TSSLE -- TSX-Plus Single Line E MACRO V05.05 Wednesday 18-Jan-89 14:16
Table of contents


```
    2-    1       GTSLCH -- Get next character from single-line editor
    3-    1       SLINCH -- See if any input characters need to be processed
    4-    1       SLINIT -- Initialize for new get line
    6-    1       ORDCHR -- Process ordinary characters
    7-    1       REGCHR -- Process normal characters
    8-    1       CKUAC  -- Check for user-defined activation characters
    9-    1       CKRDTM -- Check for read time-out character
   10-    1       CSCHK  -- Check for start of control sequence
   11-    1       EPCH1  -- Accrue a terminal control sequence
   12-    1       CSFIN  -- Process terminal control sequence
   14-    1       TCUP   -- Up-arrow processing
   15-    1       TCDOWN -- Down-arrow processing
   16-    1       TCLEFT -- Left-arrow processing
   17-    1       TCRITE -- Right-arrow processing
   18-    1       TCPF1  -- PF1 processing
   19-    1       TCPF2  -- PF2 processing
   20-    1       TCPF3  -- PF3 processing
   21-    1       TCPF4  -- PF4 processing
   22-    1       TCENTR -- Enter key processing
   22-   19       TCINVL -- Invalid function key
   23-    1       KEY0   -- Key "0" processing
   24-    1       KEY1   -- Key "1" processing
   25-    1       KEY2   -- Key "2" processing
   26-    1       KEY3   -- Key "3" processing
   27-    1       KEY4   -- Key "4" processing
   28-    1       KEY5   -- Key "5" processing
   29-    1       KEY6   -- Key "6" processing
   30-    1       KEY7   -- Key "7" processing
   31-    1       KEY8   -- Key "8" processing
   32-    1       KEY9   -- Key "9" processing
   33-    1       KEYCOM -- Key "," processing
   34-    1       KEYMIN -- Key "-" processing
   35-    1       KBS    -- Save a command
   36-    1       KBX    -- Recall the saved command
   37-    1       E1     -- E1 processing
   38-    1       E2     -- E2 processing
   39-    1       E3     -- E3 processing
   40-    1       E4     -- E4 processing
   41-    1       E5     -- E5 processing
   42-    1       E6     -- E6 processing
   43-    1       F6     -- F6 processing
   44-    1       F7     -- F7 processing
   45-    1       F8     -- F8 processing
   46-    1       F9     -- F9 processing
   47-    1       F10    -- F10 processing
   48-    1       F11    -- F11 processing
   49-    1       F12    -- F12 processing
   50-    1       F13    -- F13 processing
   51-    1       F14    -- F14 processing
   52-    1       F15    -- F15 processing
   53-    1       F16    -- F16 processing
   54-    1       F17    -- F17 processing
   55-    1       F18    -- F18 processing
   56-    1       F19    -- F19 processing
   57-    1       F20    -- F20 processing
   58-    1       KEYDOT -- Key "." processing
   59-    1       DOCTRL -- Process control characters
```

TSSLE -- TSX-Plus Single Line E MACRO V05.05 Wednesday 18-Jan-89 14:16
Table of contents

60- 1 ICPCR -- Carriage-return processing
61- 1 ICPLF -- Line-feed processing
62- 1 ICPBS -- Backspace processing
63- 1 ICPESC -- Escape processing
64- 1 ICPCTA -- Control-A processing
65- 1 ICPCTC -- Control-C processing
66- 1 ICPCTR -- Control-R processing
67- 1 ICPCTU -- Control-U processing
68- 1 ICPCTZ -- Control-Z processing
69- 1 ICPRUB -- Rubout processing
70- 1 ICPNUL -- Null processing
71- 1 INWAIT -- Wait for input characters
72- 1 SAVLIN -- Save current input line
73- 1 LINFIN -- Terminate input line
74- 1 DELLFT -- Delete character to left of cursor
75- 1 SLMVUP -- Move up to previous stored command
76- 1 SLMVDN -- Move down to previous stored command
77- 1 RSTLIN -- Restore a full line
79- 1 OSTRIK -- Overstrike
80- 1 INSERT -- Insert string into edit buffer
81- 1 PAINT -- Redisplay current edit line
82- 1 MAXLEN -- Compute maximum allowed line length
83- 1 COLCLC -- Calculate column position of a character
84- 1 CHRPOS -- Move cursor to a specific character
85- 1 CURPOS -- Move cursor to a specified column
86- 1 CSRRIT -- Generate control sequence to move cursor right
87- 1 CSRLFT -- Generate control sequence to move cursor left
88- 1 CHKDLM -- Check for word delimiters
89- 1 CVTLC -- Convert lower-case characters to upper-case
90- 1 ECOCTL -- Echo a control character
90- 27 RNGBEL -- Ring bell to signal error
90- 39 AKEYON -- Turn on alternate keypad mode
91- 1 ECHO   -- Send character to the terminal
92- 1 KEYCK -- Check to see if key is defined
93- 1 KEYSRC -- See if terminal key is defined by user
94- 1 KEYSUB -- Substitute a defined string for a key
95- 1 CHK52 -- Determine if terminal is a VT52

```
                                        .TITLE  TSSLE -- TSX-Plus Single Line Editor
                                        .ENABL  LC
                                        .ENABL  AMA
                                        .DSABL  GBL
                                        .CSECT  TSSLE
000000                          TSSLE:  .RAD50  /SLE/           ;Overlay region id
000000  074245                  ;
                                ;  TSSLE is the TSX-Plus system overlay module that contains routines
                                ;  to implement the single line editor.
                                ;
                                ;  Copyright (c) 1983, 1984, 1985, 1986, 1987, 1988.
                                ;  S&H Computer Systems, Inc.
                                ;  Nashville, Tennessee USA
                                ;  All rights reserved.
                                ;
                                ;  Global definitions
                                ;
                                        .GLOBL  GTSLCH
                                ;
                                ;  Global references
                                ;
                                        .GLOBL  SLOVER,SLRPTR,SLCYC1,SLCYC2,$scca,lsw5
                                        .GLOBL  $VBELL,$PWKEY,LSW11,LNPRIM,$V52EM,SLSPTR,SLDOWN
                                        .GLOBL  VVLSCH,$CTRLW,MAXSEC,DOSWIT,$1ESC,$WDISP,WINDSP
                                        .GLOBL  KT$LET,KT$GLT,$SLLET,LWINDO,VVPWCH,WINPRT
                                        .GLOBL  VKEYMX,KD$COD,KD$FLG,KD$TXT,KD$$SZ,KF$ECO,KF$TRM
                                        .GLOBL  KT$NRM,KT$GLD,KC$E1,KC$E2,KC$E3,KC$E4,KC$E5,KC$E6
                                        .GLOBL  KC$F6,KC$F7,KC$F8,KC$F9,KC$F10,KC$F11,KC$F12,KC$F13
                                        .GLOBL  KC$F14,KC$F15,KC$F16,KC$F17,KC$F18,KC$F19,KC$F20
                                        .GLOBL  KC$MIN,KC$ENT,KC$COM,KC$DOT,KC$PF1,KC$PF2,KC$PF3,KC$PF4
                                        .GLOBL  KC$KP0,KC$KP1,KC$KP2,KC$KP3,KC$KP4,KC$KP5,KC$KP6
                                        .GLOBL  KC$KP7,KC$KP8,KC$KP9,KC$LFT,KC$RIT,KC$UP,KC$DWN,VPAR6
                                        .GLOBL  SLCBUF,SIGWAT,LSW6,CTRLQ,LF,LFWLIM,LTRMTP,KEYPAR
                                        .GLOBL  LOGFLG,LF$IN,SLLBUF,LRBFIL,LCOL,INTPRI,LHIPCT
                                        .GLOBL  SLOPTR,SLECOL,LSPACT,BKSPAC,SLCX,VT52,TAB,$XSTOP
                                        .GLOBL  SLSCOL,SLCCOL,PR7,LSW3,STOP,LTTPAR,$NOIN,LOGCR,CHKABT
                                        .GLOBL  SLGOLD,VINTIO,LACTIV,QHDSPN,KPAR6,CR,PSW
                                        .GLOBL  LOGCHR,LNSPAC,S$INWT,SLEBUF,SLMXLN,BELL,LINPNT
                                        .GLOBL  OVRHC,$DBGMD,SLDBUF,LAFSIZ,DELCHR,$DISCN
                                        .GLOBL  ESC,QUECHR,$NOLF,$LC,LSW2,LCBIT,LJSW,$SLINI,LSW7
                                        .GLOBL  LSCCA,SLCR,$ECHO,LRTCHR,$SLKED,SLBACK,SLSBUF,LINCNT
                                        .GLOBL  VT100,LITIME,VQUAN1,LSTATE,S$TTFN,ENQTL,$SUCF,LSW9
                                        .GLOBL  $NOINT,SLLPTR,SLLEND,RUBOUT,$NDICP,LSW10
                                        .GLOBL  CSICHR,SS3CHR,SLCSBF,SLCSBX,SLCSPT,SLCSR,NEDCDI
                                        .GLOBL  $DETCH,LSW
                                ;
                                ;  Macro definitions:
                                ;
                                        .MACRO  DISABL          ;DISABLE INTERRUPTS
                                        BIS     #PR7,@#PSW
                                        .ENDM   DISABL

                                        .MACRO  ENABL           ;ENABLE INTERRUPTS
                                        BIC     INTPRI,@#PSW
                                        .ENDM   ENABL

                                        .MACRO  OCALL   ENTADD
```

```
                        .IF     B,ENTADD
                                .ERROR  ;OCALL without entry address
                        .ENDC
                        CALL    OVRHC
                        .WORD   ENTADD
                        .ENDM   OCALL
                ;
                ;  The TTMAP and TTMAPX macros are used to map kernel-mode par6 to the
                ;  terminal character buffer area.  The previous contents of par6 map
                ;  register are pushed on the stack and may be restored by using the
                ;  UNMAP or UNMAPX macros.
                ;  R1 must contain the line index number of the line whose buffers
                ;  are being accessed.
                ;  The difference between the TTMAP-UNMAP macros and the TTMAPX-UNMAPX
                ;  macros is that the X-versions are more efficient but may only be
                ;  used from within interrupt service routines where we are guaranteed
                ;  to be running on the system stack.
                ;  The TTMAP and UNMAP versions of the macros must only be
                ;  used in sections of code where the interrupts are disabled.
                ;
                        .MACRO  TTMAPX
                        MOV     LTTPAR(R1),@#KPAR6
                        .ENDM   TTMAPX
                ;
                        .MACRO  UNMAPX
                        .ENDM   UNMAPX
                ;
                        .MACRO  TTMAP
                        MOV     @#KPAR6,MAPHLD
                        MOV     LTTPAR(R1),@#KPAR6
                        .ENDM   TTMAP
                ;
                        .MACRO  UNMAP
                        MOV     MAPHLD,@#KPAR6
                        .ENDM   UNMAP
                ;
                ;  The KEYMAP macro is used to map KPAR6 to the local region that has
                ;  the user key definitions.  Use the KEYUMP macro to restore mapping.
                ;
                        .MACRO  KEYMAP
                        MOV     @#KPAR6,MAPHLD
                        DISABL
                        MOV     KEYPAR,@#KPAR6
                        .ENDM   KEYMAP
                ;
                        .MACRO  KEYUMP
                        MOV     MAPHLD,@#KPAR6
                        ENABL
                        .ENDM   KEYUMP
                ;
                ;  The KEYCHK macro is used to call the KEYSRC routine to see if a specific
                ;  key has been defined by the user.
                ;  The argument to KEYCHK is the key code.
                ;
                        .MACRO  KEYCHK  KEYCOD
                        JSR     R5,KEYCK
                        .WORD   KEYCOD
```

```
                                        .ENDM   KEYCHK
                                ;
                                ;  Data areas
                                ;
000002  000000                MAPHLD: .WORD   0               ;Temp cell used by TTMAP macro
```

```
                                        .SBTTL  GTSLCH -- Get next character from single-line editor
                                ;-----------------------------------------------------------------------
                                ;  GTSLCH is called to get from the single-line editor the next character
                                ;  to pass to the program.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
                                ;  Outputs:
                                ;   R0 = Next character to pass to program.
                                ;
000004  010246                  GTSLCH: MOV     R2,-(SP)
                                ;
                                ;  Perform initialization if this is the first call for this line
                                ;
000006  032761  000000G 000000G         BIT     #$SLINI,LSW7(R1);Have we done initialization for this line?
000014  001002                          BNE     1$              ;Br if yes
000016  004737  000176'                 CALL    SLINIT          ;Initialize for line
                                ;
                                ;  See if we have a complete line ready yet.
                                ;
000022  013702  000000G         1$:     MOV     SLOPTR,R2       ;Do we have any characters ready to go?
000026  001404                          BEQ     2$              ;Br if not
                                ;
                                ;  Get next output character
                                ;
000030  112200                          MOVB    (R2)+,R0        ;Get next output character
000032  001005                          BNE     3$              ;Br if got a character
                                ;
                                ;  We have finished passing the last line to the program.
                                ;  Initialize for a new line.
                                ;
000034  004737  000176'                 CALL    SLINIT          ;Initialize for a new line
                                ;
                                ;  There are no characters available to be passed to program.
                                ;  See if there are any pending input characters that need
                                ;  to be processed.
                                ;
000040  004737  000102'         2$:     CALL    SLINCH          ;Check for pending input characters
                                ;
                                ;  Now go back and see if we have any characters ready for output
                                ;
000044  000766                          BR      1$
                                ;
                                ;  We might have a character for the program
                                ;
000046  010237  000000G         3$:     MOV     R2,SLOPTR       ;Save updated output character pointer
                                ;
                                ;  See if we need to reset the "interactive" timer for this job
                                ;
000052  032761  000000G 000000G 5$:     BIT     #$NOINT,LSW7(R1)  ;Is program being run non-interactively?
000060  001006                          BNE     4$                ;Br if yes
000062  013761  000000G 000000G         MOV     VINTIO,LHIPCT(R1) ;Reset high-priority hit limit for job
000070  013761  000000G 000000G         MOV     VQUAN1,LITIME(R1) ;Reset interactive CPU time limit
                                ;
                                ;  Finished
                                ;
```

```
     58 000076  012602                    4$:     MOV     (SP)+,R2
     59 000100  000207                            RETURN
```

```
                                        .SBTTL  SLINCH -- See if any input characters need to be processed
                                ;-------------------------------------------------------------------------
                                ;  SLINCH is called to see if there are any characters pending in the
                                ;  terminal input buffer that need to be processed by the single
                                ;  line editor.
                                ;  This routine waits for an input character to be received, processes it,
                                ;  and then returns.
                                ;
                                ;  Inputs:
                                ;   R1 = Line index number.
                                ;
000102  010246                  SLINCH: MOV     R2,-(SP)
000104  010546                          MOV     R5,-(SP)
                                ;
                                ;  Abort the job if a detached job is trying to do input from the terminal
                                ;
000106  032761  000000G 000000G         BIT     #$DETCH,LSW(R1) ;Is this a detached job?
000114  001405                          BEQ     1$              ;Br if not
000116  052761  000000G 000000G         BIS     #$DISCN,LSW(R1) ;Abort the detached job
000124  004737  000000G                 CALL    STOP            ;Stop execution of job
                                ;
                                ;  See if there are any characters in the terminal input buffer
                                ;
000130  005761  000000G         1$:     TST     LINCNT(R1)      ;Are there any chars in input buffer?
000134  001003                          BNE     2$              ;Br if yes
                                ;
                                ;  There are no pending characters.
                                ;  Wait until we get one.
                                ;
000136  004737  007042'                 CALL    INWAIT          ;Wait for a character to arrive
000142  000772                          BR      1$              ;Go try again
                                ;
                                ;  There are pending input characters.
                                ;  Get the next character out of the input buffer.
                                ;
000144  016102  000000G         2$:     MOV     LINPNT(R1),R2   ;Get address of next character to get
000150                                  OCALL   DELCHR          ;Get character from buffer
000156  042700  177400                  BIC     #^C377,R0       ;Clear all but low order 8 bits of character
000162  010005                          MOV     R0,R5           ;Put character we got in R5
                                ;
                                ;  Process the character
                                ;
000164  004737  000350'                 CALL    PRCHAR          ;Process the character
                                ;
                                ;  Return to higher-level routine to see if we have a complete
                                ;  line ready yet.
                                ;
000170  012605                          MOV     (SP)+,R5
000172  012602                          MOV     (SP)+,R2
000174  000207                          RETURN
```

```
                                        .SBTTL  SLINIT -- Initialize for new get line
                                ;-------------------------------------------------------------------------
                                ;  SLINIT is called each time we begin to acquire a new input line.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
000176  010546                  SLINIT: MOV     R5,-(SP)
                                ;
                                ;  If SL is in KED mode, send control sequence to terminal to put
                                ;  numeric keypad into alternate mode.
                                ;
000200  032761  000000G 000000G         BIT     #$SLKED,LSW7(R1);Are we in KED mode?
000206  001402                          BEQ     1$              ;Br if not
000210  004737  011166'                 CALL    AKEYON          ;Turn on alternate keypad mode
                                ;
                                ;  Say no line is ready to be sent to program
                                ;
000214  005037  000000G         1$:     CLR     SLOPTR          ;No output line ready
                                ;
                                ;  Say Gold key (PF1) is not pending
                                ;
000220  105037  000000G                 CLRB    SLGOLD          ;Gold key not pending
                                ;
                                ;  Say carriage return not pending
                                ;
000224  105037  000000G                 CLRB    SLCR            ;Carriage return was not last character
                                ;
                                ;  Say we are not processing a terminal control sequence
                                ;
000230  005037  000000G                 CLR     SLCSR           ;Not processing terminal control sequence
                                ;
                                ;  Set direction to forward
                                ;
000234  105037  000000G                 CLRB    SLBACK          ;Set direction = forward
                                ;
                                ;  Say we are not in overstrike mode
                                ;
000240  105037  000000G                 CLRB    SLOVER          ;Not in overstrike mode
                                ;
                                ;  Clear edit buffer and initialize cursor to point to 1st character
                                ;
000244  105037  000000G                 CLRB    SLEBUF          ;Store null as 1st char in edit buffer
000250  012737  000000G 000000G         MOV     #SLEBUF,SLCX    ;Say cursor is pointing to 1st character
                                ;
                                ;  Set up cursor display position for left end of line
                                ;
000256  116100  000000G                 MOVB    LCOL(R1),R0     ;Get column number of start of field
000262  010037  000000G                 MOV     R0,SLSCOL       ;Set this as start-of-line column
000266  010037  000000G                 MOV     R0,SLECOL       ;Set this as end-of-line column
000272  010037  000000G                 MOV     R0,SLCCOL       ;Set as cursor display column
                                ;
                                ;  Initialize to point to most recent saved line
                                ;
000276  005737  000000G                 TST     SLSPTR          ;Has save-line pointer been initialized?
000302  001003                          BNE     2$              ;Br if yes
000304  012737  000000G 000000G         MOV     #SLLBUF,SLSPTR  ;Most recently saved line position
```

```
     58 000312  013737  000000G 000000G 2$:     MOV     SLSPTR,SLLPTR   ;Set up-arrow pointer to most recent line
     59                                 ;
     60                                 ;  Set flag saying initialization has been done for this line
     61                                 ;
     62 000320  052761  000000G 000000G        BIS     #$SLINI,LSW7(R1);Set initialization-done flag
     63                                 ;
     64                                 ;  See if we need to recall a line
     65                                 ;
     66 000326  013705  000000G                 MOV     SLRPTR,R5       ;Is a recall pending?
     67 000332  001404                          BEQ     9$              ;Br if not
     68 000334  004737  007704'                 CALL    RSTLIN          ;Restore the line
     69 000340  005037  000000G                 CLR     SLRPTR          ;Say recall has been done
     70                                 ;
     71                                 ;  Finished
     72                                 ;
     73 000344  012605                  9$:     MOV     (SP)+,R5
     74 000346  000207                          RETURN
```

```
                                ;-----------------------------------------------------------------------
                                ;  Process a character received from the terminal.
                                ;
                                ;  Inputs:
                                ;   R1 = Job's virtual line index number.
                                ;   R5 = Received character.
                                ;
000350  010246                  PRCHAR: MOV     R2,-(SP)
000352  010346                          MOV     R3,-(SP)
000354  010546                          MOV     R5,-(SP)
                                ;
                                ;  Determine if this character begins a terminal control sequence
                                ;
000356  004737  001144'                 CALL    CSCHK           ;Check for control sequence start
000362  103026                          BCC     9$              ;Br if start of control sequence
                                ;
                                ;  See if we are currently accepting a terminal control sequence
                                ;
000364  013700  000000G                 MOV     SLCSR,R0        ;Are we processing a control sequence?
000370  001402                          BEQ     3$              ;Br if not
000372  004710                          CALL    (R0)            ;Call routine to process this character
000374  000421                          BR      9$              ;Finished with this character
                                ;
                                ;  Determine if this is a user-defined key
                                ;
000376  005737  000000G         3$:     TST     KEYPAR          ;Are there any user-defined keys?
000402  001414                          BEQ     5$              ;Br if not
000404  010500                          MOV     R5,R0           ;Get the character
000406  105737  000000G                 TSTB    SLGOLD          ;Was gold key pressed?
000412  001403                          BEQ     8$              ;Br if not
000414  052700  000000C                 BIS     #KT$GLT*400,R0  ;Say this is a gold letter
000420  000402                          BR      2$
000422  052700  000000C         8$:     BIS     #KT$LET*400,R0  ;Say this is a regular letter
000426  004737  011272'         2$:     CALL    KEYSRC          ;See if key is user defined
000432  103402                          BCS     9$              ;Br if this is a user-defined key
                                ;
                                ;  Process ordinary characters that are not part of control sequences
                                ;  and that are not user-defined keys.
                                ;
000434  004737  000450'         5$:     CALL    ORDCHR          ;Process an ordinary character
                                ;
                                ;  Finished
                                ;
000440  012605                  9$:     MOV     (SP)+,R5
000442  012603                          MOV     (SP)+,R3
000444  012602                          MOV     (SP)+,R2
000446  000207                          RETURN
```

```
                                        .SBTTL  ORDCHR -- Process ordinary characters
                                ;-----------------------------------------------------------------
                                ;  Process ordinary characters (both control and non-control).
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number
                                ;   R5 = Received character
                                ;
000450                          ORDCHR:
                                ;
                                ;  See if Gold key was pressed
                                ;
000450  105737  0000000G              TSTB    SLGOLD          ;Was gold key pressed?
000454  001426                       BEQ     1$              ;Br if not
                                ;
                                ;  Gold key was pressed.  Check for Gold-S.
                                ;
000456  120527  000123               CMPB    R5,#'S          ;Gold-S?
000462  001403                       BEQ     2$              ;Br if yes
000464  120527  000163               CMPB    R5,#'s          ;Gold-s?
000470  001005                       BNE     3$              ;Br if not
000472  105037  0000000G       2$:     CLRB    SLGOLD          ;Reset Gold flag
000476  004737  004462'               CALL    KBS             ;Gold-S ==> Save command
000502  000443                       BR      9$
                                ;
                                ;  Check for Gold-X.
                                ;
000504  120527  000130         3$:     CMPB    R5,#'X          ;Gold-X?
000510  001403                       BEQ     10$             ;Br if yes
000512  120527  000170               CMPB    R5,#'x          ;Gold-x?
000516  001005                       BNE     1$              ;Br if not
000520  105037  0000000G       10$:    CLRB    SLGOLD          ;Reset Gold flag
000524  004737  004514'               CALL    KBX             ;Gold-X ==> Recall command
000530  000430                       BR      9$
                                ;
                                ;  If debugger is in control, bypass some special character processing
                                ;
000532  032761  0000000G 0000000G 1$:     BIT     #$DBGMD,LSW6(R1);Is debugging program in control?
000540  001006                       BNE     6$              ;If yes then bypass some checking
                                ;
                                ;  See if this is a user-defined activation character
                                ;
000542  004737  000774'               CALL    CKUAC           ;See if this is a user-defined activation char
000546  103021                       BCC     9$              ;Br if it is a user-defined activation char
                                ;
                                ;  See if this is a read-time activation character
                                ;
000550  004737  001074'               CALL    CKRDTM          ;See if this is a read time-out character
000554  103016                       BCC     9$              ;Br if it is
                                ;
                                ;  Determine if this is a normal or control character
                                ;
000556  020527  000037         6$:     CMP     R5,#37          ;Is this a normal or control char?
