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

FILEID**CTDVMSPEC

```
  CCCCCCCC  TTTTTTTTTT  DDDDDDDD    VV      VV  MM      MM    SSSSSSSS  PPPPPPPP    EEEEEEEEEE    CCCCCCCC
  CCCCCCCC  TTTTTTTTTT  DDDDDDDD    VV      VV  MM      MM    SSSSSSSS  PPPPPPPP    EEEEEEEEEE    CCCCCCCC
CC              TT      DD      DD  VV      VV  MMMM  MMMM  SS        PP      PP  EE          CC
CC              TT      DD      DD  VV      VV  MMMM  MMMM  SS        PP      PP  EE          CC
CC              TT      DD      DD  VV      VV  MM  MM  MM  SS        PP      PP  EE          CC
CC              TT      DD      DD  VV      VV  MM  MM  MM  SS        PP      PP  EE          CC
CC              TT      DD      DD  VV      VV  MM      MM    SSSSSS    PPPPPPPP    EEEEEEEE    CC
CC              TT      DD      DD  VV      VV  MM      MM    SSSSSS    PPPPPPPP    EEEEEEEE    CC
CC              TT      DD      DD  VV      VV  MM      MM        SS  PP          EE          CC
CC              TT      DD      DD  VV      VV  MM      MM        SS  PP          EE          CC
CC              TT      DD      DD    VV  VV    MM      MM        SS  PP          EE          CC                ....
CC              TT      DD      DD    VV  VV    MM      MM        SS  PP          EE          CC                ....
  CCCCCCCC      TT      DDDDDDDD        VV      MM      MM  SSSSSSSS  PP          EEEEEEEEEE    CCCCCCCC      ....
  CCCCCCCC      TT      DDDDDDDD        VV      MM      MM  SSSSSSSS  PP          EEEEEEEEEE    CCCCCCCC      ....

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

Table of contents

```
0000     1          .TITLE  CTDVMSPEC - CTDRIVER VMS specific protocol extensions
0000     2          .IDENT  'V04-000'
0000     3  ; *** DEBUG = 1
0000     4  .IF DF DEBUG
0000     5  .PRINT 0 ; CTVMSMODE being assembled WITH debug code
0000     6  .IFF
0000     7  ;*** .PRINT 0 ; CTVMSMODE being assembled WITHOUT debug code
0000     8  .ENDC
0000     9  ;
0000    10  ;****************************************************************************
0000    11  ;*                                                                          *
0000    12  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    13  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000    14  ;*  ALL RIGHTS RESERVED.                                                    *
0000    15  ;*                                                                          *
0000    16  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    17  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    18  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    19  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    20  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    21  ;*  TRANSFERRED.                                                            *
0000    22  ;*                                                                          *
0000    23  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    24  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    25  ;*  CORPORATION.                                                            *
0000    26  ;*                                                                          *
0000    27  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    28  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    29  ;*                                                                          *
0000    30  ;*                                                                          *
0000    31  ;****************************************************************************
0000    32  ;
0000    33
0000    34  ;++
0000    35  ;
0000    36  ; FACILITY:
0000    37  ;
0000    38  ;       VAX/VMS remote terminal driver
0000    39  ;
0000    40  ; ABSTRACT:
0000    41  ;
0000    42  ;       This module contains code to format and decode VMS to VMS specific
0000    43  ;       protocol used with the CTERM protocol.
0000    44  ;
0000    45  ; ENVIRONMENT:
0000    46  ;
0000    47  ;--
0000    48  ;
0000    49  ; AUTHOR: Jake VanNoy, CREATION DATE: 30-Dec-1982
0000    50  ;
0000    51  ; MODIFIED BY:
0000    52  ;
0000    53  ;       V03-002 JLV0290         Jake VanNoy             28-JUL-1983
0000    54  ;               Add upline broadcasts.
0000    55  ;
0000    56  ;       V03-001 JLV0262         Jake VanNoy             26-MAY-1983
0000    57  ;               Add missing branch in post processing.
```

