```
RRRRRRRRRRRR     TTTTTTTTTTTTTTT  PPPPPPPPPPPP        AAAAAAAAA     DDDDDDDDDDDD
RRRRRRRRRRRR     TTTTTTTTTTTTTTT  PPPPPPPPPPPP        AAAAAAAAA     DDDDDDDDDDDD
RRRRRRRRRRRR     TTTTTTTTTTTTTTT  PPPPPPPPPPPP        AAAAAAAAA     DDDDDDDDDDDD
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
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRRRRRRRRRRR           TTT        PPPPPPPPPPPP     AAA         AAA  DDD         DDD
RRRRRRRRRRRR           TTT        PPPPPPPPPPPP     AAA         AAA  DDD         DDD
RRRRRRRRRRRR           TTT        PPPPPPPPPPPP     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  DDDDDDDDDDDD
RRR         RRR        TTT        PPP              AAA         AAA  DDDDDDDDDDDD
RRR         RRR        TTT        PPP              AAA         AAA  DDDDDDDDDDDD
```

```
  CCCCCCCC  TTTTTTTTTT    SSSSSSSS  EEEEEEEEEE  NN      NN    SSSSSSSS  EEEEEEEEEE  RRRRRRRR    TTTTTTTTTT
  CCCCCCCC  TTTTTTTTTT    SSSSSSSS  EEEEEEEEEE  NN      NN    SSSSSSSS  EEEEEEEEEE  RRRRRRRR    TTTTTTTTTT
CC              TT      SS          EE          NN      NN  SS          EE          RR      RR      TT
CC              TT      SS          EE          NN      NN  SS          EE          RR      RR      TT
CC              TT      SS          EE          NNNN    NN  SS          EE          RR      RR      TT
CC              TT      SS          EE          NNNN    NN  SS          EE          RR      RR      TT
CC              TT        SSSSSS    EEEEEEEE    NN  NN  NN    SSSSSS    EEEEEEEE    RRRRRRRR        TT
CC              TT        SSSSSS    EEEEEEEE    NN  NN  NN    SSSSSS    EEEEEEEE    RRRRRRRR        TT
CC              TT              SS  EE          NN    NNNN          SS  EE          RR  RR          TT
CC              TT              SS  EE          NN    NNNN          SS  EE          RR  RR          TT
CC              TT              SS  EE          NN      NN          SS  EE          RR    RR        TT           ....
CC              TT              SS  EE          NN      NN          SS  EE          RR    RR        TT           ....
  CCCCCCCC      TT      SSSSSSSS    EEEEEEEEEE  NN      NN  SSSSSSSS    EEEEEEEEEE  RR      RR      TT           ....
  CCCCCCCC      TT      SSSSSSSS    EEEEEEEEEE  NN      NN  SSSSSSSS    EEEEEEEEEE  RR      RR      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

CT
V0

```
0000          .TITLE  CTSENSERT - RTPAD/CTERM SENSE CHARACTERISTICS
0000          .IDENT  'V04-000'
0000          .ENABLE SUPPRESSION
0000          .DISABLE GLOBAL
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ;
0000    ; FACILITY:
0000    ;
0000    ;       CTERM Remote terminal protocol driver
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       This module is called to map a characteristics item list received
0000    ;       from the net into VMS terminal characteristics.
0000    ;
0000    ; ENVIRONMENT:
0000    ;
0000    ;
0000    ;
0000    ;--
0000    ;
0000    ; AUTHOR: Jake VanNoy, CREATION DATE: 23-Aug-1982
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-001 JLV0297         Jake VanNoy             28-JUL-1983
0000    ;               Change characteristics symbols.
