$SL_TA_END	= 00000010		
TTSM_DS_RIS	= 00000010			TTY\$SL_TA_GET	= 00000004		
TTSM_HOSTSYNC	= 00000010			TTY\$SL_TA_PUT	= 00000000		
TTSM_MBXDSABL	= 00010000			TTY\$SL_WB_BLINK	= 00000004		
TTSM_NOECHO	= 00000002			TTY\$SL_WB_DATA	= 00000030		
TTSM_PASSALL	= 00000001			TTY\$SL_WB_FLINK	= 00000000		

YCI
VOI

TTYSUB
Symbol table

```

TTYSL_WB_IRP          = 00000024
TTYSL_WB_NEXT        = 0000001C
TTYSMAINT            = 00000A4F  RG    02
TTYSM_PC_DMAENA     = 00000002
TTYSM_PC_XOFENA     = 00000040
TTYSM_ST_AUTOP      = 00020000
TTYSM_ST_BACKSPACE  = 00000020
TTYSM_ST_BADESC     = 00000100
TTYSM_ST_CTRL0      = 00000001
TTYSM_ST_CTRLR      = 00040000
TTYSM_ST_CTSLOW     = 20000000
TTYSM_ST_DCL        = 00000002
TTYSM_ST_EDITING    = 00100000
TTYSM_ST_EDITREAD   = 00000200
TTYSM_ST_EOL        = 00000100
TTYSM_ST_ESC        = 00000080
TTYSM_ST_ESC_O      = 00004000
TTYSM_ST_MULTI      = 00000040
TTYSM_ST_NOFLTR     = 00000040
TTYSM_ST_OVERSTRIKE = 00800000
TTYSM_ST_OVRFLO     = 00010000
TTYSM_ST_POWER      = 00000001
TTYSM_ST_PRE        = 04000000
TTYSM_ST_PROMPT     = 00000020
TTYSM_ST_QUOTING    = 00400000
TTYSM_ST_RDVERIFY   = 00000400
TTYSM_ST_READ       = 00001000
TTYSM_ST_RECALL     = 00000800
TTYSM_ST_REFRSH     = 00000400
TTYSM_ST_SKIPCRLF   = 00080000
TTYSM_ST_SKIPPLF    = 00002000
TTYSM_ST_TERMNORM   = 01000000
TTYSM_ST_TYPFUL     = 00001000
TTYSNOTIFY          = 0000031D  RG    02
TTYSPOWERACTION     = 00000A11  RG    02
TTYSPURGE_AHEAD     = 0000036B  RG    02
TTYSPURGE_AHEAD     = 0000038D  RG    02
TTYSPURGE_AHEAD     = *****  X    02
TTYSPURGE_AHEAD     = 00000492  RG    02
TTYSPURGE_AHEAD     = 00000A5C  RG    02
TTYSPURGE_AHEAD     = 00000503  R    02
TTYSPURGE_AHEAD     = 000002B5  RG    02
TTYSPURGE_AHEAD     = 00000568  RG    02
TTYSPURGE_AHEAD     = 00000A75  RG    02
TTYSPURGE_AHEAD     = 00000A82  R    02
TTYSPURGE_AHEAD     = *****  X    02
TTYSPURGE_AHEAD     = 00000A8F  RG    02
TTYSPURGE_AHEAD     = *****  X    02
TTYSPURGE_AHEAD     = 00000535  R    02
TTYSPURGE_AHEAD     = 00000006
TTYSPURGE_AHEAD     = 00000002
TTYSPURGE_AHEAD     = 00000001
TTYSPURGE_AHEAD     = 00000005
TTYSPURGE_AHEAD     = 00000003
TTYSPURGE_AHEAD     = 00000004
TTYSPURGE_AHEAD     = 00000000
TTYSPURGE_AHEAD     = 00000002

```

```

TTYSV_PC_NOTIME     = 00000000
TTYSV_ST_CTRL0      = 00000001
TTYSV_ST_CTSLOW     = 0000001D
TTYSV_ST_PASALL     = 00000002
TTYSV_ST_TYPFUL     = 0000000C
TTYSV_SX_AUTOP      = 00000031
TTYSV_SX_BACKSPACE  = 00000005
TTYSV_SX_BADESC     = 00000028
TTYSV_SX_CTRL0      = 00000020
TTYSV_SX_CTRLR      = 00000032
TTYSV_SX_DEL        = 00000021
TTYSV_SX_EDITING    = 00000034
TTYSV_SX_EDITREAD   = 00000009
TTYSV_SX_EOL        = 00000008
TTYSV_SX_ESC        = 00000027
TTYSV_SX_ESC_O      = 0000002E
TTYSV_SX_MULTI      = 00000006
TTYSV_SX_NOECHO     = 00000025
TTYSV_SX_NOFLTR     = 00000026
TTYSV_SX_OVERSTRIKE = 00000037
TTYSV_SX_OVRFLO     = 00000030
TTYSV_SX_PASALL     = 00000022
TTYSV_SX_POWER      = 00000000
TTYSV_SX_PRE        = 0000003A
TTYSV_SX_PROMPT     = 00000025
TTYSV_SX_QUOTING    = 00000036
TTYSV_SX_RDVERIFY   = 0000000A
TTYSV_SX_READ       = 0000000C
TTYSV_SX_RECALL     = 0000000B
TTYSV_SX_RECONNECT  = 0000003C
TTYSV_SX_REFRSH     = 0000002A
TTYSV_SX_SKIPCRLF   = 00000033
TTYSV_SX_SKIPPLF    = 0000002D
TTYSV_SX_TERMNORM   = 00000038
TTYSV_SX_TYPFUL     = 0000002C
TTYSV_SX_WRITE      = 00000007
TTYSV_SX_WRTALL     = 00000024
TTYSWRITEPOST       = *****  X    02
TTYSWRITEPOST       = *****  X    02
TTYSWRITEPOST       = 0000003A
TTYSWRITEPOST       = 00000030
TTYSWRITEPOST       = 00000036
TTYSWRITEPOST       = 0000003C
TTYSWRITEPOST       = 0000000C
TTYSWRITEPOST       = 00000014
TTYSWRITEPOST       = 00000008
TTYSWRITEPOST       = 0000002A
TTYSWRITEPOST       = 00000028
TTYSWRITEPOST       = 00000AA2  RG    02
TTYSWRITEPOST       = 00000AD2  RG    02
TTYSWRITEPOST       = 00000EEB  RG    02
TTYSWRITEPOST       = 0000005F
TTYSWRITEPOST       = 00000041
TTYSWRITEPOST       = 0000005E
TTYSWRITEPOST       = 0000000B
TTYSWRITEPOST       = 000000EC
TTYSWRITEPOST       = 000000F0