```
                     0000     58 ;
                     0000     59 ;**
                     0000     60
                     0000     61         .SBTTL  DECLARATIONS
                     0000     62 ;
                     0000     63 ; INCLUDE FILES:
                     0000     64 ;
                     0000     65
                     0000     66         $DDBDEF                 ; device data block
                     0000     67         $IRPDEF                 ; I/O request packet
                     0000     68         $TSADEF                 ; cterm protocol
                     0000     69         $TT2DEF                 ; for DCL bit hacking
                     0000     70         $UCBDEF                 ; unit control block
                     0000     71         $TTYUCBDEF              ; terminal UCB extension
                     0000     72
                     0000     73 ; MACROS:
                     0000     74 ;
                     0000     75
                     0000     76 ;
                     0000     77 ; EQUATED SYMBOLS:
                     0000     78 ;
                     0000     79
00000000             0000     80 P1 = 0
00000004             0000     81 P2 = 4
00000008             0000     82 P3 = 8
0000000C             0000     83 P4 = 12
00000010             0000     84 P5 = 16
00000014             0000     85 P6 = 20
                     0000     86
                00000000     87         .PSECT $$$115_DRIVER,LONG
                     0000     88
                     0000     89 ;
                     0000     90 ; OWN STORAGE:
                     0000     91 ;
                     0000     92
```

```
                     0000     94          .SBTTL  CT_VMS_SETMODE - Packet VMS SET MODE
                     0000     95
                     0000     96 ;++
                     0000     97 ;
                     0000     98 ; FUNCTIONAL DESCRIPTION:
                     0000     99 ;
                     0000    100 ;       A mode message #10 is formatted with generic QIO format.
                     0000    101 ;
                     0000    102 ; CALLING SEQUENCE:
                     0000    103 ;       BSBW    CT_VMS_SETMODE
                     0000    104 ;
                     0000    105 ; INPUT PARAMETERS:
                     0000    106 ;
                     0000    107 ;       R1 - address of characteristics buffer
                     0000    108 ;       R2 - length of characteristics buffer (8 or 12)
                     0000    109 ;       R3 - IRP
                     0000    110 ;       R5 - UCB
                     0000    111 ;
                     0000    112 ; IMPLICIT INPUTS:
                     0000    113 ;       NONE
                     0000    114 ;
                     0000    115 ; OUTPUT PARAMETERS:
                     0000    116 ;
                     0000    117 ;       R2 - address of byte after message
                     0000    118 ;
                     0000    119 ; IMPLICIT OUTPUTS:
                     0000    120 ;       NONE
                     0000    121 ;
                     0000    122 ; COMPLETION CODES:
                     0000    123 ;       NONE
                     0000    124 ;
                     0000    125 ; SIDE EFFECTS:
                     0000    126 ;       NONE
                     0000    127 ;
                     0000    128 ;--
                     0000    129
                     0000    130 CT_VMS_SETMODE::
                     0000    131
        59   51   7D 0000    132         MOVQ    R1,R9                   ; Move R1,R2 to R9,R10
51   00000064 8F   D0 0003    133         MOVL    #<CTP$W_VMS_PLEN+<5*<6+4>>>,R1 ; overhead size
        51   5A   C0 000A    134         ADDL    R10,R1                  ; plus 8 or 12
           FFF0'  30 000D    135         BSBW    FDT_ALLOC_MESSAGE       ; Get a CTP
                     0010    136
           09   9B 0010    137         MOVZBW  #PRO$C_DATA,-
         26 A2      0012    138                 CTP$B_PRO_MSGTYPE(R2)   ; Set mode message
           0F   9B 0014    139         MOVZBW  #CTP$C_MT_VMSQIO,-
         2A A2      0016    140                 CTP$B_MSGTYPE(R2)       ; Set VAX generic qio message
           0F   90 0018    141         MOVB    #CTP$C_MT_VMSQIO,-
      0000'C3       001A    142                 IRP$B_CT_RESPTYPE(R3)   ; Mark as VMS QIO message
         50 A3   D0 001D    143         MOVL    IRP$L_SEQNUM(R3),-
         2C A2      0020    144                 CTP$L_VMS_REQID(R2)     ; Set response from server
         38 A3   D4 0022    145         CLRL    IRP$L_MEDIA(R3)         ; but no return data address (IOSB only)
         20 A3   B0 0025    146         MOVW    IRP$W_FUNC(R3),-
         30 A2      0028    147                 CTP$W_VMS_FUNC(R2)      ; Set qio function code
      50 32 A2   9E 002A    148         MOVAB   CTP$W_VMS_PLEN(R2),R0   ; Address of first parameter
                     002E    149         ;
                     002E    150         ; Do P1
```