0000    ;
0000    ;**
0000
```

```
                   0000          .SBTTL  DECLARATIONS
                   0000  ;
                   0000  ; INCLUDE FILES
                   0000  ;
                   0000          $DCDEF
                   0000          $SSDEF
                   0000          $TTDEF
                   0000          $TT2DEF
                   0000          $TSADEF
                   0000
                   0000  ;
                   0000  ; MACROS:
                   0000  ;
                   0000
                   0000  .MACRO  CHAR    LABEL
                   0000          .WORD           char_type!char_count
                   0000          .WORD           LABEL-SENSE_ROUTINES
                   0000  char_count = char_count + 1
                   0000  .ENDM   CHAR
                   0000
                   0000  .MACRO  BAUDTAB BPS
                   0000          .WORD   BPS
                   0000  .ENDM   BAUDTAB
                   0000
                   0000  .MACRO  TERMTAB TERM,VALUE
                   0000          .BYTE   VALUE
                   0000          .ASCIC  /TERM/
                   0000  .ENDM   TERMTAB
                   0000
                   0000  .MACRO  SENSETAB CHAR
                   0000          .WORD           char_type!char
                   0000  .ENDM   SENSETAB
                   0000
                   0000
                   0000  ;
                   0000  ; EQUATED SYMBOLS:
                   0000  ;
                   0000
00000000           0000  physical = ch$c_physical@8            ; 0 leftshifted 1 byte
00000100           0000  logical =  ch$c_logical@8             ; 1 leftshifted 1 byte
00000200           0000  cterm   =  ch$c_cterm@8               ; 2 leftshifted 1 byte
                   0000
00000000           0000  TT_BUF   = 0
00000004           0000  TT_CHAR1 = 4
00000008           0000  TT_CHAR2 = 8
00000000           0000  TT_IOSB  = 0
                   0000
                   0000  ;
                   0000  ; OWN STORAGE:
                   0000  ;
                   0000
         00000000        .PSECT RTPAD,NOWRT
                   0000
                   0000  SENSE_TABLE:
                   0000
00000000           0000  char_type = physical
00000001           0000  char_count = 1
```

```
          0000          CHAR     INPUT_SPEED              ; 1
          0004          CHAR     OUTPUT_SPEED             ; 2
          0008          CHAR     CHARACTER_SIZE           ; 3
          000C          CHAR     PARITY_ENABLE            ; 4
          0010          CHAR     PARITY_TYPE              ; 5
          0014          CHAR     MODEM_PRESENT            ; 6
          0018          CHAR     AUTOBAUD_DETECT          ; 7
          001C          CHAR     MANAGEMENT_GUARANTEED    ; 8
          0020          CHAR     SWITCH_CHAR_1            ; 9
          0024          CHAR     SWITCH_CHAR_2            ; 10
          0028
00000100  0028  char_type = logical
00000001  0028  char_count = 1
          0028
          0028          CHAR     MODE_WRITING_ALLOWED     ; 1
          002C          CHAR     TERMINAL_TYPE            ; 2
          0030          CHAR     TERMINAL_SUBTYPE         ; 3
          0034          CHAR     OUTPUT_FLOW_CONTROL      ; 4
          0038          CHAR     OUTPUT_PAGE_STOP         ; 5
          003C          CHAR     FLOW_CHAR_PASS_THRU      ; 6
          0040          CHAR     INPUT_FLOW_CONTROL       ; 7
          0044          CHAR     LOSS_NOTIFICATION        ; 8
          0048          CHAR     LINE_WIDTH               ; 9
          004C          CHAR     PAGE_LENGTH              ; 10
          0050          CHAR     STOP_LENGTH              ; 11
          0054          CHAR     CR_FILL                  ; 12
          0058          CHAR     LF_FILL                  ; 13
          005C          CHAR     WRAP                     ; 14
          0060          CHAR     HORIZONTAL_TAB           ; 15
          0064          CHAR     VERTICAL_TAB             ; 16
          0068          CHAR     FORM_FEED                ; 17
          006C
00000200  006C  char_type = cterm
00000001  006C  char_count = 1
          006C
          006C          CHAR     IGNORE_INPUT             ; 1
          0070          CHAR     CHAR_ATTRIBUTES          ; 2
          0074          CHAR     CONTROL_O_PASS_THRU      ; 3
          0078          CHAR     RAISE_INPUT              ; 4
          007C          CHAR     NORMAL_ECHO              ; 5
          0080  ;***    CHAR     REQUEST_PROCESSING       ; 6
          0080          CHAR     INPUT_ESCAPE_ENABLE      ; 7
          0084          CHAR     OUTPUT_ESCAPE_ENABLE     ; 8
          0088          CHAR     INPUT_COUNT_STATE        ; 9
          008C          CHAR     AUTO_PROMPT              ; 10
          0090          CHAR     ERROR_PROCESSING         ; 11
          0094
    FFFF  0094          .WORD    -1                       ; End of list (any negative number)
          0096
          0096  SENSE_BAUD:                               ; Table must be in ascending order
          0096          BAUDTAB 0                         ; 0
