F 6

```
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
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
```

_S

Ps
--

. (

FILEID**RSTSRT

D 15

```
RRRRRRRR      SSSSSSSS  TTTTTTTTTT    SSSSSSSS  RRRRRRRR    TTTTTTTTTT
RRRRRRRR      SSSSSSSS  TTTTTTTTTT    SSSSSSSS  RRRRRRRR    TTTTTTTTTT
RR      RR  SS              TT      SS          RR      RR      TT
RR      RR  SS              TT      SS          RR      RR      TT
RR      RR  SS              TT      SS          RR      RR      TT
RR      RR  SS              TT      SS          RR      RR      TT
RRRRRRRR      SSSSSS        TT        SSSSSS    RRRRRRRR        TT
RRRRRRRR      SSSSSS        TT        SSSSSS    RRRRRRRR        TT
RR  RR              SS      TT              SS  RR  RR          TT
RR  RR              SS      TT              SS  RR  RR          TT
RR    RR            SS      TT              SS  RR    RR        TT          ....
RR    RR            SS      TT              SS  RR    RR        TT          ....
RR      RR  SSSSSSSS        TT      SSSSSSSS    RR      RR      TT          ....
RR      RR  SSSSSSSS        TT      SSSSSSSS    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

```
0000   .TITLE  RSTSRT  RSTS/E Remote Terminal Protocol
0000   .IDENT  'V04-000'
0000
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   ; FACILITY: DECnet remote terminals
0000   ;
0000   ; ABSTRACT:
0000   ;
0000   ;       Handles the RSTS/E DECnet remote terminal protocol.
0000   ;
0000   ; ENVIRONMENT: VMS - USER MODE
0000   ;
0000   ; AUTHOR: Mark Bramhall            CREATION DATE: 29-Jan-1980
0000   ;
0000   ; MODIFICATION HISTORY:
0000   ;
0000   ;       V03-009         MHB0099         M. H. Bramhall          10-Jan-1984
0000   ;                       Fix automatic login for RSTS/E and TOPS-20.
0000   ;
0000   ;       V03-008         MHB0094         M. H. Bramhall           7-Mar-1983
0000   ;                       Added 8-bit terminal support.
0000   ;
0000   ;       V03-007         MHB0082         M. H. Bramhall           2-Sep-1982
0000   ;                       Fix terminal hang up looping problem.
0000   ;                       Add TOPS-20 PASSALL mode.
0000   ;
0000   ;       V03-006         WMC0045         Wayne Cardoza           26-Feb-1982
0000   ;                       Add flag to detect spurious $WAKE.
0000   ;
0000   ;       V03-005         MHB0080         M. H. Bramhall          25-Jan-1982
0000   ;                       Restore terminal characteristics upon exit.
0000   ;
```

```
0000    ;       V03-004         MHB0073         M. H. Bramhall          11-May-1981
0000    ;                       Make logging file be implied carriage control format.
0000    ;
0000    ;       V03-003         MHB0072         M. H. Bramhall          20-Apr-1981
0000    ;                       Added logging file capability.
0000    ;
0000    ;       V03-002         MHB0065         M. H. Bramhall          23-Sep-1980
0000    ;                       A large re-write to correct many problems.
0000    ;                       Added indirect file capability.
0000    ;--
```

```
                0000   .SBTTL  Declarations
                0000
                0000   .DEFAULT DISPLACEMENT WORD
                0000
                0000            $DIBDEF                                 ; Device information buffer defs
                0000            $DSCDEF                                 ; Descriptor defs
                0000            $TTDEF                                  ; Terminal characteristics defs
                0000
                0000   .SBTTL  Macros
                0000
                0000            .MACRO  QUIT       STATUS
                0000            .IF     NB         <STATUS>
                0000                    MOVL       STATUS, R0
                0000            .ENDC   ; NB       <STATUS>
                0000                    BRW        QUIT
                0000            .ENDM   QUIT
                0000
                0000            .MACRO  QUIT_NOT_ABORT   ?OK
                0000                    BLBS       R0, OK
                0000                    BRW        ABORT_QUIT
                0000                    OK:
                0000            .ENDM   QUIT_NOT_ABORT
                0000
                0000            .MACRO  QUIT_ON_ERROR    ?OK
                0000                    BLBS       R0, OK
                0000                    QUIT
                0000                    OK:
                0000            .ENDM   QUIT_ON_ERROR
                0000
                0000   .SBTTL  R/W and R-O data areas
                0000
            00000000   .PSECT  _RSTSRT, LONG
                0000
                0000   TOPS20:                                          ; TOPS-20 flag
          00    0000            .BYTE   0
                0001
                0001   EIGHT_BIT:                                       ; 8-bit terminal flag
          00    0001            .BYTE   0
                0002
                0002   CTRLC_CNT:                                       ; Count -1 of pending CTRL/C's
    FFFFFFFF    0002            .LONG   -1
                0006
                0006   CTRLC_SENT:                                      ; CTRL/C sent flag
          00    0006            .BYTE   0
                0007
                0007   CTRLC_MATCH:                                     ; Match string for CTRL/C's
       43 5E    0007            .ASCII  /^C/
                0009
                0009   OUTPUT_WAIT:                                     ; Terminal output wait flag
          00    0009            .BYTE   0
                000A
    00000200    000A   LINKBUFLEN      =       512                      ; Length of link read buffer
                000A
                000A   LINKIOSB:                                        ; Link read I/O status block
    00000012    000A            .BLKQ
                0012
                0012   LINKBUFFER:                                      ; Link read buffer
```

```
          00000212  0012                  .BLKB   LINKBUFLEN
                    0212
          000000FF  0212    TERBUFLEN     =       255             ; Length of terminal read buffer
                    0212
                    0212    TERIOSB:                              ; Terminal read I/O status block
          0000021A  0212                  .BLKQ
                    021A
                05  021A                  .BYTE   5               ;  TYPE = 5 => terminal data
          0000021D  021B                  .BLKW                   ;  LENGTH = message's length
          0000021E  021D                  .BLKB                   ;  DATA = data's length
                    021E    TERBUFFER:                            ; Terminal read buffer
          0000031D  021E                  .BLKB   TERBUFLEN
                    031D
                    031D    ECHOBUFFER:                           ; Terminal read echo buffer
          0000031F  031D                  .BLKB   2
             0A 0D  031F                  .BYTE   13, 10
                    0321
                    0321    READ_MODE:                            ; Terminal read mode
             0000'  0321                  .WORD   IO$_READVBLK!IO$M_TRMNOECHO!IO$M_DSABLMBX
                    0323
                    0323    TERMASK:                              ; Terminal read terminator mask
0000032B'00000020'  0323                  .LONG   20$-10$, 10$    ;  Long form terminator mask:
          FFFFFDFE  032B    10$:          .LONG   ^C<<1@0>!<1@9>> ;    All  of   0 -31 except <NUL>, <TAB>
          00000000  032F                  .LONG   0               ;    None of  32- 63
          00000000  0333                  .LONG   0               ;    None of  64- 95
          00000000  0337                  .LONG   0               ;    None of  96-127
          FFFFFFFF  033B                  .LONG   ^C<0>           ;    All  of 128-159
          00000001  033F                  .LONG   1@0             ;    None of 160-191 except <10/0>
          00000000  0343                  .LONG   0               ;    None of 192-223
          80000000  0347                  .LONG   1@31            ;    None of 224-255 except <15/15>
                    034B    20$:
                    034B
                    034B    ASCII_TRIM:                           ; Translation table for 7-bit ASCII
                    034B                  .REPT   256
                    034B                  .BYTE   <.-ASCII_TRIM>&127
                00  034B                  .ENDR
                    044B
                    044B    CTRLC_IOSB:                           ; CTRL/C message I/O status block
          00000453  044B                  .BLKQ
                    0453
                    0453    CTRLC_MSG:                            ; CTRL/C message
                05  0453    10$:          .BYTE   5               ;  TYPE = 5 => terminal data
             0005'  0454                  .WORD   30$-10$         ;  LENGTH = message's length
               01'  0456                  .BYTE   30$-20$         ;  DATA = data's length
                03  0457    20$:          .BYTE   ^A/C/-64
                    0458    30$:
                    0458
          00000100  0458    CMDBUFLEN     =       256             ; Length of command input buffer
                    0458
                    0458    CMDIOSB:                              ; Command input I/O status block
          00000460  0458                  .BLKQ
                    0460
                    0460    CMDBUFFER:                            ; Command input buffer
          00000560  0460                  .BLKB   CMDBUFLEN
                    0560
                    0560    CMDMASK:                              ; Command read terminator mask
00000568'00000020'  0560                  .LONG   20$-10$, 10$    ;  Long form terminator mask:
```

```
                    FFFFFFFF  0568   10$:     .LONG    ^C<0>                          ;   All  of   0- 31
                    00000000  056C            .LONG    0                              ;   None of  32- 63
                    00000000  0570            .LONG    0                              ;   None of  64- 95
                    80000000  0574            .LONG    1@31                           ;   None of  96-127 except <DEL>
                    FFFFFFFF  0578            .LONG    ^C<0>                          ;   All  of 128-159
                    00000001  057C            .LONG    1@0                            ;   None of 160-191 except <10/0>
                    00000000  0580            .LONG    0                              ;   None of 192-223
                    80000000  0584            .LONG    1@31                           ;   None of 224-255 except <15/15>
                              0588   20$:
                              0588
                              0588   CMDPROMPT:                                       ; Command input prompt
                    02000000  0588            .LONG    <DSC$K_CLASS_D>@<DSC$B_CLASS*8>
                    00000000  058C            .LONG    0
                              0590
                              0590   CRLFPROMPT:                                      ; <CR><LF> for utility string
OA OD 00000598'010E0000'      0590            .ASCID   <13><10>
                              059A
                              059A   ERASEPROMPT:                                     ; String to erase the prompting
      000005A2'0000032A'      059A            .LONG    20$-10$, 10$
                              05A2   10$:
                              05A2   .REPT    <1+6+2>+5+CMDBUFLEN                     ; _nnnnnn::REM> command...
                              05A2            .BYTE    8, 32, 8
                    08 20 08  05A2   .ENDR
                              08CC   20$:
                              08CC
                              08CC   CMDHELP:                                         ; Command help message
                    02000000  08CC            .LONG    <DSC$K_CLASS_D>@<DSC$B_CLASS*8>
                    00000000  08D0            .LONG    0
                              08D4
                              08D4   CMDODT:                                          ; ODT -or- PASSALL command
                    000008D6  08D4            .BLKW
                              08D6
                              08D6   .ENABLE LSB
                              08D6
                              08D6   CNTLBF:                                          ; CONTROL message
                          02  08D6   10$:     .BYTE    2                              ;  TYPE = 2 => CONTROL
                       0006'  08D7            .WORD    40$-10$                        ;  LENGTH = message's length
                         01'  08D9            .BYTE    30$-20$                        ;  MENU = menu's length
                          01  08DA   20$:     .BYTE    1@0
                              08DB   30$:
                              08DB   CNTLBF_ECHOFLG:                                  ; Echo flag ( <0> )
                          00  08DB            .BYTE    0
                              08DC   40$:
                              08DC
                              08DC   .DISABLE LSB
                              08DC
                    00000028  08DC   LINKMBXBUFLEN   =       40                       ; Length of link mailbox buffer
                              08DC
                              08DC   LINKMBXIOSB:                                     ; Link mailbox I/O status block
                    000008E4  08DC            .BLKQ
                              08E4
                              08E4   LINKMBXBUFFER:                                   ; Link mailbox buffer
                    0000090C  08E4            .BLKB    LINKMBXBUFLEN
                              090C
                    00000028  090C   TERMMBXBUFLEN   =       40                       ; Length of terminal mailbox buffer
                              090C
                              090C   TERMMBXIOSB:                                     ; Terminal mailbox I/O status block
```

```
                        00000914  090C              .BLKQ
                                  0914
                                  0914  TERMMBXBUFFER:                         ; Terminal mailbox buffer
                        0000093C  0914              .BLKB   TERMMBXBUFLEN
                                  093C
                                  093C  RMSMSGVEC:                             ; Model $PUTMSG vector for RMS msg's
                        00000002  093C              .LONG   2
               00000000 00000000  0940              .LONG   0, 0
                                  0948
                                  0948  NETMBXVEC:                             ; $PUTMSG vector for mailbox msg's
                        00000003  0948              .LONG   3
                        00000000' 094C              .LONG   REM$_NETMBX
                        00000001  0950              .LONG   1
                        00000000  0954              .LONG   0
                                  0958
                                  0958  CONFBF:                                ; Initial CONFIG message
                              01  0958              .BYTE   1                  ;  TYPE = 1 => CONFIG
                           0003'  0959              .WORD   CONFBF_LEN         ;  LENGTH = message's length
                        00000003  095B  CONFBF_LEN          =       .-CONFBF
                                  095B
                                  095B  .ENABLE LSB
                                  095B
                                  095B  TTYPBF:                                ; Initial CONTROL message
                              02  095B  10$:        .BYTE   2                  ;  TYPE = 2 => CONTROL
                           000B'  095C              .WORD   40$-10$            ;  LENGTH = message's length
                             01'  095E              .BYTE   30$-20$            ;  MENU = menu's length
                              1C  095F  20$:        .BYTE   <1@2>!<1@3>!<1@4>
                                  0960  30$:
                                  0960  TTYPBF_WIDTH:                          ; Terminal's width ( <2> )
                            0000  0960              .WORD   0
                                  0962  TTYPBF_TYPE:                           ; Terminal's type ( <3> )
                            0000  0962              .WORD   0
                                  0964  TTYPBF_FILL:                           ; Terminal's fill factor ( <4> )
                            0000  0964              .WORD   0
                                  0966  40$:
                                  0966
                                  0966  .DISABLE LSB
                                  0966
                                  0966  REMPROMPT:                             ; Main prompt for forming prompt string
  20 3E 4D 45 52 0000096E'010E0000'  0966           .ASCID  /REM> /
                                  0973
                                  0973  HELPPROMPT:                            ; Help text for forming help string
                                  0973              .ASCID -
            0D 0000097B'010E0000'  0973                                                          <13><10>-
69 6C 61 56 20 20 20 20 20 20 20 0A  097C          %       Valid commands:%                      <13><10>-
0D 3A 73 64 6E 61 6D 6D 6F 63 20 64  0988
                              0D 0A  0994                                                        <13><10>-
20 20 20 20 20 20 20 20 20 78 5E 0A  0996          %^x         Send a CTRL/x to remote node%     <13><10>-
2F 4C 52 54 43 20 61 20 64 6E 65 53  09A2
20 65 74 6F 6D 65 72 20 6F 74 20 78  09AE
                  0D 65 64 6F 6E  09BA
20 5D 45 55 4E 49 54 4E 5B 4F 43 0A  09BF          %CO[NTINUE] Return to terminal mode%      <13><10>-
65 74 20 6F 74 20 6E 72 75 74 65 52  09CB
0D 65 64 6F 6D 20 6C 61 6E 69 6D 72  09D7
20 20 20 20 5D 45 53 4F 5B 4C 43 0A  09E3          %CL[OSE]    Close any logging file%       <13><10>-
6F 6C 20 79 6E 61 20 65 73 6F 6C 43  09EF
   0D 65 6C 69 66 20 67 6E 69 67 67  09FB
```