```
                     002E   151          ;
    80   5A   B0   002E   152          MOVW    R10,(R0)+                       ; Length
    80   01   B0   0031   153          MOVW    #CTP$M_VMS_REF,(R0)+            ; Pass by reference
    80   01   B0   0034   154          MOVW    #1,(R0)+                        ; P1
    80   69   7D   0037   155          MOVQ    (R9),(R0)+                      ; first 8 bytes
40  A5   89   7D   003A   156          MOVQ    (R9)+,UCB$B_DEVCLASS(R5)        ; copy into UCB
    5A   0C   D1   003E   157          CMPL    #12,R10                         ; Is there 12?
         07   12   0041   158          BNEQ    20$                             ; Branch if not
    80   69   D0   0043   159          MOVL    (R9),(R0)+                      ; Last 4 bytes
48  A5   69   D0   0046   160          MOVL    (R9),UCB$L_DEVDEPND2(R5)        ; copy into UCB
                     004A   161 20$:
                     004A   162          ;
                     004A   163          ; Do P2
                     004A   164          ;
    80   04   D0   004A   165          MOVL    #4,(R0)+                        ; Length
    80   02   B0   004D   166          MOVW    #2,(R0)+                        ; P2
    80   5A   D0   0050   167          MOVL    R10,(R0)+                       ; 8 or 12
                     0053   168          ;
                     0053   169          ; Do P3
                     0053   170          ;
    80   04   D0   0053   171          MOVL    #4,(R0)+                        ; Length
    80   03   B0   0056   172          MOVW    #3,(R0)+                        ; P3
80    08 AC   D0   0059   173          MOVL    P3(AP),(R0)+                    ; parameter
                     005D   174          ;
                     005D   175          ; Do P4
                     005D   176          ;
    80   04   D0   005D   177          MOVL    #4,(R0)+                        ; Length
    80   04   B0   0060   178          MOVW    #4,(R0)+                        ; P4
80    0C AC   D0   0063   179          MOVL    P4(AP),(R0)+                    ; parameter
                     0067   180          ;
                     0067   181          ; Do P5
                     0067   182          ;
    80   04   D0   0067   183          MOVL    #4,(R0)+                        ; Length
    80   05   B0   006A   184          MOVW    #5,(R0)+                        ; P5
80    10 AC   D0   006D   185          MOVL    P5(AP),(R0)+                    ; parameter
                     0071   186
    52   50   D0   0071   187          MOVL    R0,R2                           ; Set address of next byte
                05   0074   188          RSB                                     ; return
```