000562  101003                       BHI     4$              ;Br if not control character
000564  004737  005406'               CALL    DOCTRL          ;Process a control character
000570  000410                       BR      9$
000572  120527  0000000G       4$:     CMPB    R5,#RUBOUT      ;Is this a rubout character?
```

```
     58 000576  001003                          BNE     7$              ;Br if not
     59 000600  004737  006764'                 CALL    ICPRUB          ;Process rubout character
     60 000604  000402                          BR      9$
     61 000606  004737  000614'       7$:      CALL    REGCHR          ;Process a normal character
     62                                ;
     63                                ;  Finished
     64                                ;
     65 000612  000207                 9$:      RETURN
```

```
                                        .SBTTL  REGCHR -- Process normal characters
                                ;-----------------------------------------------------------------
                                ;  Process normal (non-control) characters.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line index number.
                                ;   R5 = Current input character.
                                ;
000614  010246                  REGCHR: MOV     R2,-(SP)
000616  010346                          MOV     R3,-(SP)
000620  010446                          MOV     R4,-(SP)
000622  010546                          MOV     R5,-(SP)
                                ;
                                ;  See if this is part of a request to switch to a virtual line
                                ;
000624  032761  000000G 000000G         BIT     #$CTRLW,LSW3(R1);Was ctrl-W the last character?
000632  001427                          BEQ     1$              ;Br if not
000634  042761  000000G 000000G         BIC     #$CTRLW,LSW3(R1);Say ctrl-W no longer the last char
000642  162705  000060                  SUB     #'0,R5          ;Convert digit to binary value
000646  002445                          BLT     9$              ;Br if char after ctrl-W not a digit
000650  020527  000000G                 CMP     R5,#MAXSEC      ;Don't exceed max line # allowed
000654  003042                          BGT     9$              ;Br if too big
000656                                  OCALL   DOSWIT          ;Switch to a virtual line
000664  052761  000000G 000000G         BIS     #$VBELL,LSW9(R1);Suppress bell ring on next SL read
000672  032761  000000G 000000G         BIT     #$WDISP,LSW6(R1);Do we need to redisplay our window?
000700  001430                          BEQ     9$              ;Br if not
000702                                  OCALL   WINDSP          ;Redisplay window for job
000710  000424                          BR      9$
                                ;
                                ;  Translate character to upper case
                                ;
000712  110500                  1$:     MOVB    R5,R0           ;Get character
000714  004737  011066'                 CALL    CVTLC           ;Convert lower-case to upper-case
                                ;
                                ;  Insert character into buffer
                                ;
000720  105737  000000G                 TSTB    SLOVER          ;Are we in overstrike mode?
000724  001403                          BEQ     2$              ;Br if not
000726  004737  010066'                 CALL    OSTRIK          ;Overstrike
000732  000405                          BR      3$
000734  012702  000000G         2$:     MOV     #SLCBUF,R2      ;Point to character buffer
000740  110012                          MOVB    R0,(R2)         ;Store char into insert buffer
000742  004737  010140'                 CALL    INSERT          ;Insert character to left of cursor
                                ;
                                ;  Say there is no deleted character being held and not in terminal
                                ;  control sequence.
                                ;
000746  105037  000000G         3$:     CLRB    SLCBUF          ;No deleted character being held
000752  005037  000000G                 CLR     SLCSR           ;We are not in a terminal control sequence
000756  105037  000000G                 CLRB    SLGOLD          ;Gold key (PF1) has not been pressed
                                ;
                                ;  Finished
                                ;
000762  012605                  9$:     MOV     (SP)+,R5
000764  012604                          MOV     (SP)+,R4
000766  012603                          MOV     (SP)+,R3
000770  012602                          MOV     (SP)+,R2
```

```
     58 000772  000207                              RETURN
```

```
                                        .SBTTL  CKUAC  -- Check for user-defined activation characters
                                ;-----------------------------------------------------------------------
                                ;  CKUAC is called to determine if the current input character is a
                                ;  user-defined activation character.  If it is, it is processed here.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R5 = Current input character
                                ;
                                ;  Outputs:
                                ;   C-flag cleared ==> This is a user-defined activation character.
                                ;   C-flag set     ==> This is not a user-defined activation char.
                                ;
000774  010246                  CKUAC:  MOV     R2,-(SP)
000776  010346                          MOV     R3,-(SP)
                                ;
                                ;  See if there are any user-defined activation characters
                                ;
001000  016102  000000G                 MOV     LNSPAC(R1),R2   ;Get number of user-defined activation chars
001004  001427                          BEQ     8$              ;Br if there are none
                                ;
                                ;  There are some user-defined activation characters.
                                ;  Translate input character to upper-case if that is wanted.
                                ;
001006  010500                          MOV     R5,R0           ;Get current character
001010  004737  011066'                 CALL    CVTLC           ;Convert to upper-case if needed
                                ;
                                ;  Search for input character in table of activation characters.
                                ;
001014  016103  000000G                 MOV     LSPACT(R1),R3   ;Get pointer to table of activation chars
001020  120023                  1$:     CMPB    R0,(R3)+        ;Is this an activation character?
001022  001402                          BEQ     2$              ;Br if yes
001024  077203                          SOB     R2,1$           ;Loop if more to check
001026  000416                          BR      8$              ;This is not a user-defined activation char
                                ;
                                ;  This character is a user-defined activation character.
                                ;  Store character at end of line and signal line-completed.
                                ;
001030  010005                  2$:     MOV     R0,R5           ;Get converted character to R5
001032  004737  007172'                 CALL    SAVLIN          ;Save input line before storing activation chr
001036  013703  000000G                 MOV     SLCX,R3         ;Point to cursor character
001042  105723                  3$:     TSTB    (R3)+           ;Search for null at end of line
001044  001376                          BNE     3$              ;Loop till null found
001046  005303                          DEC     R3              ;Point to the null character
001050  110523                          MOVB    R5,(R3)+        ;Store activation char as last char in field
001052  105023                          CLRB    (R3)+           ;Store null at end
001054  004737  007346'                 CALL    LINFIN          ;Say input line is finished
001060  000241                          CLC                     ;Say this is an activation character
001062  000401                          BR      9$
                                ;
                                ;  Character is not a user-defined activation character.
                                ;
001064  000261                  8$:     SEC                     ;Say this is not a user-defined activation chr
                                ;
                                ;  Finished
                                ;
001066  012603                  9$:     MOV     (SP)+,R3
```

```
     58 001070  012602                          MOV     (SP)+,R2
     59 001072  000207                          RETURN
```

```
 1                                                      .SBTTL  CKRDTM -- Check for read time-out character
 2                                              ;-----------------------------------------------------------------------
 3                                              ;  CKRDTM is called to determine if the current input character is a
 4                                              ;  read time-out character.  If it is, the current input line is terminated.
 5                                              ;
 6                                              ;  Inputs:
 7                                              ;   R1 = Job index number.
 8                                              ;   R5 = Current input character.
 9                                              ;
10                                              ;  Outputs:
11                                              ;   C-flag cleared ==> This is a read time-out character.
12                                              ;
13 001074                                      CKRDTM:
14                                              ;
15                                              ;  See if program has specified a read time-out
16                                              ;
17 001074  016100  0000006                             MOV     LRTCHR(R1),R0   ;Is there a read time-out specified?
18 001100  001417                                      BEQ     8$              ;Br if not
19                                              ;
20                                              ;  See if current character is time-out character
21                                              ;
22 001102  120500                                      CMPB    R5,R0           ;Is current character the time-out char?
23 001104  001015                                      BNE     8$              ;Br if not
24                                              ;
25                                              ;  Read time-out has occurred.
26                                              ;  Terminate current input line.
27                                              ;
28 001106  013700  0000006                             MOV     SLCX,R0         ;Get current cursor character index
29 001112  105720                              1$:     TSTB    (R0)+           ;Search for end of line
30 001114  001376                                      BNE     1$              ;Loop till null hit
31 001116  005300                                      DEC     R0              ;Point to null
32 001120  110520                                      MOVB    R5,(R0)+        ;Store time-out character at end
33 001122  105010                                      CLRB    (R0)            ;Put null at end of string
34 001124  004737  007346'                            CALL    LINFIN          ;Terminate the input line
35                                              ;
36                                              ;  Clear the time-out character flag
37                                              ;
38 001130  005061  0000006                             CLR     LRTCHR(R1)      ;Say we have processed the time-out character
39 001134  000241                                      CLC                     ;Say this was the timeout character
40 001136  000401                                      BR      9$
41                                              ;
42                                              ;  This is not a read time-out character
43                                              ;
44 001140  000261                              8$:     SEC                     ;Signal not time-out character
45                                              ;
46                                              ;  Finished
47                                              ;
48 001142  000207                              9$:     RETURN
```

```
                                        .SBTTL  CSCHK  -- Check for start of control sequence
                                ;-------------------------------------------------------------------------
                                ;  CSCHK is called to determine if the current character begins a
                                ;  terminal control sequence.
                                ;
                                ;  Inputs:
                                ;   R1 = Line index number.
                                ;   R5 = Received character.
                                ;
                                ;  Outputs:
                                ;   C-flag cleared ==> Start of control sequence.
                                ;   C-flag set     ==> Not start of control sequence.
                                ;
001144                          CSCHK:
                                ;
                                ;  See if this is the start of an escape sequence
                                ;
001144  120527  0000000                 CMPB    R5,#ESC         ;Is this character escape?
001150  001003                          BNE     1$              ;Br if not
001152  004737  006274'                 CALL    ICPESC          ;Begin escape sequence
001156  000413                          BR      8$
                                ;
                                ;  See if this character is CSI, beginning of VT200 control sequence
                                ;
001160  120527  0000000         1$:     CMPB    R5,#CSICHR      ;CSI character?
001164  001003                          BNE     2$              ;Br if not
001166  004737  006326'                 CALL    ICPCSI          ;Start VT200 control sequence
001172  000405                          BR      8$
                                ;
                                ;  See if this character is SS3, beginning of VT200 control sequence
                                ;
001174  120527  0000000         2$:     CMPB    R5,#SS3CHR      ;SS3 character?
001200  001004                          BNE     7$              ;Br if not
001202  004737  006352'                 CALL    ICPSS3          ;Start VT200 control sequence
                                ;
                                ;  We began a control sequence
                                ;
001206  000241                  8$:     CLC                     ;Signal control sequence began
001210  000401                          BR      9$
                                ;
                                ;  We did not begin a control sequence
                                ;
001212  000261                  7$:     SEC                     ;Signal that control sequence did not begin
                                ;
                                ;  Finished
                                ;
001214  000207                  9$:     RETURN
```

```
                                        .SBTTL  EPCH1  -- Accrue a terminal control sequence
                                ;
                                ;  The routines in this section implement a finite state machine
                                ;  which accrues the characters of a terminal control sequence.
                                ;  While we are accruing a control sequence, the cell SLCSR contains
                                ;  the address of the routine to be called to process the next character.
                                ;
                                ;-------------------------------------------------------------------------
                                ;  EPCH1 is called when we receive the first character following escape
                                ;  in a VT100 escape sequence.
                                ;
001216  013700  000000G         EPCH1:  MOV     SLCSPT,R0       ;Get pointer into buffer
001222  110520                  MOVB    R5,(R0)+        ;Accrue the character
001224  010037  000000G         MOV     R0,SLCSPT       ;Save new pointer
001230  012737  001304' 000000G MOV     #EPCH2,SLCSR    ;Set address of routine for next character
001236  000207                  RETURN
                                ;-------------------------------------------------------------------------
                                ;  EP52 is called when we receive the first character following escape
                                ;  in a VT52 escape sequence.
                                ;
001240  013700  000000G         EP52:   MOV     SLCSPT,R0       ;Get pointer into buffer
                                ;
                                ;  If this character is "?" then we are receiving a two character
                                ;  control sequence. Otherwise this is a single character control sequence.
                                ;
001244  120527  000077                  CMPB    R5,#'?          ;Alternate keypad leadin character?
001250  001010                  BNE     1$              ;Br if not
                                ;
                                ;  This is a two character control sequence.
                                ;  Convert "?" to "O" to make the sequence compatible with a VT100.
                                ;
001252  112720  000117                  MOVB    #'O,(R0)+       ;Store "O" as 1st char of sequence
001256  010037  000000G                 MOV     R0,SLCSPT       ;Store new buffer pointer
001262  012737  001304' 000000G         MOV     #EPCH2,SLCSR    ;Set address of routine to process more chars
001270  000404                  BR      9$
                                ;
                                ;  This is a single character control sequence
                                ;
001272  110520                  1$:     MOVB    R5,(R0)+        ;Store character into buffer
001274  105020                  CLRB    (R0)+           ;This terminates the control sequence
001276  004737  001356'                 CALL    CSFIN           ;End of control sequence
                                ;
                                ;  Finished
                                ;
001302  000207                  9$:     RETURN

                                ;-------------------------------------------------------------------------
                                ;  EPCH2 process characters after the 1st character of a control sequence.
                                ;
001304  013700  000000G         EPCH2:  MOV     SLCSPT,R0       ;Get pointer to control sequence buffer
                                ;
                                ;  Make sure we are not about to overflow the control sequence buffer
                                ;
001310  020027  000000G                 CMP     R0,#SLCSBX      ;Is buffer full?
001314  103405                  BLO     1$              ;Br if not
001316  004737  011154'                 CALL    RNGBEL          ;Ring the bell
```

```
     58 001322  005037  000000G                 CLR     SLCSR           ;Say not processing a control sequence
     59 001326  000412                          BR      9$
     60                                 ;
     61                                 ;  Store character into control sequence buffer
     62                                 ;
     63 001330  110520                  1$:     MOVB    R5,(R0)+        ;Store character into control sequence buffer
     64                                 ;
     65                                 ;  Determine if this is the terminating character of the control sequence
     66                                 ;
     67 001332  120527  000100                  CMPB    R5,#100         ;Is this the final char of control seq?
     68 001336  103404                          BLO     2$              ;Br if not
     69 001340  105020                          CLRB    (R0)+           ;Terminate the control sequence
     70 001342  004737  001356'                 CALL    CSFIN           ;Process the control sequence
     71 001346  000402                          BR      9$
     72                                 ;
     73                                 ;  This is not the final character of the control sequence.
     74                                 ;  Continue accruing characters.
     75                                 ;
     76 001350  010037  000000G         2$:     MOV     R0,SLCSPT       ;Save new buffer pointer
     77                                 ;
     78                                 ;  Finished
     79                                 ;
     80 001354  000207                  9$:     RETURN
```

```
                                        .SBTTL  CSFIN  -- Process terminal control sequence
                                ;-------------------------------------------------------------------
                                ;  CSFIN is called when we have finished accruing a terminal control
                                ;  sequence.
                                ;
                                ;  Inputs:
                                ;   R1 = Line index number.
                                ;   SLCSBF = Control sequence in asciz form.
                                ;
001356  010246                  CSFIN:  MOV     R2,-(SP)
                                ;
                                ;  Say we are no longer processing a control sequence
                                ;
001360  005037  000000G                 CLR     SLCSR           ;No longer accruing a control sequence
                                ;
                                ;  Begin loop to search for the control sequence
                                ;
001364  012705  001456'                 MOV     #CSTBL,R5       ;Get pointer to control sequence table
                                ;
                                ;  Compare accrued control sequence with sequence stored in table
                                ;
001370  012702  000000G         1$:     MOV     #SLCSBF,R2      ;Point to accrued control sequence
001374  122225                  2$:     CMPB    (R2)+,(R5)+     ;Compare the strings
001376  001004                          BNE     3$              ;Br if mismatch
001400  105765  177777                  TSTB    -1(R5)          ;Was that the last char of both strings?
001404  001373                          BNE     2$              ;Br if not
001406  000415                          BR      4$              ;We have a match
                                ;
                                ;  Strings do not match -- Move on to next table entry
                                ;
001410  005305                  3$:     DEC     R5              ;Point to last char compared in table
001412  105725                  5$:     TSTB    (R5)+           ;Search for null at end of asciz string
001414  001376                          BNE     5$
001416  062705  000003                  ADD     #3,R5           ;Bound up and skip over following word
001422  042705  000001                  BIC     #1,R5           ;Get to word boundary
001426  020527  002206'                 CMP     R5,#CSEND       ;Checked all entries in table?
001432  103756                          BLO     1$              ;Loop if not
                                ;
                                ;  We could not find the sequence in our table
                                ;
001434  004737  011154'                 CALL    RNGBEL          ;Ring bell
001440  000404                          BR      9$
                                ;
                                ;  We found control sequence in our table.
                                ;  Call processing routine.
                                ;
001442  005205                  4$:     INC     R5              ;Bound up to next word following string
001444  042705  000001                  BIC     #1,R5           ;Get to word boundary
001450  004735                          CALL    @(R5)+          ;Call processing routine
                                ;
                                ;  Finished
                                ;
001452  012602                  9$:     MOV     (SP)+,R2
001454  000207                          RETURN
```

```
                                ;
                                ;  Table of terminal control sequences and associated processing routines
                                ;
                                        .MACRO  ESCSEQ  STRING,RTN
                                        .ASCIZ  /STRING/
                                        .EVEN
                                        .WORD   RTN
                                        .ENDM   ESCSEQ
                                ;
001456                          CSTBL:
                                        ESCSEQ  <[A>,TCUP               ;Up arrow
                                        ESCSEQ  <[B>,TCDOWN             ;Down arrow
                                        ESCSEQ  <[C>,TCRITE             ;Right arrow
                                        ESCSEQ  <[D>,TCLEFT             ;Left arrow
                                        ESCSEQ  <OM>,TCENTR             ;Enter (treat like carriage return)
                                        ESCSEQ  <OP>,TCPF1              ;PF1
                                        ESCSEQ  <OQ>,TCPF2              ;PF2
                                        ESCSEQ  <OR>,TCPF3              ;PF3
                                        ESCSEQ  <OS>,TCPF4              ;PF4
                                        ESCSEQ  <Ol>,KEYCOM             ;Comma
                                        ESCSEQ  <Om>,KEYMIN             ;Minus sign
                                        ESCSEQ  <On>,KEYDOT             ;Period
                                        ESCSEQ  <Op>,KEY0               ;0
                                        ESCSEQ  <Oq>,KEY1               ;1
                                        ESCSEQ  <Or>,KEY2               ;2
                                        ESCSEQ  <Os>,KEY3               ;3
                                        ESCSEQ  <Ot>,KEY4               ;4
                                        ESCSEQ  <Ou>,KEY5               ;5
                                        ESCSEQ  <Ov>,KEY6               ;6
                                        ESCSEQ  <Ow>,KEY7               ;7
                                        ESCSEQ  <Ox>,KEY8               ;8
                                        ESCSEQ  <Oy>,KEY9               ;9
                                        ESCSEQ  <[1~>,E1                ;E1  -- Find
                                        ESCSEQ  <[2~>,E2                ;E2  -- Insert here
                                        ESCSEQ  <[3~>,E3                ;E3  -- Remove
                                        ESCSEQ  <[4~>,E4                ;E4  -- Select
                                        ESCSEQ  <[5~>,E5                ;E5  -- Prev screen
                                        ESCSEQ  <[6~>,E6                ;E6  -- Next screen
                                        ESCSEQ  <[17~>,F6               ;F6
                                        ESCSEQ  <[18~>,F7               ;F7
                                        ESCSEQ  <[19~>,F8               ;F8
                                        ESCSEQ  <[20~>,F9               ;F9
                                        ESCSEQ  <[21~>,F10              ;F10
                                        ESCSEQ  <[23~>,F11              ;F11 -- ESC
                                        ESCSEQ  <[24~>,F12              ;F12 -- BS
                                        ESCSEQ  <[25~>,F13              ;F13 -- Line feed
                                        ESCSEQ  <[26~>,F14              ;F14
                                        ESCSEQ  <[28~>,F15              ;F15 -- Help
                                        ESCSEQ  <[29~>,F16              ;F16 -- Do
                                        ESCSEQ  <[31~>,F17              ;F17
                                        ESCSEQ  <[32~>,F18              ;F18
                                        ESCSEQ  <[33~>,F19              ;F19
                                        ESCSEQ  <[34~>,F20              ;F20
                                        ESCSEQ  <OA>,TCUP               ;Up arrow     (Cursor key mode set)
                                        ESCSEQ  <OB>,TCDOWN             ;Down arrow   (Cursor key mode set)
                                        ESCSEQ  <OC>,TCRITE             ;Right arrow  (Cursor key mode set)
                                        ESCSEQ  <OD>,TCLEFT             ;Left arrow   (Cursor key mode set)
```

```
     58 002146                                  ESCSEQ  <A>,TCUP              ;Up arrow      (VT52)
     59 002152                                  ESCSEQ  <B>,TCDOWN            ;Down arrow    (VT52)
     60 002156                                  ESCSEQ  <C>,TCRITE            ;Right arrow   (VT52)
     61 002162                                  ESCSEQ  <D>,TCLEFT            ;Left arrow    (VT52)
     62 002166                                  ESCSEQ  <P>,TCPF1             ;PF1           (VT52 numeric)
     63 002172                                  ESCSEQ  <Q>,TCPF2             ;PF2           (VT52 numeric)
     64 002176                                  ESCSEQ  <R>,TCPF3             ;PF3           (VT52 numeric)
     65 002202                                  ESCSEQ  <S>,TCPF4             ;PF4           (VT52 numeric)
     66 002206                          CSEND:
```

```
                                        .SBTTL  TCUP    -- Up-arrow processing
                                ;-----------------------------------------------------------------------
                                ;  Process the up-arrow key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
002206  010546                 TCUP:   MOV     R5,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
002210                                  KEYCHK  KC$UP           ;See if this is a user-defined key
002216  103450                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  If gold key was pressed recall through cycle
                                ;
002220  105737  0000000G                TSTB    SLGOLD          ;Was gold key pressed?
002224  001426                         BEQ     3$              ;Br if not
002226  105037  0000000G                CLRB    SLGOLD          ;Clear gold key flag
002232  005737  0000000G                TST     SLCYC1          ;Is a cycle defined?
002236  001421                         BEQ     3$              ;Br if not
002240  013705  0000000G                MOV     SLLPTR,R5       ;Point to last recalled line
002244  020537  0000000G                CMP     R5,SLCYC2       ;Reached end of the cycle?
002250  001003                         BNE     4$              ;Br if not
002252  013705  0000000G                MOV     SLCYC1,R5       ;Loop back to start of cycle
002256  000402                         BR      5$
002260  004737  007614'         4$:     CALL    SLMVDN          ;Move to next saved line
002264  010537  0000000G         5$:     MOV     R5,SLLPTR       ;Remember last recalled line
002270  010537  0000000G                MOV     R5,SLSPTR       ;Say this is last saved line
002274  004737  007704'                 CALL    RSTLIN          ;Restore the line
002300  000417                         BR      9$
                                ;
                                ;  Get pointer to saved line
                                ;
002302  013705  0000000G        3$:     MOV     SLLPTR,R5       ;Point to saved-line buffer
                                ;
                                ;  If last recall was with down arrow, skip over a command
                                ;
002306  105737  0000000G                TSTB    SLDOWN          ;Last operation down arrow?
002312  001404                         BEQ     1$              ;Br if not
002314  105037  0000000G                CLRB    SLDOWN          ;Reset direction indicator
002320  004737  007526'                 CALL    SLMVUP          ;Move up 1 line
                                ;
                                ;  Restore the saved line
                                ;
002324  004737  007704'         1$:     CALL    RSTLIN          ;Restore the line
                                ;
                                ;  Update up-arrow pointer to point to next saved line
                                ;
002330  004737  007526'                 CALL    SLMVUP          ;Move up to next line
002334  010537  0000000G                MOV     R5,SLLPTR       ;Save pointer for next up-arrow
                                ;
                                ;  Finished
                                ;
002340  012605                  9$:     MOV     (SP)+,R5
002342  000207                         RETURN
```

```
                                        .SBTTL  TCDOWN -- Down-arrow processing
                                ;-----------------------------------------------------------------------
                                ;  Process the down-arrow key -- Move forward to next saved command.
                                ;
002344  010446                 TCDOWN: MOV     R4,-(SP)
002346  010546                         MOV     R5,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
002350                                  KEYCHK  KC$DWN          ;See if this is a user-defined key
002356  103454                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  See if Gold key was pressed
                                ;
002360  105737  000000G                 TSTB    SLGOLD          ;Was gold key pressed?