```
20 20 20 5D 50 2F 4C 52 5B 54 43 0A  0A06            %CT[RL/P]  Send a CTRL/P to remote node%       <13><10>-
2F 4C 52 54 43 20 61 20 64 6E 65 53  0A12
20 65 74 6F 6D 65 72 20 6F 74 20 50  0A1E
               0D 65 64 6F 6E  0A2A
20 20 20 20 20 5D 54 49 5B 58 45 0A  0A2F            %EX[IT]    Exit remote terminal session%       <13><10>-
20 65 74 6F 6D 65 72 20 74 69 78 45  0A3B
73 65 73 20 6C 61 6E 69 6D 72 65 74  0A47
               0D 6E 6F 69 73  0A53
20 20 20 20 20 5D 50 4C 5B 45 48 0A  0A58            %HE[LP]    Type this help message%             <13><10>-
65 68 20 73 69 68 74 20 65 70 79 54  0A64
   0D 65 67 61 73 73 65 6D 20 70 6C  0A70
20 65 6C 69 66 20 5D 47 5B 4F 4C 0A  0A7B            %LO[G] file Create and use the logging file%   <13><10>-
75 20 64 6E 61 20 65 74 61 65 72 43  0A87
69 67 67 6F 6C 20 65 68 74 20 65 73  0A93
         0D 65 6C 69 66 20 67 6E  0A9F
20 20 20 5D 4C 41 4D 52 5B 4F 4E 0A  0AA7            %NO[RMAL]  Restore NORMAL mode%                <13><10>
4D 52 4F 4E 20 65 72 6F 74 73 65 52  0AB3
      0A 0D 65 64 6F 6D 20 4C 41  0ABF
                                     0AC8
                                     0AC8  RSTSHELP:                                           ; Help text for RSTS/E only
                                     0AC8            .ASCID -
20 5D 54 5B 44 4F 00000AD0'010E0000' 0AC8            %OD[T]     Enter ODT mode%                     <13><10>
4F 20 72 65 74 6E 45 20 20 20 20 20  0AD6
      0A 0D 65 64 6F 6D 20 54 44  0AE2
                                     0AEB
                                     0AEB  TOPS20HELP:                                         ; Help text for TOPS-20 only
                                     0AEB            .ASCID -
41 53 53 5B 41 50 00000AF3'010E0000' 0AEB            %PA[SSALL] Enter PASSALL mode%                 <13><10>
50 20 72 65 74 6E 45 20 20 5D 4C 4C  0AF9
0D 65 64 6F 6D 20 4C 4C 41 53 53 41  0B05
                                 0A  0B11
                                     0B12
                                     0B12  HELLO:                                              ; Login command start for RSTS/E
         0D 4F 4C 4C 45 48 03 00'    0B12            .ASCIC  <^A'C'&31>/HELLO/<13>
                                 07  0B12
                                     0B1A
                                     0B1A  LOGIN:                                              ; Login command start for TOPS-20
         20 4E 49 47 4F 4C 03 00'    0B1A            .ASCIC  <^A'C'&31>/LOGIN /
                                 07  0B1A
                                     0B22
                                     0B22  CONNECT:                                            ; Connection established message vector
                          00000003   0B22            .LONG   3
                          00000000'  0B26            .LONG   REM$_REMOTE
                          00000001   0B2A            .LONG   1
                          00000000'  0B2E            .LONG   FINALPATH
                                     0B32
                                     0B32  LOGGING_FLAG:                                       ; Logging file active flag
                                 00  0B32            .BYTE   0
                                     0B33
                                     0B33  LOGGING_FDEF:                                       ; Logging file's defaults
      47 4F 4C 2E 44 41 50 54 52     0B33            .ASCII  /RTPAD.LOG/
                          00000009   0B3C  LOGGING_FDEF_LEN = .-LOGGING_FDEF
                                     0B3C
                                     0B3C            .ALIGN  LONG
                                     0B3C
                                     0B3C  LOGGING_FAB:                                        ; FAB for logging
                                     0B3C            $FAB -                                    ; Allocate a FAB
```

```
                                  0B3C                  DNA = LOGGING_FDEF, -       ;  address of logging file defaults
                                  0B3C                  DNS = LOGGING_FDEF_LEN, - ;  length of logging file defaults
                                  0B3C                  FAC = PUT, -                ;  we'll be doing PUT's
                                  0B3C                  FOP = SQO, -                ;  we'll only do sequential access
                                  0B3C                  ORG = SEQ, -                ;  it should be sequentially organized
                                  0B3C                  RAT = CR, -                 ;  use implied carriage control format
                                  0B3C                  RFM = VAR                   ;  use variable length records
                                  0B8C
                                  0B8C          .ALIGN  LONG
                                  0B8C
                                  0B8C  LOGGING_RAB:                                ; RAB for logging
                                  0B8C          $RAB -                              ; Allocate a RAB
                                  0B8C                  FAB = LOGGING_FAB, -        ;  use the logging file FAB
                                  0B8C                  RAC = SEQ, -                ;  we'll do sequential accesses
                                  0B8C                  ROP = WBH                   ;  we want write behind for speed
                                  0BD0
                                  0BD0  LOGGING_MSGVEC:                             ; Logging file error $PUTMSG vector
00000000 00000000 00000002        0BD0          .LONG   2, 0, 0
                                  0BDC
                        000000FF  0BDC  LOGGING_BUFLEN = 255                        ; Logging file record buffer length
                                  0BDC
                                  0BDC  LOGGING_BUFDESC:                            ; Logging file record buffer descriptor
                        00000BE4  0BDC          .BLKQ
                                  0BE4
                                  0BE4  LOGGING_BUFFER:                             ; Logging file record buffer
                        00000CE3  0BE4          .BLKB   LOGGING_BUFLEN
```

```
                         0CE3   353 .SBTTL  Initialize RSTS/E Remote Terminal Protocol
                         0CE3   354
                         0CE3   355 .SHOW   MEB
                         0CE3   356
                     00000000   357 .PSECT  PROTOTBL, BYTE, NOEXE
                         0000   358
              0001       0000   359         .WORD   1@0                         ; <0> => RSTS/E protocol
          00000000'      0002   360         .LONG   RSTSRT                      ; This is our initialization entry
                         0006   361
                     00000000   362 .PSECT  RSTSRT, NOWRT
                         0000   363
                         0000   364 RSTSRT::                                    ; Initialize RSTS/E protocol
              0FFC       0000   365         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                         0002   366
          0000'CF    D4  0002   367         CLRL    RETSTATUS                   ; Pre-clear return status
                         0006   368
                         0006   369 ; Format and send CONFIG message
                         0006   370
                         0006   371         $QIOW_S -                           ; Send the initial CONFIG message
                         0006   372                 CHAN = LINKCHAN, -          ;  on the link channel
                         0006   373                 FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                         0006   374                 IOSB = LINKIOSB, -          ;  use an IOSB
                         0006   375                 P1 = CONFBF, -              ;  pre-built CONFIG message
                         0006   376                 P2 = #CONFBF_LEN            ;   and its length
               7E    7C  0006                       CLRQ    -(SP)
               7E    7C  0008                       CLRQ    -(SP)
               03    DD  000A                       PUSHL   #CONFBF_LEN
          0958'CF    DF  000C                       PUSHAL  CONFBF
               7E    7C  0010                       CLRQ    -(SP)
          000A'CF    7F  0012                       PUSHAQ  LINKIOSB
          7E  00'    3C  0016                       MOVZWL  S^#IO$_WRITEVBLK,-(SP)
     7E   0000'CF    3C  0019                       MOVZWL  LINKCHAN,-(SP)
               00    DD  001E                       PUSHL   #0
  00000000'GF  0C    FB  0020                       CALLS   #12,G^SYS$QIOW
                         0027   377         QUIT_ON_ERROR                       ; Quit on any error
            03 50    E8  0027                       BLBS    R0, 30000$
             0304    31  002A                       BRW     QUIT
                         002D                       30000$:
     50   000A'CF    3C  002D   378         MOVZWL  LINKIOSB, R0                ; Get the I/O completion status
                         0032   379         QUIT_ON_ERROR                       ;  and quit on any error
            03 50    E8  0032                       BLBS    R0, 30001$
             02F9    31  0035                       BRW     QUIT
                         0038                       30001$:
                         0038   380
                         0038   381 ; Format and send CONTROL message
                         0038   382
     51   0004'CF    D0  0038   383         MOVL    TERMCHAR+4, R1              ; Address terminal characteristics
0960'CF     06 A1    B0  003D   384         MOVW    DIB$W_DEVBUFSIZ(R1), TTYPBF_WIDTH ; Set terminal's width
 05 08 A1      07    E1  0043   385         BBC     S^#TT$V_LOWER, DIB$L_DEVDEPEND(R1), 10$ ; Lower case?
   0962'CF     0C    A8  0048   386         BISW    #<1@2>!<1@3>, TTYPBF_TYPE ; Yes, so say so
 05 08 A1      0C    E1  004D   387 10$:    BBC     S^#TT$V_SCOPE, DIB$L_DEVDEPEND(R1), 20$ ; Scope?
   0962'CF     01    A8  0052   388         BISW    #1@0, TTYPBF_TYPE           ; Yes, so say so
 05 08 A1      08    E1  0057   389 20$:    BBC     S^#TT$V_MECHTAB, DIB$L_DEVDEPEND(R1), 30$ ; Tab?
   0962'CF     02    A8  005C   390         BISW    #1@1, TTYPBF_TYPE           ; Yes, so say so
 05 08 A1      04    E1  0061   391 30$:    BBC     S^#TT$V_HOSTSYNC, DIB$L_DEVDEPEND(R1), 40$ ; XON?
   0962'CF     10    A8  0066   392         BISW    #1@4, TTYPBF_TYPE           ; Yes, so say so
 05 08 A1      13    E1  006B   393 40$:    BBC     S^#TT$V_MECHFORM, DIB$L_DEVDEPEND(R1), 50$ ; Form feed?
```

```
         0962'CF    20   A8  0070   394          BISW    #1a5, TTYPBF_TYPE          ; Yes, so say so
       07 08 A1     0F   E1  0075   395 50$:     BBC     S^#TT$V_EIGHTBIT, DIB$L_DEVDEPEND(R1), 60$ ; 8-bit?
     0962'CF   0040 8F   A8  007A   396          BISW    #1a6, TTYPBF_TYPE          ; Yes, so say so
          50   095C'CF   3C  0081   397 60$:     MOVZWL  TTYPBF+1, R0               ; Get initial CONTROL message length
                             0086   398          $QIOW_S -                          ; Send the message
                             0086   399                  CHAN = LINKCHAN, -         ;  on the link channel
                             0086   400                  FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                             0086   401                  IOSB = LINKIOSB, -         ;  use an IOSB
                             0086   402                  P1 = TTYPBF, -             ;  use filled in message
                             0086   403                  P2 = R0                    ;   and its extracted length
                    7E   7C  0086                    CLRQ    -(SP)
                    7E   7C  0088                    CLRQ    -(SP)
                    50   DD  008A                    PUSHL   R0
               095B'CF   DF  008C                    PUSHAL  TTYPBF
                    7E   7C  0090                    CLRQ    -(SP)
               000A'CF   7F  0092                    PUSHAQ  LINKIOSB
               7E   00'  3C  0096                    MOVZWL  S^#IO$_WRITEVBLK,-(SP)
          7E   0000'CF   3C  0099                    MOVZWL  LINKCHAN,-(SP)
                    00   DD  009E                    PUSHL   #0
      00000000'GF   0C   FB  00A0                    CALLS   #12,G^SYS$QIOW
                             00A7   404          QUIT_ON_ERROR                      ; Quit on any error
                 03 50   E8  00A7                    BLBS    R0, 30002$
                  0284   31  00AA                    BRW     QUIT
                             00AD                    30002$:
          50   000A'CF   3C  00AD   405          MOVZWL  LINKIOSB, R0               ; Get the I/O completion status
                             00B2   406          QUIT_ON_ERROR                      ;  and quit on any error
                 03 50   E8  00B2                    BLBS    R0, 30003$
                  0279   31  00B5                    BRW     QUIT
                             00B8                    30003$:
                    11   11  00B8   407          BRB     XXXRT                      ; Continue in common code
```

```
                                 00BA  .SBTTL  Initialize TOPS-20 Remote Terminal Protocol
                                 00BA
                             00000006  .PSECT  PROTOTBL, BYTE, NOEXE
                                 0006
                           0008  0006          .WORD   1@3                          ; <3> => TOPS-20 protocol
                      000000BA'  0008          .LONG   TOPS20RT                     ; This is our initialization entry
                                 000C
                             000000BA  .PSECT  RSTSRT, NOWRT
                                 00BA
                                 00BA  TOPS20RT::                                   ; Initialize TOPS-20 protocol
                          0FFC   00BA          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                 00BC
            0000'CF         D4   00BC          CLRL    RETSTATUS                    ; Pre-clear return status
            0000'CF         96   00C0          INCB    TOPS20                       ; Set the TOPS-20 flag
   0321'CF  0000'8F         A8   00C4          BISW    #IO$M_NOECHO!IO$M_NOFILTR, READ_MODE ; TOPS-20 read mode
                                 00CB
                                 00CB  .SBTTL  Common protocol initialization
                                 00CB
                                 00CB  XXXRT:                                       ; Common protocol initialization
                                 00CB
                                 00CB  ; Form the command prompt and help strings
                                 00CB
             50    03       D0   00CB          MOVL    #3, R0                       ; Guess at scope type prompt (3 args)
            0966'CF         7F   00CE          PUSHAQ  REMPROMPT                    ; Last Src is 'REM> '
            0000'CF         7F   00D2          PUSHAQ  NODENAME                     ; Middle Src is the local node name
        51  0004'CF         D0   00D6          MOVL    TERMCHAR+4, R1               ; Address terminal characteristics
  0D 08 A1     0C           E0   00DB          BBS     S^#TT$V_SCOPE, DIB$L_DEVDEPEND(R1), 10$ ; Scope?
                50          D6   00E0          INCL    R0                           ; Nope, change to non-scope prompt
            0590'CF         7F   00E2          PUSHAQ  CRLFPROMPT                   ;  and use an initial a <CR><LF>
   059A'CF  0590'CF         7D   00E6          MOVQ    CRLFPROMPT, ERASEPROMPT      ; Erasing is done with <CR><LF>
            0588'CF         7F   00ED  10$:    PUSHAQ  CMDPROMPT                    ; Dst string is the real prompt
 00000000'GF       50       FB   00F1          CALLS   R0, G^STR$CONCAT             ; Go concatenate the strings
                                 00F8          QUIT_ON_ERROR                        ; Quit on any error
             03 50          E8   00F8                  BLBS    R0, 30004$
              0233          31   00FB                  BRW     QUIT
                                 00FE                  30004$:
            0AC8'CF         7F   00FE          PUSHAQ  RSTSHELP                     ; Last Src is RSTS/E specific help text
   08D4'CF  444F 8F         B0   0102          MOVW    #^A/OD/, CMDODT              ;  and the ODT/PASSALL command is OD
        0C  0000'CF         E9   0109          BLBC    TOPS20, 20$                  ; Is it really RSTS/E?
        6E  0AEB'CF         7E   010E          MOVAQ   TOPS20HELP, (SP)             ; Nope, change to TOPS-20 specific help
   08D4'CF  4150 8F         B0   0113          MOVW    #^A/PA/, CMDODT              ;  and the ODT/PASSALL command is PA
            0973'CF         7F   011A  20$:    PUSHAQ  HELPPROMPT                   ; First Src is general help text
            08CC'CF         7F   011E          PUSHAQ  CMDHELP                      ; Dst string is the real help msg
 00000000'GF       03       FB   0122          CALLS   #3, G^STR$CONCAT             ; Go concatenate the strings
                                 0129          QUIT_ON_ERROR                        ; Quit on any error
             03 50          E8   0129                  BLBS    R0, 30005$
              0202          31   012C                  BRW     QUIT
                                 012F                  30005$:
                                 012F
                                 012F  ; Format initial command string(s)
                                 012F
        53  021E'CF         9E   012F          MOVAB   TERBUFFER, R3                ; Address terminal read buffer
        56  0000'CF         3C   0134          MOVZWL  FINALACS, R6                 ; Was there a final ACS?
                38          13   0139          BEQL    50$                          ; Nope
        51  0B12'CF         9E   013B          MOVAB   HELLO, R1                    ; Yep, point to RSTS/E login command
        05  0000'CF         E9   0140          BLBC    TOPS20, 30$                  ; Really RSTS/E?
        51  0B1A'CF         9E   0145          MOVAB   LOGIN, R1                    ; TOPS-20, point to its login command
```