```
                     0075   190          .SBTTL  CT_VMS_SENSEMODE - Packet VMS SENSE MODE
                     0075   191
                     0075   192 ;++
                     0075   193 ;
                     0075   194 ; FUNCTIONAL DESCRIPTION:
                     0075   195 ;
                     0075   196 ;       A VMS specific cterm message is formatted with generic QIO format.
                     0075   197 ;
                     0075   198 ; CALLING SEQUENCE:
                     0075   199 ;       BSBW    CT_VMS_SENSEMODE
                     0075   200 ;
                     0075   201 ; INPUT PARAMETERS:
                     0075   202 ;
                     0075   203 ;       R1 - address of characteristics buffer
                     0075   204 ;       R2 - length of characteristics buffer (8 or 12)
                     0075   205 ;       R3 - IRP
                     0075   206 ;       R5 - UCB
                     0075   207 ;
                     0075   208 ; IMPLICIT INPUTS:
                     0075   209 ;       NONE
                     0075   210 ;
                     0075   211 ; OUTPUT PARAMETERS:
                     0075   212 ;
                     0075   213 ;       R2 - address of byte after message
                     0075   214 ;
                     0075   215 ; IMPLICIT OUTPUTS:
                     0075   216 ;       NONE
                     0075   217 ;
                     0075   218 ; COMPLETION CODES:
                     0075   219 ;       NONE
                     0075   220 ;
                     0075   221 ; SIDE EFFECTS:
                     0075   222 ;       NONE
                     0075   223 ;
                     0075   224 ;--
                     0075   225
                     0075   226 CT_VMS_SENSEMODE::
                     0075   227
      59    51    7D  0075   228          MOVQ    R1,R9                           ; Move R1,R2 to R9,R10
51 00000046 8F    D0  0078   229          MOVL    #<CTP$W_VMS_PLEN+<2*<6+4>>>,R1 ; overhead size
      51    5A    C0  007F   230          ADDL    R10,R1                          ; plus 8 or 12
         FF7B'  30    0082   231          BSBW    FDT_ALLOC_MESSAGE               ; Get a CTP
                     0085   232
         09    9B    0085   233          MOVZBW  #PRO$C_DATA,-
        26 A2          0087   234                  CTP$B_PRO_MSGTYPE(R2)           ; Set mode message
         0F    9B    0089   235          MOVZBW  #CTP$C_MT_VMSQIO,-
        2A A2          008B   236                  CTP$B_MSGTYPE(R2)               ; Set VAX generic qio message
         0F    90    008D   237          MOVB    #CTP$C_MT_VMSQIO,-
      0000'C3          008F   238                  IRP$B_CT_RESPTYPE(R3)           ; Mark as VMS QIO message
        50 A3    D0    0092   239          MOVL    IRP$L_SEQNUM(R3),-
        2C A2          0095   240                  CTP$L_VMS_REQID(R2)             ; Set response from server
        20 A3    B0    0097   241          MOVW    IRP$W_FUNC(R3),-
        30 A2          009A   242                  CTP$W_VMS_FUNC(R2)              ; Set qio function code
   50   32 A2    9E    009C   243          MOVAB   CTP$W_VMS_PLEN(R2),R0           ; Address of first parameter
                     00A0   244          ;
                     00A0   245          ; Do P1
                     00A0   246          ;
```

```
                80   5A   B0   00A0   247          MOVW    R10,(R0)+                ; Length (not used in RTPAD***)
                80   04   B0   00A3   248          MOVW    #CTP$M_VMS_BUFFER,(R0)+  ; Generate buffer in server, return data
                80   01   B0   00A6   249          MOVW    #1,(R0)+                 ; P1
                               00A9   250                                            ; null data region
                               00A9   251          ;
                               00A9   252          ; Do P2
                               00A9   253          ;
                80   04   D0   00A9   254          MOVL    #4,(R0)+                 ; Length
                80   02   B0   00AC   255          MOVW    #2,(R0)+                 ; P2
                80   5A   D0   00AF   256          MOVL    R10,(R0)+                ; 8 or 12
                               00B2   257
                52   50   D0   00B2   258          MOVL    R0,R2                    ; Set address of next byte
                          05   00B5   259          RSB                              ; return
```

```
                        00B6   261          .SBTTL  CT_VMSQIO_MSG - Handle Mode message
                        00B6   262
                        00B6   263 ;++
                        00B6   264 ;
                        00B6   265 ; FUNCTIONAL DESCRIPTION:
                        00B6   266 ;
                        00B6   267 ;
                        00B6   268 ;
                        00B6   269 ; CALLING SEQUENCE:
                        00B6   270 ;
                        00B6   271 ;       BSBW    CT_VMSQIO_MSG
                        00B6   272 ;
                        00B6   273 ; INPUT PARAMETERS:
                        00B6   274 ;
                        00B6   275 ;       R1 - CTP
                        00B6   276 ;       R3 - Net read IIRP
                        00B6   277 ;       R5 - CT UCB
                        00B6   278 ;
                        00B6   279 ; IMPLICIT INPUTS:
                        00B6   280 ;       NONE
                        00B6   281 ;
                        00B6   282 ; OUTPUT PARAMETERS:
                        00B6   283 ;       NONE
                        00B6   284 ;
                        00B6   285 ; IMPLICIT OUTPUTS:
                        00B6   286 ;       NONE
                        00B6   287 ;
                        00B6   288 ; COMPLETION CODES:
                        00B6   289 ;       NONE
                        00B6   290 ;
                        00B6   291 ; SIDE EFFECTS:
                        00B6   292 ;       NONE
                        00B6   293 ;
                        00B6   294 ;--
                        00B6   295
                        00B6   296 ;
                        00B6   297 ; ??? IRP queue is assumed to be in order for reads, characteristics,
                        00B6   298 ; ??? and input counts. All these packets have the FUNC bit set in the
                        00B6   299 ; ??? queued IRP. The write packets can complete asynchronously to all
                        00B6   300 ; ??? these other types, and they do not have the FUNC bit set. Search
                        00B6   301 ; ??? for the associated IRP:
                        00B6   302 ;
                        00B6   303
                        00B6   304 MODE_ERR:
                  50    D4    00B6   305          CLRL    R0                      ; return error here... (***new 8-jun-83)
                        05    00B8   306          RSB
                        00B9   307
                        00B9   308 CT_VMSQIO_MSG::
                        00B9   309
            54  00B8  C5    7E 00B9   310          MOVAQ   UCB$L_RTT_IRPFL(R5),R4  ; Look through the irps for ours
                  50    54    D0 00BE   311          MOVL    R4,R0                   ; head of queue here
                        00C1   312
                  54    64    D0 00C1   313 30$:     MOVL    (R4),R4                 ; Link through chain
                  50    54    D1 00C4   314          CMPL    R4,R0                   ; end of irps?
                        ED    13 00C7   315          BEQL    MODE_ERR                ; Yes, could not find it, hangup
                              00C9   316
      0000'C4   0F    91    00C9   317          CMPB    #CTP$C_MT_VMSQIO, -
```