          0098          BAUDTAB 50                        ; 1
          009A          BAUDTAB 75                        ; 2
          009C          BAUDTAB 110                       ; 3
          009E          BAUDTAB 134                       ; 4
          00A0          BAUDTAB 150                       ; 5
          00A2          BAUDTAB 300                       ; 6
```

```
                          00A4              BAUDTAB 600                         ; 7
                          00A6              BAUDTAB 1200                        ; 8
                          00A8              BAUDTAB 1800                        ; 9
                          00AA              BAUDTAB 2000                        ; 10
                          00AC              BAUDTAB 2400                        ; 11
                          00AE              BAUDTAB 3600                        ; 12
                          00B0              BAUDTAB 4800                        ; 13
                          00B2              BAUDTAB 7200                        ; 14
                          00B4              BAUDTAB 9600                        ; 15
                          00B6              BAUDTAB 19200                       ; 16
                          00B8
                          00B8      TERM_TABLE:                                 ; Table is in order for most likely first
                          00B8              TERMTAB VT100,  TT$_VT100
                          00BF              TERMTAB VT101,  TT$_VT101
                          00C6              TERMTAB VT102,  TT$_VT102
                          00CD              TERMTAB VT105,  TT$_VT105
                          00D4              TERMTAB VT125,  TT$_VT125
                          00DB              TERMTAB VT131,  TT$_VT131
                          00E2              TERMTAB VT132,  TT$_VT132
                          00E9              TERMTAB VT52,   TT$_VT52
                          00EF              TERMTAB VT05,   TT$_VT05
                          00F5              TERMTAB VK100,  TT$_VK100
                          00FC              TERMTAB VT173,  TT$_VT173
                          0103      ;       TERMTAB TT$_FT1
                          0103      ;       TERMTAB TT$_FT2
                          0103      ;       TERMTAB TT$_FT3
                          0103      ;       TERMTAB TT$_FT4
                          0103      ;       TERMTAB TT$_FT5
                          0103      ;       TERMTAB TT$_FT6
                          0103      ;       TERMTAB TT$_FT7
                          0103      ;       TERMTAB TT$_FT8
                          0103              TERMTAB LA36,   TT$_LA36
                          0109              TERMTAB LA12,   TT$_LA12
                          010F              TERMTAB LA34,   TT$_LA34
                          0115              TERMTAB LA38,   TT$_LA38
                          011B              TERMTAB LA12,   TT$_LA12
                          0121              TERMTAB LA100,  TT$_LA100
                          0128              TERMTAB LA24,   TT$_LA24
                          012E              TERMTAB LQP02,  TT$_LQP02
                          0135              TERMTAB VT55,   TT$_VT55
                      00  013B              .BYTE   0
                          013C
4E 57 4F 4E 4B 4E 55 00'  013C      UNKNOWN_TT:     .ascic  /UNKNOWN/
                      07  013C
                          0144
                          0144      .nlist meb
                          0144
                          0144      .SBTTL CTSENSECHAR - Map READ_CHAR into CHAR using VMS data
                          0144
                          0144      ;
                          0144      ; Inputs:
                          0144      ;       R0 - CTP
                          0144      ;       R2 - address of requested characteristics buffer (READ_CHAR)
                          0144      ;       R3 - address of buffer to be written
                          0144      ;       R9 - address of 12 buffer of current characteristics
                          0144      ;       R10 - address of iosb from operation
                          0144      ;
```

```
                        0144  ; Outputs:
                        0144  ;         R1 is buffer length from start
                        0144  ;         buffer pointed to by R3 is complete
                        0144  ;         All registers destroyed
                        0144  ;
                        0144  CTSENSECHAR::
                        0144
                53  DD  0144          PUSHL   R3                          ; Save
        51   28 A0  3C  0146          MOVZWL  CTP$W_MSGSIZE(R0),R1        ; message size
        50   2A A0  9E  014A          MOVAB   CTP$B_MSGTYPE(R0),R0        ; base address
     5B   51    50  C1  014E          ADDL3   R0,R1,R11                   ; add together for end of message address
                        0152
                        0152  10$:
            56  82  3C  0152          MOVZWL  (R2)+,R6                    ; Get parameter
            5B  52  D1  0155          CMPL    R2,R11                      ; past end? *** should be one up?