002364  001423                         BEQ     3$              ;Br if not
                                ;
                                ;  Gold-down-arrow -- Set recall cycle
                                ;
002366  105037  000000G                 CLRB    SLGOLD          ;Reset Gold flag
002372  013705  000000G                 MOV     SLLPTR,R5       ;Point to last recalled command
002376  105737  000000G                 TSTB    SLDOWN          ;Was last direction down?
002402  001002                         BNE     4$              ;Br if yes
002404  004737  007614'                 CALL    SLMVDN          ;Move down a line
002410  010537  000000G         4$:     MOV     R5,SLCYC1       ;Set pointer to 1st command in cycle
002414  013737  000000G 000000G         MOV     SLSPTR,SLCYC2   ;Save pointer to last command in cycle
002422  010537  000000G                 MOV     R5,SLSPTR       ;Say this is the last command entered
002426  010537  000000G                 MOV     R5,SLLPTR       ;Remember last recalled line
002432  000426                         BR      9$
                                ;
                                ;  If last recalled command was done with up-arrow skip over that command
                                ;
002434  013705  000000G         3$:     MOV     SLLPTR,R5       ;Point to last saved command
002440  105737  000000G                 TSTB    SLDOWN          ;Was last direction down?
002444  001005                         BNE     1$              ;Br if yes
002446  112737  000001  000000G        MOVB    #1,SLDOWN       ;We are going down now
002454  004737  007614'                 CALL    SLMVDN          ;Move down a command line
                                ;
                                ;  See if there is a command to recall
                                ;
002460  020537  000000G         1$:     CMP     R5,SLSPTR       ;Any commands left to recall?
002464  001003                         BNE     2$              ;Br if yes
002466  004737  011154'                 CALL    RNGBEL          ;Ring the bell
002472  000406                         BR      9$
                                ;
                                ;  Advance down to the next saved command
                                ;
002474  004737  007614'         2$:     CALL    SLMVDN          ;Move down to next saved command
                                ;
                                ;  Store new current line pointer
                                ;
002500  010537  000000G                 MOV     R5,SLLPTR       ;Points to last line recalled
                                ;
                                ;  Restore the saved line
                                ;
002504  004737  007704'                 CALL    RSTLIN          ;Restore line pointed to by R5
                                ;
```

```
                                        ;  Finished
                                        ;
002510  012605                  9$:       MOV     (SP)+,R5
002512  012604                            MOV     (SP)+,R4
002514  000207                            RETURN
```

```
                                        .SBTTL  TCLEFT -- Left-arrow processing
                                ;-----------------------------------------------------------------------
                                ;  Process left-arrow key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
002516  010446                  TCLEFT: MOV     R4,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
002520                                  KEYCHK  KC$LFT          ;See if this is a user-defined key
002526  103422                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  If Gold key was pressed, move to left end of buffer
                                ;
002530  105737  0000000G                TSTB    SLGOLD          ;Was Gold key pressed?
002534  001407                          BEQ     2$              ;Br if not
002536  012704  0000000G                MOV     #SLEBUF,R4      ;Position to left end of buffer
002542  004737  010642'                 CALL    CHRPOS          ;Position cursor
002546  105037  0000000G                CLRB    SLGOLD          ;Clear gold-key flag
002552  000410                          BR      9$
                                ;
                                ;  Gold key was not pressed -- Move cursor left one character.
                                ;  Error if already at left margin.
                                ;
002554  013704  0000000G        2$:     MOV     SLCX,R4         ;Get cursor position index
002560  020427  0000000G                CMP     R4,#SLEBUF      ;Are we at left margin now?
002564  101403                          BLOS    9$              ;Br if yes -- Nothing to do
                                ;
                                ;  Move cursor left 1 character
                                ;
002566  005304                  1$:     DEC     R4              ;Move character pointer back 1 character
002570  004737  010642'                 CALL    CHRPOS          ;Position cursor correctly
                                ;
                                ;  Finished
                                ;
002574  012604                  9$:     MOV     (SP)+,R4
002576  000207                          RETURN
```

```
 1                                              .SBTTL  TCRITE -- Right-arrow processing
 2                                      ;-----------------------------------------------------------------
 3                                      ;  Process the right arrow key.
 4                                      ;
 5                                      ;  Inputs:
 6                                      ;   R1 = Job index.
 7                                      ;
 8 002600  010446                       TCRITE: MOV     R4,-(SP)
 9                                      ;
10                                      ;  See if this is a user-defined key
11                                      ;
12 002602                                      KEYCHK  KC$RIT          ;See if this is a user-defined key
13 002610  103424                              BCS     9$              ;Br if user-defined key
14                                      ;
15                                      ;  If Gold key was pressed, move to right end of line.
16                                      ;
17 002612  105737  000000G                     TSTB    SLGOLD          ;Was gold key pressed?
18 002616  001412                              BEQ     1$              ;Br if not
19 002620  013704  000000G                     MOV     SLCX,R4         ;Get current cursor pointer
20 002624  105724                       3$:    TSTB    (R4)+           ;Search for right end of line
21 002626  001376                              BNE     3$              ;Loop till null hit
22 002630  005304                              DEC     R4              ;Point to null at end
23 002632  004737  010642'                     CALL    CHRPOS          ;Position cursor on screen
24 002636  105037  000000G                     CLRB    SLGOLD          ;Clear gold-key flag
25 002642  000407                              BR      9$
26                                      ;
27                                      ;  Gold key was not pressed.  Move cursor right 1 character.
28                                      ;  Error if already at right end of line.
29                                      ;
30 002644  013704  000000G              1$:    MOV     SLCX,R4         ;Get current cursor character pointer
31 002650  105714                              TSTB    (R4)            ;Are we at right end of line now?
32 002652  001403                              BEQ     9$              ;Br if yes
33                                      ;
34                                      ;  Move cursor right 1 character.
35                                      ;
36 002654  005204                       2$:    INC     R4              ;Advance cursor 1 character
37 002656  004737  010642'                     CALL    CHRPOS          ;Move cursor on screen
38                                      ;
39                                      ;  Finished
40                                      ;
41 002662  012604                       9$:    MOV     (SP)+,R4
42 002664  000207                              RETURN
43
```

```
                                        .SBTTL  TCPF1  -- PF1 processing
                                ;-----------------------------------------------------------------------
                                ;  Process the PF1 (Gold) key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
002666                          TCPF1:
                                ;
                                ;  See if this is a user-defined key
                                ;
002666                               KEYCHK  KC$PF1          ;See if this is a user-defined key
002674  103410                       BCS     9$              ;Br if user-defined key
                                ;
                                ;  See if PF1 has been pressed previously
                                ;
002676  105737  0000006              TSTB    SLGOLD          ;Has PF1 already been pressed?
002702  001403                       BEQ     1$              ;Br if not
002704  105037  0000006              CLRB    SLGOLD          ;Clear gold-key flag
002710  000402                       BR      9$
                                ;
                                ;  Set flag saying Gold key pressed.
                                ;
002712  105237  0000006         1$:  INCB    SLGOLD          ;Remember "Gold" key was pressed
                                ;
                                ;  Finished
                                ;
002716  000207                  9$:  RETURN
```

```
      1                                       .SBTTL  TCPF2  -- PF2 processing
      2                                 ;-----------------------------------------------------------------------
      3                                 ;  Process the PF2 key.
      4                                 ;
      5                                 ;  Inputs
      6                                 ;   R1 = Job index number.
      7                                 ;
      8 002720                          TCPF2:
      9                                 ;
     10                                 ;  See if this is a user-defined key
     11                                 ;
     12 002720                                KEYCHK  KC$PF2          ;See if this is a user-defined key
     13 002726  103404                        BCS     9$              ;Br if user-defined key
     14                                 ;
     15                                 ;  PF2 is not defined if it is not a user-defined key.
     16                                 ;
     17 002730  004737  011154'               CALL    RNGBEL          ;Help support not implemented
     18 002734  105037  000000G               CLRB    SLGOLD          ;Clear gold flag
     19                                 ;
     20                                 ;  Finished
     21                                 ;
     22 002740  000207                  9$:   RETURN
```

```
      1                                       .SBTTL  TCPF3  -- PF3 processing
      2                                 ;-----------------------------------------------------------------------
      3                                 ;  Process PF3 key.
      4                                 ;
      5                                 ;  Inputs:
      6                                 ;   R1 = Job index number.
      7                                 ;
      8 002742                          TCPF3:
      9                                 ;
     10                                 ;  See if this is a user-defined key
     11                                 ;
     12 002742                                KEYCHK  KC$PF3          ;See if this is a user-defined key
     13 002750  103407                        BCS     9$              ;Br if user-defined key
     14                                 ;
     15                                 ;  If this is a VT52, treat PF3 like PF4.
     16                                 ;
     17 002752  004737  011602'               CALL    CHK52           ;Is this a VT52 terminal?
     18 002756  103405                        BCS     TCPF4           ;Br if yes
     19                                 ;
     20                                 ;  This is not a VT52
     21                                 ;
     22 002760  004737  011154'               CALL    RNGBEL          ;PF3 not valid for VT100
     23 002764  105037  000000G               CLRB    SLGOLD          ;Clear gold-key flag
     24 002770  000207                  9$:   RETURN
```

```
                                        .SBTTL  TCPF4  -- PF4 processing
                                ;-----------------------------------------------------------------------
                                ;  Process PF4 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
002772  010246                  TCPF4:  MOV     R2,-(SP)
002774  010346                          MOV     R3,-(SP)
002776  010446                          MOV     R4,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
003000                                  KEYCHK  KC$PF4          ;See if this is a user-defined key
003006  103430                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  See if Gold key was pressed
                                ;
003010  013704  000000G                 MOV     SLCX,R4         ;Get current cursor position pointer
003014  105737  000000G                 TSTB    SLGOLD          ;Was Gold key pressed?
003020  001013                          BNE     1$              ;Br if yes
                                ;
                                ;  Gold key was not pressed.
                                ;  Delete from cursor to end of line.
                                ;
003022  105714                          TSTB    (R4)            ;Is cursor at right end of line now?
003024  001421                          BEQ     9$              ;Br if yes
                                ;
                                ;  Move deleted characters to holding buffer.
                                ;
003026  010402                  2$:     MOV     R4,R2           ;Get cursor pointer
003030  012703  000000G                 MOV     #SLDBUF,R3      ;Point to delete buffer
003034  112223                  3$:     MOVB    (R2)+,(R3)+     ;Move deleted chars to holding buffer
003036  001376                          BNE     3$              ;Loop till null moved
                                ;
                                ;  Truncate line at cursor position
                                ;
003040  105014                          CLRB    (R4)            ;Truncate line
                                ;
                                ;  Display truncated line
                                ;
003042  004737  010336'                 CALL    PAINT           ;Redisplay line from cursor to right end
003046  000410                          BR      9$
                                ;
                                ;  Gold (PF4) key was pressed.  Replace test from holding buffer.
                                ;  Insert text to right of cursor position.
                                ;
003050  012702  000000G          1$:     MOV     #SLDBUF,R2      ;Get pointer to holding buffer
003054  004737  010140'                 CALL    INSERT          ;Insert text
003060  004737  010642'                 CALL    CHRPOS          ;Position cursor
003064  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
003070  012604                  9$:     MOV     (SP)+,R4
003072  012603                          MOV     (SP)+,R3
003074  012602                          MOV     (SP)+,R2
```

```
     58 003076  000207                          RETURN
```

```
                                        .SBTTL  TCENTR -- Enter key processing
                                ;-----------------------------------------------------------------------
                                ;  Treat Enter key like carriage-return line-feed.
                                ;
003100                          TCENTR:
                                ;
                                ;  See if this is a user-defined key
                                ;
003100                                  KEYCHK  KC$ENT          ;See if this is a user-defined key
003106  103406                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key
                                ;
003110  105037  000000G                 CLRB    SLGOLD          ;Reset gold key
003114  105237  000000G                 INCB    SLCR            ;Say we have received carriage-return
003120  004737  005622'                 CALL    ICPCR           ;Process carriage-return, line-feed
003124  000207                  9$:     RETURN

                                        .SBTTL  TCINVL -- Invalid function key
                                ;-----------------------------------------------------------------------
                                ;  An invalid function key was pressed.
                                ;
003126                          TCINVL:
                                ;
                                ;  Ring terminal bell
                                ;
003126  004737  011154'                 CALL    RNGBEL          ;Ring the bell
                                ;
                                ;  Clear gold-key flag
                                ;
003132  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
003136  000207                          RETURN
```

```
                        .SBTTL  KEYO   -- Key "0" processing
                ;-----------------------------------------------------------------------
                ;  Key 0 -- Bline / Open line
                ;
003140  010246  KEYO:   MOV     R2,-(SP)
003142  010346         MOV     R3,-(SP)
003144  010446         MOV     R4,-(SP)
                ;
                ;  See if this is a user-defined key
                ;
003146                  KEYCHK  KC$KP0          ;See if this is a user-defined key
003154  103426         BCS     9$              ;Br if user-defined key
                ;
                ;  See if the gold key was pressed
                ;
003156  105737  000000G        TSTB    SLGOLD          ;Was the gold key pressed?
003162  001005         BNE     1$              ;Br if yes
                ;
                ;  Gold key was not pressed.
                ;  Move cursor to front of line.
                ;
003164  012704  000000G        MOV     #SLEBUF,R4      ;Set cursor to front of line
003170  004737  010642'        CALL    CHRPOS          ;Position the cursor
003174  000416         BR      9$
                ;
                ;  Gold key was pressed.
                ;  Delete from cursor to end of line.
                ;
003176  105037  000000G 1$:     CLRB    SLGOLD          ;Clear gold-key flag
003202  013704  000000G        MOV     SLCX,R4         ;Get cursor position index
003206  105714         TSTB    (R4)            ;Are we already at the end of the line?
003210  001410         BEQ     9$              ;Br if yes
                ;
                ;  Move deleted characters to holding buffer
                ;
003212  010402         MOV     R4,R2           ;Get cursor pointer
003214  012703  000000G        MOV     #SLDBUF,R3      ;Point to holding buffer
003220  112223  3$:     MOVB    (R2)+,(R3)+     ;Move chars to holding buffer
003222  001376         BNE     3$              ;Loop till null moved
                ;
                ;  Truncate the line
                ;
003224  105014         CLRB    (R4)            ;Truncate the line at the cursor position
                ;
                ;  Display the truncated line
                ;
003226  004737  010336'        CALL    PAINT           ;Redisplay the line
                ;
                ;  Finished
                ;
003232  012604  9$:     MOV     (SP)+,R4
003234  012603         MOV     (SP)+,R3
003236  012602         MOV     (SP)+,R2
003240  000207         RETURN
```

```
                                        .SBTTL  KEY1    -- Key "1" processing
                                ;-----------------------------------------------------------------------
                                ;  Key 1 -- Word / Change case
                                ;
003242  010446                  KEY1:   MOV     R4,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
003244                                  KEYCHK  KC$KP1          ;See if this is a user-defined key
003252  103526                          BCS     20$             ;Br if user-defined key
                                ;
                                ;  See if the gold key was pressed
                                ;
003254  105737  000000G                 TSTB    SLGOLD          ;Was the gold key pressed?
003260  001053                          BNE     1$              ;Br if yes
                                ;
                                ;  Gold key was not pressed.
                                ;  Move forward/backward one word.
                                ;
003262  013704  000000G                 MOV     SLCX,R4         ;Get current cursor index
003266  105737  000000G                 TSTB    SLBACK          ;Moving forward or backward?
003272  001021                          BNE     16$             ;Br if backward
                                ;
                                ;  Move forward one word.
                                ;  See if cursor is pointing to a delimiter now.
                                ;
003274  112400                          MOVB    (R4)+,R0        ;Get char under cursor
003276  001514                          BEQ     20$             ;Br if at end of line now
003300  004737  011010'                 CALL    CHKDLM          ;Is current char a delimiter?
003304  103405                          BCS     4$              ;Br if yes
                                ;
                                ;  Skip characters in word under cursor
                                ;
003306  112400                  5$:     MOVB    (R4)+,R0        ;Get next char from word
003310  001410                          BEQ     6$              ;Br if hit end of line
003312  004737  011010'                 CALL    CHKDLM          ;Is this char a delimiter?
003316  103373                          BCC     5$              ;Br if not
                                ;
                                ;  Skip delimiters that follow the word
                                ;
003320  112400                  4$:     MOVB    (R4)+,R0        ;Get next char
003322  001403                          BEQ     6$              ;Br if hit end of line
003324  004737  011010'                 CALL    CHKDLM          ;Is this a delimiter?
003330  103773                          BCS     4$              ;Loop to skip all delimiters following word
003332  005304                  6$:     DEC     R4              ;Point to 1st char of new word
003334  000422                          BR      10$
                                ;
                                ;  Move backward one word
                                ;
003336  020427  000000G         16$:    CMP     R4,#SLEBUF      ;Are we already at left end of line?
003342  101472                          BLOS    20$             ;Br if yes
                                ;
                                ;  Skip over delimiters to left of cursor.
                                ;
003344  020427  000000G         7$:     CMP     R4,#SLEBUF      ;Are we at left end of line now?
003350  101414                          BLOS    10$             ;Br if yes
003352  114400                          MOVB    -(R4),R0        ;Get next char to left
```

```
     58 003354  004737  011010'                 CALL    CHKDLM          ;Is this a delimiter?
     59 003360  103771                          BCS     7$              ;Loop to skip over delimiters
     60                                 ;
     61                                 ;  Skip over characters in the word
     62                                 ;
     63 003362  020427  000000G         8$:     CMP     R4,#SLEBUF      ;Are we at left end of line yet?
     64 003366  101405                          BLOS    10$             ;Br if yes
     65 003370  114400                          MOVB    -(R4),R0        ;Get next char to left
     66 003372  004737  011010'                 CALL    CHKDLM          ;Is this a delimiter?
     67 003376  103371                          BCC     8$              ;Loop if not
     68                                 ;
     69                                 ;  Point to 1st char of the word
     70                                 ;
     71 003400  005204                          INC     R4              ;Point to 1st char of the word
     72                                 ;
     73                                 ;  Position cursor at start of word
     74                                 ;
     75 003402  004737  010642'         10$:    CALL    CHRPOS          ;Position cursor at 1st char of word
     76 003406  000450                          BR      20$
     77                                 ;
     78                                 ;  Gold key 1 -- Change case of character under the cursor
     79                                 ;
     80 003410  105037  000000G         1$:     CLRB    SLGOLD          ;Clear the gold-key flag
     81 003414  013704  000000G                 MOV     SLCX,R4         ;Get current cursor index
     82 003420  111400                          MOVB    (R4),R0         ;Get char under the cursor
     83 003422  001440                          BEQ     19$             ;Br if at right end of line
     84 003424  020027  000141                  CMP     R0,#141         ;Is this a lower-case letter?
     85 003430  103406                          BLO     11$             ;Br if not
     86 003432  020027  000172                  CMP     R0,#172
     87 003436  101016                          BHI     15$             ;Br if not
     88 003440  162700  000040                  SUB     #40,R0          ;Convert lower-case to upper-case
     89 003444  000410                          BR      14$
     90 003446  020027  000101         11$:    CMP     R0,#'A          ;Is this a upper-case letter?
     91 003452  103410                          BLO     15$             ;Br if not
     92 003454  020027  000132                  CMP     R0,#'Z
     93 003460  101005                          BHI     15$             ;Br if not
     94 003462  062700  000040                  ADD     #40,R0          ;Convert upper-case to lower-case
     95 003466  110014                  14$:    MOVB    R0,(R4)         ;Store converted character
     96 003470  004737  010336'                 CALL    PAINT           ;Redisplay the line
     97                                 ;
     98                                 ;  Advance cursor to next character
     99                                 ;
    100 003474  105737  000000G         15$:    TSTB    SLBACK          ;Moving forward or backward?
    101 003500  001002                          BNE     12$             ;Br if backward
    102 003502  005204                          INC     R4              ;Advance cursor pointer
    103 003504  000404                          BR      13$
    104 003506  020427  000000G         12$:    CMP     R4,#SLEBUF      ;Are we at left end of line now?
    105 003512  101401                          BLOS    13$             ;Br if yes
    106 003514  005304                          DEC     R4              ;Backup the cursor
    107 003516  004737  010642'         13$:    CALL    CHRPOS          ;Position cursor to next char
    108 003522  000402                          BR      20$
    109                                 ;
    110                                 ;  Error -- ring bell
    111                                 ;
    112 003524  004737  011154'         19$:    CALL    RNGBEL          ;Ring the bell
    113                                 ;
    114                                 ;  Finished
```

```
    115                                 ;
    116 003530  012604                  20$:    MOV     (SP)+,R4
    117 003532  000207                          RETURN
```

```
                                        .SBTTL  KEY2    -- Key "2" processing
                                ;-------------------------------------------------------------------
                                ;  Key "2" -- EOL / Del EOL
                                ;
003534  010446                  KEY2:   MOV     R4,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
003536                                  KEYCHK  KC$KP2          ;See if this is a user-defined key
003544  103427                          BCS     20$             ;Br if user-defined key
                                ;
                                ;  See if gold key was pressed
                                ;
003546  105737  0000006                 TSTB    SLGOLD          ;Was the gold key pressed?
003552  001020                          BNE     1$              ;Br if yes
                                ;
                                ;  Gold key was not pressed.
                                ;  Check which direction to move.
                                ;
003554  105737  0000006                 TSTB    SLBACK          ;Move to left or right end of line?
003560  001010                          BNE     2$              ;Br if moving to left end
                                ;
                                ;  Move to right end of line.
                                ;
003562  013704  0000006                 MOV     SLCX,R4         ;Get current cursor pointer
003566  105724                  3$:     TSTB    (R4)+           ;Search for null at the end of the line
003570  001376                          BNE     3$
003572  005304                          DEC     R4              ;Point to the null
003574  004737  010642'                  CALL    CHRPOS          ;Position cursor to end of line
003600  000411                          BR      20$
                                ;
                                ;  Move to left end of line.
                                ;
003602  012704  0000006         2$:     MOV     #SLEBUF,R4      ;Point to left end of line
003606  004737  010642'                  CALL    CHRPOS          ;Position cursor there
003612  000404                          BR      20$
                                ;
                                ;  Gold key 2 -- Delete to end of line
                                ;
003614  105037  0000006         1$:     CLRB    SLGOLD          ;Clear gold-key flag
003620  004737  002772'                  CALL    TCPF4           ;Process exactly like PF4 key
                                ;
                                ;  Finished
                                ;
003624  012604                  20$:    MOV     (SP)+,R4
003626  000207                          RETURN
```

```
                                        .SBTTL  KEY3   -- Key "3" processing
                                ;-----------------------------------------------------------------
                                ;  Key "3" -- VT100: Advance by character, VT52: invalid
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
003630  010446                  KEY3:   MOV     R4,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
003632                                  KEYCHK  KC$KP3          ;See if this is a user-defined key
003640  103432                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  Don't allow gold key with key 3.
                                ;
003642  105737  0000000G                TSTB    SLGOLD          ;Was gold key pressed?
003646  001023                          BNE     8$              ;Br if yes -- invalid
                                ;
                                ;  Key 3 is invalid for VT52
                                ;
003650  004737  011602'                 CALL    CHK52           ;Is this a VT52?
003654  103420                          BCS     8$              ;Br if it is
                                ;
                                ;  Advance/Backup 1 character
                                ;
003656  013704  0000000G                MOV     SLCX,R4         ;Get current cursor pointer
003662  105737  0000000G                TSTB    SLBACK          ;Advance or backup?
003666  001004                          BNE     1$              ;Br if backup
                                ;
                                ;  Advance one character
                                ;
003670  105714                          TSTB    (R4)            ;Are we at right end of line now?
003672  001415                          BEQ     9$              ;Br if yes
003674  005204                          INC     R4              ;Advance cursor
003676  000404                          BR      2$
                                ;
                                ;  Backup cursor one character
                                ;
003700  020427  0000000G        1$:     CMP     R4,#SLEBUF      ;Are we at left end of line now?