```
              50    81   9A  014A  30$:   MOVZBL  (R1)+, R0                       ; Get login command start size
         63   61    50   28  014D         MOVC    R0, (R1), (R3)                  ;   and move in start of login command
              57    53   D0  0151         MOVL    R3, R7                          ;    then save pointer just beyond it
    63     0004'DF  56   28  0154         MOVC    R6, @FINALACS+4, (R3)           ; Now move in the final ACS
              83    0D   90  015A         MOVB    #13, (R3)+                      ;   terminated with a <CR>
           11 0000'CF    E8  015D         BLBS    TOPS20, 50$                     ; Check for initial command if TOPS-20
         67   56    20   3A  0162         LOCC    #32, R6, (R7)                   ; Find a <SP> in the ACS
              06    12       0166         BNEQ    40$                             ; Found, go change it
         67   56    09   3A  0168         LOCC    #9, R6, (R7)                    ; No space, try to find a <TAB>
              1A    13       016C         BEQL    60$                             ; None, done
              61    3B   90  016E  40$:   MOVB    #^A/;/, (R1)                    ; Change the <SP>/<TAB> to semi-colon
              15    11       0171         BRB     60$                             ; Now done
                             0173
           0000'CF       B5  0173  50$:   TSTW    FIRSTCMD                        ; Is there an initial command line?
              0F    13       0177         BEQL    60$                             ; Nope
    63     0004'DF  0000'CF  28  0179     MOVC    FIRSTCMD, @FIRSTCMD+4, (R3) ; Yep, move it in
              83    0D   90  0181         MOVB    #13, (R3)+                      ;   terminated with a <CR>
           0000'CF       B4  0184         CLRW    FIRSTCMD                        ; Say initial command used all up
    53    0000021E'8F    C2  0188  60$:   SUBL    #TERBUFFER, R3                  ; Calculate the size of the string
              11    1A       018F         BGTRU   70$                             ; It exists, we'll send it soon
                             0191         $PUTMSG_S -                             ; Announce the connection
                             0191                 MSGVEC = CONNECT                ;   with the final path filled in
              00    DD       0191                 PUSHL   #0
              00    DD       0193                 PUSHL   #0
              00    DD       0195                 PUSHL   #0
           0B22'CF  DF       0197                 PUSHAL  CONNECT
      00000000'GF   04   FB  019B                 CALLS   #4,G^SYS$PUTMSG
                             01A2
                             01A2  ; Remember terminal's EIGHTBIT setting and set terminal to
                             01A2  ;          FULLDUPLEX, NOWRAP, NOESCAPE, and NOPASSALL
                             01A2
         51   0004'CF    D0  01A2  70$:   MOVL    TERMCHAR+4, R1                  ; Point to terminal characteristics
      04 08 A1      0F   E1  01A7         BBC     S^#TT$V_EIGHTBIT, DIB$L_DEVDEPEND(R1), 75$ ; 8-bit terminal?
           0001'CF       96  01AC         INCB    EIGHT_BIT                       ; Yes, so remember as 8-bit terminal
         7E   04 A1      7D  01B0  75$:   MOVQ    DIB$B_DEVCLASS(R1), -(SP) ; Copy the information to set
              51    6E   7E  01B4         MOVAQ   (SP), R1                        ;   and get a pointer to it
    04 A1   00100209 8F  CA  01B7         BICL    #TT$M_HALFDUP -                 ; Clear HALFDUPLEX,
                             01BF                 !TT$M_WRAP -                    ;        WRAP,
                             01BF                 !TT$M_ESCAPE -                  ;        ESCAPE,
                             01BF                 !TT$M_PASSALL, -                ;   and PASSALL modes
                             01BF                 DIB$L_DEVDEPEND-DIB$B_DEVCLASS(R1) ;  in the saved information
                             01BF         $QIOW_S -                               ; Force terminal characteristics
                             01BF                 CHAN = CNTRLCHAN, -             ;  using the control channel
                             01BF                 FUNC = S^#IO$_SETMODE, - ;   with mode setting function
                             01BF                 IOSB = TERIOSB, -               ;  use an IOSB
                             01BF                 P1 = (R1), -                    ;  new characteristics are here
                             01BF                 P2 = #8                         ;  new characteristics are this long
              7E    7C       01BF                 CLRQ    -(SP)
              7E    7C       01C1                 CLRQ    -(SP)
              08    DD       01C3                 PUSHL   #8
              61    DF       01C5                 PUSHAL  (R1)
              7E    7C       01C7                 CLRQ    -(SP)
           0212'CF  7F       01C9                 PUSHAQ  TERIOSB
              7E    00'  3C  01CD                 MOVZWL  S^#IO$_SETMODE,-(SP)
         7E   0000'CF    3C  01D0                 MOVZWL  CNTRLCHAN,-(SP)
              00    DD       01D5                 PUSHL   #0
      00000000'GF   0C   FB  01D7                 CALLS   #12,G^SYS$QIOW
```

```
        5E   08    C0 01DE  502         ADDL    #8, SP                  ; Clear up the stack
                      01E1  503         QUIT_ON_ERROR                   ; Quit on any error
           03 50   E8 01E1                      BLBS    R0, 30006$
            014A   31 01E4                      BRW     QUIT
                      01E7                      30006$:
        50 0212'CF 3C 01E7  504         MOVZWL  TERIOSB, R0             ; Get the I/O completion status
                      01EC  505         QUIT_ON_ERROR                   ;   and quit on any error
           03 50   E8 01EC                      BLBS    R0, 30007$
            013F   31 01EF                      BRW     QUIT
                      01F2                      30007$:
                      01F2  506
                      01F2  507 ; Enable CTRL/C AST's on the terminal (CTRL/Y has been taken away)
                      01F2  508
            01FF   30 01F2  509         BSBW    ENABLE_CTRLC            ; Enable CTRL/C AST's on terminal
                      01F5  510
                      01F5  511 ; Ensure a restored carriage
                      01F5  512
                      01F5  513         $QIOW_S -                       ; Output <CR><LF> to restore carriage
                      01F5  514                 CHAN = WRITECHAN, -     ;  using the write channel
                      01F5  515                 FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                      01F5  516                 P1 = ECHOBUFFER+2, -    ;  buffer containing a <CR><LF>
                      01F5  517                 P2 = #2                 ;   which is 2 characters
              7E   7C 01F5                      CLRQ    -(SP)
              7E   7C 01F7                      CLRQ    -(SP)
              02   DD 01F9                      PUSHL   #2
        031F'CF    DF 01FB                      PUSHAL  ECHOBUFFER+2
              7E   7C 01FF                      CLRQ    -(SP)
              00   DD 0201                      PUSHL   #0
        7E   00'   3C 0203                      MOVZWL  S^#IO$_WRITEVBLK,-(SP)
      7E   0000'CF 3C 0206                      MOVZWL  WRITECHAN,-(SP)
              00   DD 020B                      PUSHL   #0
 00000000'GF   0C  FB 020D                      CALLS   #12,G^SYS$QIOW
                      0214  518
                      0214  519 ; Set up for indirect file and/or initial command
                      0214  520
        0000'CF    95 0214  521         TSTB    INDFLAG                 ; Indirect file active?
              13   13 0218  522         BEQL    80$                     ; Nope
      0000'CF 01   90 021A  523         MOVB    #1, INDFLAG             ; Yep, ensure flag set to +1
 0024'CF 021E'CF   9E 021F  524         MOVAB   TERBUFFER, SYSINRAB+RAB$L_UBF ; Set indirect file data address
 0020'CF 00FE 8F   B0 0226  525         MOVW    #TERBUFLEN-1, SYSINRAB+RAB$W_USZ ;  and maximum record length
        0000'CF    B5 022D  526 80$:    TSTW    FIRSTCMD                ; Pending initial command line?
              07   13 0231  527         BEQL    90$                     ; Nope
 0000'CF 0000'CF   92 0233  528         MCOMB   INDFLAG, INDFLAG        ; Yep, set flag to -1 (or -2)
                      023A  529
                      023A  530 ; Send the command string(s) or queue up a terminal read
                      023A  531
     0214'CF   53  B0 023A  532 90$:    MOVW    R3, TERIOSB+2           ; Set size of the command string(s)
              1A   15 023F  533         BLEQ    100$                    ; No size, so no command(s) to do
     0212'CF   00' B0 0241  534         MOVW    S^#SS$_NORMAL, TERIOSB  ; Some size, set status to success
        0216'CF    D4 0246  535         CLRL    TERIOSB+4               ;   and no terminator
                      024A  536         $DCLAST_S -                     ; Fake an AST
                      024A  537                 ASTADR = TERREADAST, -  ;  for terminal read done
                      024A  538                 ASTPRM = #1             ;  saying wait for output to continue
              00   DD 024A                      PUSHL   #0
              01   DD 024C                      PUSHL   #1
        06C1'CF    DF 024E                      PUSHAL  TERREADAST
 00000000'GF   03  FB 0252                      CALLS   #3,G^SYS$DCLAST
```

```
  03   11  0259   539          BRB     110$                          ; Continue
           025B   540
05FD   30  025B   541 100$:    BSBW    TERREAD                       ; Queue up a terminal read
           025E   542
           025E   543 ; Queue up reads on the link mailbox, link, and terminal mailbox
           025E   544
0051   30  025E   545 110$:    BSBW    LINKMBXREAD                   ; Queue up a link mailbox read
07EA   30  0261   546          BSBW    LINKREAD                      ; Queue up a link read
0020   30  0264   547          BSBW    TERMMBXREAD                   ; Queue up a terminal mailbox read
       04  0267   548          RET                                   ; Return.
```

```
                          0268   550 .SBTTL  Terminal mailbox AST's
                          0268   551
                          0268   552 TERMMBXAST:                                  ; Terminal mailbox AST's
                    0FFC  0268   553         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                          026A   554
        50    090C'CF  3C 026A   555         MOVZWL  TERMMBXIOSB, R0              ; Get the I/O completion status
                          026F   556         QUIT_ON_ERROR                        ;  and quit on any error
              03 50    E8 026F                       BLBS    R0, 30008$
               00BC    31 0272                       BRW     QUIT
                          0275                       30008$:
   0000'8F    0914'CF  B1 0275   557         CMPW    TERMMBXBUFFER, #MSG$_TRMHANGUP ; Is it a terminal hangup?
                 03    13 027C   558         BEQL    10$                          ; Yep, exit right now...
                 07    10 027E   559         BSBB    TERMMBXREAD                  ; Else just queue up another read
                       04 0280   560         RET                                  ; Return.
                          0281   561
                          0281   562 10$:    QUIT    S^#SS$_NORMAL                ; Quit without any messages
           50     00'  D0 0281                       MOVL    S^#SS$_NORMAL, R0
               00AA    31 0284                       BRW     QUIT
                          0287   563
                          0287   564 TERMMBXREAD:                                 ; Queue up a terminal mailbox read
                          0287   565         $QIO_S -                             ; Queue up a request
                          0287   566                 CHAN = TERMMBXCHAN, -        ;  on the terminal mailbox channel
                          0287   567                 FUNC = S^#IO$_READVBLK, -    ;  reading obviously
                          0287   568                 IOSB = TERMMBXIOSB, -        ;  use an IOSB
                          0287   569                 ASTADR = TERMMBXAST, -       ;  come here upon completion
                          0287   570                 P1 = TERMMBXBUFFER, -        ;  read message into this buffer
                          0287   571                 P2 = #TERMMBXBUFLEN          ;   which is this long
                 7E    7C 0287                       CLRQ    -(SP)
                 7E    7C 0289                       CLRQ    -(SP)
                 28    DD 028B                       PUSHL   #TERMMBXBUFLEN
            0914'CF    DF 028D                       PUSHAL  TERMMBXBUFFER
                 00    DD 0291                       PUSHL   #0
              D2 AF    DF 0293                       PUSHAL  TERMMBXAST
            090C'CF    7F 0296                       PUSHAQ  TERMMBXIOSB
           7E     00'  3C 029A                       MOVZWL  S^#IO$_READVBLK,-(SP)
        7E    0000'CF  3C 029D                       MOVZWL  TERMMBXCHAN,-(SP)
                 00    DD 02A2                       PUSHL   #0
   00000000'GF   0C    FB 02A4                       CALLS   #12,G^SYS$QIO
                          02AB   572         QUIT_ON_ERROR                        ; Quit on any error
              03 50    E8 02AB                       BLBS    R0, 30009$
               0080    31 02AE                       BRW     QUIT
                          02B1                       30009$:
                       05 02B1   573         RSB                                  ; Exit.
```

```
                        02B2      .SBTTL  Link mailbox AST's
                        02B2
                        02B2      LINKMBXREAD:                                          ; Queue up a link mailbox read
                        02B2              $QIO_S -                                      ; Queue up a request
                        02B2                              CHAN = MAILCHAN, -            ;  on the link mailbox channel
                        02B2                              FUNC = S^#IO$_READVBLK, - ;  reading obviously
                        02B2                              IOSB = LINKMBXIOSB, -         ;  use an IOSB
                        02B2                              ASTADR = LINKMBXAST, -        ;  come here upon completion
                        02B2                              P1 = LINKMBXBUFFER, -         ;  read message into this buffer
                        02B2                              P2 = #LINKMBXBUFLEN           ;   which is this long
              7E    7C  02B2                              CLRQ    -(SP)
              7E    7C  02B4                              CLRQ    -(SP)
              28    DD  02B6                              PUSHL   #LINKMBXBUFLEN
         08E4'CF    DF  02B8                              PUSHAL  LINKMBXBUFFER
              00    DD  02BC                              PUSHL   #0
         02DE'CF    DF  02BE                              PUSHAL  LINKMBXAST
         08DC'CF    7F  02C2                              PUSHAQ  LINKMBXIOSB
          7E   00'  3C  02C6                              MOVZWL  S^#IO$_READVBLK,-(SP)
     7E  0000'CF    3C  02C9                              MOVZWL  MAILCHAN,-(SP)
              00    DD  02CE                              PUSHL   #0
 00000000'GF    0C  FB  02D0                              CALLS   #12,G^SYS$QIO
                        02D7              QUIT_ON_ERROR                                 ; Quit on any error
           03 50    E8  02D7                              BLBS    R0, 30010$
            0054    31  02DA                              BRW     QUIT
                        02DD                              30010$:
              05        02DD              RSB                                           ; Exit.
                        02DE
                        02DE      LINKMBXAST:                                           ; Link mailbox AST's
            0FFC        02DE              .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                        02E0
     50  08DC'CF    3C  02E0              MOVZWL  LINKMBXIOSB, R0                       ; Get the I/O completion status
                        02E5              QUIT_ON_ERROR                                 ;  and quit on any error
           03 50    E8  02E5                              BLBS    R0, 30011$
            0046    31  02E8                              BRW     QUIT
                        02EB                              30011$:
 0000'8F  08E4'CF   B1  02EB              CMPW    LINKMBXBUFFER, #MSG$_CONFIRM ; Simply the confirmation?
              03    12  02F2              BNEQ    10$                                   ; Nope
              BC    10  02F4              BSBB    LINKMBXREAD                           ; Yep, just queue up another read
              04        02F6              RET                                           ; Return.
                        02F7
 0000'8F  08E4'CF   B1  02F7      10$:    CMPW    LINKMBXBUFFER, #MSG$_DISCON ; Is it a valid disconnect?
              26    13  02FE              BEQL    30$                                   ; Yep, exit right now...
 0000'8F  08E4'CF   B1  0300              CMPW    LINKMBXBUFFER, #MSG$_ABORT ; Is it a link abort?
              05    12  0307              BNEQ    20$                                   ; Nope, other, go announce it
       18 0000'CF   E8  0309              BLBS    TOPS20, 30$                           ; Yep, skip announcement if TOPS-20
 0954'CF  08E4'CF   3C  030E      20$:    MOVZWL  LINKMBXBUFFER, NETMBXVEC+12 ; Store the mailbox code
                        0315              $PUTMSG_S -                                   ; Put out a message
                        0315                              MSGVEC = NETMBXVEC            ;  with the mailbox error code
              00    DD  0315                              PUSHL   #0
              00    DD  0317                              PUSHL   #0
              00    DD  0319                              PUSHL   #0
         0948'CF    DF  031B                              PUSHAL  NETMBXVEC
 00000000'GF    04  FB  031F                              CALLS   #4,G^SYS$PUTMSG
                        0326      30$:    QUIT    S^#SS$_NORMAL                         ; Quit without any messages
          50   00'  D0  0326                              MOVL    S^#SS$_NORMAL, R0
            0005    31  0329                              BRW     QUIT
                        032C
```

