RRRRRRRRRRR TTTTTTTTTTTTTTT PPPPPPPPPPP AAAAAAAAA DDDDDDDDDD
RRRRRRRRRRR TTTTTTTTTTTTTTT PPPPPPPPPPP AAAAAAAAA DDDDDDDDDD
RRRRRRRRRRR TTTTTTTTTTTTTTT PPPPPPPPPPP AAAAAAAAA DDDDDDDDDD
RRR RRR TTT PPP PPP AAA AAA DDD DDD
RRR RRR TTT PPP PPP AAA AAA DDD DDD
RRR RRR TTT PPP PPP AAA AAA DDD DDD
RRR RRR TTT PPP PPP AAA AAA DDD DDD
RRR RRR TTT PPP PPP AAA AAA DDD DDD
RRRRRRRRRRR TTT PPPPPPPPPPP AAA AAA DDD DDD
RRRRRRRRRRR TTT PPPPPPPPPPP AAA AAA DDD DDD
RRRRRRRRRRR TTT PPPPPPPPPPP AAA AAA DDD DDD
RRR RRR TTT PPP AAAAAAAAAAAAAAA DDD DDD
RRR RRR TTT PPP AAAAAAAAAAAAAAA DDD DDD
RRR RRR TTT PPP AAAAAAAAAAAAAAA DDD DDD
RRR RRR TTT PPP AAA AAA DDD DDD
RRR RRR TTT PPP AAA AAA DDD DDD
RRR RRR TTT PPP AAA AAA DDD DDD
RRR RRR TTT PPP AAA AAA DDDDDDDDDDD
RRR RRR TTT PPP AAA AAA DDDDDDDDDDD
RRR RRR TTT PPP AAA AAA DDDDDDDDDDD

```
  CCCCCCCC  TTTTTTTTTT  DDDDDDDD     SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT
  CCCCCCCC  TTTTTTTTTT  DDDDDDDD     SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT
CC              TT      DD      DD  SS        EE              TT
CC              TT      DD      DD  SS        EE              TT
CC              TT      DD      DD  SS        EE              TT
CC              TT      DD      DD  SS        EE              TT
CC              TT      DD      DD    SSSSSS  EEEEEEEE        TT
CC              TT      DD      DD    SSSSSS  EEEEEEEE        TT
CC              TT      DD      DD          SS  EE            TT
CC              TT      DD      DD          SS  EE            TT
CC              TT      DD      DD          SS  EE            TT        ....
CC              TT      DD      DD          SS  EE            TT        ....
  CCCCCCCC      TT      DDDDDDDD    SSSSSSSS    EEEEEEEEEE    TT        ....
  CCCCCCCC      TT      DDDDDDDD    SSSSSSSS    EEEEEEEEEE    TT        ....


LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
                        .TITLE  CTDSET - CTDRIVER SET MODE PROCESSING
                        .IDENT  'V04-000'
                        .DISABLE GLOBAL
00000000                .PSECT $$$115_DRIVER,LONG
                ;
                ;*****************************************************************************
                ;*                                                                           *
                ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
                ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
                ;*  ALL RIGHTS RESERVED.                                                     *
                ;*                                                                           *
                ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
                ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
                ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
                ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
                ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY       *
                ;*  TRANSFERRED.                                                             *
                ;*                                                                           *
                ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
                ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
                ;*  CORPORATION.                                                             *
                ;*                                                                           *
                ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
                ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
                ;*                                                                           *
                ;*                                                                           *
                ;*****************************************************************************
                ;
                ;
                ;++
                ;
                ; FACILITY:
                ;
                ;       CTERM Remote terminal protocol driver
                ;
                ; ABSTRACT:
                ;
                ;
                ; ENVIRONMENT:
                ;
                ;
                ;
                ;--
                ;
                ; AUTHOR: Jake VanNoy, CREATION DATE: 30-Aug-1982
                ;
                ; MODIFIED BY:
                ;
                ;       V03-002 JLV0337         Jake VanNoy             28-FEB-1984
                ;               Fix: CRFILL, LFFILL, WRAP, EIGHTBIT, SPEED_TABLE
                ;               HARDCOPY/SCOPE, MECHFORM, MECHTAB, PARITY
                ;
                ;       V03-001 JLV0289         Jake VanNoy             28-JUL-1983
                ;               Update to new characteristics selectors.