003704  101410                          BLOS    9$              ;Br if yes
003706  005304                          DEC     R4              ;Backup the cursor
003710  004737  010642'          2$:     CALL    CHRPOS          ;Display cursor at correct position
003714  000404                          BR      9$
                                ;
                                ;  Error -- Ring the bell
                                ;
003716  004737  011154'          8$:     CALL    RNGBEL          ;Ring the bell
003722  105037  0000000G                CLRB    SLGOLD          ;Clear the gold key flag
                                ;
                                ;  Finished
                                ;
003726  012604                  9$:     MOV     (SP)+,R4
003730  000207                          RETURN
```

```
                                        .SBTTL  KEY4    -- Key "4" processing
                                ;-----------------------------------------------------------------------
                                ;  Key "4" -- Set forward direction / Move to right end of line.
                                ;
003732                          KEY4:
                                ;
                                ;  See if this is a user-defined key
                                ;
003732                                  KEYCHK  KC$KP4          ;See if this is a user-defined key
003740  103410                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  See if the gold key was pressed
                                ;
003742  105737  0000006                 TSTB    SLGOLD          ;Was gold key pressed?
003746  001003                          BNE     1$              ;Br if yes
                                ;
                                ;  Gold key was not pressed -- Set forward direction
                                ;
003750  105037  0000006                 CLRB    SLBACK          ;Set forward direction
003754  000402                          BR      9$
                                ;
                                ;  Gold key was pressed -- Move to the right end of the line
                                ;
003756  004737  002600'         1$:     CALL    TCRITE          ;Process just like right-arrow key
                                ;
                                ;  Finished
                                ;
003762  000207                  9$:     RETURN
```

```
                                        .SBTTL  KEY5    -- Key "5" processing
                                ;-----------------------------------------------------------------------
                                ;  Key "5" -- Set backward direction / Move to left end of line
                                ;
003764                          KEY5:
                                ;
                                ;  See if this is a user-defined key
                                ;
003764                                  KEYCHK  KC$KP5          ;See if this is a user-defined key
003772  103411                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  See if the gold key was pressed
                                ;
003774  105737  000000G                 TSTB    SLGOLD          ;Was gold key pressed?
004000  001004                          BNE     1$              ;Br if yes
                                ;
                                ;  Gold key not pressed -- Set backward direction.
                                ;
004002  112737  000001  000000G         MOVB    #1,SLBACK       ;Set backward direction
004010  000402                          BR      9$
                                ;
                                ;  Gold key was pressed -- Move to left end of line
                                ;
004012  004737  002516'         1$:     CALL    TCLEFT          ;Treat just like left-arrow key
                                ;
                                ;  Finished
                                ;
004016  000207                  9$:     RETURN
```

```
                                        .SBTTL  KEY6    -- Key "6" processing
                                ;-----------------------------------------------------------------
                                ;  Key "6" -- VT100:  invalid,  VT52:  [un] delete character
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004020                          KEY6:
                                ;
                                ;  See if this is a user-defined key
                                ;
004020                                  KEYCHK  KC$KP6          ;See if this is a user-defined key
004026  103411                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  Determine if this is a VT52 or VT100
                                ;
004030  004737  011602'                 CALL    CHK52           ;Is this a VT52?
004034  103002                         BCC     1$              ;Br if not
                                ;
                                ;  VT52 -- Treat key 6 like VT100 comma key
                                ;
004036  000137  004154'                 JMP     KEYCOM          ;Treat like comma key
                                ;
                                ;  VT100 -- invalid
                                ;
004042  004737  011154'         1$:     CALL    RNGBEL          ;Invalid function
004046  105037  0000000                 CLRB    SLGOLD          ;Reset gold key flag
004052  000207                 9$:     RETURN
```

```
                                        .SBTTL  KEY7   -- Key "7" processing
                                ;-----------------------------------------------------------------------
                                ;  Key "7" -- No function
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number
                                ;
004054                          KEY7:
                                ;
                                ;  See if this is a user-defined key
                                ;
004054                                  KEYCHK  KC$KP7          ;See if this is a user-defined key
004062  103404                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  Not user-defined key
                                ;
004064  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004070  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004074  000207                  9$:     RETURN
```

```
      1                                       .SBTTL  KEY8   -- Key "8" processing
      2                                 ;-----------------------------------------------------------------------
      3                                 ;  Key "8" -- No function
      4                                 ;
      5                                 ;  Inputs:
      6                                 ;   R1 = Job index number
      7                                 ;
      8 004076                          KEY8:
      9                                 ;
     10                                 ;  See if this is a user-defined key
     11                                 ;
     12 004076                                KEYCHK  KC$KP8          ;See if this is a user-defined key
     13 004104  103404                        BCS     9$              ;Br if user-defined key
     14                                 ;
     15                                 ;  This is not a user-defined key
     16                                 ;
     17 004106  004737  011154'               CALL    RNGBEL          ;Ring the bell
     18 004112  105037  000000G               CLRB    SLGOLD          ;Clear gold-key flag
     19                                 ;
     20                                 ;  Finished
     21                                 ;
     22 004116  000207                  9$:   RETURN
```

```
                                        .SBTTL  KEY9   -- Key "9" processing
                                ;-----------------------------------------------------------------------
                                ;  Key "9" -- VT100:  invalid,  VT52:  [un] delete word
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004120                          KEY9:
                                ;
                                ;  See if this is a user-defined key
                                ;
004120                                  KEYCHK  KC$KP9          ;See if this is a user-defined key
004126  103411                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key
                                ;
004130  004737  011602'                 CALL    CHK52           ;Is this a VT52 terminal?
004134  103002                          BCC     1$              ;Br if not
                                ;
                                ;  VT52 -- Treat like VT100 minus key
                                ;
004136  000137  004262'                 JMP     KEYMIN          ;Treat like minus key
                                ;
                                ;  VT100 -- Invalid
                                ;
004142  004737  011154'         1$:     CALL    RNGBEL          ;Ring the bell
004146  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
004152  000207                  9$:     RETURN
```

```
                                        .SBTTL  KEYCOM -- Key "," processing
                                ;-----------------------------------------------------------------------
                                ;  Comma key -- [un] delete character
                                ;
004154  010246                 KEYCOM: MOV     R2,-(SP)
004156  010446                         MOV     R4,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
004160                                  KEYCHK  KC$COM          ;See if this is a user-defined key
004166  103432                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  See if Gold key (PF1) was pressed.
                                ;
004170  105737  000000G                 TSTB    SLGOLD          ;Was Gold key pressed?
004174  001414                         BEQ     1$              ;Br if not
                                ;
                                ;  Gold key was pressed  -- put back previously deleted character
                                ;
004176  012702  000000G                 MOV     #SLCBUF,R2      ;Point to deleted-character buffer
004202  004737  010140'                 CALL    INSERT          ;Insert the character
004206  013704  000000G                 MOV     SLCX,R4         ;Get current cursor pointer
004212  005304                         DEC     R4              ;Point to character we inserted
004214  004737  010642'                 CALL    CHRPOS          ;Position cursor to inserted character
004220  105037  000000G                 CLRB    SLGOLD          ;Clear Gold-key flag
004224  000413                         BR      9$
                                ;
                                ;  Gold key not pressed -- Delete character under the cursor
                                ;
004226  013704  000000G         1$:     MOV     SLCX,R4         ;Get current cursor pointer
004232  105714                         TSTB    (R4)            ;Is cursor at right end of line?
004234  001407                         BEQ     9$              ;Br if yes
                                ;
                                ;  Save character being deleted and then delete it
                                ;
004236  111437  000000G         2$:     MOVB    (R4),SLCBUF     ;Save character being deleted
004242  005204                         INC     R4              ;Move cursor right 1 character
004244  004737  010642'                 CALL    CHRPOS
004250  004737  007454'                 CALL    DELLFT          ;Now delete character to left of cursor
                                ;
                                ;  Finished
                                ;
004254  012604                 9$:     MOV     (SP)+,R4
004256  012602                         MOV     (SP)+,R2
004260  000207                         RETURN
```

```
                                        .SBTTL  KEYMIN -- Key "-" processing
                                ;-------------------------------------------------------------------------
                                ;  Minus key -- [un] delete word
                                ;
004262  010246                  KEYMIN: MOV     R2,-(SP)
004264  010346                          MOV     R3,-(SP)
004266  010446                          MOV     R4,-(SP)
004270  010546                          MOV     R5,-(SP)
                                ;
                                ;  See if this is a user-defined key
                                ;
004272                                  KEYCHK  KC$MIN          ;See if this is a user-defined key
004300  103463                          BCS     20$             ;Br if user-defined key
                                ;
                                ;  See if the gold key was pressed
                                ;
004302  013704  0000006                 MOV     SLCX,R4         ;Get current cursor index
004306  105737  0000006                 TSTB    SLGOLD          ;Was the gold key pressed?
004312  001041                          BNE     1$              ;Br if yes
                                ;
                                ;  Gold key was not pressed -- Delete the word under the cursor
                                ;
004314  010405                          MOV     R4,R5           ;Remember starting position of deletion
004316  112400                          MOVB    (R4)+,R0        ;Get current char
004320  001453                          BEQ     20$             ;Br if at right end of line now
004322  004737  011010'                 CALL    CHKDLM          ;Is this character a delimiter?
004326  103405                          BCS     3$              ;Br if yes
004330  112400                  2$:     MOVB    (R4)+,R0        ;Get next character
004332  001410                          BEQ     4$              ;Br if hit right end of line
004334  004737  011010'                 CALL    CHKDLM          ;Is this character a delimiter?
004340  103373                          BCC     2$              ;Br if not
                                ;
                                ;  Delete delimiters that follow the word
                                ;
004342  112400                  3$:     MOVB    (R4)+,R0        ;Get next character
004344  001403                          BEQ     4$              ;Br if hit right end of line
004346  004737  011010'                 CALL    CHKDLM          ;Is this a delimiter?
004352  103773                          BCS     3$              ;Loop if yes
004354  005304                  4$:     DEC     R4              ;Point to 1st char of following word
                                ;
                                ;  We have identified the characters to delete.
                                ;  Move the characters to a holding buffer.
                                ;
004356  010503                          MOV     R5,R3           ;Get index to 1st char to delete
004360  012702  0000006                 MOV     #SLDBUF,R2      ;Point to holding buffer
004364  112322                  5$:     MOVB    (R3)+,(R2)+     ;Move char to holding buffer?
004366  020304                          CMP     R3,R4           ;Moved all we need to delete?
004370  103775                          BLO     5$              ;Loop if not
004372  105012                          CLRB    (R2)            ;Put null at end of deleted char string
                                ;
                                ;  Delete the characters from the buffer
                                ;
004374  010503                          MOV     R5,R3           ;Get pointer to 1st char to delete
004376  112423                  6$:     MOVB    (R4)+,(R3)+     ;Move remainder of line left
004400  001376                          BNE     6$              ;Loop till null at end moved
                                ;
                                ;  Redisplay the line
```

```
                                ;
004402  010504                  MOV     R5,R4           ;Get new position of cursor
004404  004737  010336'          CALL    PAINT           ;Redisplay the line
                                ;
                                ;  Reposition cursor
                                ;
004410  004737  010642'          CALL    CHRPOS          ;Resposition cursor
004414  000415                  BR      20$
                                ;
                                ;  Gold minus key -- Replace deleted word
                                ;
004416  105037  000000G  1$:     CLRB    SLGOLD          ;Reset gold-key flag
                                ;
                                ;  Insert the deleted word
                                ;
004422  013704  000000G          MOV     SLCX,R4         ;Get current cursor index
004426  012702  000000G          MOV     #SLDBUF,R2      ;Point to buffer with deleted text
004432  004737  010140'          CALL    INSERT          ;Insert the text
004436  004737  010642'          CALL    CHRPOS          ;Position cursor to front of replaced text
004442  000402                  BR      20$
                                ;
                                ;  Error -- Ring bell
                                ;
004444  004737  011154'  19$:     CALL    RNGBEL          ;Ring bell
                                ;
                                ;  Finished
                                ;
004450  012605          20$:    MOV     (SP)+,R5
004452  012604                  MOV     (SP)+,R4
004454  012603                  MOV     (SP)+,R3
004456  012602                  MOV     (SP)+,R2
004460  000207                  RETURN
```

```
                                        .SBTTL  KBS     -- Save a command
                                ;-----------------------------------------------------------------------
                                ;  This routine is called when PF1 S is typed.  It saves a command.
                                ;
004462  010446                  KBS:    MOV     R4,-(SP)
004464  010546                          MOV     R5,-(SP)
                                ;
                                ;  Save line
                                ;
004466  105037  0000000G                CLRB    SLGOLD          ;Clear gold-key flag
004472  012704  0000000G                MOV     #SLEBUF,R4      ;Point to current line buffer
004476  012705  0000000G                MOV     #SLSBUF,R5      ;Point to save buffer
004502  112425                  2$:     MOVB    (R4)+,(R5)+     ;Save the line
004504  001376                          BNE     2$              ;Loop if more to save
                                ;
                                ;  Finished
                                ;
004506  012605                          MOV     (SP)+,R5
004510  012604                          MOV     (SP)+,R4
004512  000207                          RETURN
```

```
      1                                          .SBTTL  KBX     -- Recall the saved command
      2                                ;-----------------------------------------------------------------------
      3                                ;  This routine is called when PF1 X is typed.  It recalls the saved command.
      4                                ;
      5 004514  010546                 KBX:    MOV     R5,-(SP)
      6                                ;
      7                                ;  Insert saved line
      8                                ;
      9 004516  012705  000000G                MOV     #SLSBUF,R5      ;Point to buffer with saved line
     10 004522  004737  007704'                CALL    RSTLIN          ;Restore the line
     11                                ;
     12                                ;  Finished
     13                                ;
     14 004526  012605                         MOV     (SP)+,R5
     15 004530  000207                         RETURN
```

```
                                        .SBTTL  E1      -- E1 processing
                                ;-----------------------------------------------------------------------
                                ;  Process E1 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004532                          E1:
                                ;
                                ;  See if this is a user-defined key
                                ;
004532                               KEYCHK  KC$E1            ;See if this is a user-defined key
004540  103404                       BCS     9$               ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004542  004737  011154'              CALL    RNGBEL           ;Ring the bell
004546  105037  000000G              CLRB    SLGOLD           ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004552  000207                  9$:     RETURN
```

```
                                        .SBTTL  E2      -- E2 processing
                                ;-----------------------------------------------------------------------
                                ;  Process E2 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004554                          E2:
                                ;
                                ;  See if this is a user-defined key
                                ;
004554                                  KEYCHK  KC$E2           ;See if this is a user-defined key
004562  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004564  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004570  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004574  000207                  9$:     RETURN
```

```
                                        .SBTTL  E3      -- E3 processing
                                ;-----------------------------------------------------------------------
                                ;  Process E3 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004576                          E3:
                                ;
                                ;  See if this is a user-defined key
                                ;
004576                                  KEYCHK  KC$E3           ;See if this is a user-defined key
004604  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004606  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004612  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004616  000207                  9$:     RETURN
```

```
                                        .SBTTL  E4      -- E4 processing
                                ;-----------------------------------------------------------------------
                                ;  Process E4 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004620                          E4:
                                ;
                                ;  See if this is a user-defined key
                                ;
004620                                  KEYCHK  KC$E4           ;See if this is a user-defined key
004626  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004630  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004634  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004640  000207                  9$:     RETURN
```

```
                                        .SBTTL  E5      -- E5 processing
                                ;-------------------------------------------------------------------
                                ;  Process E5 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004642                          E5:
                                ;
                                ;  See if this is a user-defined key
                                ;
004642                                  KEYCHK  KC$E5           ;See if this is a user-defined key
004650  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004652  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004656  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004662  000207                  9$:     RETURN
```

```
                                        .SBTTL  E6      -- E6 processing
                                ;-----------------------------------------------------------------------
                                ;  Process E6 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004664                          E6:
                                ;
                                ;  See if this is a user-defined key
                                ;
004664                                  KEYCHK  KC$E6           ;See if this is a user-defined key
004672  103404                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004674  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004700  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004704  000207                  9$:     RETURN
```

```
                                        .SBTTL  F6      -- F6 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F6 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004706                          F6:
                                ;
                                ;  See if this is a user-defined key
                                ;
004706                                  KEYCHK  KC$F6           ;See if this is a user-defined key
004714  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004716  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004722  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004726  000207                  9$:     RETURN
```

```
                                        .SBTTL  F7      -- F7 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F7 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004730                          F7:
                                ;
                                ;  See if this is a user-defined key
                                ;
004730                                  KEYCHK  KC$F7           ;See if this is a user-defined key
004736  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004740  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004744  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004750  000207                  9$:     RETURN
```

```
                                        .SBTTL  F8      -- F8 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F8 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004752                          F8:
                                ;
                                ;  See if this is a user-defined key
                                ;
004752                                  KEYCHK  KC$F8           ;See if this is a user-defined key
004760  103404                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
004762  004737  011154'                 CALL    RNGBEL          ;Ring the bell
004766  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
004772  000207                  9$:     RETURN
```

```
                                        .SBTTL  F9      -- F9 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F9 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
004774                          F9:
                                ;
                                ;  See if this is a user-defined key
                                ;
                                        KEYCHK  KC$F9           ;See if this is a user-defined key
005002  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005004  004737  011154'                 CALL    RNGBEL          ;Ring the bell
005010  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005014  000207                  9$:     RETURN
```

```
 1                                          .SBTTL  F10     -- F10 processing
 2                                ;-----------------------------------------------------------------------
 3                                ;  Process F10 key.
 4                                ;
 5                                ;  Inputs:
 6                                ;   R1 = Job index number.
 7                                ;
 8 005016                         F10:
 9                                ;
10                                ;  See if this is a user-defined key
11                                ;
12 005016                                KEYCHK  KC$F10          ;See if this is a user-defined key
13 005024  103404                        BCS     9$              ;Br if user-defined key
14                                ;
15                                ;  This is not a user-defined key.
16                                ;  Invalid key.
17                                ;
18 005026  004737  011154'               CALL    RNGBEL          ;Ring the bell
19 005032  105037  000000G               CLRB    SLGOLD          ;Clear gold-key flag
20                                ;
21                                ;  Finished
22                                ;
23 005036  000207                9$:     RETURN
```

```
 1                                          .SBTTL  F11     -- F11 processing
 2                                  ;-----------------------------------------------------------------
 3                                  ;  Process F11 key.
 4                                  ;
 5                                  ;  Inputs:
 6                                  ;   R1 = Job index number.
 7                                  ;
 8 005040                           F11:
 9                                  ;
10                                  ;  See if this is a user-defined key
11                                  ;
12 005040                                  KEYCHK  KC$F11          ;See if this is a user-defined key
13 005046  103404                          BCS     9$              ;Br if user-defined key
14                                  ;
15                                  ;  This is not a user-defined key.
16                                  ;  Invalid key.
17                                  ;
18 005050  004737  011154'                 CALL    RNGBEL          ;Ring the bell
19 005054  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
20                                  ;
21                                  ;  Finished
22                                  ;
23 005060  000207                  9$:     RETURN
```

```
                                        .SBTTL  F12     -- F12 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F12 key -- BS.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005062                          F12:
                                ;
                                ;  See if this is a user-defined key
                                ;
005062                                  KEYCHK  KC$F12          ;See if this is a user-defined key
005070  103414                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  See if gold key was pressed
                                ;
005072  105737  000000G                 TSTB    SLGOLD          ;Was gold key pressed?
005076  001005                         BNE     5$              ;Br if yes
                                ;
                                ;  Gold key was not pressed
                                ;
005100  112705  000010                  MOVB    #10,R5          ;Get backspace character
005104  004737  006156'                 CALL    ICPBS           ;Process backspace character
005110  000404                         BR      9$
                                ;
                                ;  Gold key was pressed
                                ;
005112  105037  000000G         5$:     CLRB    SLGOLD          ;Reset gold key
005116  004737  011154'                 CALL    RNGBEL          ;Ring the bell
                                ;
                                ;  Finished
                                ;
005122  000207                 9$:     RETURN
```

```
                                        .SBTTL  F13     -- F13 processing
                                ;-----------------------------------------------------------------
                                ;  Process F13 key -- LF.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005124                          F13:
                                ;
                                ;  See if this is a user-defined key
                                ;
005124                                  KEYCHK  KC$F13          ;See if this is a user-defined key
005132  103414                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  See if gold key was pressed
                                ;
005134  105737  0000006                 TSTB    SLGOLD          ;Was gold key pressed?
005140  001005                         BNE     5$              ;Br if yes
                                ;
                                ;  Gold key was not pressed
                                ;
005142  012705  000012                  MOV     #12,R5          ;Get line-feed character
005146  004737  005764'                CALL    ICPLF           ;Process line-feed character
005152  000404                         BR      9$
                                ;
                                ;  Gold key was pressed
                                ;
005154  105037  0000006         5$:     CLRB    SLGOLD          ;Reset gold key
005160  004737  011154'                CALL    RNGBEL          ;Ring the bell
                                ;
                                ;  Finished
                                ;
005164  000207                  9$:     RETURN
```

```
                                        .SBTTL  F14     -- F14 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F14 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005166                          F14:
                                ;
                                ;  See if this is a user-defined key
                                ;
005166                                  KEYCHK  KC$F14          ;See if this is a user-defined key
005174  103404                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005176  004737  011154'                 CALL    RNGBEL          ;Ring the bell
005202  105037  0000000                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005206  000207                  9$:     RETURN
```

```
                                        .SBTTL  F15     -- F15 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F15 key -- Help.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005210                          F15:
                                ;
                                ;  See if this is a user-defined key
                                ;
005210                                  KEYCHK  KC$F15          ;See if this is a user-defined key
005216  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005220  004737  011154'                 CALL    RNGBEL          ;Ring the bell
005224  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005230  000207                  9$:     RETURN
```

```
                                        .SBTTL  F16     -- F16 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F16 key -- Do.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005232                          F16:
                                ;
                                ;  See if this is a user-defined key
                                ;
005232                               KEYCHK  KC$F16          ;See if this is a user-defined key
005240  103404                      BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005242  004737  011154'              CALL    RNGBEL          ;Ring the bell
005246  105037  000000G              CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005252  000207                  9$:     RETURN
```

```
                                        .SBTTL  F17     -- F17 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F17 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005254                          F17:
                                ;
                                ;  See if this is a user-defined key
                                ;
005254                                  KEYCHK  KC$F17          ;See if this is a user-defined key
005262  103404                          BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005264  004737  011154'                 CALL    RNGBEL          ;Ring the bell
005270  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005274  000207                  9$:     RETURN
```

```
                                        .SBTTL  F18     -- F18 processing
                                ;-----------------------------------------------------------------------
                                ;  Process F18 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005276                          F18:
                                ;
                                ;  See if this is a user-defined key
                                ;
005276                               KEYCHK  KC$F18          ;See if this is a user-defined key
005304  103404                      BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005306  004737  011154'              CALL    RNGBEL          ;Ring the bell
005312  105037  000000G              CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005316  000207                  9$:     RETURN
```

```
                                        .SBTTL  F19     -- F19 processing
                                ;-----------------------------------------------------------------
                                ;  Process F19 key.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
005320                          F19:
                                ;
                                ;  See if this is a user-defined key
                                ;
005320                                  KEYCHK  KC$F19          ;See if this is a user-defined key
005326  103404                         BCS     9$              ;Br if user-defined key
                                ;
                                ;  This is not a user-defined key.
                                ;  Invalid key.
                                ;
005330  004737  011154'                 CALL    RNGBEL          ;Ring the bell
005334  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
                                ;
                                ;  Finished
                                ;
005340  000207                  9$:     RETURN
```

```
 1                                          .SBTTL  F20     -- F20 processing
 2                                  ;-----------------------------------------------------------------------
 3                                  ;  Process F20 key.
 4                                  ;
 5                                  ;  Inputs:
 6                                  ;   R1 = Job index number.
 7                                  ;
 8 005342                           F20:
 9                                  ;
10                                  ;  See if this is a user-defined key
11                                  ;
12 005342                                  KEYCHK  KC$F20          ;See if this is a user-defined key
13 005350  103404                          BCS     9$              ;Br if user-defined key
14                                  ;
15                                  ;  This is not a user-defined key.
16                                  ;  Invalid key.