                11  1A  0158          BGTRU   25$                         ; Done, exit
            83  56  B0  015A          MOVW    R6,(R3)+                    ; Move into table
     57   FE9F  CF  9E  015D          MOVAB   SENSE_TABLE,R7              ; Get table address
                        0162  20$:
            87  56  B1  0162          CMPW    R6,(R7)+                    ; Compare to table entry
                07  13  0165          BEQL    30$                         ; Branch if match found
                87  B5  0167          TSTW    (R7)+                       ; Advance pointer past routine
                F7  18  0169          BGEQ    20$                         ; Loop if greater than or equal to zero
                        016B
                        016B          ; PANIC                             ; 'invalid sense mode returned'
                        016B  25$:
              0193  31  016B          BRW     POST_SENSE_EXIT             ; exit code
                        016E
                        016E  ;
                        016E  ; Dispatch to routine
                        016E  ;
                        016E  30$:
 56   0000017F'EF   9E  016E          MOVAB   SENSE_ROUTINES,R6           ; Base
            57  67  3C  0175          MOVZWL  (R7),R7                     ; get routine address
            57  56  C0  0178          ADDL2   R6,R7                       ; Add offset to base for routine address
                67  16  017B          JSB     (R7)                        ; jsb to routine
                D3  11  017D          BRB     10$                         ; Do next parameter
                        017F
                        017F  ;
                        017F  ; All these routines have R0,R1,R4,R6-R8 as scratch
                        017F  ;
                        017F
```

```
                          017F  SENSE_ROUTINES:
                          017F
                          017F  ; physical
                          017F
                          017F  INPUT_SPEED:                                      ; physical 1
      50    03 AA    9A   017F          MOVZBL  TT_IOSB+3(R10),R0                 ; Get speed
               06    13   0183          BEQL    OUTPUT_SPEED                      ; If zero, then same as output speed
               0E    10   0185          BSBB    GET_SPEED                         ; map from TT$C_BAUD_xxxx
         83    50    B0   0187          MOVW    R0,(R3)+                          ; set receive speed
                     05   018A          RSB                                       ; Return
                          018B
                          018B  OUTPUT_SPEED:                                     ; physical 2
      50    02 AA    9A   018B          MOVZBL  TT_IOSB+2(R10),R0                 ; set transmit speed
               04    10   018F          BSBB    GET_SPEED                         ; map from TT$C_BAUD_xxxx
         83    50    B0   0191          MOVW    R0,(R3)+                          ; set tranmit speed
                     05   0194          RSB                                       ; Return
                          0195  ;
                          0195  ; Local routine to map TT$C_BAUD rates into speed
                          0195  ;
                          0195  ; Input:
                          0195  ;       R0 - Baud rate in TT$C_BAUD_xxx terms
                          0195  ;
                          0195  ; Output:
                          0195  ;       R0 - baud rate in BPS
                          0195  ;
                          0195  GET_SPEED:
   51    FEFD CF     9E   0195          MOVAB   SENSE_BAUD,R1                     ; Get table
      50    6140     3C   019A          MOVZWL  (R1)[R0],R0                       ; Index in and fetch value
                     05   019E          RSB                                       ; Return
                          019F
                          019F  CHARACTER_SIZE:                                   ; physical 3
         50    07    90   019F          MOVB    #7,R0                             ; Character size
               0F    E1   01A2          BBC     #TT$V_EIGHTBIT,-
       03 04 A9           01A4                  TT_CHAR1(R9),10$                  ; check characteristic
         50    08    90   01A7          MOVB    #8,R0                             ; Character size
                          01AA
         83    50    9B   01AA  10$:    MOVZBW  R0,(R3)+                          ; Set value
                     05   01AD          RSB                                       ; Return
                          01AE
                          01AE  PARITY_ENABLE:                                    ; physical 4
               50    94   01AE          CLRB    R0                                ; Set noparity
               06    E1   01B0          BBC     #TT$V_PARITY,-
       03 06 AA           01B2                  TT_IOSB+6(R10),10$                ; Check parity
         50    01    90   01B5          MOVB    #1,R0                             ; Set parity enabled
                          01B8  10$:
         83    50    90   01B8          MOVB    R0,(R3)+
                     05   01BB          RSB                                       ; Return
                          01BC
                          01BC  PARITY_TYPE:                                      ; physical 5
         50    01    90   01BC          MOVB    #1,R0                             ; Assume even
               07    E1   01BF          BBC     #TT$V_ODD,-
       03 06 AA           01C1                  TT_IOSB+6(R10),10$                ; branch if even
         50    02    90   01C4          MOVB    #2,R0                             ; Set odd
                          01C7  10$:
         83    50    9B   01C7          MOVZBW  R0,(R3)+                          ; Set parity
                     05   01CA          RSB                                       ; Return
                          01CB
```

L 11

```
                      01CB   327 MODEM_PRESENT:                                  ; physical 6
          50    94    01CB   328          CLRB    R0                             ; Get Boolean
 03 04 A9 15    E1    01CD   329          BBC     #TT$V_MODEM, -
                      01D2   330                  TT_CHAR1(R9),10$               ; check modem bit
       50  01   90    01D2   331          MOVB    #1,R0
                      01D5   332 10$:
       83  50   90    01D5   333          MOVB    R0,(R3)+                       ; Set value
                05    01D8   334          RSB                                    ; Return
                      01D9   335
                      01D9   336 AUTOBAUD_DETECT:                                ; physical 7
          50    94    01D9   337          CLRB    R0
          01    E1    01DB   338          BBC     #TT2$V_AUTOBAUD,-
     03 08 A9         01DD   339                  TT_CHAR2(R9),10$               ; Set autobaud
       50  01   90    01E0   340          MOVB    #1,R0
                      01E3   341 10$:
       83  50   90    01E3   342          MOVB    R0,(R3)+
                05    01E6   343          RSB                                    ; Return