                ;
                ;
                ;**
```

0000 58

```
                        0000     60          .SBTTL  DECLARATIONS
                        0000     61 ;
                        0000     62 ; INCLUDE FILES:
                        0000     63 ;
                        0000     64          $TTDEF
                        0000     65          $TT2DEF
                        0000     66          $TSADEF
                        0000     67          $UCBDEF
                        0000     68          $TTYUCBDEF                      ; tty dependent UCB info
                        0000     69
                        0000     70 .EXTERNAL        CT$AB_TERM_TABLE
                        0000     71
                        0000     72 ;
                        0000     73 ; MACROS:
                        0000     74 ;
                        0000     75
                        0000     76
                        0000     77 ;
                        0000     78 ; EQUATED SYMBOLS:
                        0000     79 ;
                        0000     80
                        0000     81 ;*** TEMP
                        0000     82
                        0000     83 .save LOCAL_BLOCK
                        0000     84 .psect $ABS$,ABS
                        0000     85
00000110                0000     86 . = UCB$T_CT_DEBUG_FILL
                        0110     87
00000111                0110     88 UCB$B_CT_CRFILL:         .BLKB
00000112                0111     89 UCB$B_CT_LFFILL:         .BLKB
00000114                0112     90 UCB$W_CT_SPEED:          .BLKW
                        0114     91
                   00000000     92 .restore
                        0000     93 ;*** END TEMP
                        0000     94
00000000                0000     95 physical = ch$c_physical@8               ; 0 leftshifted one byte
00000100                0000     96 logical =  ch$c_logical@8                ; 1 leftshifted one byte
00000200                0000     97 cterm   =  ch$c_cterm@8                  ; 2 leftshifted one byte
                        0000     98
00000000                0000     99 P1       = 0
00000004                0000    100 P2       = 4
00000008                0000    101 P3       = 8
0000000C                0000    102 P4       = 12
00000010                0000    103 P5       = 16
00000014                0000    104 P6       = 20
                        0000    105
                        0000    106 ;
                        0000    107 ; OWN STORAGE:
                        0000    108 ;
                        0000    109 SPEED_TABLE:
                0000    0000    110     .WORD       0                       ; filler
                0032    0002    111     .WORD       50                      ; assumes values of TT$C_BAUD_xxx
                004B    0004    112     .WORD       75
                006E    0006    113     .WORD       110
                0086    0008    114     .WORD       134
                0096    000A    115     .WORD       150
                012C    000C    116     .WORD       300
```

```
0258  000E   117      .WORD       600
04B0  0010   118      .WORD       1200
0708  0012   119      .WORD       1800
07D0  0014   120      .WORD       2000
0960  0016   121      .WORD       2400
0E10  0018   122      .WORD       3600
12C0  001A   123      .WORD       4800
1C20  001C   124      .WORD       7200
2580  001E   125      .WORD       9600
4B00  0020   126      .WORD       19200
9600  0022   127      .WORD       38400                ; future value?
      0024   128
```

```
                                0024   130          .SBTTL  CT_SET - Map VMS data into TSA characteristics
                                0024   131
                                0024   132 ;++
                                0024   133 ;
                                0024   134 ; FUNCTIONAL DESCRIPTION:
                                0024   135 ;
                                0024   136 ;
                                0024   137 ; CALLING SEQUENCE:
                                0024   138 ;       BSBW    CT_SET
                                0024   139 ;
                                0024   140 ; INPUT PARAMETERS:
                                0024   141 ;       R2  - address of send buffer
                                0024   142 ;       R9  - UCB$L_DEVCHAR data
                                0024   143 ;       R10 - UCB$L_DEVDEPEND data
                                0024   144 ;       R11 - UCB$L_DEVDEPND2 data
                                0024   145 ;       P3(AP) - speed: (Low byte = transmit, High byte = receive)
                                0024   146 ;       P4(AP) - fill:  (Low byte = CR fill, High byte = LF fill)
                                0024   147 ;       P5(AP) - parity and frame size
                                0024   148 ;
                                0024   149 ; IMPLICIT INPUTS:
                                0024   150 ;       NONE
                                0024   151 ;
                                0024   152 ; OUTPUT PARAMETERS:
                                0024   153 ;
                                0024   154 ;       R2 - Address of byte after buffer data
                                0024   155 ;
                                0024   156 ; IMPLICIT OUTPUTS:
                                0024   157 ;       NONE
                                0024   158 ;
                                0024   159 ; COMPLETION CODES:
                                0024   160 ;       NONE
                                0024   161 ;
                                0024   162 ; SIDE EFFECTS:
                                0024   163 ;       NONE
                                0024   164 ;
                                0024   165 ;--
                                0024   166
                                0024   167
                                0024   168 CT_SET::
                    0E00 8F     BB  0024   169          PUSHR   #^M<R9,R10,R11>         ; Save input
                          56    2C A2   9E  0028   170          MOVAB   CTP$W_CH_PARAM(R2),R6   ; buffer pointer
                                002C   171 ;
                                002C   172 ; See if DEVCLASS, DEVTYPE and DEVBUFSIZ are the same
                                002C   173 ;
                          40 A5   59    D1  002C   174          CMPL    R9,UCB$B_DEVCLASS(R5)   ; Compare all at once
                                  0C    12  0030   175          BNEQ    10$                     ; Branch if not
                                0032   176 ;
                                0032   177 ; Because HARDCOPY is in TERM_BITS, check for change here
                                0032   178 ;
                       59   44 A5   5A    CD  0032   179          XORL3   R10,UCB$L_DEVDEPEND(R5),R9 ; get bits that changed...