17                                  ;
18 005352  004737  011154'                 CALL    RNGBEL          ;Ring the bell
19 005356  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
20                                  ;
21                                  ;  Finished
22                                  ;
23 005362  000207                  9$:     RETURN
```

```
      1                                       .SBTTL  KEYDOT -- Key "." processing
      2                                 ;-----------------------------------------------------------------------
      3                                 ;  Period key -- No defined function
      4                                 ;
      5 005364                          KEYDOT:
      6                                 ;
      7                                 ;  See if this is a user-defined key
      8                                 ;
      9 005364                                KEYCHK  KC$DOT          ;See if this is a user-defined key
     10 005372  103404                        BCS     9$              ;Br if user-defined key
     11                                 ;
     12                                 ;  This is not a user-defined key.
     13                                 ;  Invalid key.
     14                                 ;
     15 005374  004737  011154'              CALL    RNGBEL          ;Ring the bell
     16 005400  105037  000000G              CLRB    SLGOLD          ;Clear gold-key flag
     17                                 ;
     18                                 ;  Finished
     19                                 ;
     20 005404  000207                  9$:  RETURN
```

```
                                        .SBTTL  DOCTRL -- Process control characters
                                ;-----------------------------------------------------------------------
                                ;  DOCTRL is called from the input interrupt character processing when
                                ;  we determine that the character being processed is a control character.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line number.
                                ;   R5 = Character to process.
                                ;
005406  010246                  DOCTRL: MOV     R2,-(SP)
                                ;
                                ;  See if this is a request to print the current window
                                ;
005410  120537  000000G                 CMPB    R5,VVPWCH       ;Request to print current window?
005414  001013                          BNE     5$              ;Br if not
005416  032761  000000G 000000G         BIT     #$PWKEY,LSW11(R1);Is print window control char enabled?
005424  001407                          BEQ     5$              ;Br if not
005426  016102  000000G                 MOV     LWINDO(R1),R2   ;Is windowing enabled for this process?
005432  001404                          BEQ     5$              ;Br if not
                                ;
                                ;  Request to print the current screen window
                                ;
005434                                  OCALL   WINPRT          ;Print the window
005442  000425                          BR      9$
                                ;
                                ;  See if this is a request to switch to a virtual line
                                ;
005444  120537  000000G         5$:     CMPB    R5,VVLSCH       ;Is this char a request to switch to vir line?
005450  001013                          BNE     1$              ;Br if not
005452  032761  000000G 000000G         BIT     #$CTRLW,LSW3(R1);Was last char also ctrl-W?
005460  001004                          BNE     2$              ;Br if yes -- Treat two ctrl-W like one ctrl-W
005462  052761  000000G 000000G         BIS     #$CTRLW,LSW3(R1);Remember last character was ctrl-W
005470  000412                          BR      9$              ;Finished with this character
005472  042761  000000G 000000G 2$:     BIC     #$1ESC,LSW(R1)  ;Say last char was not escape
005500  042761  000000G 000000G 1$:     BIC     #$CTRLW,LSW3(R1);Say last char not ctrl-W
                                ;
                                ;  This is an ordinary control character
                                ;
005506  010500                          MOV     R5,R0           ;Get the control character
005510  006300                          ASL     R0              ;Convert to word table index
005512  004770  005522'                 CALL    @CTLRTN(R0)     ;Call appropriate processing routine
                                ;
                                ;  Finished
                                ;
005516  012602                  9$:     MOV     (SP)+,R2
005520  000207                          RETURN

                                ;-----------------------------------------------------------------------
                                ;  Branch table for control character processing routines.
                                ;
005522  007040'                 CTLRTN: .WORD   ICPNUL          ; 00 - NUL
005524  006376'                         .WORD   ICPCTA          ; 01 - SOH (control-A)
005526  000614'                         .WORD   REGCHR          ; 02 - STX
005530  006422'                         .WORD   ICPCTC          ; 03 - ETX (control-C)
005532  000614'                         .WORD   REGCHR          ; 04 - EOT
005534  000614'                         .WORD   REGCHR          ; 05 - ENQ
005536  000614'                         .WORD   REGCHR          ; 06 - ACK
```

```
     58 005540  000614'                    .WORD   REGCHR          ; 07 - BEL
     59 005542  006156'                    .WORD   ICPBS           ; 10 - Backspace
     60 005544  000614'                    .WORD   REGCHR          ; 11 - TAB
     61 005546  005764'                    .WORD   ICPLF           ; 12 - Line feed
     62 005550  000614'                    .WORD   REGCHR          ; 13 - VT
     63 005552  000614'                    .WORD   REGCHR          ; 14 - FF
     64 005554  005622'                    .WORD   ICPCR           ; 15 - Carriage return
     65 005556  000614'                    .WORD   REGCHR          ; 16 - SO
     66 005560  000614'                    .WORD   REGCHR          ; 17 - SI  (control-O)
     67 005562  000614'                    .WORD   REGCHR          ; 20 - DLE
     68 005564  000614'                    .WORD   REGCHR          ; 21 - DC1 (control-Q)
     69 005566  006524'                    .WORD   ICPCTR          ; 22 - DC2 (control-R)
     70 005570  000614'                    .WORD   REGCHR          ; 23 - DC3 (control-S)
     71 005572  000614'                    .WORD   REGCHR          ; 24 - DC4
     72 005574  006560'                    .WORD   ICPCTU          ; 25 - NAK (control-U)
     73 005576  000614'                    .WORD   REGCHR          ; 26 - SYN
     74 005600  006524'                    .WORD   ICPCTR          ; 27 - ETB (control-W)
     75 005602  000614'                    .WORD   REGCHR          ; 30 - CAN (control-X)
     76 005604  000614'                    .WORD   REGCHR          ; 31 - EM
     77 005606  006700'                    .WORD   ICPCTZ          ; 32 - SUB (control-Z)
     78 005610  006274'                    .WORD   ICPESC          ; 33 - ESC
     79 005612  000614'                    .WORD   REGCHR          ; 34 - FS
     80 005614  000614'                    .WORD   REGCHR          ; 35 - GS
     81 005616  000614'                    .WORD   REGCHR          ; 36 - RS
     82 005620  000614'                    .WORD   REGCHR          ; 37 - US
```

```
                                        .SBTTL  ICPCR  -- Carriage-return processing
                                ;-----------------------------------------------------------------------
                                ;  Process Carriage-return character.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line index number.
                                ;   R5 = Current input character.
                                ;
005622  010446                  ICPCR:  MOV     R4,-(SP)
                                ;
                                ;  If we have not yet gotten line-feed character, set flag and wait
                                ;  for line-feed to arrive.
                                ;
005624  105737  000000G                 TSTB    SLCR            ;Have we seen line-feed yet?
005630  001003                          BNE     5$              ;Br if yes
                                ;
                                ;  We have not seen line-feed yet. Set flag and wait for line-feed.
                                ;
005632  105237  000000G                 INCB    SLCR            ;Set flag saying we are waiting for line feed
005636  000450                          BR      9$              ;Defer processing until after line feed
                                ;
                                ;  We have received carriage return and line feed.
                                ;  See if Gold key has been pressed.
                                ;
005640  105037  000000G         5$:     CLRB    SLCR            ;Clear carriage return flag
005644  105737  000000G                 TSTB    SLGOLD          ;Has gold key been pressed?
005650  001407                          BEQ     1$              ;Br if not
                                ;
                                ;  Gold key was pressed -- Truncate line before terminating it
                                ;
005652  105037  000000G                 CLRB    SLGOLD          ;Clear gold-key flag
005656  013704  000000G                 MOV     SLCX,R4         ;Get current cursor pointer
005662  105014                          CLRB    (R4)            ;Truncate edit line
005664  004737  010336'                 CALL    PAINT           ;Redisplay line
                                ;
                                ;  Terminate input line.
                                ;
005670  004737  007172'         1$:     CALL    SAVLIN          ;Save the completed input line
005674  013704  000000G                 MOV     SLCX,R4         ;Point to cursor position
005700  105724                  2$:     TSTB    (R4)+           ;Search for end of line
005702  001376                          BNE     2$              ;Loop till null hit
005704  005304                          DEC     R4              ;Point to null
005706  112724  000000G                 MOVB    #CR,(R4)+       ;Store carriage-return
005712  112724  000000G                 MOVB    #LF,(R4)+       ;And line-feed at end of line
005716  105014                          CLRB    (R4)            ;Put null at end
                                ;
                                ;  Echo carriage-return & line-feed to terminal
                                ;
005720  112700  000000G                 MOVB    #CR,R0          ;Get carriage return
005724  004737  011210'                 CALL    ECHO            ;Echo carriage return
005730  032761  000000G 000000G         BIT     #$NOLF,LSW6(R1) ;Are we suppressing LF echoing after CR?
005736  001004                          BNE     3$              ;Br if yes
005740  112700  000000G                 MOVB    #LF,R0          ;Get line-feed character
005744  004737  011210'                 CALL    ECHO            ;Echo line feed
                                ;
                                ;  Signal line completed
                                ;
```

```
     58 005750  004737  007346'          3$:     CALL    LINFIN          ;Input line is finished
     59                                   ;
     60                                   ;  Say that cursor is at left margin
     61                                   ;
     62 005754  105061  000000G                   CLRB    LCOL(R1)        ;Say cursor is at left margin
     63                                   ;
     64                                   ;  Finished
     65                                   ;
     66 005760  012604                    9$:     MOV     (SP)+,R4
     67 005762  000207                            RETURN
```

```
                                        .SBTTL  ICPLF  -- Line-feed processing
                                ;-----------------------------------------------------------------------
                                ;  Process Line-feed input character.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line index number.
                                ;   R5 = Current input character.
                                ;
005764                          ICPLF:
                                ;
                                ;  If this is a line feed that is part of a carriage-return/line-feed pair
                                ;  then go to do carriage return processing.
                                ;
005764  105737  0000000G               TSTB    SLCR            ;Is this line feed after a carriage return?
005770  001314                         BNE     ICPCR           ;Br if yes -- Go do carriage return processing
                                ;
                                ;  This is a line feed by itself
                                ;
005772  010246                         MOV     R2,-(SP)
005774  010346                         MOV     R3,-(SP)
005776  010446                         MOV     R4,-(SP)
006000  010546                         MOV     R5,-(SP)
                                ;
                                ;  See if Gold key (PF1) was pressed before line-feed
                                ;
006002  105737  0000000G               TSTB    SLGOLD          ;Was gold key pressed?
006006  001407                         BEQ     1$              ;Br if not
                                ;
                                ;  Gold key was pressed -- Replace previously deleted word.
                                ;
006010  012702  0000000G               MOV     #SLDBUF,R2      ;Point to delete buffer
006014  004737  010140'                CALL    INSERT          ;Replace the deleted word
006020  105037  0000000G               CLRB    SLGOLD          ;Clear the gold key
006024  000447                         BR      9$
                                ;
                                ;  Gold key was not pressed -- Delete the word to the left of the cursor.
                                ;  Begin by deleting any delimiters immediately to left of cursor.
                                ;
006026  013704  0000000G       1$:     MOV     SLCX,R4         ;Get pointer to character under cursor
006032  010405                         MOV     R4,R5
006034  020527  0000000G               CMP     R5,#SLEBUF      ;Are we at left end now?
006040  101441                         BLOS    9$              ;Br if yes
006042  114500               11$:      MOVB    -(R5),R0        ;Get next character to be deleted
006044  004737  011010'                CALL    CHKDLM          ;Is this character a delimiter?
006050  103006                         BCC     4$              ;Br if not
                                ;
                                ;  Found delimiter to left of cursor.
                                ;  Delete all consecutive occurrances of the delimiter.
                                ;
006052  020527  0000000G       3$:     CMP     R5,#SLEBUF      ;Are we at left end of line now?
006056  101413                         BLOS    6$              ;Br if yes
006060  120045                         CMPB    R0,-(R5)        ;See if there are more occurrences of delim
006062  001773                         BEQ     3$              ;Br if yes -- keep deleting
006064  005205                         INC     R5
                                ;
                                ;  We have now deleted any delimiters to the left of the cursor.
                                ;  Begin to delete the word to the left of the cursor.
```

```
                                        ;
006066  020527  000000G        4$:     CMP     R5,#SLEBUF      ;Have we hit left end of line?
006072  101405                        BLOS    6$              ;Br if yes
006074  114500                        MOVB    -(R5),R0        ;Get next char to delete
006076  004737  011010'               CALL    CHKDLM          ;Is this character a delimiter?
006102  103371                        BCC     4$              ;Br if not
006104  005205                        INC     R5              ;Don't delete the delimiter
                                        ;
                                        ;  We have now identified the characters to delete.
                                        ;  Save characters being deleted in deleted-string buffer.
                                        ;
006106  010503                  6$:     MOV     R5,R3           ;Point to left-most character to delete
006110  012702  000000G                 MOV     #SLDBUF,R2      ;Point to deleted-string buffer
006114  112322                  10$:    MOVB    (R3)+,(R2)+     ;Move chars to holding buffer
006116  020304                        CMP     R3,R4           ;Moved all that need to be deleted?
006120  103775                        BLO     10$             ;Loop if not
006122  105012                        CLRB    (R2)            ;Put null at end of deleted string
                                        ;
                                        ;  Delete the characters from the buffer
                                        ;
006124  010503                        MOV     R5,R3
006126  112425                  7$:     MOVB    (R4)+,(R5)+     ;Move over characters to right of deleted ones
006130  001376                        BNE     7$
006132  010304                        MOV     R3,R4           ;Get new cursor character index
                                        ;
                                        ;  Redisplay line
                                        ;
006134  004737  010336'               CALL    PAINT           ;Redisplay from delete point to right end
                                        ;
                                        ;  Reposition cursor
                                        ;
006140  004737  010642'               CALL    CHRPOS          ;Reposition cursor
                                        ;
                                        ;  Finished
                                        ;
006144  012605                  9$:     MOV     (SP)+,R5
006146  012604                        MOV     (SP)+,R4
006150  012603                        MOV     (SP)+,R3
006152  012602                        MOV     (SP)+,R2
006154  000207                        RETURN
```

```
                                        .SBTTL  ICPBS  -- Backspace processing
                                ;-----------------------------------------------------------------------
                                ;  Process a Backspace character.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
006156  010446                  ICPBS:  MOV     R4,-(SP)
                                ;
                                ;  See if Gold key was pressed
                                ;
006160  013704  0000000G                MOV     SLCX,R4         ;Get cursor pointer
006164  105737  0000000G                TSTB    SLGOLD          ;Was gold key pressed?
006170  001423                          BEQ     1$              ;Br if not
                                ;
                                ;  Gold key was pressed.
                                ;  Exchange character under cursor with character to left of cursor.
                                ;
006172  105037  0000000G                CLRB    SLGOLD          ;Clear gold-key flag
006176  020427  0000000G                CMP     R4,#SLEBUF      ;Is cursor at left end of line?
006202  001402                          BEQ     3$              ;Br if yes
006204  105714                          TSTB    (R4)            ;Is cursor at right end of line?
006206  001003                          BNE     4$              ;Br if not
006210  004737  011154'          3$:     CALL    RNGBEL          ;Ring bell
006214  000425                          BR      9$              ;And ignore command
006216  111400                  4$:     MOVB    (R4),R0         ;Get character under cursor
006220  114464  000001                  MOVB    -(R4),1(R4)     ;Move char to left of cursor to cursor pos
006224  110014                          MOVB    R0,(R4)         ;Store old char to left of old cursor pos
006226  004737  010336'                 CALL    PAINT           ;Redisplay the line
006232  004737  010642'                 CALL    CHRPOS          ;Position cursor
006236  000414                          BR      9$
                                ;
                                ;  Gold key was not pressed.
                                ;  Exchange character under cursor with character to right of cursor.
                                ;
006240  112400                  1$:     MOVB    (R4)+,R0        ;Get current char under the cursor
006242  001762                          BEQ     3$              ;Br if we were at right end of line
006244  105714                          TSTB    (R4)            ;Is next character end of line?
006246  001760                          BEQ     3$              ;Br if yes
006250  111444                          MOVB    (R4),-(R4)      ;Move char to right of cursor under cursor
006252  110064  000001                  MOVB    R0,1(R4)
006256  004737  010336'                 CALL    PAINT           ;Redisplay the line
006262  005204                          INC     R4              ;Cursor moves right 1 character
006264  004737  010642'                 CALL    CHRPOS          ;Position cursor
                                ;
                                ;  Finished
                                ;
006270  012604                  9$:     MOV     (SP)+,R4
006272  000207                          RETURN
```

```
                                        .SBTTL  ICPESC -- Escape processing
                                ;-------------------------------------------------------------------
                                ;  Process an Escape character.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R5 = Escape character.
                                ;
006274                          ICPESC:
                                ;
                                ;  Set address of routine to be called to process 1st character after
                                ;  escape.
                                ;
006274  012700  001216'                 MOV     #EPCH1,R0       ;Routine for 1st char after escape
006300  004737  011602'                 CALL    CHK52           ;Is this a VT52 terminal?
006304  103002                         BCC     1$              ;Br if not
006306  012700  001240'                 MOV     #EP52,R0        ;Set routine for VT52
006312  010037  000000G         1$:     MOV     R0,SLCSR        ;Address of routine to process next char
006316  012737  000000G 000000G        MOV     #SLCSBF,SLCSPT  ;Set pointer to start of buffer
                                ;
                                ;  Finished
                                ;
006324  000207                          RETURN

                                ;-------------------------------------------------------------------
                                ;  Received a CSI control character from a VT200 terminal.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R5 = character.
                                ;
006326  012737  001304' 000000G ICPCSI: MOV     #EPCH2,SLCSR    ;Set address of routine to process next char
006334  012700  000000G                 MOV     #SLCSBF,R0      ;Get pointer to start of buffer
006340  112720  000133                  MOVB    #'[,(R0)+       ;Store "[" as 1st char of sequence
006344  010037  000000G                 MOV     R0,SLCSPT       ;Set pointer into control sequence buffer
006350  000207                          RETURN

                                ;-------------------------------------------------------------------
                                ;  Received a SS3 control character from a VT200 terminal.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R5 = Character.
                                ;
006352  012737  001304' 000000G ICPSS3: MOV     #EPCH2,SLCSR    ;Set address of routine to process next char
006360  012700  000000G                 MOV     #SLCSBF,R0      ;Get pointer to start of buffer
006364  112720  000117                  MOVB    #'O,(R0)+       ;Store "O" as 1st char of sequence
006370  010037  000000G                 MOV     R0,SLCSPT       ;Set pointer into control sequence buffer
006374  000207                          RETURN
```

```
                                        .SBTTL  ICPCTA -- Control-A processing
                                ;-----------------------------------------------------------------------
                                ;  Control-A -- toggle overstrike/insert mode
                                ;
006376  105737  000000G        ICPCTA: TSTB    SLOVER          ;In overstrike mode now?
006402  001004                        BNE     1$              ;Br if yes
006404  112737  000001  000000G       MOVB    #1,SLOVER       ;Enter overstrike mode
006412  000402                        BR      9$
006414  105037  000000G        1$:     CLRB    SLOVER          ;Exit overstrike mode
006420  000207                 9$:     RETURN
```

```
                                        .SBTTL  ICPCTC -- Control-C processing
                                ;-----------------------------------------------------------------------
                                ;  Process a control-C input character.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line index number.
                                ;   R5 = Current input character.
                                ;
006422                          ICPCTC:
                                ;
                                ;  If line is a .SCCA, treat control-C like control-Z
                                ;
006422  005761  000000G                 TST     LSCCA(R1)       ;Did program do a .SCCA?
006426  001402                          BEQ     3$              ;Br if not
006430  000137  006700'                 JMP     ICPCTZ          ;Treat control-C like control-Z
006434  032761  000000G 000000G 3$:     bit     #$scca,lsw5(r1) ;Suppressing control-c abort (RUN/SCCA)
006442  001401                          beq     2$              ;BR if no
006444  000207                          return

                                ;
                                ;  Clear edit buffer
                                ;
006446  012704  000000G         2$:     MOV     #SLEBUF,R4      ;Point to front of edit buffer
006452  105014                          CLRB    (R4)            ;Say buffer is empty
006454  004737  010336'                 CALL    PAINT           ;Clear line on screen
                                ;
                                ;  Echo "^C" to terminal
                                ;
006460  004737  011130'                 CALL    ECOCTL          ;Echo "^C"
                                ;
                                ;  Send "^C" to log file if we are doing logging
                                ;
006464  032737  000000G 000000G         BIT     #LF$IN,LOGFLG   ;Are we logging input characters?
006472  001407                          BEQ     1$              ;Br if not
006474  110500                          MOVB    R5,R0           ;Get control-C character
006476                                  OCALL   LOGCHR          ;Log Control-C
006504                                  OCALL   LOGCR           ;Log CR-LF
                                ;
                                ;  Stop program execution and enter Kmon
                                ;
006512  042761  000000G 000000G 1$:     BIC     #$SLINI,LSW7(R1);Say SL must reinitialize for next line
006520  004737  000000G                 CALL    STOP
```

```
                                        .SBTTL  ICPCTR -- Control-R processing
                                ;-----------------------------------------------------------------------
                                ;  Process control-R input character.
                                ;
006524  010246                  ICPCTR: MOV     R2,-(SP)
006526  010446                          MOV     R4,-(SP)
                                ;
                                ;  Save current cursor position and redisplay the line
                                ;
006530  013702  000000G                 MOV     SLCX,R2         ;Save current cursor character pointer
006534  012704  000000G                 MOV     #SLEBUF,R4      ;Redisplay entire line
006540  004737  010336'                 CALL    PAINT           ;Do the display
                                ;
                                ;  Reposition cursor
                                ;
006544  010204                          MOV     R2,R4           ;Get original cursor pointer
006546  004737  010642'                 CALL    CHRPOS          ;Reposition cursor
                                ;
                                ;  Finished
                                ;
006552  012604                          MOV     (SP)+,R4
006554  012602                          MOV     (SP)+,R2
006556  000207                          RETURN
```

```
                                        .SBTTL  ICPCTU -- Control-U processing
                                ;-----------------------------------------------------------------------
                                ;  Process a Control-U character.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line index number.
                                ;   R5 = Current input character.
                                ;
006560  010246                  ICPCTU: MOV     R2,-(SP)
006562  010346                          MOV     R3,-(SP)
006564  010446                          MOV     R4,-(SP)
                                ;
                                ;  See if Gold key (PF1) was pressed before control-U
                                ;
006566  105737  0000006                 TSTB    SLGOLD          ;Was Gold key pressed?
006572  001407                          BEQ     3$              ;Br if not
                                ;
                                ;  Gold key was pressed -- Put back previously deleted text
                                ;
006574  012702  0000006                 MOV     #SLDBUF,R2      ;Point to buffer with deleted text
006600  004737  010140'                 CALL    INSERT          ;Insert in front of cursor
006604  105037  0000006                 CLRB    SLGOLD          ;Clear gold-key flag
006610  000427                          BR      9$              ;Finished
                                ;
                                ;  Gold key was not pressed -- Delete all characters to left of cursor
                                ;  Check to see if there is anything to delete.
                                ;
006612  013704  0000006         3$:     MOV     SLCX,R4         ;Get pointer to char under cursor
006616  020427  0000006                 CMP     R4,#SLEBUF      ;Is cursor at left end now?
006622  001422                          BEQ     9$              ;Br if yes -- nothing to delete
                                ;
                                ;  Move deleted characters to holding buffer
                                ;
006624  012702  0000006                 MOV     #SLEBUF,R2      ;Point to regular edit buffer
006630  012703  0000006                 MOV     #SLDBUF,R3      ;Point to deleted-characters buffer
006634  112223                  1$:     MOVB    (R2)+,(R3)+     ;Save characters being deleted
006636  020204                          CMP     R2,R4           ;Have we saved all?
006640  103775                          BLO     1$              ;Br if not
006642  105013                          CLRB    (R3)            ;Put null at end of delete string
                                ;
                                ;  Delete characters from the buffer
                                ;
006644  012702  0000006                 MOV     #SLEBUF,R2      ;Point to start of string to delete
006650  112422                  2$:     MOVB    (R4)+,(R2)+     ;Move characters left over deleted ones
006652  001376                          BNE     2$              ;Loop till all moved
                                ;
                                ;  Redisplay the line
                                ;
006654  012704  0000006                 MOV     #SLEBUF,R4      ;Redisplay entire line
006660  004737  010336'                 CALL    PAINT
                                ;
                                ;  Reposition cursor
                                ;
006664  004737  010642'                 CALL    CHRPOS          ;Position cursor
                                ;
                                ;  Finished
                                ;
```

```
     58 006670  012604                    9$:     MOV     (SP)+,R4
     59 006672  012603                            MOV     (SP)+,R3
     60 006674  012602                            MOV     (SP)+,R2
     61 006676  000207                            RETURN
```

```
                                        .SBTTL  ICPCTZ -- Control-Z processing
                                ;-----------------------------------------------------------------------
                                ;  Process Control-Z character.