                      01E7   344
                      01E7   345 MANAGEMENT_GUARANTEED:                          ; physical 8
          83    94    01E7   346          CLRB    (R3)+                          ; clear
                05    01E9   347 10$:     RSB                                    ; Return
                      01EA   348
                      01EA   349 SWITCH_CHAR_1:                                  ; physical 9
        0111    30    01EA   350          BSBW    IGNORE_STRING                  ; Ignore this string data
                05    01ED   351          RSB                                    ; Return
                      01EE   352
                      01EE   353 SWITCH_CHAR_2:                                  ; physical 10
        010D    30    01EE   354          BSBW    IGNORE_STRING                  ; Ignore this string data
                05    01F1   355          RSB                                    ; Return
                      01F2   356
```

```
                          01F2   ; logical
                          01F2
                          01F2   MODE_WRITING_ALLOWED:                           ; logical 1
                 83   94  01F2           CLRB    (r3)+
                      05  01F4   10$:    RSB                                     ; Return
                          01F5
                          01F5   TERMINAL_TYPE:                                  ; logical 2
                          01F5
                          01F5   ; Bits are:
                          01F5   ;       0 - known/unknown
                          01F5   ;       1 - scope/hardcopy
                          01F5   ;
                 50   D4  01F5           CLRL    R0
                          01F7           ASSUME  TT$_UNKNOWN EQ 0                ; assume unknown = 0
              01 A9   95  01F7           TSTB    TT_BUF+1(R9)                    ; is it unknown?
                 03   13  01FA           BEQL    10$                             ; Branch if unknown
         50      01   88  01FC           BISB    #CTP$M_CH_KNOWN,R0              ; Set known bit
                          01FF   10$:
                 0C   E1  01FF           BBC     #TT$V_SCOPE,-
           03 04 A9       0201                   TT_CHAR1(R9),20$                ; check for scope
         50      02   88  0204           BISB    #CTP$M_CH_SCOPE,R0              ; set scope
                          0207   20$:
         83      50   9B  0207           MOVZBW  R0,(R3)+                        ; set value
                      05  020A           RSB                                     ; Return
                          020B
                          020B   TERMINAL_SUBTYPE:                               ; logical 3
                          020B
      50      01 A9   9A  020B           MOVZBL  TT_BUF+1(R9),R0                 ; Fetch terminal type byte
   51      FEA5 CF    9E  020F           MOVAB   TERM_TABLE,R1                   ; Address of table
                          0214   20$:
         54      81   9A  0214           MOVZBL  (R1)+,R4                        ; Get terminal type from table
                 0D   13  0217           BEQL    30$                             ; if zero, end of list => unknown
         54      50   91  0219           CMPB    R0,R4                           ; Compare terminal types
                 0D   13  021C           BEQL    40$                             ; Match, branch
         54      81   9A  021E           MOVZBL  (R1)+,R4                        ; get length of string
         51      54   C0  0221           ADDL    R4,R1                           ; Add to pointer
                 EE   11  0224           BRB     20$                             ; Loop
                          0226   30$:
   51      FF12 CF    9E  0226           MOVAB   UNKNOWN_TT,R1                   ;  "unknown"
                          022B   40$:
                 34   BB  022B           PUSHR   #^M<R2,R4,R5>                   ; Save registers
         54      61   9A  022D           MOVZBL  (R1),R4                         ; Get length
                 54   D6  0230           INCL    R4                              ; Include byte for count
   63    61      54   28  0232           MOVC3   R4,(R1),(R3)                    ; Move to buffer, update R3
                 34   BA  0236           POPR    #^M<R2,R4,R5>                   ; restore
                          0238
                      05  0238           RSB                                     ; Return
                          0239
                          0239   OUTPUT_FLOW_CONTROL:                            ; logical 4
                 50   94  0239           CLRB    R0                              ; assume no
                 05   E1  023B           BBC     #TT$V_TTSYNC,-
           03 04 A9       023D                   TT_CHAR1(R9),10$                ; branch if not tt sync
         50      01   90  0240           MOVB    #1,R0                           ; Set true
                          0243   10$:
         83      50   90  0243           MOVB    R0,(R3)+                        ; Set data
                      05  0246           RSB                                     ; Return
                          0247
```

```
                         0247  OUTPUT_PAGE_STOP:                                ; logical 5
         50     94       0247          CLRB    R0                               ; assume no
         0E     E1       0249          BBC     #TT$V_HOLDSCREEN,-
   03 04 A9              024B                  TT_CHAR1(R9),10$                 ; branch if not holdscreen
   50    01     90       024E          MOVB    #1,R0                            ; Set true
                         0251  10$:
     83  50     90       0251          MOVB    R0,(R3)+                         ; Set data
                05       0254          RSB                                      ; Return
                         0255
                         0255  FLOW_CHAR_PASS_THRU:                             ; logical 6
         83     94       0255          CLRB    (R3)+
                05       0257  10$:    RSB                                      ; Return
                         0258
                         0258  INPUT_FLOW_CONTROL:                              ; logical 7
         50     94       0258          CLRB    R0                               ; assume no
         04     E1       025A          BBC     #TT$V_HOSTSYNC,-
   03 04 A9              025C                  TT_CHAR1(R9),10$                 ; branch if not host sync
   50    01     90       025F          MOVB    #1,R0                            ; Set true
                         0262  10$:
     83  50     90       0262          MOVB    R0,(R3)+                         ; Set data
                05       0265          RSB                                      ; Return
                         0266
                         0266  LOSS_NOTIFICATION:                               ; logical 8
         83     94       0266          CLRB    (R3)+                            ; ignore
                         0268  10$:
                05       0268          RSB             ; Bell on loss data?