                            03 59   0C    E0  0037   180          BBS     #TT$V_SCOPE,R9,10$      ; Scope/hardcopy change - send type and bits
                                  005B    31  003B   181          BRW     DEVDEPEND               ; Ok to continue
                                0043   182 10$:
                58   59   08   08    EF  003E   183          EXTZV   #8,#8,R9,R8             ; Get terminal type
                                  16    13  0043   184          BEQL    30$                     ; Branch if setting to unknown
                          57    00000000'EF   9E  0045   185          MOVAB   CT$AB_TERM_TABLE,R7     ; Get address of terminal table
                                004C   186
```

```
                58   87   91   004C   187 20$:    CMPB    (R7)+,R8                ; Is it the same?
                     0E   13   004F   188         BEQL    50$                     ; Branch if yes
                50   87   9A   0051   189         MOVZBL  (R7)+,R0                ; Get length of string
                57   50   C0   0054   190         ADDL2   R0,R7                   ; Add to address
                     67   95   0057   191         TSTB    (R7)                    ; End of table?
                     F1   12   0059   192         BNEQ    20$                     ; Branch if no
                               005B   193 ;
                               005B   194 ; Unknown terminal, set up terminal attributes
                               005B   195 ;
                               005B   196 30$:
                     50   D4   005B   197         CLRL    R0                      ; Assume unknown/hardcopy
                     18   11   005D   198         BRB     55$                     ; continue
                               005F   199 ;
                               005F   200 ; terminal match found, copy text string of terminal type
                               005F   201 ;
                               005F   202 50$:
            86   0103   8F   B0   005F   203         MOVW    #logical!ch$c_lg_term_type,(R6)+ ; Set terminal type data
                50   67   9A   0064   204         MOVZBL  (R7),R0                 ; Get length of string
                     50   D6   0067   205         INCL    R0                      ; Add one for count
                     3C   BB   0069   206         PUSHR   #^M<R2,R3,R4,R5>        ; Save
            66   67   50   28   006B   207         MOVC3   R0,(R7),(R6)            ; Move terminal string
                56   53   D0   006F   208         MOVL    R3,R6                   ; Update buffer pointer
                     3C   BA   0072   209         POPR    #^M<R2,R3,R4,R5>        ; Save
                               0074   210
                50   01   9A   0074   211         MOVZBL  #CTP$M_CH_KNOWN,R0      ; Set known, Assume hardcopy
              03 5A   0C   E1   0077   212 55$:    BBC     #TT$V_SCOPE,R10,60$     ; Branch if scope bit clear in new char
                50   02   88   007B   213         BISB    #CTP$M_CH_SCOPE,R0      ; Set scope
                               007E   214 60$:
            86   0102   8F   B0   007E   215         MOVW    #logical!ch$c_lg_term_bits,(R6)+ ; Move terminal bits data
                 86   50   B0   0083   216         MOVW    R0,(R6)+                ; Set characteristic
                               0086   217 ;
                               0086   218 ; Now do terminal width
                               0086   219 ;
        58   59   10   10   EF   0086   220         EXTZV   #16,#16,R9,R8           ; Get width
               42 A5   58   B1   008B   221         CMPW    R8,UCB$W_DEVBUFSIZ(R5)  ; Compare to current setting
                       08   13   008F   222         BEQL    DEVDEPEND               ; Branch if equal, nothing to do
                               0091   223
            86   0109   8F   B0   0091   224         MOVW    #logical!ch$c_lg_line_width,(R6)+ ; Set line width
                 86   58   B0   0096   225         MOVW    R8,(R6)+                ; Move into buffer
                               0099   226
```

```
                                0099   228
                                0099   229 ;
                                0099   230 ; Process DEVDEPEND bits (high byte is page length)
                                0099   231 ;
                                0099   232 ;
                                0099   233 ; R9 is scratch now
                                0099   234
                                0099   235 DEVDEPEND:
58   5A   08    18   EF   0099   236          EXTZV   #24,#8,R10,R8               ; Fetch page length
 5A   FF000000 8F   CA   009E   237          BICL    #TT$M_PAGE,R10              ; Clear page length bits
       47 A5    58   91   00A5   238          CMPB    R8,UCB$L_DEVDEPEND+3(R5); Compare to current setting
                08   13   00A9   239          BEQL    10$                         ; Branch if equal
                                00AB   240
     86   010A 8F   B0   00AB   241          MOVW    #logical!ch$c_lg_page_length,(R6)+      ; Move data type
          86   58   B0   00B0   242          MOVW    R8,(R6)+                    ; Move data
                                00B3   243 10$:
      FF000000 8F   CB   00B3   244          BICL3   #TT$M_PAGE,-
       59   44 A5                00B9   245                  UCB$L_DEVDEPEND(R5),R9  ; Fetch current DEVDEPEND bits
          59   5A   CC   00BC   246          XORL    R10,R9                      ; Determine which bits changed
                                00BF   247 ;
                                00BF   248 ; CR fill and LF fill must be checked before the test for R9 = 0.