```
                     00CE   318                  IRP$B_CT_RESPTYPE(R4)   ; Does it match?
        F1   12      00CE   319          BNEQ    30$                     ; Branch if no
     2C A1   D1      00D0   320          CMPL    CTP$L_VMS_REQID(R1),-
     50 A4           00D3   321                  IRP$L_SEQNUM(R4)        ; Sequence number match?
        EA   12      00D5   322          BNEQ    30$                     ; Loop if not
                     00D7   323 ;
                     00D7   324 ; A match has been found
                     00D7   325 ;
     2C A3   D4      00D7   326          CLRL    IRP$L_SVAPTE(R3)        ; Buffer not in net iirp now
     32 A3   B0      00DA   327          MOVW    IRP$W_BCNT(R3),-
     3C A4           00DD   328                  IRP$L_IOST2(R4)         ; Set size of buffer read from net
  53    64   0F      00DF   329          REMQUE  (R4),R3                 ; Remove the CT irp from queue
2C A3   52   D0      00E2   330          MOVL    R2,IRP$L_SVAPTE(R3)     ; stick buffer there
                     00E6   331 ;
                     00E6   332 ; Function as "interrupt routine"
                     00E6   333 ;
  0200 8F   A8       00E6   334          BISW    #IRP$M_TERMIO,-
    2A A3            00EA   335                  IRP$W_STS(R3)           ; Set terminal I/O
54  38 A3   D0      00EC   336          MOVL    IRP$L_MEDIA(R3),R4      ; User buffer return address
    08   13      00F0   337          BEQL    50$                     ; Branch if no data (IOSB only)
62  38 A1   9E      00F2   338          MOVAB   CTP$T_VMS_RDATA(R1),(R2); Set address of data
04 A2   54   D0      00F6   339          MOVL    R4,4(R2)                ; Set user address to return data
                     00FA   340 50$:
                     00FA   341 ;
                     00FA   342 ; Set IRP$W_BCNT
                     00FA   343 ;
50  3C A3   3C      00FA   344          MOVZWL  IRP$L_IOST2(R3),R0      ; Set size of buffer read from net
   50   12   C2      00FE   345          SUBL2   #<CTP$T_VMS_RDATA- -
                     0101   346                  CTP$B_PRO_MSGTYPE>,R0   ; Subtract out header size
32 A3   50   B1      0101   347          CMPW    R0,IRP$W_BCNT(R3)       ; Compare to user buffer size
    04   1A      0105   348          BGTRU   80$                     ; If greater, leave user size
32 A3   50   B0      0107   349          MOVW    R0,IRP$W_BCNT(R3)       ; Set return size
                     010B   350 80$:
    30 A1   7D      010B   351          MOVQ    CTP$Q_VMS_IOSB(R1),-
    38 A3           010E   352                  IRP$L_IOST1(R3)         ; Copy IOSB
                     0110   353
  06   00   ED      0110   354          CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,-
00000000'8F  20 A3   0113   355                  IRP$W_FUNC(R3),#IO$_SENSEMODE   ; sense mode qio?
    02   12      011A   356          BNEQ    90$                     ; branch if not
    0A   10      011C   357          BSBB    SENSE_SPAWN             ; Set DCL bits ***
                     011E   358 90$:
      00000000'GF    16      011E   359          JSB     G^COM$POST              ; POST I/O
   50   01   D0      0124   360          MOVL    #1,R0                   ; success
                     05      0127   361          RSB                     ; return to FOUNDATION message code
                             0128   362
```