```
                              032C  .SBTTL  Image termination
                              032C
                              032C  .ENABLE LSB
                              032C
                              032C  ABORT_QUIT:                                           ; Possible image termination
              00'   50   D1   032C          CMPL    R0, S^#SS$_ABORT                      ; Is the link simply aborting?
                    62   13   032F          BEQL    20$                                   ; Yes, we'll wait for mailbox message
                              0331  QUIT:                                                 ; Image termination
            0000'CF      D5   0331          TSTL    RETSTATUS                             ; Do we already have a return status?
                    05   12   0335          BNEQ    10$                                   ; Yep, don't overwrite it with another
       0000'CF      50   D0   0337          MOVL    R0, RETSTATUS                         ; Set the final completion status
              073F       30   033C  10$:    BSBW    END_LOG                               ; Go close out the logging file if any
                              033F          $CANCEL_S -                                   ; Cancel any current terminal read
                              033F                  CHAN = READCHAN                       ;  on the read channel
      7E    0000'CF      3C   033F                  MOVZWL  READCHAN,-(SP)
 00000000'GF        01   FB   0344                  CALLS   #1,G^SYS$CANCEL
                              034B          $CANCEL_S -                                   ; Cancel operation(s)
                              034B                  CHAN = CNTRLCHAN                      ;  so we can reset the terminal
      7E    0000'CF      3C   034B                  MOVZWL  CNTRLCHAN,-(SP)
 00000000'GF        01   FB   0350                  CALLS   #1,G^SYS$CANCEL
      51    0004'CF      D0   0357          MOVL    TERMCHAR+4, R1                        ; Point to terminal characteristics
         7E   04    A1   7D   035C          MOVQ    DIB$B_DEVCLASS(R1), -(SP)             ; Copy the information to set
              51    6E   7E   0360          MOVAQ   (SP), R1                              ;  and get a pointer to it
                              0363          $QIOW_S -                                     ; Force terminal characteristics
                              0363                  CHAN = CNTRLCHAN, -                   ;  using the control channel
                              0363                  FUNC = S^#IO$_SETMODE, -              ;  with mode setting function
                              0363                  P1 = (R1), -                          ;  new characteristics are here
                              0363                  P2 = #8                               ;  new characteristics are this long
                    7E   7C   0363                  CLRQ    -(SP)
                    7E   7C   0365                  CLRQ    -(SP)
                    08   DD   0367                  PUSHL   #8
                    61   DF   0369                  PUSHAL  (R1)
                    7E   7C   036B                  CLRQ    -(SP)
                    00   DD   036D                  PUSHL   #0
              7E    00'  3C   036F                  MOVZWL  S^#IO$_SETMODE,-(SP)
      7E    0000'CF      3C   0372                  MOVZWL  CNTRLCHAN,-(SP)
                    00   DD   0377                  PUSHL   #0
 00000000'GF        0C   FB   0379                  CALLS   #12,G^SYS$QIOW
              5E    08   C0   0380          ADDL    #8, SP                                ; Clear up the stack
       0000'CF      01   90   0383          MOVB    #1, WAKEFLAG                          ; Indicate a legitimate $WAKE
                              0388          $WAKE_S                                       ; Wake up the mainline program
                    00   DD   0388                  PUSHL   #0
                    00   DD   038A                  PUSHL   #0
 00000000'GF        02   FB   038C                  CALLS   #2,G^SYS$WAKE
                    04        0393  20$:    RET                                           ; Return from whatever level.
                              0394
                              0394  .DISABLE LSB
```

```
                        0394     .SBTTL  CTRL/C AST's
                        0394
                        0394     CTRLCAST:                                       ; CTRL/C AST's
                  0FFC  0394             .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                        0396
             5C     10  0396             BSBB    ENABLE_CTRLC                    ; Re-enable CTRL/C's
        0002'CF     D6  0398             INCL    CTRLC_CNT                       ; CTRL/C message already in-progress?
             55     12  039C             BNEQ    40$                             ; Yep
   51   0453'CF     9E  039E     10$:    MOVAB   CTRLC_MSG, R1                   ; Address the CTRL/C message
      50   01 A1    3C  03A3             MOVZWL  1(R1), R0                       ; Get the CTRL/C message length
     06 0000'CF     E9  03A7             BLBC    TOPS20, 20$                     ; All set if RSTS/E
         51   04    C0  03AC             ADDL    #4, R1                          ; It's TOPS-20, skip the header
         50   04    C2  03AF             SUBL    #4, R0                          ;  and don't count it
                        03B2     20$:    $QIO_S -                                ; Queue up a CTRL/C link send
                        03B2                     CHAN = LINKCHAN, -              ;  using the link channel
                        03B2                     FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                        03B2                     IOSB = CTRLC_IOSB, -            ;  use an IOSB
                        03B2                     ASTADR = 30$, -                 ;  come here upon completion
                        03B2                     P1 = (R1), -                    ;  this is the message
                        03B2                     P2 = R0                         ;   with this size
             7E     7C  03B2                     CLRQ    -(SP)
             7E     7C  03B4                     CLRQ    -(SP)
             50     DD  03B6                     PUSHL   R0
             61     DF  03B8                     PUSHAL  (R1)
             00     DD  03BA                     PUSHL   #0
        03E1'CF     DF  03BC                     PUSHAL  30$
        044B'CF     7F  03C0                     PUSHAQ  CTRLC_IOSB
          7E   00'  3C  03C4                     MOVZWL  S^#IO$_WRITEVBLK,-(SP)
   7E   0000'CF     3C  03C7                     MOVZWL  LINKCHAN,-(SP)
             00     DD  03CC                     PUSHL   #0
    00000000'GF     0C  FB  03CE                 CALLS   #12,G^SYS$QIO
                        03D5             QUIT_NOT_ABORT                          ; Quit on non-link-abort error
          03 50     E8  03D5                     BLBS    R0, 30012$
           FF51     31  03D8                     BRW     ABORT_QUIT
                        03DB                     30012$:
   0006'CF    01    90  03DB             MOVB    #1, CTRLC_SENT                  ; Say a CTRL/C message was sent
                    04  03E0             RET                                     ; Return pending CTRL/C completion.
                        03E1
                        03E1     30$:                                            ; CTRL/C message sent AST
                  0FFC  03E1             .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
   50   044B'CF     3C  03E3             MOVZWL  CTRLC_IOSB, R0                  ; Get the I/O completion status
                        03E8             QUIT_NOT_ABORT                          ; Quit on non-link-abort error
          03 50     E8  03E8                     BLBS    R0, 30013$
           FF3E     31  03EB                     BRW     ABORT_QUIT
                        03EE                     30013$:
     AB 0002'CF     F4  03EE             SOBGEQ  CTRLC_CNT, 10$                  ; Do it again if more CTRL/C's pending
                    04  03F3     40$:    RET                                     ; Return.
                        03F4
                        03F4     .SBTTL  CTRL/C AST enable
                        03F4
                        03F4     ENABLE_CTRLC:                                   ; CTRL/C AST enable
                        03F4             $QIOW_S -                               ; Enable CTRL/C AST's
                        03F4                     CHAN = CNTRLCHAN, -             ;  using the control channel
                        03F4                     FUNC = #IO$_SETMODE!IO$M_CTRLCAST, - ;  function for AST
                        03F4                     P1 = CTRLCAST                   ;  come here on CTRL/C AST's
             7E     7C  03F4                     CLRQ    -(SP)
             7E     7C  03F6                     CLRQ    -(SP)
```

```
                  00  DD  03F8                  PUSHL   #0
               97 AF  DF  03FA                  PUSHAL  CTRLCAST
                  7E  7C  03FD                  CLRQ    -(SP)
                  00  DD  03FF                  PUSHL   #0
      7E   0000'8F    3C  0401                  MOVZWL  #IO$_SETMODE!IO$M_CTRLCAST,-(SP)
      7E   0000'CF    3C  0406                  MOVZWL  CNTR[CHAN,-(SP)
                  00  DD  040B                  PUSHL   #0
   00000000'GF    0C  FB  040D                  CALLS   #12,G^SYS$QIOW
                          0414  677     QUIT_ON_ERROR                   ; Quit on any error
               03 50  E8  0414                  BLBS    R0, 30014$
                FF17  31  0417                  BRW     QUIT
                          041A                  30014$:
                      05  041A  678     RSB                             ; Exit.
```

```
                              041B  .SBTTL  Terminal read AST's
                              041B
                              041B  .ENABLE LSB
                              041B
                  50     D5   041B  10$:    TSTL    R0                      ; Anything to delete?
                  27     13   041D          BEQL    20$                     ; Nope
          045A'CF 51     A2   041F          SUBW    R1, CMDIOSB+2           ; Yep, delete from the buffer
                  51     03   C4    0424    MULL    #3, R1                  ; Form the erase count
                              0427          $QIOW_S -                       ; Write the screen deletion
                              0427                  EFN = #1, -             ;  we really want to wait
                              0427                  CHAN = READCHAN, -      ;  use the read channel
                              0427                  FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                              0427                  P1 = @ERASEPROMPT+4, -  ;  the erase sequence
                              0427                  P2 = R1                 ;  for this many characters
                  7E     7C   0427          CLRQ    -(SP)
                  7E     7C   0429          CLRQ    -(SP)
                  51     DD   042B          PUSHL   R1
          059E'DF        DF   042D          PUSHAL  @ERASEPROMPT+4
                  7E     7C   0431          CLRQ    -(SP)
                  00     DD   0433          PUSHL   #0
             7E   00'    3C   0435          MOVZWL  S^#IO$_WRITEVBLK,-(SP)
        7E   0000'CF     3C   0438          MOVZWL  READCHAN,-(SP)
                  01     DD   043D          PUSHL   #1
      00000000'GF 0C     FB   043F          CALLS   #12,G^SYS$QIOW
        50   045A'CF     3C   0446  20$:    MOVZWL  CMDIOSB+2, R0           ; Get the updated data size
             7E   50     B0   044B          MOVW    R0, -(SP)               ;  and save it
51    000000FF 8F 50     C3   044E          SUBL3   R0, #CMDBUFLEN-1, R1    ; Form left over buffer size
50    00000460'8F        C0   0456          ADDL    #CMDBUFFER, R0          ;  and starting buffer address
                              045D          $QIOW_S -                       ; Re-ask for a command to decode
                              045D                  EFN = #1, -             ;  we really want to wait
                              045D                  CHAN = READCHAN, -      ;  use the read channel
                              045D                  FUNC = <#IO$_READVBLK-  ;  read function
                              045D                          !IO$M_NOFILTR!IO$M_TRMNOECHO>, - ;  w/o filtering
                              045D                  IOSB = CMDIOSB, -       ;  use an IOSB
                              045D                  P1 = (R0), -            ;  into this part of the buffer
                              045D                  P2 = R1, -              ;  with this remaining size
                              045D                  P4 = #CMDMASK           ;  use the command terminator mask
                  7E     7C   045D          CLRQ    -(SP)
      00000560'8F        DD   045F          PUSHL   #CMDMASK
                  00     DD   0465          PUSHL   #0
                  51     DD   0467          PUSHL   R1
                  60     DF   0469          PUSHAL  (R0)
                  7E     7C   046B          CLRQ    -(SP)
          0458'CF        7F   046D          PUSHAQ  CMDIOSB
        7E   0000'8F     3C   0471          MOVZWL  #IO$_READVBLK                          !IO$M_NOFILTR!IO$M_TRMNOECHO
        7E   0000'CF     3C   0476          MOVZWL  READCHAN,-(SP)
                  01     DD   047B          PUSHL   #1
      00000000'GF 0C     FB   047D          CALLS   #12,G^SYS$QIOW
          045A'CF 8E     A0   0484          ADDW    (SP)+, CMDIOSB+2        ; Correct the data read size
                  45     11   0489          BRB     40$                     ;  and go try, try again...
                              048B
                              048B  CTRLP:                                  ; CTRL/P intercept, command decode
        50   0588'CF     3C   048B          MOVZWL  CMDPROMPT, R0           ; Get length of the command prompt
        51   0000'8F     3C   0490          MOVZWL  #IO$_READPROMPT!IO$M_TRMNOECHO, R1 ; Set read function
        02   059A'CF     B1   0495          CMPW    ERASEPROMPT, #2         ; Doing fancy scope things?
                  05     1B   049A          BLEQU   30$                     ; Nope
        51   0000'8F     A8   049C          BISW    #IO$M_NOFILTR, R1       ; Yep, handle <DEL> ourselves
```

```
                     04A1   716 30$:    $QIOW_S -                             ; Ask for a command to decode
                     04A1   717                 EFN = #1, -                   ;  we really want to wait
                     04A1   718                 CHAN = READCHAN, -            ;  use the read channel
                     04A1   719                 FUNC = R1, -                  ;  using the correct function
                     04A1   720                 IOSB = CMDIOSB, -             ;  use an IOSB
                     04A1   721                 P1 = CMDBUFFER, -             ;  read into command buffer
                     04A1   722                 P2 = #CMDBUFLEN-1, -          ;   which is this long less terminator
                     04A1   723                 P4 = #CMDMASK, -              ;  use the command terminator mask
                     04A1   724                 P5 = CMDPROMPT+4, -           ;  pointer to command prompt
                     04A1   725                 P6 = R0                       ;   and its size
             50  DD  04A1               PUSHL   R0
        058C'CF  DD  04A3               PUSHL   CMDPROMPT+4
    00000560'8F  DD  04A7               PUSHL   #CMDMASK
             00  DD  04AD               PUSHL   #0
    000000FF 8F  DD  04AF               PUSHL   #CMDBUFLEN-1
        0460'CF  DF  04B5               PUSHAL  CMDBUFFER
             7E  7C  04B9               CLRQ    -(SP)
        0458'CF  7F  04BB               PUSHAQ  CMDIOSB
          7E 51  3C  04BF               MOVZWL  R1,-(SP)
     7E 0000'CF  3C  04C2               MOVZWL  READCHAN,-(SP)
             01  DD  04C7               PUSHL   #1
 00000000'GF 0C  FB  04C9               CALLS   #12,G^SYS$QIOW
          77 50  E9  04D0   726 40$:    BLBC    R0, 90$                       ; Quit on any error
    00' 0458'CF  B1  04D3   727         CMPW    CMDIOSB, S^#SS$_NORMAL        ; Normal completion?
             70  12  04D8   728         BNEQ    90$                           ; Nope, quit
     50 045A'CF  3C  04DA   729         MOVZWL  CMDIOSB+2, R0                 ; Get size of data read
  7F 8F 045C'CF  91  04DF   730         CMPB    CMDIOSB+4, #127               ; Was terminator <DEL>?
             06  12  04E5   731         BNEQ    60$                           ; Nope
          51 01  D0  04E7   732         MOVL    #1, R1                        ; Yep, set deletion size to one
           FF2E  31  04EA   733 50$:    BRW     10$                           ; Go handle scope <DEL> and CTRL/U...
                     04ED   734
     15 045C'CF  91  04ED   735 60$:    CMPB    CMDIOSB+4, #^A/U/-64          ; Was terminator CTRL/U?
             0A  12  04F2   736         BNEQ    70$                           ; Nope
          51 50  D0  04F4   737         MOVL    R0, R1                        ; Yep, is the CTRL/U the only thing?
             F1  12  04F7   738         BNEQ    50$                           ; Not the only, go do deletions
     0460'CF 18  90  04F9   739         MOVB    #^A/X/-64, CMDBUFFER          ; The only, replace it with a CTRL/X
     51 059A'CF  B0  04FE   740 70$:    MOVW    ERASEPROMPT, R1               ; Get string size that erases prompt
          02 51  B1  0503   741         CMPW    R1, #2                        ; Is it the fancy scope erase?
             09  1B  0506   742         BLEQU   80$                           ; Nope, use it as is
  51 0588'CF 50  A1  0508   743         ADDW3   R0, CMDPROMPT, R1             ; Yep, calculate erase size
          51 03  A4  050E   744         MULW    #3, R1                        ; That *3 is the real erase string
          51 51  3C  0511   745 80$:    MOVZWL  R1, R1                        ; Extract size into a longword
                     0514   746         $QIOW_S -                             ; Write out prompt erase
                     0514   747                 EFN = #1, -                   ;  we really want to wait
                     0514   748                 CHAN = READCHAN, -            ;  using the read channel
                     0514   749                 FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                     0514   750                 P1 = @ERASEPROMPT+4, -        ;  pointer to prompt erase
                     0514   751                 P2 = R1                       ;  and its size
             7E  7C  0514               CLRQ    -(SP)
             7E  7C  0516               CLRQ    -(SP)
             51  DD  0518               PUSHL   R1
        059E'DF  DF  051A               PUSHAL  @ERASEPROMPT+4
             7E  7C  051E               CLRQ    -(SP)
             00  DD  0520               PUSHL   #0
         7E 00'  3C  0522               MOVZWL  S^#IO$_WRITEVBLK,-(SP)
     7E 0000'CF  3C  0525               MOVZWL  READCHAN,-(SP)
             01  DD  052A               PUSHL   #1
```