                                ;
                                ;  Inputs:
                                ;   R1 = Virtual line index number.
                                ;   R5 = Current input character.
                                ;
006700  010446                  ICPCTZ: MOV     R4,-(SP)
                                ;
                                ;  Move cursor to end of line
                                ;
006702  013704  000000G         1$:     MOV     SLCX,R4         ;Get current cursor pointer
006706  105724                  2$:     TSTB    (R4)+           ;Search for null at end of line
006710  001376                          BNE     2$              ;Loop till null hit
006712  005304                          DEC     R4              ;Point to null
006714  004737  010642'                  CALL    CHRPOS          ;Position cursor to end of line
                                ;
                                ;  Echo "^Z" to terminal
                                ;
006720  004737  011130'                  CALL    ECOCTL          ;Echo "^Z"
                                ;
                                ;  Echo carriage return / line feed
                                ;
006724  112700  000000G                 MOVB    #CR,R0          ;Get carriage return
006730  004737  011226'                  CALL    ECHO2           ;Echo it
006734  112700  000000G                 MOVB    #LF,R0          ;Get line feed
006740  004737  011226'                  CALL    ECHO2           ;Echo it
                                ;
                                ;  Save input line up to (but not including) control-Z
                                ;
006744  004737  007172'                  CALL    SAVLIN          ;Save input line
                                ;
                                ;  Insert control-Z into buffer
                                ;
006750  110524                          MOVB    R5,(R4)+        ;Insert control-Z into buffer
006752  105014                          CLRB    (R4)            ;Put null at end of buffer
                                ;
                                ;  We have finished input line
                                ;
006754  004737  007346'                  CALL    LINFIN          ;Say we have finished input line
                                ;
                                ;  Finished
                                ;
006760  012604                          MOV     (SP)+,R4
006762  000207                          RETURN
```

```
                                        .SBTTL  ICPRUB -- Rubout processing
                                ;-----------------------------------------------------------------------
                                ;  Process a rubout character.
                                ;
006764  010246                  ICPRUB: MOV     R2,-(SP)
                                ;
                                ;  See if Gold key (PF1) was pressed before rubout.
                                ;
006766  105737  0000000G                TSTB    SLGOLD          ;Was Gold key pressed?
006772  001407                          BEQ     1$              ;Br if not
                                ;
                                ;  Gold key was pressed -- Put back previously deleted character
                                ;
006774  012702  0000000G                MOV     #SLCBUF,R2      ;Point to deleted-character buffer
007000  004737  010140'                 CALL    INSERT          ;Insert in line
007004  105037  0000000G                CLRB    SLGOLD          ;Clear the Gold-key flag
007010  000411                          BR      9$              ;Finished
                                ;
                                ;  Gold key was not pressed -- Delete character to left of cursor
                                ;
007012  013702  0000000G        1$:     MOV     SLCX,R2         ;Get pointer to character under cursor
007016  020227  0000000G                CMP     R2,#SLEBUF      ;Is cursor at left end of line?
007022  101404                          BLOS    9$              ;Br if yes
                                ;
                                ;  Save character being deleted and then delete it
                                ;
007024  114237  0000000G        2$:     MOVB    -(R2),SLCBUF    ;Save character being deleted
007030  004737  007454'                 CALL    DELLFT          ;Delete character to left of cursor
                                ;
                                ;  Finished
                                ;
007034  012602                  9$:     MOV     (SP)+,R2
007036  000207                          RETURN
```

```
      1                                       .SBTTL  ICPNUL -- Null processing
      2                                 ;-----------------------------------------------------------------------
      3                                 ;  Process a null character
      4                                 ;
      5 007040  000207                  ICPNUL: RETURN
```

```
                                        .SBTTL  INWAIT -- Wait for input characters
                                ;-----------------------------------------------------------------
                                ;  INWAIT is called to wait for input characters.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number
                                ;
007042                          INWAIT:
                                ;
                                ;  If we previously suspended input from silo buffer to prevent
                                ;  input buffer overrun, reenable input now.
                                ;
007042  042761  000000G 000000G        BIC     #$XSTOP,LSW6(R1);Reenable input from silo buffer
007050  052761  000000G 000000G        BIS     #$NDICP,LSW10(R1);Say line needs input character servicing
007056  005237  000000G                INC     NEDCDI          ;Say input character processing needed
                                ;
                                ;  Suspend job until activation character is received.
                                ;
007062                          1$:     OCALL   SIGWAT          ;Signal virtual line wait condition
007070  042761  000000G 000000G        BIC     #$NOIN,LSW3(R1) ;Allow input to be accepted for line
007076  042761  000000G 000000G        BIC     #$SUCF,LSW9(R1) ;No longer in startup command file
007104  004737  000000G          2$:   CALL    CHKABT          ;See if job has been aborted
007110                                  DISABL                  ;;;** Disable **
007116  005761  000000G                TST     LACTIV(R1)      ;;;Got any activation chars yet?
007122  001017                        BNE     3$              ;;;Br if yes
007124  032761  000000G 000000G        BIT     #$NOINT,LSW7(R1);;;Is program being run non-interactively?
007132  001006                        BNE     4$              ;;;Br if yes
007134  013761  000000G 000000G        MOV     VINTIO,LHIPCT(R1);;;Reset high-priority hit limit for job
007142  013761  000000G 000000G        MOV     VQUAN1,LITIME(R1);;;Reset interactive CPU time limit
007150  012700  000000G          4$:   MOV     #S$INWT,R0      ;;;Get waiting-for-input state
007154  004737  000000G                CALL    QHDSPN          ;;;Suspend job and wait for activation char
007160  000751                        BR      2$              ;Go check again
                                ;
                                ;  There are input characters available
                                ;
007162                          3$:     ENABL                   ;** Enable **
                                ;
                                ;  Finished
                                ;
007170  000207                          RETURN
```

```
                                        .SBTTL  SAVLIN -- Save current input line
                                ;-----------------------------------------------------------------------
                                ;  SAVLIN is called to save the current input line as the "last line"
                                ;  which can be recalled later by use of the up-arrow key.
                                ;
007172  010246                  SAVLIN: MOV     R2,-(SP)
007174  010346                          MOV     R3,-(SP)
                                ;
                                ;  If line is null, there is nothing to save
                                ;
007176  012702  000000G                 MOV     #SLEBUF,R2      ;Point to current line buffer
007202  105712                          TSTB    (R2)            ;Is current line null?
007204  001450                          BEQ     9$              ;Br if yes -- Don't save a null line
                                ;
                                ;  If current line matches last line, don't push saved lines
                                ;
007206  013703  000000G                 MOV     SLSPTR,R3       ;Point to most recently saved line
007212  020327  000000G         4$:     CMP     R3,#SLLEND      ;Gone past end of save buffer?
007216  103402                          BLO     6$              ;Br if not
007220  012703  000000G                 MOV     #SLLBUF,R3      ;Wrap to top of buffer
007224  121223                  6$:     CMPB    (R2),(R3)+      ;Does next char in cur line match saved line?
007226  001003                          BNE     5$              ;Br if not
007230  105722                          TSTB    (R2)+           ;Have we reached end of current line?
007232  001367                          BNE     4$              ;Loop if not
007234  000434                          BR      9$              ;Line is same as last saved, don't save again
                                ;
                                ;  We want to save the current line in the save buffer.
                                ;  Copy to save buffer above previous saved line.
                                ;
007236  105722                  5$:     TSTB    (R2)+           ;Find null at end of current command
007240  001376                          BNE     5$
007242  013703  000000G                 MOV     SLSPTR,R3       ;Point to start of last saved line
007246  020327  000000G         8$:     CMP     R3,#SLLBUF      ;Gone past top of save buffer?
007252  101002                          BHI     10$             ;Br if not
007254  012703  000000G                 MOV     #SLLEND,R3      ;Wrap around to end of save buffer
007260  114243                  10$:    MOVB    -(R2),-(R3)     ;Copy new line into save buffer
007262  020227  000000G                 CMP     R2,#SLEBUF      ;Copied all of new line?
007266  101367                          BHI     8$              ;Br if not
007270  010337  000000G                 MOV     R3,SLSPTR       ;Save pointer to start of most recent line
                                ;
                                ;  Now null out the remainder of any command we may have partially covered
                                ;
007274  020327  000000G         11$:    CMP     R3,#SLLBUF      ;Hit top of save buffer?
007300  101002                          BHI     12$             ;Br if not
007302  012703  000000G                 MOV     #SLLEND,R3      ;Wrap around to end of save buffer
007306  105743                  12$:    TSTB    -(R3)           ;Reached null that terminates prev line?
007310  001402                          BEQ     13$             ;Br if yes
007312  105013                          CLRB    (R3)            ;Null out remainder of saved line
007314  000767                          BR      11$
                                ;
                                ;  Reset recall cycle
                                ;
007316  005037  000000G         13$:    CLR     SLCYC1          ;Say no recall cycle active
007322  005037  000000G                 CLR     SLCYC2
                                ;
                                ;  Set recall pointer to recall last saved line
                                ;
```

```
     58 007326  013737  000000G 000000G 9$:     MOV     SLSPTR,SLLPTR   ;Set ptr to next command to recall
     59 007334  105037  000000G                 CLRB    SLDOWN          ;Say down-arrow was not last command
     60                                 ;
     61                                 ;  Finished
     62                                 ;
     63 007340  012603                          MOV     (SP)+,R3
     64 007342  012602                          MOV     (SP)+,R2
     65 007344  000207                          RETURN
```

```
                                        .SBTTL  LINFIN -- Terminate input line
                                ;-----------------------------------------------------------------------
                                ;  LINFIN is called when an input line is completely acquired.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
007346  010246                  LINFIN: MOV     R2,-(SP)
007350  010346                          MOV     R3,-(SP)
                                ;
                                ;  Make sure job priority is not higher than normal (non-single character
                                ;  activation) input complete.
                                ;
007352  026127  000000G 000000G         CMP     LSTATE(R1),#S$TTFN;Is prio higher than normal TT input?
007360  103004                          BHIS    3$              ;Br if not
007362  012700  000000G               MOV     #S$TTFN,R0      ;Lower priority of this job
007366  004737  000000G               CALL    ENQTL           ;Requeue at tail of S$TTFN state queue
                                ;
                                ;  If terminal logging is wanted, log the line now
                                ;
007372  032737  000000G 000000G 3$:     BIT     #LF$IN,LOGFLG   ;Are we logging input characters?
007400  001410                          BEQ     5$              ;Br if not
007402  012702  000000G               MOV     #SLEBUF,R2      ;Point to line buffer
007406  112200                  4$:     MOVB    (R2)+,R0        ;Get next character from the line
007410  001404                          BEQ     5$              ;Br when null hit
007412                                  OCALL   LOGCHR          ;Log the character
007420  000772                          BR      4$
                                ;
                                ;  Set up pointer to character string to be passed to program
                                ;
007422  012737  000000G 000000G 5$:     MOV     #SLEBUF,SLOPTR  ;Set pointer for GTSLCH routine
                                ;
                                ;  Set cursor position at end of line
                                ;
007430  113761  000000G 000000G         MOVB    SLECOL,LCOL(R1) ;Remember cursor position at end of line
                                ;
                                ;  Clear field width activation and field width limit values
                                ;
007436  005061  000000G               CLR     LFWLIM(R1)      ;No field width limit now
007442  005061  000000G               CLR     LAFSIZ(R1)      ;No field width activation now
                                ;
                                ;  Finished
                                ;
007446  012603                          MOV     (SP)+,R3
007450  012602                          MOV     (SP)+,R2
007452  000207                          RETURN
```

```
                                        .SBTTL  DELLFT -- Delete character to left of cursor
                                ;-----------------------------------------------------------------------
                                ;  DELLFT is called to delete the character to the left of the current
                                ;  cursor position and redisplay any characters to the right of the
                                ;  deleted character.
                                ;  The cursor is left positioned on the character that was to the right
                                ;  of the deleted character.
                                ;
                                ;  Inputs:
                                ;   SLCX   = Pointer to character under the cursor.
                                ;   SLCCOL = Column number where cursor is positioned.
                                ;
007454  010346                  DELLFT: MOV     R3,-(SP)
007456  010446                          MOV     R4,-(SP)
                                ;
                                ;  If cursor is already at left-most character then there is nothing
                                ;  to do.
                                ;
007460  013704  000000G                 MOV     SLCX,R4         ;Get pointer to character under cursor
007464  020427  000000G                 CMP     R4,#SLEBUF      ;Is cursor at left end?
007470  001413                          BEQ     9$              ;Br if at left-mode char -- Nothing to delete
                                ;
                                ;  Remove deleted character from buffer
                                ;
007472  005304                          DEC     R4              ;Point to the deleted character
007474  111437  000000G                 MOVB    (R4),SLCBUF     ;Save deleted character
007500  010403                          MOV     R4,R3
007502  116323  000001          1$:     MOVB    1(R3),(R3)+     ;Move chars left over deleted char
007506  001375                          BNE     1$
                                ;
                                ;  Redisplay any characters to the right of the deleted character
                                ;
007510  004737  010336'                 CALL    PAINT           ;Redisplay chars to left of deleted char
                                ;
                                ;  Reposition cursor to character that was to right of one deleted.
                                ;
007514  004737  010642'                 CALL    CHRPOS          ;Reposition cursor
                                ;
                                ;  Finished
                                ;
007520  012604                  9$:     MOV     (SP)+,R4
007522  012603                          MOV     (SP)+,R3
007524  000207                          RETURN
```

```
                                        .SBTTL  SLMVUP -- Move up to previous stored command
                                ;-----------------------------------------------------------------------
                                ;  This routine returns a pointer to the previous saved command moving
                                ;  in an up-arrow direction.
                                ;
                                ;  Inputs:
                                ;   R5 = Pointer to current command.
                                ;
                                ;  Outputs:
                                ;   R5 = Pointer to previous command.
                                ;
007526  010446                  SLMVUP: MOV     R4,-(SP)
                                ;
                                ;  Skip up to the null at the end of the current command
                                ;
007530  010504                          MOV     R5,R4
007532  020427  0000006         1$:     CMP     R4,#SLLEND      ;Hit end of buffer?
007536  103402                          BLO     2$              ;Br if not
007540  012704  0000006                 MOV     #SLLBUF,R4      ;Wrap around to the top
007544  105724                  2$:     TSTB    (R4)+           ;Reached null at end of saved command?
007546  001371                          BNE     1$              ;Br if not
                                ;
                                ;  Now skip over nulls at the end of the command
                                ;
007550  020427  0000006         3$:     CMP     R4,#SLLEND      ;Hit end of buffer?
007554  103402                          BLO     4$              ;Br if not
007556  012704  0000006                 MOV     #SLLBUF,R4      ;Wrap around to the top
007562  020405                  4$:     CMP     R4,R5           ;Wrapped around to current cmd?
007564  001411                          BEQ     9$              ;Br if nothing to recall
007566  105724                          TSTB    (R4)+           ;More nulls to skip?
007570  001767                          BEQ     3$              ;Br if so
                                ;
                                ;  Point to 1st character of command
                                ;
007572  020427  0000006         5$:     CMP     R4,#SLLBUF      ;At top of buffer now?
007576  101002                          BHI     6$              ;Br if not
007600  012704  0000006                 MOV     #SLLEND,R4      ;Wrap around to bottom
007604  005304                  6$:     DEC     R4              ;Point to 1st char of next command
                                ;
                                ;  Finished
                                ;
007606  010405                          MOV     R4,R5           ;Return pointer in R5
007610  012604                  9$:     MOV     (SP)+,R4
007612  000207                          RETURN
```

```
                                        .SBTTL  SLMVDN -- Move down to previous stored command
                                ;-----------------------------------------------------------------------
                                ;  This routine returns a pointer to the previous saved command moving
                                ;  in a down-arrow direction.
                                ;
                                ;  Inputs:
                                ;   R5 = Pointer to current command.
                                ;
                                ;  Outputs:
                                ;   R5 = Pointer to previous command.
                                ;
007614  010446                  SLMVDN: MOV     R4,-(SP)
                                ;
                                ;  Skip over nulls in front of current command
                                ;
007616  010504                          MOV     R5,R4
007620  020427  000000G         1$:     CMP     R4,#SLLBUF      ;At top of buffer now?
007624  101002                          BHI     2$              ;Br if not
007626  012704  000000G                 MOV     #SLLEND,R4      ;Wrap around to the bottom
007632  105744                  2$:     TSTB    -(R4)           ;Is this character null?
007634  001003                          BNE     3$              ;Br if not
007636  020405                          CMP     R4,R5           ;Wrapped around to beginning?
007640  001367                          BNE     1$              ;Loop if not
007642  000416                          BR      9$              ;Nothing to recall
                                ;
                                ;  Now skip back to null at front of the prev command
                                ;
007644  020427  000000G         3$:     CMP     R4,#SLLBUF      ;At top of buffer now?
007650  101002                          BHI     4$              ;Br if not
007652  012704  000000G                 MOV     #SLLEND,R4      ;Wrap around to the bottom
007656  105744                  4$:     TSTB    -(R4)           ;Is this character null?
007660  001371                          BNE     3$              ;Loop if not
                                ;
                                ;  Now move forward to the 1st character of next command
                                ;
007662  005204                  5$:     INC     R4              ;Point to next char
007664  020427  000000G                 CMP     R4,#SLLEND      ;Past end of buffer?
007670  103402                          BLO     6$              ;Br if not
007672  012704  000000G                 MOV     #SLLBUF,R4      ;Wrap around to the top
                                ;
                                ;  Finished
                                ;
007676  010405                  6$:     MOV     R4,R5           ;Return pointer in R5
007700  012604                  9$:     MOV     (SP)+,R4
007702  000207                          RETURN
```

```
                                        .SBTTL  RSTLIN -- Restore a full line
                                ;-----------------------------------------------------------------
                                ;  Restore a full line.
                                ;
                                ;  Inputs:
                                ;   R5 = Pointer to asciz string to be restored.
                                ;
007704  010246                  RSTLIN: MOV     R2,-(SP)
007706  010346                          MOV     R3,-(SP)
007710  010446                          MOV     R4,-(SP)
007712  010546                          MOV     R5,-(SP)
007714  010502                          MOV     R5,R2           ;Save pointer to line start
007716  010504                          MOV     R5,R4           ; "
                                ;
                                ;  See if there is anything to restore
                                ;
007720  105712                          TSTB    (R2)            ;Do we have a saved command?
007722  001003                          BNE     1$              ;Br if yes
007724  004737  011154'                 CALL    RNGBEL          ;Ring bell if not
007730  000441                          BR      9$
                                ;
                                ;  Count number of characters in saved line
                                ;
007732  005005                  1$:     CLR     R5              ;Count # chars in line
007734  004737  010046'          5$:     CALL    RSTCHR          ;Get next char from line buffer
007740  001402                          BEQ     6$              ;Br if yes
007742  005205                          INC     R5              ;Count # chars in saved line
007744  000773                          BR      5$              ;Keep searching for the end of the line
                                ;
                                ;  See if saved line will fit
                                ;
007746  004737  010522'          6$:     CALL    MAXLEN          ;Determine max allowed line length
007752  020500                          CMP     R5,R0           ;Will saved line fit?
007754  101403                          BLOS    4$              ;Br if yes
007756  004737  011154'                 CALL    RNGBEL          ;Ring bell if not
007762  000424                          BR      9$
                                ;
                                ;  Move saved line to edit buffer
                                ;
007764  012703  000000G         4$:     MOV     #SLEBUF,R3      ;Point to edit buffer
007770  010402                          MOV     R4,R2           ;Point to start of saved line
007772  004737  010046'          2$:     CALL    RSTCHR          ;Get next char to be restored
007776  110023                          MOVB    R0,(R3)+        ;Move saved char to edit buffer
010000  001374                          BNE     2$              ;Loop until null moved
                                ;
                                ;  Display the line
                                ;
010002  012704  000000G                 MOV     #SLEBUF,R4      ;Display entire line
010006  004737  010336'                 CALL    PAINT
                                ;
                                ;  See if we should activate due to field width being filled
                                ;
010012  016102  000000G                 MOV     LAFSIZ(R1),R2   ;Was field width activation specified?
010016  001406                          BEQ     9$              ;Br if not
010020  020502                          CMP     R5,R2           ;Does saved line = field width?
010022  103404                          BLO     9$              ;Br if not
010024  004737  007172'                 CALL    SAVLIN          ;Save the complete line
```

```
     58 010030  004737  007346'                    CALL    LINFIN          ;Input line is complete
     59                                 ;
     60                                 ;  Finished
     61                                 ;
     62 010034  012605                  9$:        MOV     (SP)+,R5
     63 010036  012604                             MOV     (SP)+,R4
     64 010040  012603                             MOV     (SP)+,R3
     65 010042  012602                             MOV     (SP)+,R2
     66 010044  000207                             RETURN
```

```
                                ;+-----------------------------------------------------------------------------
                                ;  Get the next character from a saved line and handle buffer wraparound.
                                ;
                                ;  Inputs:
                                ;   R2 = Points to next character to get.
                                ;
                                ;  Outputs:
                                ;   R0 = Character we got.
                                ;   R2 = Pointer to next character to get.
                                ;   Condition codes set for value returned in R0.
                                ;
010046                          RSTCHR:
                                ;
                                ;  Get the character we want to return
                                ;
010046  112200                          MOVB    (R2)+,R0        ;Get character to be returned
                                ;
                                ;  See if we are getting from within SLLBUF and need to worry about wrap.
                                ;
010050  020227  000000G                 CMP     R2,#SLLEND      ;Are we at end of SLLBUF?
010054  001002                          BNE     9$              ;Br if not
010056  012702  000000G                 MOV     #SLLBUF,R2      ;Wrap around to top of SLLBUF
                                ;
                                ;  Finished
                                ;
010062  005700                  9$:     TST     R0              ;Set sign for value in R0
010064  000207                          RETURN
```

```
                                        .SBTTL  OSTRIK -- Overstrike
                                ;-------------------------------------------------------------------------
                                ;  Replace the character under the cursor with a new (overstriking) character.
                                ;
                                ;  Inputs:
                                ;   R0 = New (overstriking) character.
                                ;   R1 = Job index number.
                                ;
010066  010246                  OSTRIK: MOV     R2,-(SP)
010070  010446                          MOV     R4,-(SP)
                                ;
                                ;  If we are positioned to the end of the line, do an insert
                                ;
010072  013704  0000006                 MOV     SLCX,R4         ;Get pointer to char under cursor
010076  105714                          TSTB    (R4)            ;At end of line now?
010100  001006                          BNE     1$              ;Br if not
010102  012702  0000006                 MOV     #SLCBUF,R2      ;Point to char buffer
010106  110012                          MOVB    R0,(R2)         ;Store char into insert buffer
010110  004737  010140'                 CALL    INSERT          ;Insert the character
010114  000406                          BR      9$
                                ;
                                ;  We are positioned in the middle of the line.
                                ;  Replace the character being overstruck.
                                ;
010116  110014                  1$:     MOVB    R0,(R4)         ;Replace char being overstruck
010120  004737  010336'                 CALL    PAINT           ;Redisplay the line
                                ;
                                ;  Reposition the cursor
                                ;
010124  005204                          INC     R4              ;Position past overstrike
010126  004737  010642'                 CALL    CHRPOS          ;Reposition cursor
                                ;
                                ;  Finished
                                ;
010132  012604                  9$:     MOV     (SP)+,R4
010134  012602                          MOV     (SP)+,R2
010136  000207                          RETURN
```

```
                                        .SBTTL  INSERT -- Insert string into edit buffer
                                ;-------------------------------------------------------------------
                                ;  INSERT is called to insert an asciz text string into the edit string.
                                ;  The text is inserted immediately to the left of the cursor position.
                                ;  After the insertion is made, the line is redisplayed and the cursor
                                ;  is left pointing to the right of the inserted string.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R2 = Pointer to asciz string to be inserted.