```

TTYSUB
Symbol table

UCBSB_TT_DS_RCV = 00000124
UCBSB_TT_OUTYPE = 00000108
UCBSB_TT_PARITY = 000000F8
UCBSL_AMB = 00000060
UCBSL_CRB = 00000024
UCBSL_DEVCHAR = 00000038
UCBSL_DEVCHAR2 = 0000003C
UCBSL_DEVDEPEND = 00000044
UCBSL_DEVDEPND2 = 00000048
UCBSL_DUETIME = 0000006C
UCBSL_FPC = 0000000C
UCBSL_FR3 = 00000010
UCBSL_FR4 = 00000014
UCBSL_IOQFL = 0000004C
UCBSL_IRP = 00000058
UCBSL_PDT = 00000084
UCBSL_PID = 0000002C
UCBSL_STS = 00000064
UCBSL_SVAPTE = 00000078
UCBSL_TL_BANDQUE = 0000009C
UCBSL_TL_CTLPID = 000000A4
UCBSL_TL_CTRLC = 00000094
UCBSL_TL_CTRLY = 00000090
UCBSL_TL_OUTBAND = 00000098
UCBSL_TL_PHYUCB = 000000A0
UCBSL_TT_DECHA; = 000000C8
UCBSL_TT_DECHAR = 000000C4
UCBSL_TT_LOGUCB = 000000C0
UCBSL_TT_MULT I = 000000D8
UCBSL_TT_PORT = 00000118
UCBSL_TT_RDUE = 00000080
UCBSL_TT_RTIMOU = 00000084
UCBSL_TT_TYPAHD = 000000E4
UCBSL_TT_WBLINK = 000000D0
UCBSL_TT_WFLINK = 000000CC
UCBSL_TT_WRTBUF = 000000D4
UCBSM_BSY = 00000100
UCBSM_DELETEUCB = 00010000
UCBSM_INT = 00000002
UCBSM_JOB = 00000001
UCBSM_ONLINE = 00000010
UCBSM_POWER = 00000020
UCBSM_TIM = 00000001
UCBSM_TT_HANGUP = 00000008
UCBSM_TT_LEN = 00000018
UCBSM_TT_NOTIF = 00000004
UCBSM_TT_TIMO = 00000002
UCBSQ_TL_BRKTHRU = 000000A8
UCBSQ_TT_STATE = 00000088
UCBSV_BSY = 00000008
UCBSV_JOB = 00000000
UCBSV_POWER = 00000005
UCBSV_TT_DISPERR = 00000001
UCBSV_TT_NOTIF = 00000002
UCBSV_TT_PARTY = 00000006
UCBSV_TT_TIMO = 00000001
UCBSV_TT_USERFRAME = 00000002

UCBSW_BOFF = 0000007C
UCBSW_DEVBUFSIZ = 00000042
UCBSW_DEVSTS = 00000068
UCBSW_ERRCNT = 00000082
UCBSW_REFC = 0000005C
UCBSW_STS = 00000064
UCBSW_TT_CURSOR = 000000FC
UCBSW_TT_DESPEE = 000000E8
UCBSW_TT_DS_ST = 00000126
UCBSW_TT_DS_TIM = 00000128
UCBSW_TT_HOCD = 00000108
UCBSW_TT_MULTILEN = 000000DC
UCBSW_TT_PRTCTL = 00000122
UCBSW_TT_SPEED = 000000F4
UNLINKFORK = 00000849 R 02
UNSOL = 000008E7 R 02
VECSL_IDB = 00000008
VERIFY = 00000EB7 R 02
VTSUCB = ***** X 02
WO = 00000000
WT = 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
\$AB\$\$	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
_\$255\$DUA18:[SYS.OBJ]LIB.MLB;1	28
-\$255\$DUA18:[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\$:TT;SUB/OBJ=OBJ\$:TTYSUB MSRC\$:TTYSUB/UPDATE=(BUGS:TTYSUB)+EXECMLS/LIB

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

YCDRIVER
LIS

TYSTRSTP
LIS

TTYSUB
LIS

The image shows a grid of 16 columns and 16 rows of source code listings. Each cell contains a small window of text, likely representing a single line or a small block of code from a larger program. The text is dense and appears to be a mix of comments and code. The overall layout is a structured grid of these small code snippets.