```
 00000000'GF   0C  FB  052C                CALLS   #12,G^SYS$QIOW
     50  0460'CF   9E  0533   752          MOVAB   CMDBUFFER, R0                   ; Address command buffer
 51  60  2020 8F   AB  0538   753          BICW3   #<32@8>!32, (R0), R1            ; Do a cheap(!) upper case convert
     5845 8F  51   B1  053E   754          CMPW    R1, #^A/EX/                     ; Is it EXIT?
              05   13  0543   755          BEQL    90$                             ; Yes, quit
         1A   60   91  0545   756          CMPB    (R0), #^A/Z/-64                 ; CTRL/Z typed?
              57   12  0548   757          BNEQ    120$                            ; Nope, go decode real commands
                       054A   758 90$:     QUIT    S^#SS$_NORMAL                   ; Quit.
         50   00'  D0  054A                MOVL    S^#SS$_NORMAL, R0
            FDE1   31  054D                BRW     QUIT
                       0550   759
     51  0004'CF   D0  0550   760 100$:    MOVL    TERMCHAR+4, R1                  ; Point to terminal characteristics
     7E   04  A1   7D  0555   761          MOVQ    DIB$B_DEVCLASS(R1), -(SP)       ; Copy the information to set
         51   6E   7E  0559   762          MOVAQ   (SP), R1                        ;  and get a pointer to it
 04 A1  00100209 8F CA 055C   763          BICL    #TT$M_HALFDUP -                 ; Clear HALFDUPLEX,
                       0564   764                  !TT$M_WRAP -                    ;       WRAP,
                       0564   765                  !TT$M_ESCAPE -                  ;       ESCAPE,
                       0564   766                  !TT$M_PASSALL, -                ;   and PASSALL modes
                       0564   767                  DIB$L_DEVDEPEND-DIB$B_DEVCLASS(R1) ;  in the saved information
   04 08DB'CF      E9  0564   768          BLBC    CNTLBF_ECHOFLG, 110$            ; Is it NORMAL mode?
   04 A1      01   C8  0569   769          BISL    #TT$M_PASSALL, -                ; Nope, PASSALL mode, set PASSALL mode
                       056D   770                  DIB$L_DEVDEPEND-DIB$B_DEVCLASS(R1) ;  in the saved information
                       056D   771 110$:    $QIOW_S -                               ; Force terminal characteristics
                       056D   772                  EFN = #1, -                     ;  we really want to wait
                       056D   773                  CHAN = READCHAN, -              ;  using the read channel
                       056D   774                  FUNC = S^#IO$_SETMODE, - ;       with mode setting function
                       056D   775                  IOSB = CMDIOSB, -               ;  use an IOSB
                       056D   776                  P1 = (R1), -                    ;  new characteristics are here
                       056D   777                  P2 = #8                         ;  new characteristics are this long
              7E   7C  056D                CLRQ    -(SP)
              7E   7C  056F                CLRQ    -(SP)
              08   DD  0571                PUSHL   #8
              61   DF  0573                PUSHAL  (R1)
              7E   7C  0575                CLRQ    -(SP)
         0458'CF   7F  0577                PUSHAQ  CMDIOSB
         7E   00'  3C  057B                MOVZWL  S^#IO$_SETMODE,-(SP)
     7E  0000'CF   3C  057E                MOVZWL  READCHAN,-(SP)
              01   DD  0583                PUSHL   #1
 00000000'GF   0C  FB  0585                CALLS   #12,G^SYS$QIOW
         5E   08   C0  058C   778          ADDL    #8, SP                          ; Clear up the stack
                       058F   779          QUIT_ON_ERROR                           ; Quit on any error
           03 50   E8  058F                        BLBS    R0, 30015$
            FD9C   31  0592                        BRW     QUIT
                       0595                        30015$:
     50  0458'CF   3C  0595   780          MOVZWL  CMDIOSB, R0                     ; Get the I/O completion status
                       059A   781          QUIT_ON_ERROR                           ;  and quit on any error
           03 50   E8  059A                        BLBS    R0, 30016$
            FD91   31  059D                        BRW     QUIT
                       05A0                        30016$:
              05   05A0       782          RSB                                     ; Exit.
                       05A1   783
   08DB'CF    01   90  05A1   784 120$:    MOVB    #1@0, CNTLBF_ECHOFLG            ; Guess at ODT/PASSALL mode
   08D4'CF    51   B1  05A6   785          CMPW    R1, CMDODT                      ; Was that a good guess?
              0C   13  05AB   786          BEQL    130$                            ; Yes, it's ODT/PASSALL mode
   08DB'CF    02   90  05AD   787          MOVB    #1@1, CNTLBF_ECHOFLG            ; Now guess at NORMAL mode
   4F4E 8F    51   B1  05B2   788          CMPW    R1, #^A/NO/                     ; A good guess this time?
              49   12  05B7   789          BNEQ    180$                            ; No, go decode other commands
```

```
      92 0000'CF    E8  05B9  130$:   BLBS    TOPS20, 100$                    ; Do differently for TOPS-20
0321'CF    0000'8F  AA  05BE          BICW    #IO$M_NOECHO!IO$M_NOFILTR, READ_MODE ; Set NORMAL read mode
      07 08DB'CF    E9  05C5          BLBC    CNTLBF_ECHOFLG, 140$            ; A good guess?
0321'CF    0000'8F  A8  05CA          BISW    #IO$M_NOECHO!IO$M_NOFILTR, READ_MODE ; Set ODT read mode
   50    08D7'CF    3C  05D1  140$:   MOVZWL  CNTLBF+1, R0                    ; Else get the message's length
                        05D6          $QIO_S -                                ; Send the CONTROL message
                        05D6                  CHAN = LINKCHAN, -              ;  using the link channel
                        05D6                  FUNC = S^#IO$_WRITEVBLK, -      ;   writing obviously
                        05D6                  P1 = CNTLBF, -                  ;  buffer is the CONTROL message
                        05D6                  P2 = R0                         ;    which is this long
                7E  7C  05D6                  CLRQ    -(SP)
                7E  7C  05D8                  CLRQ    -(SP)
                50  DD  05DA                  PUSHL   R0
            08D6'CF DF  05DC                  PUSHAL  CNTLBF
                7E  7C  05E0                  CLRQ    -(SP)
                00  DD  05E2                  PUSHL   #0
          7E   00'  3C  05E4                  MOVZWL  S^#IO$_WRITEVBLK,-(SP)
   7E    0000'CF    3C  05E7                  MOVZWL  LINKCHAN,-(SP)
                00  DD  05EC                  PUSHL   #0
00000000'GF     0C  FB  05EE                  CALLS   #12,G^SYS$QIO
                        05F5          QUIT_NOT_ABORT                          ; Quit on non-link-abort error
           03 50    E8  05F5                  BLBS    R0, 30017$
            FD31    31  05F8                  BRW     ABORT_QUIT
                        05FB                  30017$:
                    05  05FB  150$:   RSB                                     ; Exit.
                        05FC
            04ED    31  05FC  160$:   BRW     START_LOG                       ; Go off to start logging
                        05FF
            047C    31  05FF  170$:   BRW     END_LOG                         ; Go off to end logging
                        0602
   4F43 8F      51  B1  0602  180$:   CMPW    R1, #^A/CO/                     ; Is it CONTINUE?
                F2  13  0607          BEQL    150$                            ; Yes, so just continue
        0D      60  91  0609          CMPB    (R0), #13                       ; <CR> typed?
                ED  13  060C          BEQL    150$                            ; Yes, so just continue
   4F4C 8F      51  B1  060E          CMPW    R1, #^A/LO/                     ; Is it LOG?
                E7  13  0613          BEQL    160$                            ; Yes, so go off to start logging
   4C43 8F      51  B1  0615          CMPW    R1, #^A/CL/                     ; Is it CLOSE?
                E3  13  061A          BEQL    170$                            ; Yes, so go off to end logging
   5443 8F      51  B1  061C          CMPW    R1, #^A/CT/                     ; Is it CTRL/P?
                3A  13  0621          BEQL    190$                            ; Yep, go send a CTRL/P
        51      60  9A  0623          MOVZBL  (R0), R1                        ; Pick up the first character
        20      60  91  0626          CMPB    (R0), #32                       ; Is it a control character?
                35  1F  0629          BLSSU   200$                            ; Yes, send the control character
     51    01   A0  9A  062B          MOVZBL  1(R0), R1                       ; Pick up next character for CTRL/x
     5E 8F      60  91  062F          CMPB    (R0), #^A/^/                    ; Is it CTRL/x?
                2B  13  0633          BEQL    200$                            ; Yep, so send the "x" as a control
   50    08CC'CF    7D  0635          MOVQ    CMDHELP, R0                     ; Get command help prompt descriptor
        50      50  3C  063A          MOVZWL  R0, R0                          ;  and fix the help text's length
                        063D          $QIOW_S -                               ; Write out some help
                        063D                  EFN = #1, -                     ;  we really want to wait
                        063D                  CHAN = READCHAN, -              ;  using the read channel
                        063D                  FUNC = S^#IO$_WRITEVBLK, -      ;   writing obviously
                        063D                  P1 = (R1), -                    ;  pointer to help text
                        063D                  P2 = R0                         ;  and its size
                7E  7C  063D                  CLRQ    -(SP)
                7E  7C  063F                  CLRQ    -(SP)
                50  DD  0641                  PUSHL   R0
```

RST
Sym
RAB
RAB
RAB
RAB
REA
REA
REM
REM
REM
RET
RMS
RMS
RST
RST
SS$
SS$
SS$
SS$
STA
STR
SYS
SYS
SYS
SYS
SYS
SYS
SYS
SYS
SYS
SYS
SYS
SYS
SYS
TER
TER
TER
TER
TER
TER
TER
TER
TER
TER
TER
TER
TOP
TOP
TOP
TT$
TT$
TT$
TT$
TT$
TT$
TT$
TT$

```
                61   DF  0643                     PUSHAL  (R1)
                7E   7C  0645                     CLRQ    -(SP)
                00   DD  0647                     PUSHL   #0
            7E  00'  3C  0649                     MOVZWL  S^#IO$_WRITEVBLK,-(SP)
       7E    0000'CF 3C  064C                     MOVZWL  READCHAN,-(SP)
                01   DD  0651                     PUSHL   #1
     00000000'GF 0C  FB  0653                     CALLS   #12,G^SYS$QIOW
              FE2E   31  065A                BRW      CTRLP                       ; Now go ask again...
                         065D
            51   10  9A  065D  190$:         MOVZBL   #^A/P/-64, R1               ; Pick up a CTRL/P to send
   60    51   E0 8F  8B  0660  200$:         BICB3    #^C<31>, R1, (R0)           ; Put control character in echo buffer
                51   01  D0  0665            MOVL     #1, R1                      ;  and set echo size to one
              6746   60  90  0668            MOVB     (R0), (R7)[R6]              ; Add control character to message
                56   D6  066C                INCL     R6                          ;  and count it
            10   60  91  066E                CMPB     (R0), #^A/P/-64             ; Did we add a CTRL/P?
                15   12  0671                BNEQ     210$                        ; Nope
  0F 0321'CF     00'  E0  0673               BBS      S^#IO$V_NOFILTR, READ_MODE, 210$ ; Are we in NORMAL mode?
              6746   0D  90  0679            MOVB     #13, (R7)[R6]               ; Add a <CR> to message
                56   D6  067D                INCL     R6                          ;  and count it
  02 A0     0A0D 8F  B0  067F                MOVW     #<10@8>!13, 2(R0)           ; Put <CR><LF> in echo buffer
                51   03  D0  0685            MOVL     #3, R1                      ;  and set echo size to three
  32 0321'CF     00'  E0  0688  210$:        BBS      S^#IO$V_NOECHO, READ_MODE, 240$ ; Are we echoing?
                07   60  91  068E            CMPB     (R0), #7                    ; Less than <BEL>?
                05   1F  0691                BLSSU    220$                        ; Yes, make into uparrow format
                0D   60  91  0693            CMPB     (R0), #13                   ; <CR> or less?
                0B   1B  0696                BLEQU    230$                        ; Yes, <BEL> through <CR> echo same
   60    80   40 8F  89  0698  220$:         BISB3    #64, (R0)+, (R0)            ; Make the control a graphic
            70   5E 8F  90  069D             MOVB     #^A/^/, -(R0)               ;  and add an uparrow prefix
                51   D6  06A1                INCL     R1                          ; Add one to the echo count
                         06A3  230$:         $QIOW_S -                            ; Write out the echo
                         06A3                        EFN = #1, -                  ;  we really want to wait
                         06A3                        CHAN = READCHAN, -           ;  using the read channel
                         06A3                        FUNC = S^#IO$_WRITEVBLK, -   ;   writing obviously
                         06A3                        P1 = (R0), -                 ;  the echo
                         06A3                        P2 = R1                      ;  with its size
                7E   7C  06A3                     CLRQ    -(SP)
                7E   7C  06A5                     CLRQ    -(SP)
                51   DD  06A7                     PUSHL   R1
                60   DF  06A9                     PUSHAL  (R0)
                7E   7C  06AB                     CLRQ    -(SP)
                00   DD  06AD                     PUSHL   #0
            7E  00'  3C  06AF                     MOVZWL  S^#IO$_WRITEVBLK,-(SP)
       7E    0000'CF 3C  06B2                     MOVZWL  READCHAN,-(SP)
                01   DD  06B7                     PUSHL   #1
     00000000'GF 0C  FB  06B9                     CALLS   #12,G^SYS$QIOW
                05       06C0  240$:         RSB                                  ; Exit.
                         06C1
                         06C1  .DISABLE LSB
                         06C1
                         06C1  .ENABLE LSB
                         06C1
                         06C1  TERREADAST:                                        ; Terminal read done AST
              0FFC       06C1                .WORD    ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                         06C3
  48 0321'CF     00'  E1  06C3               BBC      S^#IO$V_NOFILTR, READ_MODE, 10$ ; Are we in ODT read mode?
 00010000'8F  0212'CF  D1  06C9              CMPL     TERIOSB, #<1@16>!SS$_NORMAL ; Normal completion of 1 char?
                3D   12  06D2                BNEQ     10$                         ; Nope, just go process it
```