                                ;
010140  010246                  INSERT: MOV     R2,-(SP)
010142  010346                          MOV     R3,-(SP)
010144  010446                          MOV     R4,-(SP)
010146  010546                          MOV     R5,-(SP)
                                ;
                                ;  Count number of characters in string being inserted
                                ;
010150  010203                          MOV     R2,R3           ;Get pointer to insert string
010152  105723                  1$:     TSTB    (R3)+           ;Search for null at end of string
010154  001376                          BNE     1$              ;Loop till null hit
010156  160203                          SUB     R2,R3           ;Get length of string
010160  005303                          DEC     R3              ;Don't count null at end
010162  003003                          BGT     7$              ;Br if there are some characters to insert
                                ;
                                ;  There are no characters in string being inserted
                                ;
010164  004737  011154'                 CALL    RNGBEL          ;Ring the bell
010170  000455                          BR      9$              ;We are finished
                                ;
                                ;  Locate the end of the current edit line
                                ;
010172  013705  000000G         7$:     MOV     SLCX,R5         ;Get pointer to character under cursor
010176  010504                          MOV     R5,R4
010200  105724                  2$:     TSTB    (R4)+           ;Search for null at end of string
010202  001376                          BNE     2$
                                ;
                                ;  See if there is enough free space remaining in edit buffer to do
                                ;  the insertion.
                                ;
010204  010405                          MOV     R4,R5           ;Get pointer past null
010206  162705  000001G                 SUB     #SLEBUF+1,R5    ;Calc number of characters in current string
010212  060305                          ADD     R3,R5           ;Add # characters in insert string
010214  004737  010522'                 CALL    MAXLEN          ;Compute maximum allowed line length
010220  020500                          CMP     R5,R0           ;Compare line with insert with max allowed
010222  101403                          BLOS    3$              ;Br if ok
                                ;
                                ;  There is not enough space for the insertion
                                ;
010224  004737  011154'                 CALL    RNGBEL          ;Ring bell
010230  000435                          BR      9$
                                ;
                                ;  There is enough space to do the insertion.
                                ;  Move characters over to make room for the insertion.
                                ;
010232  010405                  3$:     MOV     R4,R5           ;Pointer past null at end of current string
010234  060305                          ADD     R3,R5           ;Add number of characters to be inserted
```

```
     58 010236  114445                4$:     MOVB    -(R4),-(R5)     ;Move characters to right
     59 010240  020437  000000G             CMP     R4,SLCX         ;Have we moved enough?
     60 010244  101374                      BHI     4$              ;Br if not
     61                                ;
     62                                ;  Insert the string
     63                                ;
     64 010246  010405                5$:     MOV     R4,R5           ;Get pointer to insertion point
     65 010250  112225                6$:     MOVB    (R2)+,(R5)+     ;Insert the string
     66 010252  077302                      SOB     R3,6$
     67                                ;
     68                                ;  Redisplay the line
     69                                ;
     70 010254  004737  010336'             CALL    PAINT           ;Redisplay the edit line
     71                                ;
     72                                ;  Reposition the cursor
     73                                ;
     74 010260  010504                      MOV     R5,R4           ;Get desired cursor position
     75 010262  004737  010642'             CALL    CHRPOS          ;Reposition cursor
     76                                ;
     77                                ;  Count total number of characters in the line and see if we should
     78                                ;  activate due to field-width activation.
     79                                ;
     80 010266  016102  000000G             MOV     LAFSIZ(R1),R2   ;Is field-width activation in effect?
     81 010272  001414                      BEQ     9$              ;Br if not
     82 010274  013703  000000G             MOV     SLCX,R3         ;Point to cursor position
     83 010300  105723                8$:     TSTB    (R3)+           ;Is there another character in buffer?
     84 010302  001376                      BNE     8$              ;Loop till null hit
     85 010304  162703  000001G             SUB     #SLEBUF+1,R3    ;Calc total number of characters in string
     86 010310  020302                      CMP     R3,R2           ;Have we reached activation size yet?
     87 010312  103404                      BLO     9$              ;Br if not
     88 010314  004737  007172'             CALL    SAVLIN          ;Save input line
     89 010320  004737  007346'             CALL    LINFIN          ;Completed input line
     90                                ;
     91                                ;  Finished
     92                                ;
     93 010324  012605                9$:     MOV     (SP)+,R5
     94 010326  012604                       MOV     (SP)+,R4
     95 010330  012603                       MOV     (SP)+,R3
     96 010332  012602                       MOV     (SP)+,R2
     97 010334  000207                       RETURN
```

```
                                        .SBTTL  PAINT  -- Redisplay current edit line
                                ;-------------------------------------------------------------------------
                                ;  PAINT is called to redisplay all or a portion of the current edit line.
                                ;  The cursor is left pointing beyond the right end of the line.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R4 = Pointer to character where redisplay is to begin.
                                ;
                                ;  Outputs:
                                ;   SLCX   = Pointer to null at end of edit string.
                                ;   SLCCOL = Cursor position at end of line.
                                ;
010336  010246                  PAINT:  MOV     R2,-(SP)
010340  010346                          MOV     R3,-(SP)
010342  010446                          MOV     R4,-(SP)
010344  010546                          MOV     R5,-(SP)
                                ;
                                ;  Position cursor to point where display is to begin
                                ;
010346  004737  010564'                 CALL    COLCLC          ;Calculate position where redisplay begins
010352  004737  010664'                 CALL    CURPOS          ;Position cursor there (R3 has column number)
                                ;
                                ;  Begin loop to display each character.
                                ;
010356  112400                  1$:     MOVB    (R4)+,R0        ;Get next character to display
010360  001427                          BEQ     5$              ;Br if hit end of string
010362  120027  000000G                 CMPB    R0,#TAB         ;Is this a tab?
010366  001015                          BNE     3$              ;Br if not tab
                                ;
                                ;  Character being displayed is a tab.
                                ;  Convert to spaces.
                                ;
010370  010300                          MOV     R3,R0           ;Save position at start of tab field
010372  062703  000010                  ADD     #8.,R3          ;Calculate position beyond end of tab field
010376  042703  000007                  BIC     #7,R3
010402  010302                          MOV     R3,R2
010404  160002                          SUB     R0,R2           ;Get # spaces needed to simulate the tab
010406  112700  000040                  MOVB    #' ,R0          ;Get a space character
010412  004737  011210'          2$:     CALL    ECHO            ;Send the space
010416  077203                          SOB     R2,2$           ;Send as many as needed for tab
010420  000756                          BR      1$
                                ;
                                ;  Character is not a tab
                                ;
010422  004737  011210'          3$:     CALL    ECHO            ;Send the character to the terminal
010426  120027  000037                  CMPB    R0,#37          ;Is this a printing character?
010432  101751                          BLOS    1$              ;Br if not
010434  005203                          INC     R3              ;Advance column number
010436  000747                          BR      1$
                                ;
                                ;  We finished displaying all characters in buffer.
                                ;  See if we need to display blanks to wipe out deleted characters at end.
                                ;
010440  005304                  5$:     DEC     R4              ;Point back to null at end of line
010442  010437  000000G                 MOV     R4,SLCX         ;Save index where we want to leave cursor
010446  013702  000000G                 MOV     SLECOL,R2       ;Get old end-of-line column number
```

```
     58 010452  160302                        SUB     R3,R2           ;Do we need to erase any old chars?
     59 010454  003411                        BLE     7$              ;Br if not
     60                                 ;
     61                                 ;  Display blanks to erase any characters that used to be at end of line
     62                                 ;
     63 010456  010204                        MOV     R2,R4           ;Save number of columns being erased
     64 010460  116100  000000G               MOVB    LRBFIL(R1),R0   ;Get rubout-filler character for this line
     65 010464  004737  011210'       6$:     CALL    ECHO            ;Send the space
     66 010470  077203                        SOB     R2,6$           ;Loop for as many as needed
     67                                 ;
     68                                 ;  Now leave cursor positioned at end of line
     69                                 ;
     70 010472  004737  010776'       8$:     CALL    CSRLFT          ;Move cursor 1 column left
     71 010476  077403                        SOB     R4,8$           ;Move back over erased characters
     72                                 ;
     73                                 ;  Save information about column positions.
     74                                 ;
     75 010500  010337  000000G       7$:     MOV     R3,SLCCOL       ;Cursor position
     76 010504  010337  000000G               MOV     R3,SLECOL       ;End-of-line column
     77                                 ;
     78                                 ;  Finished
     79                                 ;
     80 010510  012605                9$:     MOV     (SP)+,R5
     81 010512  012604                        MOV     (SP)+,R4
     82 010514  012603                        MOV     (SP)+,R3
     83 010516  012602                        MOV     (SP)+,R2
     84 010520  000207                        RETURN
```

```
                                        .SBTTL  MAXLEN -- Compute maximum allowed line length
                                ;-----------------------------------------------------------------------
                                ;  MAXLEN is called to compute the maximum length allowed for an
                                ;  edit line.  It takes into consideration the following constraints:
                                ;   1. Size of edit line buffer.
                                ;   2. Field width limit size.
                                ;   3. Field with activation size.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
                                ;  Outputs:
                                ;   R0 = Maximum line length.
                                ;
010522  010246                  MAXLEN: MOV     R2,-(SP)
                                ;
                                ;  Start with edit buffer length.
                                ;
010524  012700  000000G                 MOV     #SLMXLN,R0      ;Length of edit buffer
                                ;
                                ;  See if a field width limit is in effect.
                                ;
010530  016102  000000G                 MOV     LFWLIM(R1),R2   ;Is a field width limit in effect?
010534  001403                          BEQ     1$              ;Br if not
010536  020002                          CMP     R0,R2           ;Is current limit smaller?
010540  101401                          BLOS    1$              ;Br if yes
010542  010200                          MOV     R2,R0           ;Use field width limit
                                ;
                                ;  See if field width activation is in effect.
                                ;
010544  016102  000000G         1$:     MOV     LAFSIZ(R1),R2   ;Is field width activation in effect?
010550  001403                          BEQ     2$              ;Br if not
010552  020002                          CMP     R0,R2           ;Is current limit smaller?
010554  101401                          BLOS    2$              ;Br if yes
010556  010200                          MOV     R2,R0           ;Use field activation width as limit
                                ;
                                ;  Finished
                                ;
010560  012602                  2$:     MOV     (SP)+,R2
010562  000207                          RETURN
```

```
                                        .SBTTL  COLCLC -- Calculate column position of a character
                                ;-----------------------------------------------------------------------
                                ;  COLCLC is called to compute the display column position of a specified
                                ;  character in the edit buffer.
                                ;
                                ;  Inputs:
                                ;   R4 = Pointer to character whose position is to be computed.
                                ;
                                ;  Outputs:
                                ;   R3 = Column position of the character.
                                ;
010564  010246                  COLCLC: MOV     R2,-(SP)
                                ;
                                ;  Initialize character info
                                ;
010566  012702  000000G                 MOV     #SLEBUF,R2      ;Get pointer to start of edit buffer
010572  013703  000000G                 MOV     SLSCOL,R3       ;Get col # of left-most character
                                ;
                                ;  Loop through all characters counting columns
                                ;
010576  020204                  1$:     CMP     R2,R4           ;Have we reached character of interest?
010600  103016                          BHIS    4$              ;Br if yes
                                ;
                                ;  Get next character to check
                                ;
010602  112200                          MOVB    (R2)+,R0        ;Get next character
010604  120027  000000G                 CMPB    R0,#TAB         ;Is character a tab?
010610  001005                          BNE     2$              ;Br if not
                                ;
                                ;  Character is tab.  Advance to next tab stop.
                                ;
010612  062703  000010                  ADD     #8.,R3          ;Advance to next tab stop
010616  042703  000007                  BIC     #7,R3
010622  000765                          BR      1$
                                ;
                                ;  Character is not tab.
                                ;  See if it is a printing character.
                                ;
010624  120027  000037          2$:     CMPB    R0,#37          ;Is this a printing or control character?
010630  101762                          BLOS    1$              ;Br if control character
010632  005203                          INC     R3              ;Count one column for the character
010634  000760                          BR      1$              ;Go process more characters
                                ;
                                ;  Finished
                                ;
010636  012602                  4$:     MOV     (SP)+,R2
010640  000207                          RETURN
```

```
                                        .SBTTL  CHRPOS -- Move cursor to a specific character
                                ;-----------------------------------------------------------------------
                                ;  CHRPOS is called to position the cursor to a specified character in the
                                ;  edit buffer.
                                ;
                                ;  Inputs:
                                ;   R4 = Pointer to character to which cursor is to be positioned.
                                ;
010642  010346                  CHRPOS: MOV     R3,-(SP)
                                ;
                                ;  Calculate position of desired character
                                ;
010644  004737  010564'                 CALL    COLCLC          ;Calculate col position of character
                                ;
                                ;  Position cursor there
                                ;
010650  004737  010664'                 CALL    CURPOS          ;Position cursor
                                ;
                                ;  Remember which character cursor is pointing to
                                ;
010654  010437  000000G                 MOV     R4,SLCX         ;Save current cursor pointer
                                ;
                                ;  Finished
                                ;
010660  012603                          MOV     (SP)+,R3
010662  000207                          RETURN
```

```
                                        .SBTTL  CURPOS -- Move cursor to a specified column
                                ;-----------------------------------------------------------------------
                                ;  CURPOS is called to move the cursor to a specified column.
                                ;
                                ;  Inputs:
                                ;   R3 = Position to which cursor is to be moved.
                                ;   SLCCOL = Current cursor position.
                                ;
                                ;  Outputs:
                                ;   SLCCOL = Position where cursor has been positioned.
                                ;
010664  010246                  CURPOS: MOV     R2,-(SP)
                                ;
                                ;  Calculate how many columns cursor needs to be moved
                                ;
010666  010302                          MOV     R3,R2           ;Get desired column position
010670  163702  0000006                 SUB     SLCCOL,R2       ;Calculate number of columns to move
010674  001413                          BEQ     9$              ;Br if no movement necessary
010676  002404                          BLT     2$              ;Br if need to move left
                                ;
                                ;  Move cursor right
                                ;
010700  004737  010730'         1$:     CALL    CSRRIT          ;Move cursor right 1 column
010704  077203                          SOB     R2,1$           ;Loop if need to move more
010706  000404                          BR      4$
                                ;
                                ;  Move cursor left
                                ;
010710  005402                  2$:     NEG     R2              ;Get positive column count
010712  004737  010776'         3$:     CALL    CSRLFT          ;Move cursor left 1 column
010716  077203                          SOB     R2,3$           ;Loop if need to move more
                                ;
                                ;  Finished move
                                ;
010720  010337  0000006         4$:     MOV     R3,SLCCOL       ;Save new cursor position
                                ;
                                ;  Finished
                                ;
010724  012602                  9$:     MOV     (SP)+,R2
010726  000207                          RETURN
```

```
                                        .SBTTL  CSRRIT -- Generate control sequence to move cursor right
                                ;-----------------------------------------------------------------------
                                ;  CSRRIT is called to generate the terminal control sequence to move
                                ;  the cursor right one column.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
010730  010246                  CSRRIT: MOV     R2,-(SP)
                                ;
                                ;  Get control sequence based on terminal type
                                ;
010732  012702  010766'                 MOV     #MRS100,R2      ;Assume VT100 terminal
010736  004737  011602'                 CALL    CHK52           ;Is this a VT52?
010742  103002                          BCC     1$              ;Br if not
010744  012702  010772'                 MOV     #MRS52,R2       ;Get VT52 control sequence
                                ;
                                ;  Send control sequence to terminal
                                ;
010750  112200                  1$:     MOVB    (R2)+,R0        ;Get next character from sequence
010752  001403                          BEQ     9$              ;Br if hit end
010754  004737  011210'                 CALL    ECHO            ;Send to terminal
010760  000773                          BR      1$              ;Continue sending
                                ;
                                ;  Finished
                                ;
010762  012602                  9$:     MOV     (SP)+,R2
010764  000207                          RETURN
                                ;
                                ;  Control sequences
                                ;
010766      0000    133     103 MRS100: .BYTE   ESC,'[,'C,0     ;VT100
010771      000
010772      0000    103     000 MRS52:  .BYTE   ESC,'C,0
                                .EVEN
```

```
                                        .SBTTL  CSRLFT -- Generate control sequence to move cursor left
                                ;-----------------------------------------------------------------------
                                ;  CSRLFT is called to generate the terminal control sequence to move
                                ;  the cursor left one column.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
 010776  112700  0000006        CSRLFT: MOVB    #BKSPAC,R0      ;Use backspace character to move to left
 011002  004737  011210'                CALL    ECHO
                                ;
                                ;  Finished
                                ;
 011006  000207                         RETURN
```

```
                                        .SBTTL  CHKDLM -- Check for word delimiters
                                ;-------------------------------------------------------------------
                                ;  Check to see if a character is a word delimiter.
                                ;
                                ;  Inputs:
                                ;   R0 = Character to check.
                                ;
                                ;  Outputs:
                                ;   C-flag set     ==> Character is a delimiter.
                                ;   C-flag cleared ==> Character is not a delimiter.
                                ;
011010  010146                  CHKDLM: MOV     R1,-(SP)
011012  010246                          MOV     R2,-(SP)
                                ;
                                ;  Null is always a delimiter
                                ;
011014  105700                          TSTB    R0              ;Is character null?
011016  001407                          BEQ     2$              ;Br if yes -- this is a delimiter
                                ;
                                ;  Search table of delimiters for the character
                                ;
011020  012701  011052'                 MOV     #WRDDLM,R1      ;Point to table of delimiters
011024  112102                  1$:     MOVB    (R1)+,R2        ;Get next delimiter character
011026  001405                          BEQ     3$              ;Br if hit end of table
011030  120002                          CMPB    R0,R2           ;Is this character the delimiter?
011032  001401                          BEQ     2$              ;Br if yes
011034  000773                          BR      1$              ;Keep checking
                                ;
                                ;  Character is a delimiter
                                ;
011036  000261                  2$:     SEC                     ;Signal that this is a delimiter
011040  000401                          BR      9$
                                ;
                                ;  Character is not a delimiter
                                ;
011042  000241                  3$:     CLC                     ;Signal that this is not a delimiter
                                ;
                                ;  Finished
                                ;
011044  012602                  9$:     MOV     (SP)+,R2
011046  012601                          MOV     (SP)+,R1
011050  000207                          RETURN
                                ;
                                ;  Table of word delimiters
                                ;
011052      040     000Q    054 WRDDLM: .BYTE   ' ,TAB,',,'=,'/,':,'.,'[,'(,'),0 ;Word delimiters (null=end)
    011055      075     057     072
    011060      056     133     050
    011063      051     000
                                        .EVEN
```

```
                                        .SBTTL  CVTLC  -- Convert lower-case characters to upper-case
                                ;-----------------------------------------------------------------------
                                ;  CVTLC is called to convert lower-case characters to upper case
                                ;  if requested by the currently running program.
                                ;
                                ;  Inputs:
                                ;   R0 = Input character.
                                ;   R1 = Job index number.
                                ;
                                ;  Outputs:
                                ;   R0 = Converted character.
                                ;
011066  032761  0000006 0000006 CVTLC:  BIT     #$LC,LSW2(R1)   ;Was "SET TT LC" done?
011074  001404                          BEQ     1$              ;Br if not
011076  032761  0000006 0000006         BIT     #LCBIT,LJSW(R1) ;Does program want lower case characters?
011104  001010                          BNE     2$              ;Br if yes -- lower-case chars ok
                                ;
                                ;  Translate lower-case character to upper-case
                                ;
011106  120027  000141          1$:     CMPB    R0,#141         ;Is this a lower case letter?
011112  103405                          BLO     2$              ;Br if not
011114  120027  000172                  CMPB    R0,#172
011120  101002                          BHI     2$              ;Br if not lower-case letter
011122  042700  000040                  BIC     #40,R0          ;Convert lower-case to upper-case
                                ;
                                ;  Finished
                                ;
011126  000207                  2$:     RETURN
```

```
                                        .SBTTL  ECOCTL -- Echo a control character
                                ;-----------------------------------------------------------------------
                                ;  ECOCTL is called to echo a control character by printing "^x" where
                                ;  "x" is the character.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R5 = Control character.
                                ;
011130                          ECOCTL:
                                ;
                                ;  Echo "^"
                                ;
011130  112700  000136                  MOVB    #'^,R0          ;Get up-arrow
011134  004737  011210'                 CALL    ECHO            ;Echo it
                                ;
                                ;  Echo character
                                ;
011140  110500                          MOVB    R5,R0           ;Get control character
011142  062700  000100                  ADD     #100,R0         ;Convert to upper-case character
011146  004737  011210'                 CALL    ECHO            ;Echo it
                                ;
                                ;  Finished
                                ;
011152  000207                          RETURN

                                        .SBTTL  RNGBEL -- Ring bell to signal error
                                ;-----------------------------------------------------------------------
                                ;  RNGBEL is called to send a bell character to the terminal to signal
                                ;  an error condition.
                                ;
011154  112700  000000G         RNGBEL: MOVB    #BELL,R0        ;Get bell character
011160  004737  011226'                 CALL    ECHO2           ;Send to terminal
                                ;
                                ;  Finished
                                ;
011164  000207                          RETURN

                                        .SBTTL  AKEYON -- Turn on alternate keypad mode
                                ;-----------------------------------------------------------------------
                                ;  AKEYON is called to enable alternate keypad mode.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;
011166                          AKEYON:
                                ;
                                ;  Send control sequence to terminal
                                ;
011166  112700  000000G                 MOVB    #ESC,R0         ;Send Escape as 1st char
011172  004737  011226'                 CALL    ECHO2           ;Send to terminal
011176  112700  000075                  MOVB    #'=,R0          ;Send equal sign as 2nd char
011202  004737  011226'                 CALL    ECHO2
                                ;
                                ;  Finished
                                ;
011206  000207                          RETURN
```

```
                                        .SBTTL  ECHO   -- Send character to the terminal
                                ;-------------------------------------------------------------------------
                                ;  ECHO is called to send a character to the terminal.
                                ;  ECHO2 always echoes the character even if character echoing is turned off.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number.
                                ;   R0 = Character to be sent (preserved).
                                ;
011210                          ECHO:
                                ;
                                ;  See if character echoing is wanted.
                                ;
011210  032761  000000G 000000G         BIT     #$ECHO,LSW2(R1) ;Is character echoing wanted?
011216  001402                          BEQ     9$              ;Br if not
011220  004737  011226'                 CALL    ECHO2           ;Echo the character
011224  000207                  9$:     RETURN
                                ;-----------------------------
                                ;  We want character echoing
                                ;
011226  010046                  ECHO2:  MOV     R0,-(SP)        ;Save character being echoed
                                ;
                                ;  Send character to terminal
                                ;
011230                                  OCALL   QUECHR          ;Send char to terminal
                                ;
                                ;  Finished
                                ;
011236  012600                          MOV     (SP)+,R0        ;Recover character
011240  000207                  9$:     RETURN
```

```
                                        .SBTTL  KEYCK  -- Check to see if key is defined
                                ;-----------------------------------------------------------------
                                ;  KEYCK is the routine called through the use of the KEYCHK macro.
                                ;  It sets up information correctly in registers and calls KEYSRC.
                                ;
                                ;  Form of the call:
                                ;       JSR     R5,KEYCK
                                ;       .WORD   key_code
                                ;
011242                          KEYCK:
                                ;
                                ;  Pick up the key code.
                                ;
011242  012500                          MOV     (R5)+,R0        ;Get key code
                                ;
                                ;  See if gold key has been pressed
                                ;
011244  105737  0000006                 TSTB    SLGOLD          ;Has gold key been pressed?
011250  001403                          BEQ     1$              ;Br if not
011252  052700  0000040                 BIS     #KT$GLD*400,R0  ;Set key type to gold
011256  000402                          BR      2$
011260  052700  0000000         1$:     BIS     #KT$NRM*400,R0  ;Set normal key type
                                ;
                                ;  See if this key is defined by user
                                ;
011264  004737  011272'         2$:     CALL    KEYSRC
                                ;
                                ;  Finished
                                ;
011270  000205                          RTS     R5              ;Return to caller
```

```
                                        .SBTTL  KEYSRC -- See if terminal key is defined by user
                                ;-------------------------------------------------------------------
                                ;  Determine if a specific terminal key is defined by the user.
                                ;  If so, perform the key substitution.
                                ;
                                ;  Inputs:
                                ;   R0 = Key code to be searched for.
                                ;   R1 = Job index number.