                                00BF   249 ;
                                00BF   250 CRFILL:
       50   0C AC   9A   00BF   251          MOVZBL  P4(AP),R0                   ; Fetch CRFILL
       09 59    0A   E0   00C3   252          BBS     #TT$V_CRFILL,R9,10$         ; Branch if it has changed
                14   13   00C7   253          BEQL    LFFILL                      ; or parameter = zero
     0110 C5    50   91   00C9   254          CMPB    R0,UCB$B_CT_CRFILL(R5)      ; compare to old value
                0D   13   00CE   255          BEQL    LFFILL                      ; branch if no change
     86   010C 8F   B0   00D0   256 10$:     MOVW    #logical!ch$c_lg_cr_fill,(R6)+  ; Set data type
          86   50   B0   00D5   257          MOVW    R0,(R6)+                    ; CR FILL count
     0110 C5    50   90   00D8   258          MOVB    R0,UCB$B_CT_CRFILL(R5)      ; save value
                                00DD   259
                                00DD   260 LFFILL:
       50   0D AC   9A   00DD   261          MOVZBL  P4+1(AP),R0                 ; Fetch LFFILL
       09 59    0B   E0   00E1   262          BBS     #TT$V_LFFILL,R9,10$         ; Branch if it has changed
                14   13   00E5   263          BEQL    20$                         ; or parameter = zero
     0111 C5    50   91   00E7   264          CMPB    R0,UCB$B_CT_LFFILL(R5)      ; compare to old value
                0D   13   00EC   265          BEQL    20$                         ; branch if no change
     86   010D 8F   B0   00EE   266 10$:     MOVW    #logical!ch$c_lg_lf_fill,(R6)+  ; Set data type
          86   50   B0   00F3   267          MOVW    R0,(R6)+                    ; LF FILL count
     0111 C5    50   90   00F6   268          MOVB    R0,UCB$B_CT_LFFILL(R5)      ; save value
                                00FB   269 20$:
                59   D5   00FB   270          TSTL    R9                          ; test for changes
                03   12   00FD   271          BNEQ    PASSALL                     ; Branch if not zero
              00F7   31   00FF   272          BRW     DEVDEPND2                   ; Optimization, skip this section
                                0102   273
                                0102   274 PASSALL:
                                0102   275
                                0102   276 NOECHO:
       0B 59    01   E1   0102   277          BBC     #TT$V_NOECHO,R9,-
                                0106   278                  NOTYPEAHD           ; Branch if it hasn't changed
     86   0205 8F   B0   0106   279          MOVW    #cterm!ch$c_ct_normal_echo,(R6)+                ; Set data type
          51   01   9A   010B   280          MOVZBL  #TT$V_NOECHO,R1             ; Set bit
              00D0   30   010E   281          BSBW    TEST_BIT_NEG                ; test bit, negative sense
                                0111   282
                                0111   283 NOTYPEAHD:
                                0111   284
```

```
                     0111   285 ESCAPE:
 19 59   03   E1     0111   286          BBC     #TT$V_ESCAPE,R9,-
                     0115   287                  HOSTSYNC                ; Branch if it hasn't changed
        50   94     0115   288          CLRB    R0
 03 5A   03   E1     0117   289          BBC     #TT$V_ESCAPE,R10,10$    ; branch if setting to noescape
    50   01   90     011B   290          MOVB    #1,R0                   ; Set true
                     011E   291 10$:
86  0206 8F   B0     011E   292          MOVW    #cterm!ch$c_ct_input_esc,(R6)+          ; Set escape input
    86   50   90     0123   293          MOVB    R0,(R6)+                ; true/false
86  0207 8F   B0     0126   294          MOVW    #cterm!ch$c_ct_output_esc,(R6)+         ; Set escape output
    86   50   90     012B   295          MOVB    R0,(R6)+                ; true/false
                     012E   296
                     012E   297 HOSTSYNC:                               ; TSA Input-flow-control
 0B 59   04   E1     012E   298          BBC     #TT$V_HOSTSYNC,R9,-
                     0132   299                  TTSYNC                  ; Branch if it hasn't changed
86  0107 8F   B0     0132   300          MOVW    #logical!ch$c_lg_input_flow,(R6)+       ; Set data type
    51   04   9A     0137   301          MOVZBL  #TT$V_HOSTSYNC,R1       ; Set bit
         0098 30     013A   302          BSBW    TEST_BIT                ; test bit
                     013D   303
                     013D   304 TTSYNC:                                 ; TSA Output-flow-control
 0B 59   05   E1     013D   305          BBC     #TT$V_TTSYNC,R9,-
                     0141   306                  SCRIPT                  ; Branch if it hasn't changed
86  0104 8F   B0     0141   307          MOVW    #logical!ch$c_lg_output_flow,(R6)+      ; Set data type
    51   05   9A     0146   308          MOVZBL  #TT$V_TTSYNC,R1
         0089 30     0149   309          BSBW    TEST_BIT                ; test bit
                     014C   310
                     014C   311 SCRIPT:
 0B 59   06   E1     014C   312          BBC     #TT$V_SCRIPT,R9,-