                         0269
                         0269  LINE_WIDTH:                                      ; logical 9
 83   02 A9     B0       0269          MOVW    TT_BUF+2(R9),(R3)+               ; Line width
                05       026D          RSB                                      ; Return
                         026E
                         026E  PAGE_LENGTH:                                     ; logical 10
 83   07 A9     9B       026E          MOVZBW  TT_CHAR1+3(R9),(R3)+             ; Page length
                05       0272          RSB                                      ; Return
                         0273
                         0273  STOP_LENGTH:                                     ; logical 11
         83     B4       0273          CLRW    (R3)+                            ; ** ignore
                05       0275          RSB                                      ; Return
                         0276
                         0276  CR_FILL:                                         ; logical 12
         50     B4       0276          CLRW    R0                               ; Assume no cr fill
         0A     E1       0278          BBC     #TT$V_CRFILL,-
   04 04 A9              027A                  TT_CHAR1(R9),10$                 ; branch if off
 50   06 AA     B0       027D          MOVW    TT_IOSB+6(R10),R0                ; set on
                         0281  10$:
     83  50     B0       0281          MOVW    R0,(R3)+
                05       0284          RSB                                      ; Return
                         0285
                         0285  LF_FILL:                                         ; logical 13
         50     B4       0285          CLRW    R0                               ; Assume no lf fill
         0B     E1       0287          BBC     #TT$V_LFFILL,-
   04 04 A9              0289                  TT_CHAR1(R9),10$                 ; branch if off
 50   07 AA     B0       028C          MOVW    TT_IOSB+7(R10),R0                ; set on
                         0290  10$:
     83  50     B0       0290          MOVW    R0,(R3)+
                05       0293          RSB                                      ; Return
                         0294
```

```
                  0294   WRAP:                                           ; logical 14
   50    01   B0  0294           MOVW    #1,R0                          ; assume no wrap
         09   E1  0297           BBC     #TT$V_WRAP,-
03 04 A9          0299                   TT_CHAR1(R9),10$               ; branch if no wrap
   50    04   B0  029C           MOVW    #4,R0                          ; Set wrap
                  029F   10$:
   83    50   B0  029F           MOVW    R0,(R3)+                       ; Set data
              05  02A2           RSB                                    ; Return
                  02A3
                  02A3   HORIZONTAL_TAB:                                ; logical 15
   50    02   B0  02A3           MOVW    #2,R0                          ; Assume no mechtab
         08   E1  02A6           BBC     #TT$V_MECHTAB,-
03 04 A9          02A8                   TT_CHAR1(R9),10$               ; Set mechtab
   50    01   B0  02AB           MOVW    #1,R0                          ; set mechtab
                  02AE   10$:
   83    50   B0  02AE           MOVW    R0,(R3)+                       ; set data
              05  02B1           RSB                                    ; Return
                  02B2
                  02B2   VERTICAL_TAB:                                  ; logical 16
         83   B4  02B2           CLRW    (R3)+                          ; ignore
              05  02B4           RSB                                    ; Return
                  02B5
                  02B5   FORM_FEED:                                     ; logical 17
   50    02   B0  02B5           MOVW    #2,R0                          ; Assume no mechform
         13   E1  02B8           BBC     #TT$V_MECHFORM,-
03 04 A9          02BA                   TT_CHAR1(R9),10$               ; Set mechform
   50    01   B0  02BD           MOVW    #1,R0                          ; set mechform
                  02C0   10$:
   83    50   B0  02C0           MOVW    R0,(R3)+                       ; set data
              05  02C3           RSB                                    ; Return
                  02C4
                  02C4
```

```
                  02C4  ; cterm
                  02C4
                  02C4  IGNORE_INPUT:                               ; cterm 1
          83  94  02C4          CLRB    (R3)+                       ; Ignore
              05  02C6  10$:    RSB                                 ; Return
                  02C7
                  02C7  CHAR_ATTRIBUTES:                            ; cterm 2
                  02C7
                  02C7