```
          0216'CF   D5  06D4   871         TSTL    TERIOSB+4               ; Did we get any terminator?
               37   12  06D8   872         BNEQ    10$                     ; Yep, also just go process it
                        06DA   873         $QIOW_S -                       ; Else get the remaining data (if any)
                        06DA   874                 EFN = #1, -             ;  we really want to wait
                        06DA   875                 CHAN = READCHAN, -      ;  using the read channel obviously
                        06DA   876                 FUNC = <#IO$_READVBLK!IO$M_TRMNOECHO-
                        06DA   877                         !IO$M_NOECHO!IO$M_NOFILTR-
                        06DA   878                         !IO$M_TIMED>, - ;  using the read type ahead mode
                        06DA   879                 IOSB = TERIOSB, -       ;  (re-)use the IOSB
                        06DA   880                 P1 = TERBUFFER+1, -     ;  use remainder of terminal buffer
                        06DA   881                 P2 = #TERBUFLEN-1-1, -  ;   which is this long w/o terminator
                        06DA   882                 P4 = #TERMASK           ;  use the terminator mask
               7E   7C  06DA                   CLRQ    -(SP)
      00000323'8F   DD  06DC                   PUSHL   #TERMASK
               00   DD  06E2                   PUSHL   #0
      000000FD 8F   DD  06E4                   PUSHL   #TERBUFLEN-1-1
          021F'CF   DF  06EA                   PUSHAL  TERBUFFER+1
               7E   7C  06EE                   CLRQ    -(SP)
          0212'CF   7F  06F0                   PUSHAQ  TERIOSB
      7E  0000'8F   3C  06F4                   MOVZWL  #IO$_READVBLK!IO$M_TRMNOECHO                    !IO$M_NOECHO
      7E  0000'CF   3C  06F9                   MOVZWL  READCHAN,-(SP)
               01   DD  06FE                   PUSHL   #1
  00000000'GF  0C   FB  0700                   CALLS   #12,G^SYS$QIOW
                        0707   883         QUIT_ON_ERROR                   ; Quit on any error
            03 50   E8  0707                   BLBS    R0, 30018$
             FC24   31  070A                   BRW     QUIT
                        070D                   30018$:
          0214'CF   B6  070D   884         INCW    TERIOSB+2               ; Count the data character of 1st read
      50  0212'CF   3C  0711   885 10$:    MOVZWL  TERIOSB, R0             ; Get the I/O completion status
      0000'8F  50   B1  0716   886         CMPW    R0, #SS$_CONTROLY       ; Completed under CTRL/Y?
               09   12  071B   887         BNEQ    20$                     ; Nope
 0216'CF  00010019 8F   D0  071D   888     MOVL    #<1@16>!<^A/Y/-64>, TERIOSB+4 ; Yep, set CTRL/Y terminator
      0000'8F  50   B1  0726   889 20$:    CMPW    R0, #SS$_CONTROLC       ; Completed under CTRL/C?
               14   12  072B   890         BNEQ    30$                     ; Nope
          0212'CF   7C  072D   891         CLRQ    TERIOSB                 ; Yep, force nothing read at all
      05  0002'CF   D1  0731   892         CMPL    CTRLC_CNT, #5           ; Too many pending CTRL/C's?
               09   19  0736   893         BLSS    30$                     ; Not yet
 0216'CF  00010010 8F   D0  0738   894     MOVL    #<1@16>!<^A/P/-64>, TERIOSB+4 ; Yep, set CTRL/P terminator
      56  0214'CF   3C  0741   895 30$:    MOVZWL  TERIOSB+2, R6           ; Get amount of data read
      57  021E'CF   9E  0746   896         MOVAB   TERBUFFER, R7           ; Address the read data
      50  0216'CF   9A  074B   897         MOVZBL  TERIOSB+4, R0           ; Get the terminator if any
               0C   13  0750   898         BEQL    40$                     ; There was none...
          6746  50  90  0752   899         MOVB    R0, (R7)[R6]            ; There is one, store it in the buffer
            10  50  91  0756   900         CMPB    R0, #^A/P/-64           ; Is it CTRL/P?
               05   12  0759   901         BNEQ    50$                     ; Nope
             FD2D   30  075B   902         BSBW    CTRLP                   ; Handle that as an intercept
               76   11  075E   903 40$:    BRB     100$                    ; Continue
                        0760   904
  6E 0321'CF   00'  E0  0760   905 50$:    BBS     S^#IO$V_NOECHO, READ_MODE, 90$ ; Are we echoing?
         6A 50 07   E0  0766   906         BBS     #7, R0, 90$             ; Don't echo C1 controls, 10/0, 15/15
            14  50  91  076A   907         CMPB    R0, #^A/T/-64           ; Is it CTRL/T?
               65   13  076D   908         BEQL    90$                     ; Yep, that's never echoed
            19  50  91  076F   909         CMPB    R0, #^A/Y/-64           ; Is it CTRL/Y?
               60   13  0772   910         BEQL    90$                     ; Yep, that's never echoed
            51  01  3C  0774   911         MOVZWL  #1, R1                  ; Set echo size to one initially
 031D'CF   0A24 8F  B0  0777   912         MOVW    #<10@8>!^A/$/, ECHOBUFFER ;  and load echo buffer w/ $<LF>
            1B  50  91  077E   913         CMPB    R0, #27                 ; Is it <ESC>?
```

**F

```
                    32  13  0781   914          BEQL    80$                    ; Yes, that echoes as a dollar sign
          031D'CF   0D  90  0783   915          MOVB    #13, ECHOBUFFER        ; Now load echo buffer w/ <CR><LF>
               0A   50  91  0788   916          CMPB    R0, #10                ; Is it <LF>?
                    26  13  078B   917          BEQL    70$                    ; Yes, that echoes as <CR><LF>
          031D'CF   50  90  078D   918          MOVB    R0, ECHOBUFFER         ; Now load echo buffer w/ terminator
               07   50  91  0792   919          CMPB    R0, #7                 ; Less than <BEL>?
                    07  1F  0795   920          BLSSU   60$                    ; Yes, make into uparrow format
               0D   50  91  0797   921          CMPB    R0, #13                ; <CR> or less?
                    17  13  079A   922          BEQL    70$                    ; It's <CR>, go echo as <CR><LF>
                    17  1F  079C   923          BLSSU   80$                    ; Less that <CR>, echo as itself
       031D'CF  5E  8F  90  079E   924 60$:     MOVB    #^A/^/, ECHOBUFFER     ; Put uparrow prefix into echo buffer
  031E'CF  50   40  8F  89  07A4   925          BISB3   #64, R0, ECHOBUFFER+1  ;  then make the control a graphic
               1A   50  91  07AB   926          CMPB    R0, #^A/Z/-64          ; A CTRL/Z?
                    03  12  07AE   927          BNEQ    70$                    ; Nope
               51   02  C0  07B0   928          ADDL    #2, R1                 ; Yep, echo as ^Z<CR><LF>
                    51  D6  07B3   929 70$:     INCL    R1                     ; Add one to the echo count
                        07B5   930 80$:     $QIO_S -                       ; Queue up the echo
                        07B5   931                  CHAN = WRITECHAN, -    ;  on the write channel
                        07B5   932                  FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                        07B5   933                  P1 = ECHOBUFFER, -     ;  from the echo buffer
                        07B5   934                  P2 = R1                ;   with this size
               7E   7C  07B5                    CLRQ    -(SP)
               7E   7C  07B7                    CLRQ    -(SP)
                    51  DD  07B9                PUSHL   R1
          031D'CF   DF  07BB                    PUSHAL  ECHOBUFFER
               7E   7C  07BF                    CLRQ    -(SP)
               00   DD  07C1                    PUSHL   #0
          7E   00'  3C  07C3                    MOVZWL  S^#IO$_WRITEVBLK,-(SP)
     7E   0000'CF   3C  07C6                    MOVZWL  WRITECHAN,-(SP)
               00   DD  07CB                    PUSHL   #0
    00000000'GF  0C  FB  07CD                   CALLS   #12,G^SYS$QIO
               56   D6  07D4   935 90$:     INCL    R6                     ; Count the terminator in data count
               56   D5  07D6   936 100$:    TSTL    R6                     ; Any data at all?
               55   13  07D8   937          BEQL    130$                   ; Nope
       0B 0000'CF   E8  07DA   938          BLBS    TOPS20, 110$           ; Yep, TOPS-20 mode?
               77   56  90  07DF   939          MOVB    R6, -(R7)              ; Set length of data
                        07E2   940 ASSUME   TERBUFLEN LE 255
               56   04  C0  07E2   941          ADDL    #4, R6                 ; Calculate length of message
               77   56  B0  07E5   942          MOVW    R6, -(R7)              ;  and set it
                    57  D7  07E8   943          DECL    R7                     ; Back up to the TYPE byte
                        07EA   944 110$:    $QIO_S -                       ; Queue up a link write
                        07EA   945                  CHAN = LINKCHAN, -     ;  using the link channel
                        07EA   946                  FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                        07EA   947                  IOSB = TERIOSB, -      ;  use an IOSB
                        07EA   948                  ASTADR = 120$, -       ;  come here upon completion
                        07EA   949                  ASTPRM = 4(AP), -      ;   forwarding the output wait flag
                        07EA   950                  P1 = (R7), -           ;  send this message buffer
                        07EA   951                  P2 = R6                ;   which is this long
               7E   7C  07EA                    CLRQ    -(SP)
               7E   7C  07EC                    CLRQ    -(SP)
                    56  DD  07EE                PUSHL   R6
                    67  DF  07F0                PUSHAL  (R7)
               04 AC    DD  07F2                PUSHL   4(AP)
          0815'CF   DF  07F5                    PUSHAL  120$
          0212'CF   7F  07F9                    PUSHAQ  TERIOSB
          7E   00'  3C  07FD                    MOVZWL  S^#IO$_WRITEVBLK,-(SP)
     7E   0000'CF   3C  0800                    MOVZWL  LINKCHAN,-(SP)
```

```
                   00  DD 0805                  PUSHL   #0
       00000000'GF 0C  FB 0807                  CALLS   #12,G^SYS$QIO
                          080E          QUIT_NOT_ABORT                  ; Quit on non-link-abort error
             03 50     E8 080E                  BLBS    R0, 30019$
              FB18     31 0811                  BRW     ABORT_QUIT
                          0814                  30019$:
                   04     0814          RET                             ; Return pending link write complete.
                          0815
                          0815     120$:                                ; Link write done AST
              0FFC        0815          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
     50    0212'CF     3C 0817          MOVZWL  TERIOSB, R0             ; Get the I/O completion status
                          081C          QUIT_NOT_ABORT                  ; Quit on non-link-abort error
             03 50     E8 081C                  BLBS    R0, 30020$
              FB0A     31 081F                  BRW     ABORT_QUIT
                          0822                  30020$:
          09 04 AC     E9 0822          BLBC    4(AP), 130$             ; Should we wait for output done?
       0009'CF  01     90 0826          MOVB    #1, OUTPUT_WAIT         ; Yep, set output wait flag
                04     11 082B          BRB     140$                    ;  and wait for it...
                          082D
                          082D     OUTPUTWAITAST:                       ; Output wait done AST
              0FFC        082D          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                2A     10 082F     130$: BSBB   TERREAD                 ; Do a terminal (re-)read
                04        0831     140$: RET                            ; Return.
                          0832
                          0832     .DISABLE LSB
                          0832
                          0832     .ENABLE LSB
                          0832
           0000'CF     B5 0832     10$:  TSTW   FIRSTCMD                ; Initial command, have we used it?
                1A     13 0836          BEQL    20$                     ; It's been used, go check indirect
 021E'CF  0004'DF  0000'CF 28 0838      MOVC    FIRSTCMD, @FIRSTCMD+4, TERBUFFER ; Move in command
           0000'CF     B4 0842          CLRW    FIRSTCMD                ;  then say it's been used up
 0214'CF  53  0000021E'8F C3 0846       SUBL3   #TERBUFFER, R3, TERIOSB+2 ; Calculate the command's size
                26     11 0850          BRB     30$                     ;  and go use it
                          0852
 0000'CF   0000'CF     92 0852     20$:  MCOMB  INDFLAG, INDFLAG        ; Flip the flop
                75     13 0859          BEQL    70$                     ; No indirect, clear up and continue
                          085B     TERREAD:                             ; Queue up a terminal read
           0000'CF     95 085B          TSTB    INDFLAG                 ; Initial command or indirect file?
                7B     13 085F          BEQL    80$                     ; Neither
                CF     19 0861          BLSS    10$                     ; Initial command
                          0863          $GET -                          ; Get next record
                          0863                  RAB = SYSINRAB          ;  from the indirect file
           0000'CF     DF 0863          PUSHAL  SYSINRAB
       00000000'GF 01  FB 0867          CALLS   #$$.TMP1,G^SYS$GET
             2E 50     E9 086E          BLBC    R0, 50$                 ; Go check out any error
  0214'CF  0022'CF     B0 0871          MOVW    SYSINRAB+RAB$W_RSZ, TERIOSB+2 ; Set the data character size
   03 0321'CF  00'     E0 0878     30$:  BBS    S^#IO$V_NOFILTR, READ_MODE, 40$ ; Already in ODT mode?
              00A2     30 087E          BSBW    100$                    ; Nope, go shift to ODT mode
      0212'CF  00'     B0 0881     40$:  MOVW   S^#SS$_NORMAL, TERIOSB  ; Set completion status as normal
 0216'CF   0001000D 8F D0 0886          MOVL    #<1@16>!13, TERIOSB+4   ; Set length 1 terminator of <CR>
                          088F          $DCLAST_S -                     ; Fake an AST
                          088F                  ASTADR = TERREADAST, -  ;  for terminal read done
                          088F                  ASTPRM = #1             ;  saying wait for output to continue
                00     DD 088F                  PUSHL   #0
                01     DD 0891                  PUSHL   #1
           FE2A CF     DF 0893                  PUSHAL  TERREADAST
```

```
00000000'GF    03   FB  0897                      CALLS   #3,G^SYS$DCLAST
               05   089E   996          RSB                                   ; Exit.
                    089F   997
00000000'8F    50   D1  089F   998 50$:   CMPL    R0, #RMS$_EOF               ; Are we just done with the indirect?
               1D   13  08A6   999          BEQL    60$                         ; Simple EOF, we're done
     0940'CF   50   D0  08A8  1000          MOVL    R0, RMSMSGVEC+4             ; Load status code into message vector
0944'CF   000C'CF   D0  08AD  1001          MOVL    SYSINRAB+RAB$L_STV, RMSMSGVEC+8 ;  and load the STV value
                        08B4  1002          $PUTMSG_S -                         ; Put out a message
                        08B4  1003                  MSGVEC = RMSMSGVEC          ;  to announce the RMS error
               00   DD  08B4                        PUSHL   #0
               00   DD  08B6                        PUSHL   #0
               00   DD  08B8                        PUSHL   #0
          093C'CF   DF  08BA                        PUSHAL  RMSMSGVEC
00000000'GF    04   FB  08BE                        CALLS   #4,G^SYS$PUTMSG
                        08C5  1004 60$:   $CLOSE -                            ; Close out
                        08C5  1005                  FAB = SYSINFAB              ;  the indirect file
          0000'CF   DF  08C5                        PUSHAL  SYSINFAB
00000000'GF    01   FB  08C9                        CALLS   #$$.TMP1,G^SYS$CLOSE
          0000'CF   94  08D0  1006 70$:   CLRB    INDFLAG                     ; Say nothing special is active
02 0321'CF     00'  E1  08D4  1007          BBC     S^#IO$V_NOFILTR, READ_MODE, 80$ ; Already in NORMAL mode?
               5A   10  08DA  1008          BSBB    110$                        ; Nope, go shift (back) to NORMAL mode
     51   00FE 8F   3C  08DC  1009 80$:   MOVZWL  #TERBUFLEN-1, R1            ; Get buffer size w/o a terminator
03 0321'CF     00'  E1  08E1  1010          BBC     S^#IO$V_NOFILTR, READ_MODE, 90$ ; All set unless ODT mode
          51   01   3C  08E7  1011          MOVZWL  #1, R1                      ; Set buffer size to one of ODT mode
     33 0000'CF     E8  08EA  1012 90$:   BLBS    WAKEFLAG, 95$               ; Don't queue another read if quitting
                        08EF  1013          $QIO_S -                            ; Queue up a terminal read
                        08EF  1014                  CHAN = READCHAN, -          ;  using the read channel obviously
                        08EF  1015                  FUNC = READ_MODE, -         ;  using the correct read mode
                        08EF  1016                  IOSB = TERIOSB, -           ;  use an IOSB
                        08EF  1017                  ASTADR = TERREADAST, -      ;  come here upon completion
                        08EF  1018                  ASTPRM = #0, -              ;  saying no wait for output
                        08EF  1019                  P1 = TERBUFFER, -           ;  read into this buffer
                        08EF  1020                  P2 = R1, -                  ;   which is this long this time
                        08EF  1021                  P4 = #TERMASK               ;  use the terminator mask
               7E   7C  08EF                        CLRQ    -(SP)
    00000323'8F     DD  08F1                        PUSHL   #TERMASK
               00   DD  08F7                        PUSHL   #0
               51   DD  08F9                        PUSHL   R1
          021E'CF   DF  08FB                        PUSHAL  TERBUFFER
               00   DD  08FF                        PUSHL   #0
          FDBC CF   DF  0901                        PUSHAL  TERREADAST
          0212'CF   7F  0905                        PUSHAQ  TERIOSB
     7E   0321'CF   3C  0909                        MOVZWL  READ_MODE,-(SP)
     7E   0000'CF   3C  090E                        MOVZWL  READCHAN,-(SP)
               00   DD  0913                        PUSHL   #0
00000000'GF    0C   FB  0915                        CALLS   #12,G^SYS$QIO
                        091C  1022          QUIT_ON_ERROR                       ; Quit on any error
            03 50   E8  091C                        BLBS    R0, 30021$
             FA0F   31  091F                        BRW     QUIT
                        0922                        30021$:
               05   0922  1023 95$:   RSB                                   ; Exit.
                    0923  1024
     49 0000'CF     E8  0923  1025 100$:  BLBS    TOPS20, 130$                ; Mode always correct if TOPS-20
0321'CF   0000'8F   A8  0928  1026          BISW    #IO$M_NOECHO!IO$M_NOFILTR, READ_MODE ; Set ODT read mode
  08DB'CF      01   90  092F  1027          MOVB    #1@0, CNTLBF_ECHOFLG        ;  and indicate ODT in CONTROL msg
               11   11  0934  1028          BRB     120$                        ; Go send the message
                        0936  1029
```