                     0150   313                  LOWER                   ; Branch if it hasn't changed
86  0209 8F   B0     0150   314          MOVW    #cterm!ch$c_ct_auto_prompt,(R6)+        ; Set data type
    51   06   9A     0155   315          MOVZBL  #TT$V_SCRIPT,R1
         007A 30     0158   316          BSBW    TEST_BIT                ; test bit
                     015B   317
                     015B   318 LOWER:                                  ; Cterm raise-input
 0B 59   07   E1     015B   319          BBC     #TT$V_LOWER,R9,-
                     015F   320                  MECHTAB                 ; Branch if it hasn't changed
86  0204 8F   B0     015F   321          MOVW    #cterm!ch$c_ct_raise_input,(R6)+ ; Set data type
    51   07   9A     0164   322          MOVZBL  #TT$V_LOWER,R1          ; set bit number
         0077 30     0167   323          BSBW    TEST_BIT_NEG            ; test bit, negative sense
                     016A   324
                     016A   325 MECHTAB:                                ; Logical horizontal-tab
 12 59   08   E1     016A   326          BBC     #TT$V_MECHTAB,R9,-
                     016E   327                  WRAP                    ; Branch if it hasn't changed
86  010F 8F   B0     016E   328          MOVW    #logical!ch$c_lg_hor_tab,(R6)+  ; Set data type
    50   02   B0     0173   329          MOVW    #2,R0                   ; Assume no mechtab
 03 5A   08   E1     0176   330          BBC     #TT$V_MECHTAB,R10,-
                     017A   331                  10$                     ; Branch if right
    50   01   B0     017A   332          MOVW    #1,R0                   ; Set mechtab
                     017D   333 10$:
    86   50   B0     017D   334          MOVW    R0,(R6)+                ; Move data
                     0180   335
                     0180   336 WRAP:
 04 59   09   E0     0180   337          BBS     #TT$V_WRAP,R9,10$
 19 59   0C   E1     0184   338          BBC     #TT$V_SCOPE,R9,SCOPE    ; Branch if neither hasn changed
86  010E 8F   B0     0188   339 10$:     MOVW    #logical!ch$c_lg_wrap,(R6)+     ; Set data type
    50   04   B0     018D   340          MOVW    #4,R0                   ; Assume regulate software wrap
 0A 5A   09   E0     0190   341          BBS     #TT$V_WRAP,R10,30$      ; branch if right
```

```
                50   01   B0  0194   342          MOVW    #1,R0                   ; Set No wrap
          03 5A 0C   E0   0197   343          BBS     #TT$V_SCOPE,R10,30$     ; Branch if scope
                50   02   B0  019B   344          MOVW    #2,R0                   ; Set wrap/truncate - because of hardcopy
                                019E   345 30$:
                86   50   B0  019E   346          MOVW    R0,(R6)+                ; Set data in buffer
                                01A1   347
                                01A1   348 SCOPE:  ; Handled above in DEVCHAR code
                                01A1   349
                                01A1   350 REMOTE:
                                01A1   351
                                01A1   352 HOLDSCREEN:
          0B 59 0E   E1   01A1   353          BBC     #TT$V_HOLDSCREEN,R9,-
                                01A5   354                  EIGHTBIT                ; Branch if it hasn't changed
       86   0105 8F   B0   01A5   355          MOVW    #logical!ch$c_lg_page_stop,(R6)+        ; Set data type
                51   0E   9A   01AA   356          MOVZBL  #TT$V_HOLDSCREEN,R1     ; set bit number
                   0025   30   01AD   357          BSBW    TEST_BIT                ; test bit
                                01B0   358
                                01B0   359 EIGHTBIT:
          09 59 0F   E1   01B0   360          BBC     #TT$V_EIGHTBIT,R9,-
                                01B4   361                  MBXDSABL                ; Branch if it hasn't changed
                86   0B   B0   01B4   362          MOVW    #physical!ch$c_ph_eightbit,(R6)+ ; Set data type
                51   0F   9A   01B7   363          MOVZBL  #TT$V_EIGHTBIT,R1       ; set bit number
                   0018   30   01BA   364          BSBW    TEST_BIT                ; test bit
                                01BD   365
                                01BD   366 MBXDSABL:
                                01BD   367
                                01BD   368 NOBRDCST:
                                01BD   369
                                01BD   370 READSYNC:
                                01BD   371
                                01BD   372 MECHFORM:
          12 59 13   E1   01BD   373          BBC     #TT$V_MECHFORM,R9,-
                                01C1   374                  HALFDUP                 ; Branch if it hasn't changed
       86   0111 8F   B0   01C1   375          MOVW    #logical!ch$c_lg_form_feed,(R6)+        ; Set data type
                50   02   B0   01C6   376          MOVW    #2,R0                   ; Assume no mechform
          03 5A 13   E1   01C9   377          BBC     #TT$V_MECHFORM,R10,-
                                01CD   378                  10$                     ; Branch if right