              05  02C7          RSB
                  02C8
                  02C8  CONTROL_O_PASS_THRU:                        ; cterm 3
          83  94  02C8          CLRB    (R3)+                       ; Ignore
              05  02CA  10$:    RSB                                 ; Return
                  02CB
                  02CB  RAISE_INPUT:                                ; cterm 4
          50  94  02CB          CLRB    R0                          ; Assume lower case (i.e. NO RAISE)
          07  E0  02CD          BBS     #TT$V_LOWER,-
    03 04 A9      02CF                  TT_CHAR1(R9),10$            ; Branch if NO RAISE
   50     01  90  02D2          MOVB    #1,R0                       ; Set RAISE
                  02D5  10$:
   83     50  90  02D5          MOVB    R0,(R3)+                    ; set data
              05  02D8          RSB                                 ; Return
                  02D9
                  02D9  NORMAL_ECHO:                                ; cterm 5
          50  94  02D9          CLRB    R0                          ; (note opposite sense)
          01  E0  02DB          BBS     #TT$V_NOECHO,-
    03 04 A9      02DD                  TT_CHAR1(R9),10$            ; check noecho
   50     01  90  02E0          MOVB    #1,R0
                  02E3  10$:
   83     50  90  02E3          MOVB    R0,(R3)+                    ; set data
              05  02E6          RSB                                 ; Return
                  02E7
                  02E7  ;*** REQUEST_PROCESSING:                            ; cterm 6
                  02E7
                  02E7  INPUT_ESCAPE_ENABLE:                        ; cterm 7
                  02E7  OUTPUT_ESCAPE_ENABLE:                       ; cterm 8
          50  94  02E7          CLRB    R0
          03  E1  02E9          BBC     #TT$V_ESCAPE,-
    03 04 A9      02EB                  TT_CHAR1(R9),10$
   50     01  90  02EE          MOVB    #1,R0                       ; Set escape
                  02F1  10$:
   83     50  90  02F1          MOVB    R0,(R3)+
              05  02F4          RSB                                 ; Return
                  02F5
                  02F5  INPUT_COUNT_STATE:                          ; cterm 9
          83  B4  02F5          CLRW    (R3)+                       ; Ignore
              05  02F7          RSB
                  02F8
                  02F8  AUTO_PROMPT:                                ; cterm 10
          83  94  02F8          CLRB    (R3)+                       ; Ignore
              05  02FA  10$:    RSB                                 ; Return
                  02FB
                  02FB  ERROR_PROCESSING:                           ; cterm 11
          83  94  02FB          CLRB    (R3)+                       ; Ignore
              05  02FD  10$:    RSB                                 ; Return
                  02FE
```

```
            02FE    562
            02FE    563 IGNORE_STRING:
  83    94  02FE    564             CLRB    (R3)+
        05  0300    565             RSB
            0301    566
```

```
                    0301   568 POST_SENSE_EXIT:
                    0301   569
                    0301   570          ;
                    0301   571          ; Return R1 as correct length of characteristics
                    0301   572          ;
            51 8ED0 0301   573          POPL    R1                      ; start of buffer
51   53     51   C3 0304   574          SUBL3   R1,R3,R1                ; R1 = R3 - R1
                 05 0308   575          RSB
                    0309   576
                    0309   577 .end
```

```
AUTOBAUD_DETECT                  000001D9 R     02
AUTO_PROMPT                      000002F8 R     02
CH$C_CTERM                     = 00000002
CH$C_LOGICAL                   = 00000001
CH$C_PHYSICAL                  = 00000000
CHARACTER_SIZE                   0000019F R     02
CHAR_ATTRIBUTES                  000002C7 R     02
CHAR_COUNT                     = 0000000B
CHAR_TYPE                      = 00000200
CONTROL_O_PASS_THRU              000002C8 R     02
CR_FILL                          00000276 R     02
CTERM                          = 00000200
CTP$B_MSGTYPE                  = 0000002A
CTP$M_CH_KNOWN                 = 00000001
CTP$M_CH_SCOPE                 = 00000002
CTP$W_MSGSIZE                  = 00000028
CTSENSECHAR                      00000144 RG    02
ERROR_PROCESSING                 000002FB R     02
FLOW_CHAR_PASS_THRU              00000255 R     02
FORM_FEED                        000002B5 R     02
GET_SPEED                        00000195 R     02
HORIZONTAL_TAB                   000002A3 R     02
IGNORE_INPUT                     000002C4 R     02
IGNORE_STRING                    000002FE R     02
INPUT_COUNT_STATE                000002F5 R     02
INPUT_ESCAPE_ENABLE              000002E7 R     02