```
                     0128   364          .SBTTL  SENSE_SPAWN  Sense for spawn
                     0128   365
                     0128   366 ;       Sense special characteristics bits for DCL spawn command.
                     0128   367 ;       Return bits for ctrl/c ast, outofband ast and associated mailbox.
                     0128   368 ;       These bits may be reused later and are not for customer application
                     0128   369 ;       consumption.
                     0128   370 ;
                     0128   371 ; inputs:
                     0128   372 ;       r1 -> CTP
                     0128   373 ;
                     0128   374
                     0128   375 SENSE_SPAWN:
   50    40 A1   9E  0128   376          MOVAB   CTP$T_VMS_RDATA+8(R1),R0 ; Set address of data
                     012C   377
60    0200 8F   AA  012C   378          BICW    #TT2$M_DCL_MAILBX,(R0)  ; Reset mailbox bit
      60 A5     D5  0131   379          TSTL    UCB$L_AMB(R5)           ; Any associated mailbox?
         05     13  0134   380          BEQL    10$                     ; No
60    0200 8F   A8  0136   381          BISW    #TT2$M_DCL_MAILBX,(R0)  ; Yes, so set characteristic
                     013B   382 10$:
                  05 013B   383          RSB
```

```
                                013C   385 .SBTTL  CT_VMS_BRDCST - Handle upline dump of broadcast text
                                013C   386 ;+
                                013C   387 ; CT_VMS_BRDCST -
                                013C   388 ;
                                013C   389 ; Deliver broadcast message to the mailbox.
                                013C   390 ;
                                013C   391 ; If RTPAD receives a broadcast message in it's mailbox, it dumps it up
                                013C   392 ; to the remote terminal and it gets here for delivery to the real
                                013C   393 ; application broadcast mailbox.
                                013C   394 ;
                                013C   395 ; The unit number and name of the device is fixed up in the packet first.
                                013C   396 ;
                                013C   397 ;-
                                013C   398
                                013C   399 CT_VMS_BRDCST::
                                013C   400
            38 48 A5    04   E1 013C   401         BBC     #TT2$V_BRDCSTMBX, -                ; If we are allowing mailbox
                                0141   402                 UCB$L_DEVDEPND2(R5),10$            ; to receive the messages
                    60 A5   D5 0141   403         TSTL    UCB$L_AMB(R5)                      ; and we have a mailbox
                       33   13 0144   404         BEQL    10$                                ; Nope
            2F A1    54 A5   B0 0146   405         MOVW    UCB$W_UNIT(R5), -                  ; Then fix the unit number
                                014B   406                 CTP$W_BR_UNITNUM(R1)               ; in the message
                52    28 A5   D0 014B   407         MOVL    UCB$L_DDB(R5), R2                  ; and get the proper name of
      50    14 A2    04   00   EF 014F   408         EXTZV   #0, #4, DDB$T_NAME(R2), R0         ; this device for the message
                       50   D6 0155   409         INCL    R0                                 ; including the count
                                0157   410
                       3F   BB 0157   411         PUSHR   #^M<R0, R1, R2, R3, R4, R5>        ; Copy the new name and
31 A1   10   00   14 A2   50   2C 0159   412         MOVC5   R0, DDB$T_NAME(R2), #0, -          ; clobber the remainder of the
                                0161   413                 #CTP$S_BR_DEVNAME, -               ; stuff in the fixed length
                                0161   414                 CTP$T_BR_DEVNAME(R1)               ; field
                       3F   BA 0161   415         POPR    #^M<R0, R1, R2, R3, R4, R5>        ; restore the regs
                                0163   416
                       38   BB 0163   417         PUSHR   #^M<R3, R4, R5>                    ; Save a few
         53    28 A1   3C   0165   418         MOVZWL  CTP$W_MSGSIZE(R1), R3              ; Size of the message
         54    2D A1   9E   0169   419         MOVAB   CTP$W_BR_MSGCODE(R1), R4           ; Address of the message
         55    60 A5   D0   016D   420         MOVL    UCB$L_AMB(R5), R5                  ; Mailbox ucb address
00000000'GF          16   0171   421         JSB     G^EXE$WRTMAILBOX                   ; Write the message to it
                       38   BA 0177   422         POPR    #^M<R3, R4, R5>                    ; and ignore the errors
                                0179   423
                       05   0179   424 10$:     RSB
                                017A   425
                                017A   426 .END
```