                50   01   B0   01CD   379          MOVW    #1,R0                   ; Set mechform
                                01D0   380 10$:
                86   50   B0   01D0   381          MOVW    R0,(R6)+                ; Move data
                                01D3   382
                                01D3   383 HALFDUP:
                                01D3   384
                                01D3   385 MODEM:
                                01D3   386
                     24   11   01D3   387          BRB     DEVDEPND2               ; Goto DEVDEPEND 2 checking
                                01D5   388
                                01D5   389 TEST_BIT:
                50   94   01D5   390          CLRB    R0                      ; Assume off
          02 5A 51   E1   01D7   391          BBC     R1,R10,10$              ; Branch if not setting bit
                50   96   01DB   392          INCB    R0                      ; Set on
                                01DD   393 10$:
                86   50   90   01DD   394          MOVB    R0,(R6)+                ; Move data
                     05   01E0   395          RSB
                                01E1   396
                                01E1   397 TEST_BIT_NEG:
                50   94   01E1   398          CLRB    R0                      ; Assume off
```

```
02 5A   51   E0  01E3   399          BBS     R1,R10,10$              ; Branch if setting bit
        50   96  01E7   400          INCB    R0                      ; Set on
                 01E9   401 10$:
   86   50   90  01E9   402          MOVB    R0,(R6)+                ; Move data
             05  01EC   403          RSB
                 01ED   404
                 01ED   405 TEST_BIT_DEV2:                            ; DEVDEPEND 2 version of TEST_BIT
        50   94  01ED   406          CLRB    R0                      ; Assume off
02 5B   51   E1  01EF   407          BBC     R1,R11,10$              ; Branch if not setting bit
        50   96  01F3   408          INCB    R0                      ; Set on
                 01F5   409 10$:
   86   50   90  01F5   410          MOVB    R0,(R6)+                ; Move data
             05  01F8   411          RSB
```

```
                                01F9    413 ;
                                01F9    414 ; Process DEVDEPND2 bits
                                01F9    415 ;
                                01F9    416 ;
                                01F9    417
                                01F9    418 DEVDEPND2:
                                01F9    419
59   48 A5   D0                01F9    420         MOVL    UCB$L_DEVDEPND2(R5),R9  ; Fetch current DEVDEPEND 2 bits
   59   5B   CC                01FD    421         XORL    R11,R9                  ; Determine which bits changed
        0D   13                0200    422         BEQL    CTSET_SPEED             ; Branch if zero - optimization
                                0202    423
                                0202    424 LOCALECHO:
                                0202    425
                                0202    426 AUTOBAUD:
09 59   01   E1                0202    427         BBC     #TT2$V_AUTOBAUD,R9,-
                                0206    428                 HANGUP                  ; Is autobaud changing?
   86   07   B0                0206    429         MOVW    #physical!ch$c_ph_autobaud,(R6)+        ; Set data type
   51   01   9A                0209    430         MOVZBL  #TT2$v_AUTOBAUD,R1      ; Set bit number
      FFDE   30                020C    431         BSBW    TEST_BIT_DEV2           ; test bit
                                020F    432
                                020F    433 HANGUP:
                                020F    434
                                020F    435 MODHANGUP:
                                020F    436
                                020F    437 BRDCSTMBX:
                                020F    438
                                020F    439 XON:
                                020F    440
                                020F    441 DMA:
                                020F    442
                                020F    443 ALTYPEAHD:
                                020F    444
                                020F    445 SETSPEED:
                                020F    446
                                020F    447 ANSICRT:
                                020F    448
                                020F    449 REGIS:
                                020F    450
                                020F    451 BLOCK:
                                020F    452
                                020F    453 AVO:
                                020F    454
                                020F    455 EDIT:
                                020F    456
                                020F    457 DECCRT:
                                020F    458
```

```
                                 020F    460
                                 020F    461 CTSET_SPEED:
                                 020F    462
          50    08 AC    D0      020F    463          MOVL    P3(AP),R0               ; Speed?