                                ;
                                ;  Outputs:
                                ;   C-flag set     ==> Key was defined; processing has been done for it.
                                ;   C-flag cleared ==> Key was not defined.
                                ;
011272                          KEYSRC:
                                ;
                                ;  See if there are any user-defined keys
                                ;
011272  005737  000000G                 TST     KEYPAR          ;Are there any user-defined keys?
011276  001404                          BEQ     5$              ;Br if not
011300  032761  000000G 000000G         BIT     #$SLLET,LSW7(R1);Is substitution wanted?
011306  001002                          BNE     4$              ;Br if yes
011310  000241                  5$:     CLC                     ;Signal failure on return
011312  000207                          RETURN
                                ;
                                ;  There are some user-defined keys
                                ;
011314  010246                  4$:     MOV     R2,-(SP)
011316  010346                          MOV     R3,-(SP)
                                ;
                                ;  Begin loop to search for the key definition
                                ;
011320  013703  000000G                 MOV     VKEYMX,R3       ;Get max # key definitions allowed
011324  012702  000000G                 MOV     #VPAR6,R2       ;Point to virtual address of region base
011330                                  KEYMAP                  ;;;Map to key region
011352  020062  000000G         1$:     CMP     R0,KD$COD(R2)   ;;;Is this entry for the key?
011356  001413                          BEQ     2$              ;;;Br if yes
011360  062702  000000G                 ADD     #KD$$SZ,R2      ;;;Point to next entry
011364  077306                          SOB     R3,1$           ;;;Loop if more to check
011366                                  KEYUMP                  ;Restore par 6 mapping
                                ;
                                ;  Key is not defined
                                ;
011402  000241                  3$:     CLC                     ;Signal failure on return
011404  000415                          BR      9$
                                ;
                                ;  We found the key definition
                                ;
011406  116203  000000G         2$:     MOVB    KD$FLG(R2),R3   ;;;Get control flags for key def
011412                                  KEYUMP                  ;Restore par 6 mapping
011426  105037  000000G                 CLRB    SLGOLD          ;Clear gold key flag
                                ;
                                ;  Call routine which substitutes the key definition string
                                ;
011432  004737  011446'                 CALL    KEYSUB          ;Do the substitution
011436  000261                          SEC                     ;Set flag saying we did a key definition
                                ;
                                ;  Finished
```

```
     58                                         ;
     59 011440  012603                  9$:    MOV     (SP)+,R3
     60 011442  012602                         MOV     (SP)+,R2
     61 011444  000207                         RETURN
```

```
                                        .SBTTL  KEYSUB -- Substitute a defined string for a key
                                ;-----------------------------------------------------------------------
                                ;  A user-defined key has been identified.
                                ;  Insert the defined text into the command buffer.
                                ;
                                ;  Inputs:
                                ;   R1 = Current job index number
                                ;   R2 = Pointer to key definition block
                                ;   R3 = Control flags from definition block
                                ;
011446  010246                  KEYSUB: MOV     R2,-(SP)
011450  010546                          MOV     R5,-(SP)
011452  016146  000000G                MOV     LSW2(R1),-(SP)  ;Save current LSW2 flag settings
                                ;
                                ;  See if we should suppress display of this string
                                ;
011456  132703  000000G                 BITB    #KF$ECO,R3      ;Should we echo the string?
011462  001003                          BNE     1$              ;Br if yes
011464  042761  000000G 000000G        BIC     #$ECHO,LSW2(R1) ;Suppress echo of string
                                ;
                                ;  Move the characters from the string definition to input line
                                ;
011472  062702  000000G         1$:     ADD     #KD$TXT,R2      ;Point to key def string
                                ;
                                ;  Get next character from definition string
                                ;
011476                          2$:     KEYMAP                  ;;;Map to definition buffer
011520  112205                          MOVB    (R2)+,R5        ;;;Get next char from definition string
011522                                  KEYUMP                  ;Restore par 6 mapping
011536  105705                          TSTB    R5              ;Did we reach the end of the string?
011540  001403                          BEQ     5$              ;Br if yes
                                ;
                                ;  Process the character as an ordinary received character
                                ;
011542  004737  000450'                 CALL    ORDCHR          ;Process the character
011546  000753                          BR      2$              ;Go see if there are more characters
                                ;
                                ;  We reached the end of the string.
                                ;  See if we should terminate the input line.
                                ;
011550  132703  000000G         5$:     BITB    #KF$TRM,R3      ;Should we terminate the input line?
011554  001405                          BEQ     9$              ;Br if not
                                ;
                                ;  Terminate the current input line
                                ;
011556  112737  000001  000000G        MOVB    #1,SLCR         ;Say CR-LF received
011564  004737  005622'                 CALL    ICPCR           ;Pretend we received a carriage-return
                                ;
                                ;  Finished
                                ;
011570  012661  000000G         9$:     MOV     (SP)+,LSW2(R1)  ;Restore echo flag
011574  012605                          MOV     (SP)+,R5
011576  012602                          MOV     (SP)+,R2
011600  000207                          RETURN
```

```
                                        .SBTTL  CHK52  -- Determine if terminal is a VT52
                                ;-------------------------------------------------------------------
                                ;  Determine if the terminal is a VT52.
                                ;
                                ;  Inputs:
                                ;   R1 = Job index number
                                ;
                                ;  Outputs:
                                ;   C-flag set if this is a VT52
                                ;   All registers are preserved.
                                ;
011602  010246                  CHK52:  MOV     R2,-(SP)
                                ;
                                ;  See if the terminal type is VT52
                                ;
011604  032761  000000G 000000G         BIT     #VT52,LTRMTP(R1);Is the terminal type VT52?
011612  001010                          BNE     5$              ;Br if yes
                                ;
                                ;  See if we are emulating a VT52
                                ;
011614  116102  000000G                 MOVB    LNPRIM(R1),R2   ;Get primary job index number
011620  032762  000000G 000000G         BIT     #$V52EM,LSW11(R2);Are we emulating a VT52?
011626  001002                          BNE     5$              ;Br if yes
                                ;
                                ;  This is not a VT52
                                ;
011630  000241                          CLC                     ;Signal not VT52
011632  000401                          BR      9$
                                ;
                                ;  This is a VT52
                                ;
011634  000261                  5$:     SEC                     ;Signal that this is a VT52
                                ;
                                ;  Finished
                                ;
011636  012602                  9$:     MOV     (SP)+,R2
011640  000207                          RETURN
        000001                          .END
Errors detected:  0

*** Assembler statistics


Work  file  reads:  0
Work  file writes:  0
Size of work file:  8356 Words  ( 33 Pages)
Size of core pool:  18176 Words  ( 71 Pages)
Operating  system:  RT-11

Elapsed time:  00:01:43.49
,LP:TSSLE=DK:TSSLE/C/N:SYM
```

Cross reference table (CREF V05.05)

$1ESC	1-24	59-34										
$CTRLW	1-24	7-16	7-18	59-30	59-32	59-35						
$DBGMD	1-39	6-38										
$DETCH	1-45	3-17										
$DISCN	1-39	3-19										
$ECHO	1-41	91-14	94-19									
$LC	1-40	89-13										
$NDICP	1-43	71-14										
$NOIN	1-36	71-20										
$NOINT	1-43	2-51	71-26									
$NOLF	1-40	60-51										
$PWKEY	1-23	59-16										
$SCCA	1-22	65-16										
$SLINI	1-40	2-16	4-62	65-41								
$SLKED	1-41	4-13										
$SLLET	1-25	93-20										
$SUCF	1-42	71-21										
$V52EM	1-23	95-22										
$VBELL	1-23	7-24										
$WDISP	1-24	7-25										
$XSTOP	1-35	71-13										
AKEYON	4-15	90-46#										
BELL	1-38	90-32										
BKSPAC	1-35	87-9										
CHK52	20-17	26-22	29-17	32-17	63-15	86-14	95-12#					
CHKABT	1-36	71-22										
CHKDLM	24-29	24-36	24-43	24-58	24-66	34-26	34-30	34-37	61-44	61-62	88-12#	
CHRPOS	16-20	16-34	17-23	17-37	21-50	23-23	24-75	24-107	25-29	25-35	26-43	33-24
	33-38	34-64	34-76	61-89	62-30	62-44	66-17	67-54	68-17	74-37	79-31	80-75
	84-9#											
CKRDTM	6-48	9-13#										
CKUAC	6-43	8-14#										
COLCLC	81-21	83-12#	84-13									
CR	1-37	60-43	60-49	68-25								
CSCHK	5-14	10-14#										
CSEND	12-36	13-66#										
CSFIN	11-42	11-70	12-10#									
CSICHR	1-44	10-25										
CSRLFT	81-70	85-30	87-9#									
CSRRIT	85-23	86-9#										
CSTBL	12-18	13-10#										
CTLRTN	59-41	59-51#										
CTRLQ	1-33											
CURPOS	81-22	84-17	85-12#									
CVTLC	7-33	8-26	89-13#									
DELCHR	1-39	3-37										
DELLFT	33-39	69-28	74-13#									
DOCTRL	6-55	59-10#										
DOSWIT	1-24	7-23										
E1	13-33	37-8#										
E2	13-34	38-8#										
E3	13-35	39-8#										
E4	13-36	40-8#										
E5	13-37	41-8#										
E6	13-38	42-8#										
ECHO	60-50	60-54	81-40	81-46	81-65	86-22	87-10	90-15	90-21	91-10#		

Cross reference table (CREF V05.05)

```
ECHO2    68-26    68-28    90-33    90-51    90-53    91-16    91-22#
ECOCTL   65-29    68-21    90-10#
ENQTL     1-42    73-17
EP52     11-22#   63-17
EPCH1    11-12#   63-14
EPCH2    11-15    11-35    11-51#   63-32    63-45
ESC       1-40    10-18    86-32    86-33    90-50
F10      13-43    47-8#
F11      13-44    48-8#
F12      13-45    49-8#
F13      13-46    50-8#
F14      13-47    51-8#
F15      13-48    52-8#
F16      13-49    53-8#
F17      13-50    54-8#
F18      13-51    55-8#
F19      13-52    56-8#
F20      13-53    57-8#
F6       13-39    43-8#
F7       13-40    44-8#
F8       13-41    45-8#
F9       13-42    46-8#
GTSLCH    1-18     2-12#
ICPBS    49-24    59-59    62-8#
ICPCR    22-16    59-64    60-9#    61-15    94-47
ICPCSI   10-27    63-32#
ICPCTA   59-52    64-5#
ICPCTC   59-54    65-9#
ICPCTR   59-69    59-74    66-5#
ICPCTU   59-72    67-9#
ICPCTZ   59-77    65-15    68-9#
ICPESC   10-20    59-78    63-9#
ICPLF    50-24    59-61    61-9#
ICPNUL   59-51    70-5#
ICPRUB    6-59    69-5#
ICPSS3   10-34    63-45#
INSERT    7-43    21-49    33-21    34-75    61-32    67-21    69-15    79-19    80-12#
INTPRI    1-34    71-36    93-39    93-49    94-29
INWAIT    3-30    71-8#
KBS       6-23    35-5#
KBX       6-33    36-5#
KC$COM    1-30    33-10
KC$DOT    1-30    58-9
KC$DWN    1-32    15-10
KC$E1     1-27    37-12
KC$E2     1-27    38-12
KC$E3     1-27    39-12
KC$E4     1-27    40-12
KC$E5     1-27    41-12
KC$E6     1-27    42-12
KC$ENT    1-30    22-9
KC$F10    1-28    47-12
KC$F11    1-28    48-12
KC$F12    1-28    49-12
KC$F13    1-28    50-12
KC$F14    1-29    51-12
```

Cross reference table (CREF V05.05)

```
KC$F15    1-29      52-12
KC$F16    1-29      53-12
KC$F17    1-29      54-12
KC$F18    1-29      55-12
KC$F19    1-29      56-12
KC$F20    1-29      57-12
KC$F6     1-28      43-12
KC$F7     1-28      44-12
KC$F8     1-28      45-12
KC$F9     1-28      46-12
KC$KP0    1-31      23-11
KC$KP1    1-31      24-9
KC$KP2    1-31      25-9
KC$KP3    1-31      26-12
KC$KP4    1-31      27-9
KC$KP5    1-31      28-9
KC$KP6    1-31      29-12
KC$KP7    1-32      30-12
KC$KP8    1-32      31-12
KC$KP9    1-32      32-12
KC$LFT    1-32      16-12
KC$MIN    1-30      34-12
KC$PF1    1-30      18-12
KC$PF2    1-30      19-12
KC$PF3    1-30      20-12
KC$PF4    1-30      21-14
KC$RIT    1-32      17-12
KC$UP     1-32      14-12
KD$$SZ    1-26      93-37
KD$COD    1-26      93-35
KD$FLG    1-26      93-48
KD$TXT    1-26      94-23
KEY0     13-23      23-5#
KEY1     13-24      24-5#
KEY2     13-25      25-5#
KEY3     13-26      26-8#
KEY4     13-27      27-5#
KEY5     13-28      28-5#
KEY6     13-29      29-8#
KEY7     13-30      30-8#
KEY8     13-31      31-8#
KEY9     13-32      32-8#
KEYCK    14-12      15-10     16-12     17-12     18-12     19-12     20-12     21-14     22-9      23-11     24-9      25-9
         26-12      27-9      28-9      29-12     30-12     31-12     32-12     33-10     34-12     37-12     38-12     39-12
         40-12      41-12     42-12     43-12     44-12     45-12     46-12     47-12     48-12     49-12     50-12     51-12
         52-12      53-12     54-12     55-12     56-12     57-12     58-9      92-10#
KEYCOM   13-20      29-22     33-5#
KEYDOT   13-22      58-5#
KEYMIN   13-21      32-22     34-5#
KEYPAR    1-33       5-26     93-18     93-34     94-27
KEYSRC    5-34      92-26     93-14#
KEYSUB   93-54      94-11#
KF$ECO    1-26      94-17
KF$TRM    1-26      94-41
KPAR6     1-37      93-34     93-34*    93-39*    93-49*    94-27     94-27*    94-29*
KT$GLD    1-27      92-20
```

Cross reference table (CREF V05.05)

```
KT$GLT    1-25      5-31
KT$LET    1-25      5-33
KT$NRM    1-27     92-22
LACTIV    1-37     71-24
LAFSIZ    1-39     73-40*    77-53     80-80     82-31
LCBIT     1-40     89-15
LCOL      1-34      4-48     60-62*    73-35*
LF        1-33     60-44     60-53     68-27
LF$IN     1-34     65-33     73-21
LFWLIM    1-33     73-39*    82-23
LHIPCT    1-34      2-53*    71-28*
LINCNT    1-41      3-24
LINFIN    8-47      9-34     60-58     68-41     73-8#     77-58     80-89
LINPNT    1-38      3-36
LITIME    1-42      2-54*    71-29*
LJSW      1-40     89-15
LNPRIM    1-23     95-21
LNSPAC    1-38      8-19
LOGCHR    1-38     65-36     73-26
LOGCR     1-36     65-37
LOGFLG    1-34     65-33     73-21
LRBFIL    1-34     81-64
LRTCHR    1-41      9-17      9-38*
LSCCA     1-41     65-13
LSPACT    1-35      8-30
LSTATE    1-42     73-14
LSW       1-45      3-17      3-19*     59-34*
LSW10     1-43     71-14*
LSW11     1-23     59-16     95-22
LSW2      1-40     89-13     91-14     94-13     94-19*    94-51*
LSW3      1-36      7-16      7-18*    59-30     59-32*    59-35*    71-20*
LSW5      1-22     65-16
LSW6      1-33      6-38      7-25     60-51     71-13*
LSW7      1-40      2-16      2-51      4-13      4-62*    65-41*    71-26     93-20
LSW9      1-42      7-24*    71-21*
LTRMTP    1-33     95-16
LTTPAR    1-36
LWINDO    1-25     59-18
MAPHLD    1-119#   93-34*    93-39     93-49     94-27*    94-29
MAXLEN   77-32     80-44     82-15#
MAXSEC    1-24      7-21
MRS100   86-13     86-32#
MRS52    86-16     86-33#
NEDCDI    1-44     71-15*
ORDCHR    5-40      6-9#     94-35
OSTRIK    7-39     79-9#
OVRHC     1-39      3-37      7-23      7-27     59-23     65-36     65-37     71-19     73-26     91-26
PAINT    21-42     23-47     24-96     34-60     60-34     61-85     62-29     62-42     65-25     66-12     67-50     74-33
         77-49     79-26     80-70     81-14#
PR7       1-36     71-23     93-34     94-27
PRCHAR    3-43      5-8#
PSW       1-37     71-23*    71-36*    93-34*    93-39*    93-49*    94-27*    94-29*
QHDSPN    1-37     71-31
QUECHR    1-40     91-26
REGCHR    6-61      7-9#     59-53     59-55     59-56     59-57     59-58     59-60     59-62     59-63     59-65     59-66
         59-67     59-68     59-70     59-71     59-73     59-75     59-76     59-79     59-80     59-81     59-82
```

Cross reference table (CREF V05.05)

RNGBEL	11-57	12-41	15-43	19-17	20-22	22-27	24-112	26-48	29-26	30-17	31-17	32-26
	34-81	37-18	38-18	39-18	40-18	41-18	42-18	43-18	44-18	45-18	46-18	47-18
	48-18	49-30	50-30	51-18	52-18	53-18	54-18	55-18	56-18	57-18	58-15	62-24
	77-19	77-35	80-28	80-50	90-32#							
RSTCHR	77-25	77-42	78-12#									
RSTLIN	4-68	14-30	14-46	15-56	36-10	77-8#						
RUBOUT	1-43	6-57										
S$INWT	1-38	71-30										
S$TTFN	1-42	73-14	73-16									
SAVLIN	8-40	60-38	68-32	72-6#	77-57	80-88						
SIGWAT	1-33	71-19										
SLBACK	1-41	4-35*	24-21	24-100	25-20	26-28	27-19*	28-19*				
SLCBUF	1-33	7-41	7-48*	33-20	33-36*	69-14	69-27*	74-26*	79-17			
SLCCOL	1-36	4-51*	81-75*	85-17	85-35*							
SLCR	1-41	4-27*	22-15*	60-14	60-19*	60-25*	61-14	94-46*				
SLCSBF	1-44	12-22	63-19	63-33	63-46							
SLCSBX	1-44	11-55										
SLCSPT	1-44	11-12	11-14*	11-22	11-34*	11-51	11-76*	63-19*	63-35*	63-48*		
SLCSR	1-44	4-31*	5-19	7-49*	11-15*	11-35*	11-58*	12-14*	63-18*	63-32*	63-45*	
SLCX	1-35	4-44*	8-41	9-28	16-27	17-19	17-30	21-19	23-30	24-20	24-81	25-25
	26-27	33-22	33-30	34-17	34-73	60-32	60-39	61-39	62-12	66-10	67-28	68-13
	69-21	74-19	79-14	80-33	80-59	80-82	81-56*	84-21*				
SLCYC1	1-22	14-20	14-25	15-25*	72-53*							
SLCYC2	1-22	14-23	15-26*	72-54*								
SLDBUF	1-39	21-32	21-48	23-37	34-45	34-74	61-31	61-70	67-20	67-35		
SLDOWN	1-23	14-39	14-41*	15-22	15-34	15-36*	72-59*					
SLEBUF	1-38	4-43*	4-44	16-19	16-28	23-22	24-50	24-55	24-63	24-104	25-34	26-40
	35-11	61-41	61-50	61-59	62-20	65-23	66-11	67-29	67-34	67-43	67-49	69-22
	72-11	72-37	73-23	73-31	74-20	77-40	77-48	80-42	80-85	83-16		
SLECOL	1-35	4-50*	73-35	81-57	81-76*							
SLGOLD	1-37	4-23*	5-29	6-13	6-22*	6-32*	7-50*	14-17	14-19*	15-15	15-20*	16-17
	16-21*	17-17	17-24*	18-17	18-19*	18-24*	19-18*	20-23*	21-20	21-51*	22-14*	22-31*
	23-16	23-29*	24-14	24-80*	25-14	25-40*	26-17	26-49*	27-14	28-14	29-27*	30-18*
	31-18*	32-27*	33-15	33-25*	34-18	34-69*	35-10*	37-19*	38-19*	39-19*	40-19*	41-19*
	42-19*	43-19*	44-19*	45-19*	46-19*	47-19*	48-19*	49-18	49-29*	50-18	50-29*	51-19*
	52-19*	53-19*	54-19*	55-19*	56-19*	57-19*	58-16*	60-26	60-31*	61-26	61-33*	62-13
	62-19*	67-15	67-22*	69-9	69-16*	92-18	93-50*					
SLINCH	2-39	3-12#										
SLINIT	2-18	2-33	4-8#									
SLLBUF	1-34	4-57	72-20	72-33	72-43	75-19	75-27	75-35	76-17	76-28	76-39	78-22
SLLEND	1-43	72-18	72-35	72-45	75-17	75-25	75-37	76-19	76-30	76-37	78-20	
SLLPTR	1-43	4-58*	14-22	14-28*	14-35	14-51*	15-21	15-28*	15-33	15-52*	72-58*	
SLMVDN	14-27	15-24	15-37	15-48	76-12#							
SLMVUP	14-42	14-50	75-12#									
SLMXLN	1-38	82-19										
SLOPTR	1-35	2-22	2-47*	4-19*	73-31*							
SLOVER	1-22	4-39*	7-37	64-5	64-7*	64-9*						
SLRPTR	1-22	4-66	4-69*									
SLSBUF	1-41	35-12	36-9									
SLSCOL	1-36	4-49*	83-17									
SLSPTR	1-23	4-55	4-57*	4-58	14-29*	15-26	15-27*	15-41	72-17	72-32	72-39*	72-58
SS3CHR	1-44	10-32										
STOP	1-36	3-20	65-42									
TAB	1-35	81-28	83-27	88-46								
TCDOWN	13-12	13-55	13-59	15-5#								
TCENTR	13-15	22-5#										

Cross reference table (CREF V05.05)

```
TCINVL  22-23#
TCLEFT  13-14     13-57     13-61     16-8#     28-24
TCPF1   13-16     13-62     18-8#
TCPF2   13-17     13-63     19-8#
TCPF3   13-18     13-64     20-8#
TCPF4   13-19     13-65     20-18     21-8#     25-41
TCRITE  13-13     13-56     13-60     17-8#     27-24
TCUP    13-11     13-54     13-58     14-8#
TSSLE    1-6#
VINTIO   1-37      2-53     71-28
VKEYMX   1-26     93-32
VPAR6    1-32     93-33
VQUAN1   1-42      2-54     71-29
VT100    1-42
VT52     1-35     95-16
VVLSCH   1-24     59-28
VVPWCH   1-25     59-14
WINDSP   1-24      7-27
WINPRT   1-25     59-23
WRDDLM  88-22     88-46#
```

Cross reference table (CREF V05.05)

DISABL	1-49#	71-23	93-34	94-27								
ENABL	1-53#	71-36	93-39	93-49	94-29							
ESCSEQ	13-4#	13-11	13-12	13-13	13-14	13-15	13-16	13-17	13-18	13-19	13-20	13-21
	13-22	13-23	13-24	13-25	13-26	13-27	13-28	13-29	13-30	13-31	13-32	13-33
	13-34	13-35	13-36	13-37	13-38	13-39	13-40	13-41	13-42	13-43	13-44	13-45
	13-46	13-47	13-48	13-49	13-50	13-51	13-52	13-53	13-54	13-55	13-56	13-57
	13-58	13-59	13-60	13-61	13-62	13-63	13-64	13-65				
KEYCHK	1-112#	14-12	15-10	16-12	17-12	18-12	19-12	20-12	21-14	22-9	23-11	24-9
	25-9	26-12	27-9	28-9	29-12	30-12	31-12	32-12	33-10	34-12	37-12	38-12
	39-12	40-12	41-12	42-12	43-12	44-12	45-12	46-12	47-12	48-12	49-12	50-12
	51-12	52-12	53-12	54-12	55-12	56-12	57-12	58-9				
KEYMAP	1-97#	93-34	94-27									
KEYUMP	1-103#	93-39	93-49	94-29								
OCALL	1-57#	3-37	7-23	7-27	59-23	65-36	65-37	71-19	73-26	91-26		
TTMAP	1-85#											
TTMAPX	1-78#											
UNMAP	1-90#											
UNMAPX	1-82#											