INPUT_FLOW_CONTROL               00000258 R     02
INPUT_SPEED                      0000017F R     02
LF_FILL                          00000285 R     02
LINE_WIDTH                       00000269 R     02
LOGICAL                        = 00000100
LOSS_NOTIFICATION                00000266 R     02
MANAGEMENT_GUARANTEED            000001E7 R     02
MODEM_PRESENT                    000001CB R     02
MODE_WRITING_ALLOWED             000001F2 R     02
NORMAL_ECHO                      000002D9 R     02
OUTPUT_ESCAPE_ENABLE             000002E7 R     02
OUTPUT_FLOW_CONTROL              00000239 R     02
OUTPUT_PAGE_STOP                 00000247 R     02
OUTPUT_SPEED                     0000018B R     02
PAGE_LENGTH                      0000026E R     02
PARITY_ENABLE                    000001AE R     02
PARITY_TYPE                      000001BC R     02
PHYSICAL                       = 00000000
POST_SENSE_EXIT                  00000301 R     02
RAISE_INPUT                      000002CB R     02
SENSE_BAUD                       00000096 R     02
SENSE_ROUTINES                   0000017F R     02
SENSE_TABLE                      00000000 R     02
STOP_LENGTH                      00000273 R     02
SWITCH_CHAR_1                    000001EA R     02
SWITCH_CHAR_2                    000001EE R     02
TERMINAL_SUBTYPE                 0000020B R     02
TERMINAL_TYPE                    000001F5 R     02
TERM_TABLE                       000000B8 R     02
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$V_MODEM                     = 00000015
TT$V_NOECHO                    = 00000001
TT$V_ODD                       = 00000007
TT$V_PARITY                    = 00000006
TT$V_SCOPE                     = 0000000C
TT$V_TTSYNC                    = 00000005
TT$V_WRAP                      = 00000009
TT$_LA100                      = 00000025
TT$_LA12                       = 00000024
TT$_LA24                       = 00000025
TT$_LA34                       = 00000022
TT$_LA36                       = 00000020
TT$_LA38                       = 00000023
TT$_LQP02                      = 00000026
TT$_UNKNOWN                    = 00000000
TT$_VK100                      = 00000002
TT$_VT05                       = 00000001
TT$_VT100                      = 00000060
TT$_VT101                      = 00000061
TT$_VT102                      = 00000062
TT$_VT105                      = 00000063
TT$_VT125                      = 00000064
TT$_VT131                      = 00000065
TT$_VT132                      = 00000066
TT$_VT173                      = 00000003
TT$_VT52                       = 00000040
TT$_VT55                       = 00000041
TT2$V_AUTOBAUD                 = 00000001
TT_BUF                         = 00000000
TT_CHAR1                       = 00000004
TT_CHAR2                       = 00000008
TT_IOSB                        = 00000000
UNKNOWN_TT                       0000013C R     02
VERTICAL_TAB                     000002B2 R     02
WRAP                             00000294 R     02
```

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

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (     0.)  00 (   0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00000000  (     0.)  01 (   1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
RTPAD                           00000309  (   777.)  02 (   2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

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

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     32    00:00:00.05    00:00:01.46
Command processing                122    00:00:00.50    00:00:03.53
Pass 1                            367    00:00:08.00    00:00:35.16
Symbol table sort                   0    00:00:01.37    00:00:03.69
Pass 2                            106    00:00:01.64    00:00:06.41
Symbol table output                13    00:00:00.07    00:00:00.07
Psect synopsis output               1    00:00:00.02    00:00:00.02
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              643    00:00:11.65    00:00:50.34

The working set limit was 1500 pages.
68631 bytes (135 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1274 non-local and 33 local symbols.
577 source lines were read in Pass 1, producing 16 object records in Pass 2.
17 pages of virtual memory were used to define 16 macros.

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

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[RTPAD.OBJ]RTPAD.MLB;1                    1
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     8
TOTALS (all libraries)                                9

1303 GETS were required to define 9 macros.

There were no errors, warnings or information messages.

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CTDSENSE
LIS

CTDVMSPEC
LIS

CTSENSERT
LIS

RSTSRT
LIS

CTERMRT
LIS

DTE_DF03
LIS

CTSETRT
LIS

CTDSET
LIS