```
COM$POST                        ********    X   02
CTP$B_MSGTYPE                 = 0000002A
CTP$B_PRO_MSGTYPE             = 00000026
CTP$C_MT_VMSQIO               = 0000000F
CTP$L_VMS_REQID               = 0000002C
CTP$M_VMS_BUFFER              = 00000004
CTP$M_VMS_REF                 = 00000001
CTP$Q_VMS_IOSB                = 00000030
CTP$S_BR_DEVNAME              = 00000010
CTP$T_BR_DEVNAME              = 00000031
CTP$T_VMS_RDATA               = 00000038
CTP$W_BR_MSGCODE              = 0000002D
CTP$W_BR_UNITNUM              = 0000002F
CTP$W_MSGSIZE                 = 00000028
CTP$W_VMS_FUNC                = 00000030
CTP$W_VMS_PLEN                = 00000032
CT_VMSQIO_MSG                   000000B9 RG     02
CT_VMS_BRDCST                   0000013C RG     02
CT_VMS_SENSEMODE                00000075 RG     02
CT_VMS_SETMODE                  00000000 RG     02
DDB$T_NAME                    = 00000014
EXE$RTMAILBOX                   ********    X   02
FDT_ALLOC_MESSAGE               ********    X   02
IO$_SENSEMODE                   ********    X   02
IRP$B_CT_RESPTYPE               ********    X   02
IRP$L_IOST1                   = 00000038
IRP$L_IOST2                   = 0000003C
IRP$L_MEDIA                   = 00000038
IRP$L_SEQNUM                  = 00000050
IRP$L_SVAPTE                  = 0000002C
IRP$M_TERMIO                  = 00000200
IRP$S_FCODE                   = 00000006
IRP$V_FCODE                   = 00000000
IRP$W_BCNT                    = 00000032
IRP$W_FUNC                    = 00000020
IRP$W_STS                     = 0000002A
MODE_ERR                        000000B6 R      02
P1                            = 00000000
P2                            = 00000004
P3                            = 00000008
P4                            = 0000000C
P5                            = 00000010
P6                            = 00000014
PRO$C_DATA                    = 00000009
SENSE_SPAWN                     00000128 R      02
TT2$M_DCL_MAILBX              = 00000200
TT2$V_BRDCSTMBX               = 00000004
UCB$B_DEVCLASS                = 00000040
UCB$L_AMB                     = 00000060
UCB$L_DDB                     = 00000028
UCB$L_DEVDEPND2               = 00000048
UCB$L_RTT_IRPFL               = 000000B8
UCB$W_UNIT                    = 00000054
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$$$115_DRIVER	0000017A (378.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.06	00:00:00.37
Command processing	129	00:00:00.50	00:00:03.75
Pass 1	369	00:00:08.04	00:00:27.93
Symbol table sort	0	00:00:01.44	00:00:03.29
Pass 2	79	00:00:01.57	00:00:03.74
Symbol table output	7	00:00:00.06	00:00:00.06
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	617	00:00:11.69	00:00:39.17

The working set limit was 1500 pages.
67161 bytes (132 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1350 non-local and 7 local symbols.
426 source lines were read in Pass 1, producing 14 object records in Pass 2.
13 pages of virtual memory were used to define 12 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	4
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	9

1412 GETS were required to define 9 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:CTDVMSPEC/OBJ=OBJ$:CTDVMSPEC MSRC$:CTDVMSPEC/UPDATE=(ENH$:CTDVMSPEC)+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