                   20    13      0213    464          BEQL    CTSET_PARITY            ; nope, continue
             51    50    9A      0215    465          MOVZBL  R0,R1                   ; Get transmit speed
       5B    FDE4 CF    9E      0218    466          MOVAB   W^SPEED_TABLE,R11       ;  table address
             86    01    B0      021D    467          MOVW    #physical!ch$c_ph_in_speed,(R6)+ ; set data type
          86    6B41    B0      0220    468          MOVW    (R11)[R1],(R6)+         ; set speed
    51    50    08    08      EF      0224    469          EXTZV   #8,#8,R0,R1             ; receive speed
                   03    12      0229    470          BNEQU   10$                     ; branch if non-zero
             51    50    9A      022B    471          MOVZBL  R0,R1                   ; same as transmit speed
                                 022E    472 10$:
             86    02    B0      022E    473          MOVW    #physical!ch$c_ph_out_speed,(R6)+ ; set data type
          86    6B41    B0      0231    474          MOVW    (R11)[R1],(R6)+         ; set speed
                                 0235    475
                                 0235    476 CTSET_PARITY:
                                 0235    477
          51    10 AC    3C      0235    478          MOVZWL  P5(AP),R1               ; Fetch parameter
          20 51    05    E1      0239    479          BBC     #TT$V_ALTRPAR,R1,CTSET_EXIT ; Ignore if ALTER not flagged
             86    04    B0      023D    480          MOVW    #physical!ch$c_ph_parity_enable,(R6)+
                   50    94      0240    481          CLRB    R0                      ; Assume off
          02 51    06    E1      0242    482          BBC     #TT$V_PARITY,R1,10$     ; Branch if not setting bit
                   50    96      0246    483          INCB    R0                      ; Set on
             86    50    90      0248    484 10$:     MOVB    R0,(R6)+                ; Move data
                   10    13      024B    485          BEQL    CTSET_EXIT              ; Ignore type if NOPARITY
                                 024D    486          ;
                                 024D    487          ; Now, set parity type
                                 024D    488          ;
          86    05    B0      024D    489          MOVW    #physical!ch$c_ph_parity_type,(R6)+
          50    01    B0      0250    490          MOVW    #ch$c_parity_even,R0    ; assume even
    03 51    07    E1      0253    491          BBC     #TT$V_ODD,R1,20$        ; branch if even
          50    02    B0      0257    492          MOVW    #ch$c_parity_odd,R0     ; set odd
          86    50    B0      025A    493 20$:     MOVW    R0,(R6)+                ; load value
                                 025D    494
                                 025D    495 CTSET_EXIT:
                                 025D    496
                                 025D    497          ;
                                 025D    498          ; Check if any characteristics changed
                                 025D    499          ;
             50    D4      025D    500          CLRL    R0                      ; Assume no changes
    51    2C A2    9E      025F    501          MOVAB   CTP$W_CH_PARAM(R2),R1   ; Original buffer pointer
          56    51    D1      0263    502          CMPL    R1,R6                   ; compare
                02    13      0266    503          BEQL    10$                     ; branch if no changes
                50    D6      0268    504          INCL    R0                      ; Set low bit
                                 026A    505 10$:
          52    56    D0      026A    506          MOVL    R6,R2                   ; Set output
          0E00 8F     BA      026D    507          POPR    #^M<R9,R10,R11>         ; Restore input
                05    0271    508          RSB                             ; Return
                                 0272    509
                                 0272    510
                                 0272    511 .END     ; CTSET
```

```
ALTYPEAHD                       0000020F  R      01
ANSICRT                         0000020F  R      01
AUTOBAUD                        00000202  R      01
AVO                             0000020F  R      01
BLOCK                           0000020F  R      01
BRDCSTMBX                       0000020F  R      01
CH$C_CTERM                    = 00000002
CH$C_CT_AUTO_PROMPT           = 00000009
CH$C_CT_INPUT_ESC             = 00000006
CH$C_CT_NORMAL_ECHO           = 00000005
CH$C_CT_OUTPUT_ESC            = 00000007
CH$C_CT_RAISE_INPUT           = 00000004
CH$C_LG_CR_FILL               = 0000000C
CH$C_LG_FORM_FEED             = 00000011
CH$C_LG_HOR_TAB               = 0000000F
CH$C_LG_INPUT_FLOW            = 00000007
CH$C_LG_LF_FILL               = 0000000D
CH$C_LG_LINE_WIDTH            = 00000009
CH$C_LG_OUTPUT_FLOW           = 00000004
CH$C_LG_PAGE_LENGTH           = 0000000A
CH$C_LG_PAGE_STOP             = 00000005
CH$C_LG_TERM_BITS             = 00000002
CH$C_LG_TERM_TYPE             = 00000003
CH$C_LG_WRAP                  = 0000000E
CH$C_LOGICAL                  = 00000001
CH$C_PARITY_EVEN              = 00000001
CH$C_PARITY_ODD               = 00000002
CH$C_PHYSICAL                 = 00000000
CH$C_PH_AUTOBAUD              = 00000007
CH$C_PH_EIGHTBIT              = 0000000B
CH$C_PH_IN_SPEED              = 00000001