```
        36 0000'CF  E8 0936  1030 110$:   BLBS    TOPS20, 130$              ; Mode always correct if TOPS-20
  0321'CF   0000'8F  AA 093B  1031         BICW    #IO$M_NOECHO!IO$M_NOFILTR, READ_MODE ; Set NORMAL read mode
     08DB'CF    02   90 0942  1032         MOVB    #1@1, CNTLBF_ECHOFLG      ;  and Indicate NORMAL in CONTROL msg
       50   08D7'CF  3C 0947  1033 120$:   MOVZWL  CNTLBF+1, R0              ; Get the message's length
                        094C  1034         $QIO_S -                          ; Send the CONTROL message
                        094C  1035                 CHAN = LINKCHAN, -        ;  using the link channel
                        094C  1036                 FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                        094C  1037                 P1 = CNTLBF, -            ;  buffer is the CONTROL message
                        094C  1038                 P2 = R0                   ;   which is this long
                  7E  7C 094C                  CLRQ    -(SP)
                  7E  7C 094E                  CLRQ    -(SP)
                  50  DD 0950                  PUSHL   R0
             08D6'CF  DF 0952                  PUSHAL  CNTLBF
                  7E  7C 0956                  CLRQ    -(SP)
                  00  DD 0958                  PUSHL   #0
            7E   00'  3C 095A                  MOVZWL  S^#IO$_WRITEVBLK,-(SP)
       7E   0000'CF   3C 095D                  MOVZWL  LINKCHAN,-(SP)
                  00  DD 0962                  PUSHL   #0
     00000000'GF  0C  FB 0964                  CALLS   #12,G^SYS$QIO
                        096B  1039         QUIT_NOT_ABORT                    ; Quit on non-link-abort error
               03 50  E8 096B                  BLBS    R0, 30022$
                F9BB  31 096E                  BRW     ABORT_QUIT
                        0971                   30022$:
                  05     0971  1040 130$:   RSB                               ; Exit.
                        0972  1041
                        0972  1042 .DISABLE LSB
```

```
                        0972      .SBTTL  Link read AST's
                        0972
                        0972      LINKREADAST:                                        ; Link read done AST
                 0FFC   0972              .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                        0974
       50   000A'CF   3C 0974              MOVZWL  LINKIOSB, R0                        ; Get the I/O completion status
                        0979              QUIT_NOT_ABORT                              ; Quit on non-link-abort error
             03 50   E8 0979                      BLBS    R0, 30023$
              F9AD   31 097C                      BRW     ABORT_QUIT
                        097F                      30023$:
       56   000C'CF   3C 097F              MOVZWL  LINKIOSB+2, R6                      ; Get size of the read message
                57   13 0984              BEQL    50$                                 ; No size??
       57   0012'CF   9E 0986              MOVAB   LINKBUFFER, R7                      ; Get a pointer to read message
         58 0000'CF   E8 098B              BLBS    TOPS20, 70$                         ; Always just data if TOPS-20
             05  67   91 0990              CMPB    (R7), #5                            ; Data message?
                 4B   13 0993              BEQL    60$                                 ; Yep
             02  67   91 0995              CMPB    (R7), #2                            ; Control message?
                 43   12 0998              BNEQ    50$                                 ; Nope, other, punt on it
       06    01  A7   B1 099A              CMPW    1(R7), #6                           ; Yep, is size at least 6?
                 3D   1F 099E              BLSSU   50$                                 ; Too small, also punt on it
    37 0321'CF   00'  E0 09A0              BBS     S^#IO$V_NOFILTR, READ_MODE, 50$ ; Also punt if in ODT mode
    01     05  A7     91 09A6              CMPB    5(R7), #1@0                         ; Which way is echo changing?
                 08   13 09AA              BEQL    10$                                 ; Echo is turning off
    2B 0321'CF   00'  E5 09AC              BBCC    S^#IO$V_NOECHO, READ_MODE, 50$ ; Echo ON; is it already on?
                 06   11 09B2              BRB     20$                                 ; Not already on, correct the read
                        09B4
    23 0321'CF   00'  E2 09B4      10$:    BBSS    S^#IO$V_NOECHO, READ_MODE, 50$ ; Echo OFF; is it already off?
                        09BA      20$:    $CANCEL_S -                                 ; Cancel the current terminal read
                        09BA                      CHAN = READCHAN             ;  on the read channel
       7E   0000'CF   3C 09BA                      MOVZWL  READCHAN,-(SP)
   00000000'GF   01   FB 09BF                      CALLS   #1,G^SYS$CANCEL
                 15   11 09C6              BRB     50$                                 ; Now queue up another link read
                        09C8
                        09C8      30$:                                                ; Terminal write completed AST
                 0FFC   09C8              .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
    0D 0009'CF   00   E5 09CA      40$:    BBCC    #0, OUTPUT_WAIT, 50$        ; Say terminal output is done
                        09D0              $DCLAST_S -                                 ; Something waiting, fake an AST
                        09D0                      ASTADR = OUTPUTWAITAST      ;  for terminal output waiter
                 7E   7C 09D0                      CLRQ    -(SP)
          FE57 CF     DF 09D2                      PUSHAL  OUTPUTWAITAST
   00000000'GF   03   FB 09D6                      CALLS   #3,G^SYS$DCLAST
                 6F   10 09DD      50$:    BSBB    LINKREAD                            ; Queue up another link read
                      04 09DF              RET                                         ; Return.
                        09E0
             57  04   C0 09E0      60$:    ADDL    #4, R7                              ; Skip pointer over the header
             56  04   C2 09E3              SUBL    #4, R6                              ;  and don't count it in the size
                 F5   1B 09E6              BLEQU   50$                                 ; Nothing left??
            0002'CF   D5 09E8      70$:    TSTL    CTRLC_CNT                           ; Pending CTRL/C's?
                 EF   18 09EC              BGEQ    50$                                 ; Yep, toss this output...
       09 0001'CF     E8 09EE              BLBS    EIGHT_BIT, 75$                      ; An 8-bit terminal?
 56    034B'CF   00  67   56   2E 09F3     MOVTC   R6, (R7), #0, ASCII_TRIM, R6, (R7) ; Nope, form 7-bit ASCII
                      67 09FB
             016E     30 09FC      75$:    BSBW    DO_LOG                              ; Go do logging if logging is active
       22 0006'CF     00   E5 09FF         BBCC    #0, CTRLC_SENT, 80$                 ; A recent CTRL/C sent?
 67   56   0007'CF    02   39 0A05         MATCHC  #2, CTRLC_MATCH, R6, (R7) ; Try to find the "^C" in text
                 19   12 0A0C              BNEQ    80$                                 ; Not there??
             56  52   7D 0A0E              MOVQ    R2, R6                              ; Found, copy desc just beyond it
```

```
         01  56  D1  0A11       CMPL    R6, #1                  ; What's the remaining length?
             B4  1F  0A14       BLSSU   40$                     ; It's zero, nothing remains, skip it
             0F  13  0A16       BEQL    80$                     ; It's one, just go do that character
    0A0D 8F  67  B1  0A18       CMPW    (R7), #<10a8>!13        ; It's two or greater, is it <CR><LF>?
             08  12  0A1D       BNEQ    80$                     ; Nope
         57  02  C0  0A1F       ADDL    #2, R7                  ; Yep, skip pointer over the <CR><LF>
         56  02  C2  0A22       SUBL    #2, R6                  ;  and don't count it
             A3  13  0A25       BEQL    40$                     ; Skip it if nothing now left
                     0A27  80$: $QIO_S -                        ; Queue up a terminal write
                     0A27               CHAN = WRITECHAN, -     ;  on the write channel
                     0A27               FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                     0A27               ASTADR = 30$, -         ;  come here upon completion
                     0A27               P1 = (R7), -            ;  use this for the data
                     0A27               P2 = R6                 ;   which is this long
             7E  7C  0A27               CLRQ    -(SP)
             7E  7C  0A29               CLRQ    -(SP)
             56  DD  0A2B               PUSHL   R6
             67  DF  0A2D               PUSHAL  (R7)
             00  DD  0A2F               PUSHL   #0
          94 AF  DF  0A31               PUSHAL  30$
             00  DD  0A34               PUSHL   #0
         7E  00' 3C  0A36               MOVZWL  S^#IO$_WRITEVBLK,-(SP)
   7E  0000'CF   3C  0A39               MOVZWL  WRITECHAN,-(SP)
             00  DD  0A3E               PUSHL   #0
 00000000'GF 0C  FB  0A40               CALLS   #12,G^SYS$QIO
                     0A47       QUIT_ON_ERROR                   ; Quit on any error
          03 50  E8  0A47               BLBS    R0, 30024$
           F8E4  31  0A4A               BRW     QUIT
                     0A4D               30024$:
                 04  0A4D       RET                             ; Return pending write completion.
                     0A4E
                     0A4E  LINKREAD:                            ; Queue up a link read
                     0A4E       $QIO_S -                        ; Queue up a link read
                     0A4E               CHAN = LINKCHAN, -      ;  using the link channel obviously
                     0A4E               FUNC = S^#IO$_READVBLK, - ;  reading obviously
                     0A4E               IOSB = LINKIOSB, -      ;  use an IOSB
                     0A4E               ASTADR = LINKREADAST, - ;  come here upon completion
                     0A4E               P1 = LINKBUFFER, -      ;  read into this buffer
                     0A4E               P2 = #LINKBUFLEN        ;   which is this long
             7E  7C  0A4E               CLRQ    -(SP)
             7E  7C  0A50               CLRQ    -(SP)
    00000200 8F  DD  0A52               PUSHL   #LINKBUFLEN
        0012'CF  DF  0A58               PUSHAL  LINKBUFFER
             00  DD  0A5C               PUSHL   #0
        FF10 CF  DF  0A5E               PUSHAL  LINKREADAST
        000A'CF  7F  0A62               PUSHAQ  LINKIOSB
         7E  00' 3C  0A66               MOVZWL  S^#IO$_READVBLK,-(SP)
   7E  0000'CF   3C  0A69               MOVZWL  LINKCHAN,-(SP)
             00  DD  0A6E               PUSHL   #0
 00000000'GF 0C  FB  0A70               CALLS   #12,G^SYS$QIO
                     0A77       QUIT_NOT_ABORT                  ; Quit on non-link-abort error
          03 50  E8  0A77               BLBS    R0, 30025$
           F8AF  31  0A7A               BRW     ABORT_QUIT
                     0A7D               30025$:
                 05  0A7D       RSB                             ; Exit.
```

```
                         0A7E  .SBTTL  Logging file handling
                         0A7E
                         0A7E  .ENABLE LSB
                         0A7E
                         0A7E  END_LOG:                                          ; Close out logging file if any
          0B32'CF   95   0A7E           TSTB    LOGGING_FLAG                     ; Is logging active?
               67   13   0A82           BEQL    40$                              ; No
          0B32'CF   94   0A84           CLRB    LOGGING_FLAG                     ; Yes, but no longer
          0BDC'CF   B5   0A88           TSTW    LOGGING_BUFDESC                  ; Anything left in the buffer?
               02   13   0A8C           BEQL    5$                               ; Nope
               17   10   0A8E           BSBB    DUMP_LOG                         ; Yep, go dump last record in log
                         0A90  5$:      $CLOSE -                                 ; Close
                         0A90                   FAB = LOGGING_FAB                ;  the logging file
          0B3C'CF   DF   0A90           PUSHAL  LOGGING_FAB
      00000000'GF   01   FB   0A94      CALLS   #$$.TMPT,G^SYS$CLOSE
            4D 50   E8   0A9B           BLBS    R0, 40$                          ; Go exit if success
 0BD8'CF  0B48'CF   D0   0A9E  10$:     MOVL    LOGGING_FAB+FAB$L_STV, LOGGING_MSGVEC+8 ; Set RMS FAB value
               2E   11   0AA5           BRB     30$                              ;  and go announce the error
                         0AA7
                         0AA7  DUMP_LOG:                                         ; Dump the logging record buffer
 0BAE'CF  0BDC'CF   B0   0AA7           MOVW    LOGGING_BUFDESC, LOGGING_RAB+RAB$W_RSZ ; Set record length
 0BB4'CF  0BE4'CF   9E   0AAE           MOVAB   LOGGING_BUFFER, LOGGING_RAB+RAB$L_RBF ; Set buffer address
                         0AB5           $PUT -                                   ; Put a record
                         0AB5                   RAB = LOGGING_RAB                ;  to the logging file
          0B8C'CF   DF   0AB5           PUSHAL  LOGGING_RAB
      00000000'GF   01   FB   0AB9      CALLS   #$$.TMPT,G^SYS$PUT
          0BDC'CF   D4   0AC0           CLRL    LOGGING_BUFDESC                  ; Clear out the
 0BE0'CF  0BE3'CF   9E   0AC4           MOVAB   LOGGING_BUFFER-1, LOGGING_BUFDESC+4 ;  buffer descriptor
            1D 50   E8   0ACB           BLBS    R0, 40$                          ; Go exit if success
 0BD8'CF  0B98'CF   D0   0ACE  20$:     MOVL    LOGGING_RAB+RAB$L_STV, LOGGING_MSGVEC+8 ; Set RMS RAB value
       0BD4'CF  50  D0   0AD5  30$:     MOVL    R0, LOGGING_MSGVEC+4             ; Set the error status in vector
                         0ADA           $PUTMSG_S -                              ; $PUTMSG the error into real log file
                         0ADA                   MSGVEC = LOGGING_MSGVEC          ;  using the built-up message vector
               00   DD   0ADA                   PUSHL   #0
               00   DD   0ADC                   PUSHL   #0
               00   DD   0ADE                   PUSHL   #0
          0BD0'CF   DF   0AE0                   PUSHAL  LOGGING_MSGVEC
      00000000'GF   04   FB   0AE4              CALLS   #4,G^SYS$PUTMSG
                    05   0AEB  40$:     RSB                                      ; Exit.
                         0AEC
                         0AEC  START_LOG:                                        ; Start up a logging file
 60   045A'CF   20  3A   0AEC           LOCC    #32, CMDIOSB+2, (R0)             ; Isolate any specified file name
               04   13   0AF2           BEQL    50$                              ; Nothing there...
               50   D7   0AF4           DECL    R0                               ; Else don't count the space
               51   D6   0AF6           INCL    R1                               ;  and skip the pointer over it
       0B70'CF  50  90   0AF8  50$:     MOVB    R0, LOGGING_FAB+FAB$B_FNS        ; Set the file name's size
       0B68'CF  61  9E   0AFD           MOVAB   (R1), LOGGING_FAB+FAB$L_FNA      ;  and the file name's address
             FF79   30   0B02           BSBW    END_LOG                          ; First close any old logging file
                         0B05           $CREATE -                                ; Try a create
                         0B05                   FAB = LOGGING_FAB                ;  of the logging file
          0B3C'CF   DF   0B05           PUSHAL  LOGGING_FAB
      00000000'GF   01   FB   0B09      CALLS   #$$.TMPT,G^SYS$CREATE
            8B 50   E9   0B10           BLBC    R0, 10$                          ; Forget logging if any error...
                         0B13           $CONNECT -                               ; Else connect
                         0B13                   RAB = LOGGING_RAB                ;  to the logging file
          0B8C'CF   DF   0B13           PUSHAL  LOGGING_RAB
      00000000'GF   01   FB   0B17      CALLS   #$$.TMPT,G^SYS$CONNECT
```