CH$C_PH_OUT_SPEED             = 00000002
CH$C_PH_PARITY_ENABLE         = 00000004
CH$C_PH_PARITY_TYPE           = 00000005
CRFILL                          000000BF  R      01
CT$AB_TERM_TABLE                ********    X    01
CTERM                         = 00000200
CTP$M_CH_KNOWN                = 00000001
CTP$M_CH_SCOPE                = 00000002
CTP$W_CH_PARAM                = 0000002C
CTSET_EXIT                      0000025D  R      01
CTSET_PARITY                    00000235  R      01
CTSET_SPEED                     0000020F  R      01
CT_SET                          00000024  RG     01
DECCRT                          0000020F  R      01
DEVDEPEND                       00000099  R      01
DEVDEPND2                       000001F9  R      01
DMA                             0000020F  R      01
EDIT                            0000020F  R      01
EIGHTBIT                        000001B0  R      01
ESCAPE                          00000111  R      01
HALFDUP                         000001D3  R      01
HANGUP                          0000020F  R      01
HOLDSCREEN                      000001A1  R      01
HOSTSYNC                        0000012E  R      01
LFFILL                          000000DD  R      01
LOCALECHO                       00000202  R      01
LOGICAL                       = 00000100
LOWER                           0000015B  R      01
MBXDSABL                        000001BD  R      01
MECHFORM                        000001BD  R      01
MECHTAB                         0000016A  R      01
MODEM                           000001D3  R      01
MODHANGUP                       0000020F  R      01
NOBRDCST                        000001BD  R      01
NOECHO                          00000102  R      01
NOTYPEAHD                       00000111  R      01
P1                            = 00000000
P2                            = 00000004
P3                            = 00000008
P4                            = 0000000C
P5                            = 00000010
P6                            = 00000014
PASSALL                         00000102  R      01
PHYSICAL                      = 00000000
READSYNC                        000001BD  R      01
REGIS                           0000020F  R      01
REMOTE                          000001A1  R      01
SCOPE                           000001A1  R      01
SCRIPT                          0000014C  R      01
SETSPEED                        0000020F  R      01
SPEED_TABLE                     00000000  R      01
TEST_BIT                        000001D5  R      01
TEST_BIT_DEV2                   000001ED  R      01
TEST_BIT_NEG                    000001E1  R      01
TT$M_PAGE                     = FF000000
TT$V_ALTRPAR                  = 00000005
TT$V_CRFILL                   = 0000000A
TT$V_EIGHTBIT                 = 0000000F
TT$V_ESCAPE                   = 00000003
TT$V_HOLDSCREEN               = 0000000E
TT$V_HOSTSYNC                 = 00000004
TT$V_LFFILL                   = 0000000B
TT$V_LOWER                    = 00000007
TT$V_MECHFORM                 = 00000013
TT$V_MECHTAB                  = 00000008
TT$V_NOECHO                   = 00000001
TT$V_ODD                      = 00000007
TT$V_PARITY                   = 00000006
TT$V_SCOPE                    = 0000000C
TT$V_SCRIPT                   = 00000006
TT$V_TTSYNC                   = 00000005
TT$V_WRAP                     = 00000009
TT2$V_AUTOBAUD                = 00000001
TTSYNC                          0000013D  R      01
UCB$B_CT_CRFILL                 00000110
UCB$B_CT_LFFILL                 00000111
UCB$B_DEVCLASS                = 00000040
UCB$L_DEVDEPEND               = 00000044
UCB$L_DEVDEPND2               = 00000048
UCB$T_CT_DEBUG_FILL           = 00000110
UCB$W_CT_SPEED                  00000112
UCB$W_DEVBUFSIZ               = 00000042
WRAP                            00000180  R      01
```

```
XON                             0000020F R      01
```

Psect synopsis

```
PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
. ABS .                         00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$$$115_DRIVER                   00000272  (  626.)  01 (  1.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC LONG
$ABS$                           00000114  (  276.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
```

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.07	00:00:00.41
Command processing	120	00:00:00.55	00:00:01.66
Pass 1	373	00:00:07.79	00:00:17.12
Symbol table sort	0	00:00:01.46	00:00:03.52
Pass 2	93	00:00:01.53	00:00:03.01
Symbol table output	15	00:00:00.06	00:00:00.21
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	637	00:00:11.49	00:00:25.96

```
The working set limit was 1650 pages.
68046 bytes (133 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1353 non-local and 22 local symbols.
511 source lines were read in Pass 1, producing 14 object records in Pass 2.
12 pages of virtual memory were used to define 11 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SHRLIB]REM.MLB;1	0
_255DUA28:[RTPAD.OBJ]RTPAD.MLB;1	1
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	8

```
1368 GETS were required to define 8 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:CTDSET/OBJ=OBJ$:CTDSET MSRC$:CTDSET/UPDATE=(ENH$:CTDSET)+EXECML$/LIB+LIB$:RTPAD/LIB+SHRLIB$:REM/LIB
```

CTDSENSE
LIS

CTDVMSPEC
LIS

CTSENSERT
LIS

RSTSRT
LIS

CTERMRT
LIS

DTE_DF03
LIS

CTSETRT
LIS

CTDSET
LIS