```
           AD 50   E9  0B1E        BLBC    R0, 20$                         ; Forget logging if any error...
     0B32'CF  01   90  0B21        MOVB    #1, LOGGING_FLAG                ; All set, indicate logging active
        0BDC'CF    D4  0B26        CLRL    LOGGING_BUFDESC                 ; Set up an empty
  0BE0'CF  0BE3'CF 9E  0B2A        MOVAB   LOGGING_BUFFER-1, LOGGING_BUFDESC+4 ;  buffer descriptor
       36 0000'CF  E8  0B31        BLBS    TOPS20, 60$                     ; All set if TOPS-20
     08DB'CF  01   90  0B36        MOVB    #1@0, CNTLBF_ECHOFLG            ; Set ODT mode on CONTROL message
  0321'CF  0000'8F A8  0B3B        BISW    #IO$M_NOECHO!IO$M_NOFILTR, READ_MODE ; Set ODT read mode
       50 08D7'CF  3C  0B42        MOVZWL  CNTLBF+1, R0                    ; Get the message's length
                       0B47        $QIO_S -                                ; Send the CONTROL message
                       0B47                CHAN = LINKCHAN, -              ;  using the link channel
                       0B47                FUNC = S^#IO$_WRITEVBLK, - ;  writing obviously
                       0B47                P1 = CNTLBF, -                  ;  buffer is the CONTROL message
                       0B47                P2 = R0                         ;   which is this long
               7E  7C  0B47                CLRQ    -(SP)
               7E  7C  0B49                CLRQ    -(SP)
               50  DD  0B4B                PUSHL   R0
          08D6'CF  DF  0B4D                PUSHAL  CNTLBF
               7E  7C  0B51                CLRQ    -(SP)
               00  DD  0B53                PUSHL   #0
           7E  00' 3C  0B55                MOVZWL  S^#IO$_WRITEVBLK,-(SP)
       7E  0000'CF 3C  0B58                MOVZWL  LINKCHAN,-(SP)
               00  DD  0B5D                PUSHL   #0
  00000000'GF  0C  FB  0B5F                CALLS   #12,G^SYS$QIO
                       0B66        QUIT_NOT_ABORT                          ; Quit on non-link-abort error
            03 50  E8  0B66                BLBS    R0, 30026$
             F7C0  31  0B69                BRW     ABORT_QUIT
                       0B6C                30026$:
                   05  0B6C 60$:   RSB                                     ; Exit.
                       0B6D
                       0B6D .DISABLE LSB
                       0B6D
                       0B6D DO_LOG:                                        ; Do logging if logging is active
          0B32'CF  95  0B6D        TSTB    LOGGING_FLAG                    ; Is logging active?
               4D  13  0B71        BEQL    70$                             ; No
            54 56  7D  0B73        MOVQ    R6, R4                          ; Yes, copy the data descriptor
            52 85  9A  0B76 10$:   MOVZBL  (R5)+, R2                       ; Get the next character
            0A 52  91  0B79        CMPB    R2, #10                         ; Is it a <LF>?
               13  12  0B7C        BNEQ    20$                             ; Nope
          0BDC'CF  B5  0B7E        TSTW    LOGGING_BUFDESC                 ; Anything left in the record buffer?
               36  13  0B82        BEQL    50$                             ; Nothing left, go dump the buffer
       0D 0BE0'DF  91  0B84        CMPB    @LOGGING_BUFDESC+4, #13 ; Was the <LF> preceeded by a <CR>?
               2F  12  0B89        BNEQ    50$                             ; No, go dump the buffer
          0BDC'CF  B7  0B8B        DECW    LOGGING_BUFDESC                 ; Yes, remove the <CR> from buffer
               29  11  0B8F        BRB     50$                             ;  and then go dump the buffer
                       0B91
            0D 52  91  0B91 20$:   CMPB    R2, #13                         ; Is it a <CR>?
               06  12  0B94        BNEQ    30$                             ; Nope
          0BDC'CF  B5  0B96        TSTW    LOGGING_BUFDESC                 ; Putting <CR> into an empty buffer?
               21  13  0B9A        BEQL    60$                             ; Would've, but don't, just ignore it
  00FF 8F 0BDC'CF  B1  0B9C 30$:   CMPW    LOGGING_BUFDESC, #LOGGING_BUFLEN ; Room in buffer for another?
               03  1F  0BA3        BLSSU   40$                             ; Room exists, go buffer character
             FEFF  30  0BA5        BSBW    DUMP_LOG                        ; Else dump the partial record buffer
          0BDC'CF  B6  0BA8 40$:   INCW    LOGGING_BUFDESC                 ; Say one more character in buffer
          0BE0'CF  D6  0BAC        INCL    LOGGING_BUFDESC+4               ; Bump the record buffer pointer
       0BE0'DF 52  90  0BB0        MOVB    R2, @LOGGING_BUFDESC+4          ;  and store the character in buffer
            0C 52  91  0BB5        CMPB    R2, #12                         ; Did we just store a <FF>?
               03  12  0BB8        BNEQ    60$                             ; Nope
```

```
    FEEA  30  0BBA  50$:    BSBW    DUMP_LOG         ; Dump record buffer to logging file
   B6 54  F5  0BBD  60$:    SOBGTR  R4, 10$          ; Loop for all of the data...
          05  0BC0  70$:    RSB                      ; Exit.
              0BC1
              0BC1          .END
```

```
$$.TAB               = 00000B8C R     02
$$.TABEND            = 00000BD0 R     02
$$.TMP               = 00000400
$$.TMP1              = 00000001
$$.TMP2              = 000000AF
$$T1                 = 00000001
ABORT_QUIT             0000032C R     04
ASCII_TRIM             0000034B R     02
CMDBUFFER              00000460 R     02
CMDBUFLEN            = 00000100
CMDHELP                000008CC R     02
CMDIOSB                00000458 R     02
CMDMASK                00000560 R     02
CMDODT                 000008D4 R     02
CMDPROMPT              00000588 R     02
CNTLBF                 000008D6 R     02
CNTLBF_ECHOFLG         000008DB R     02
CNTRLCHAN              ********   X   04
CONFBF                 00000958 R     02
CONFBF_LEN           = 00000003
CONNECT                00000B22 R     02
CRLFPROMPT             00000590 R     02
CTRLCAST               00000394 R     04
CTRLC_CNT              00000002 R     02
CTRLC_IOSB             0000044B R     02
CTRLC_MATCH            00000007 R     02
CTRLC_MSG              00000453 R     02
CTRLC_SENT             00000006 R     02
CTRLP                  0000048B R     04
DIB$B_DEVCLASS       = 00000004
DIB$L_DEVDEPEND      = 00000008
DIB$W_DEVBUFSIZ      = 00000006
DO_LOG                 00000B6D R     04
DSC$B_CLASS          = 00000003
DSC$K_CLASS_D        = 00000002
DUMP_LOG               00000AA7 R     04
ECHOBUFFER             0000031D R     02
EIGHT_BIT              00000001 R     02
ENABLE_CTRLC           000003F4 R     04
END_LOG                00000A7E R     04
ERASEPROMPT            0000059A R     02
FAB$B_FNS            = 00000034
FAB$C_BID            = 00000003
FAB$C_BLN            = 00000050
FAB$C_SEQ            = 00000000
FAB$C_VAR            = 00000002
FAB$L_ALQ            = 00000010
FAB$L_FNA            = 0000002C
FAB$L_FOP            = 00000004
FAB$L_STV            = 0000000C
FAB$V_CHAN_MODE      = 00000002
FAB$V_CR             = 00000001
FAB$V_FILE_MODE      = 00000004
FAB$V_LNM_MODE       = 00000000
FAB$V_PUT            = 00000000
FAB$V_SQO            = 00000006
FAB$W_GBC            = 00000048
FINALACS               ********   X   04
FINALPATH              ********   X   02
FIRSTCMD               ********   X   04
HELLO                  00000B12 R     02
HELPPROMPT             00000973 R     02
INDFLAG                ********   X   04
IO$M_CTRLCAST          ********   X   04
IO$M_DSABLMBX          ********   X   02
IO$M_NOECHO            ********   X   04
IO$M_NOFILTR           ********   X   04
IO$M_TIMED             ********   X   04
IO$M_TRMNOECHO         ********   X   02
IO$V_NOECHO            ********   X   04
IO$V_NOFILTR           ********   X   04
IO$_READPROMPT         ********   X   04
IO$_READVBLK           ********   X   02
IO$_SETMODE            ********   X   04
IO$_WRITEVBLK          ********   X   04
LINKBUFFER             00000012 R     02
LINKBUFLEN           = 00000200
LINKCHAN               ********   X   04
LINKIOSB               0000000A R     02
LINKMBXAST             000002DE R     04
LINKMBXBUFFER          000008E4 R     02
LINKMBXBUFLEN        = 00000028
LINKMBXIOSB            000008DC R     02
LINKMBXREAD            000002B2 R     04
LINKREAD               00000A4E R     04
LINKREADAST            00000972 R     04
LOGGING_BUFDESC        00000BDC R     02
LOGGING_BUFFER         00000BE4 R     02
LOGGING_BUFLEN       = 000000FF
LOGGING_FAB            00000B3C R     02
LOGGING_FDEF           00000B33 R     02
LOGGING_FDEF_LEN     = 00000009
LOGGING_FLAG           00000B32 R     02
LOGGING_MSGVEC         00000BD0 R     02
LOGGING_RAB            00000B8C R     02
LOGIN                  00000B1A R     02
MAILCHAN               ********   X   04
MSG$_ABORT             ********   X   04
MSG$_CONFIRM           ********   X   04
MSG$_DISCON            ********   X   04
MSG$_TRMHANGUP         ********   X   04
NETMBXVEC              00000948 R     02
NODENAME               ********   X   04
OUTPUTWAITAST          0000082D R     04
OUTPUT_WAIT            00000009 R     02
QUIT                   00000331 R     04
RAB$B_RAC            = 0000001E
RAB$C_BID            = 00000001
RAB$C_BLN            = 00000044
RAB$C_SEQ            = 00000000
RAB$L_CTX            = 00000018
RAB$L_RBF            = 00000028
RAB$L_ROP            = 00000004
RAB$L_STV            = 0000000C
```

```
RAB$L_UBF          = 00000024
RAB$V_WBH          = 0000000A
RAB$W_RSZ          = 00000022
RAB$W_USZ          = 00000020
READCHAN             ********    X    04
READ_MODE            00000321 R       02
REM$_NETMBX          ********    X    02
REM$_REMOTE          ********    X    02
REMPROMPT            00000966 R       02
RETSTATUS            ********    X    04
RMS$_EOF             ********    X    04
RMSMSGVEC            0000093C R       02
RSTSHELP             00000AC8 R       02
RSTSRT               00000000 RG      04
SS$_ABORT            ********    X    04
SS$_CONTROLC         ********    X    04
SS$_CONTROLY         ********    X    04
SS$_NORMAL           ********    X    04
START_LOG            00000AEC R       04
STR$CONCAT           ********    X    04
SYS$CANCEL           ********   GX    04
SYS$CLOSE            ********   GX    04
SYS$CONNECT          ********   GX    04
SYS$CREATE           ********   GX    04
SYS$DCLAST           ********   GX    04
SYS$GET              ********   GX    04
SYS$PUT              ********   GX    04
SYS$PUTMSG           ********   GX    04
SYS$QIO              ********   GX    04
SYS$QIOW             ********   GX    04
SYS$WAKE             ********   GX    04
SYSINFAB             ********    X    04
SYSINRAB             ********    X    04
TERBUFFER            0000021E R       02
TERBUFLEN          = 000000FF
TERIOSB              00000212 R       02
TERMASK              00000323 R       02
TERMCHAR             ********    X    04
TERMMBXAST           00000268 R       04
TERMMBXBUFFER        00000914 R       02
TERMMBXBUFLEN      = 00000028
TERMMBXCHAN          ********    X    04
TERMMBXIOSB          0000090C R       02
TERMMBXREAD          00000287 R       04
TERREAD              0000085B R       04
TERREADAST           000006C1 R       04
TOPS20               00000000 R       02
TOPS20HELP           00000AEB R       02
TOPS20RT             000000BA RG      04
TT$M_ESCAPE        = 00000008
TT$M_HALFDUP       = 00100000
TT$M_PASSALL       = 00000001
TT$M_WRAP          = 00000200
TT$V_EIGHTBIT      = 0000000F
TT$V_HOSTSYNC      = 00000004
TT$V_LOWER         = 00000007
TT$V_MECHFORM      = 00000013
TT$V_MECHTAB       = 00000008
TT$V_SCOPE         = 0000000C
TTYPBF               0000095B R       02
TTYPBF_FILL          00000964 R       02
TTYPBF_TYPE          00000962 R       02
TTYPBF_WIDTH         00000960 R       02
WAKEFLAG             ********    X    04
WRITECHAN            ********    X    04
XXXRT                000000CB R       04
```

```
                                                 +----------------+
                                                 ! 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
 RSTSRT                           00000CE3  (  3299.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC LONG
PROTOTBL                          0000000C  (    12.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC BYTE
RSTSRT                            00000BC1  (  3009.)  04 (  4.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD  NOWRT NOVEC BYTE

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

Phase                        Page faults    CPU Time        Elapsed Time
-----                        -----------    --------        ------------
Initialization                        34    00:00:00.04     00:00:01.02
Command processing                   122    00:00:00.45     00:00:02.28
Pass 1                               459    00:00:10.91     00:00:42.31
Symbol table sort                      0    00:00:00.82     00:00:01.56
Pass 2                               299    00:00:03.19     00:00:12.95
Symbol table output                   22    00:00:00.12     00:00:00.52
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                 940    00:00:15.55     00:01:00.66

The working set limit was 1950 pages.
128123 bytes (251 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 722 non-local and 147 local symbols.
1212 source lines were read in Pass 1, producing 35 object records in Pass 2.
40 pages of virtual memory were used to define 34 macros.

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

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[RTPAD.OBJ]RTPAD.MLB;1                        0
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                            0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                        28
TOTALS (all libraries)                                   28

951 GETS were required to define 28 macros.

There were no errors, warnings or information messages.

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

0333 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CTDSENSE
LIS

CTDVMSPEC
LIS

CTSENSERT
LIS

RSTSRT
LIS

CTERMRT
LIS

DTE_DF03
LIS

CTSETRT
LIS

CTDSET
LIS

0334 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RTMSG
LIS

RSXRT
LIS

RTDEF
LIS

RTDTE
LIS

RTPAD
LIS

RTLOGIO
LIS