DECUS
PROGRAM LIBRARY

DECUS NO.	8-573
TITLE	EDITS - A PS/8 EDITOR FOR NON-STORAGE SCOPE DISPLAY
AUTHOR	Ray Smith
COMPANY	M.I.T. Laboratory for Nuclear Science Cambridge, Massachusetts
DATE	August 1, 1972
SOURCE LANGUAGE	PAL-8

EDITS-A PS/8 Editor for A Non-storage Scope Display

AUTHOR: RAY SMITH, ADDRESS: 24-004 M.I.T., Camb., Mass.

COMPANY: LAB FOR NUCLEAR SCIENCE, M.I.T.

DATE: July 6 , 1972

Edits is a modified version of the PS/8 EDITOR (VERSION EDIT.002). EDITS displays on the CRT scope the portion of text surrounding the current line. Although this program was initally designed to use a hardware character generator, small modifications have been made to allow either hardware or software character generators. Software character generation is much slower and therefore creates much more flicker.

EDITS allows the user to choose whether or not to display line numbers. The user also has the choice between truncating or "wrapping-around" long lines.

AC0: 1 - Long lines are "wrapped-around" and continue idented on the line below.

0 - Lines are truncated at the right edge of the scope.

AC1: 1 - Small line numbers are displayed, except for the current line, to the left of the line.

0 - No line numbers are displayed.

I. PROGRAM CHANGES

A few command changes have been made to facilitate the use of the scope display in editing. They are the following:

1. There is no T command
2. There is an nV command which moves the current line to line n.
3. The G command has been changed. The new G command initiates an intra-buffer search identical to that of the J command, except that the buffers passed over are not output.

EXAMPLE: Looking for location 5762 in listing file PROG.LS

```
.R EDITS          / NOTE:  -Means carriage return
                           -Computer print out is underlined
*   DTA2:PROG.LS  / no output file needed
#G
$ 5762            / if 5762 is not unique, may need to repeat G
#                 / if found, scope should show line if not
                  / found after passing through whole program
                  / will type "?".
```

4. When specifying the input/output one can use the = option to choose the character size and/or the number of text lines displayed. The form of this command is = nn...n, where nn... is an octal number of 8 or less digits, which represents a two word (23 bit) number. The high order word is the character size 1-7. The low order word is the number of lines before and after the current line which are displayed. Zeroes or nulls give default values.

   The defaults are:

   Size = 3 - allows approx 50 (decimal) character per line.
   Number of lines +/-=7 - gives 7 (octal) + current line +7 (octal) = 15 (decimal) lines displayed

   EXAMPLES

```
.R EDITS ↵
*OUT←IN↵                    / give default size and lines
*OUT←IN=2↵                  / gives default size, 3, and a
                            / total of 5 lines
*OUT←IN=12↵                 / gives default size,3, and 12 (8)
                            / +1+12(8)=21(10) lines
*OUT←IN=20024↵              / gives 41(10)lines with size 2
                            / - approx 85 character lines)
*OUT←IN=70002↵              / gives 5 lines or ridiculously large
                            / (size 7) characters.
```

   NOTE: Variable character size assumes EPIC, LNS hardware. See Section IV.

II. PROGRAM ASSEMBLY

NOTE: PUT SOURCE TAPE ON DECTAPE AND ASSIGN NAME DSK.

A. To assemble EDITS with supplied software character generation:

```
.R PAL8↵
*EDITS ←DSK: SEDSYM, DEDIT, EDITS, END/L ↵
*DSK: DISCHR (ALTMODE)↵
.SAVE EDITS: 200 = 2001↵
```

B. To assemble EDITS using your own hardware:

1. Create a file like SEDSYM defining IOT's which conform with your hardware. If you don't have variable size leave LAPC, LAPCX,LAPCY as NOP's.

   If you need some software routines, assemble them in field 1 below 16600 and set PIRA= to location lower than your software. If you need no additional space set PIRA=6600.

III. PROGRAM LAYOUT

EDITS loads into 0-7577, PIRA-1000-1600.
EDITS run in 0-7577, 10100-10120, (PIRA-1000) -17577
NOTES: 10000-11777 need not be saved when calling USR.
Lots of room for "once only" code around 2600.

One may be able to get some code in field 0 above 4000.

IV. PROGRAM PHILOSOPHY

EDITS displays text on the CRT while waiting for teletype input or output. (133, OUTL in EDITS.PA). When display bounds are set up, the display processor, (DEDIT,PA), displays plus and minus NLINES from the current line of the text buffer. If the current line is close to a text boundary the display extends, if possible. 2 NLINES + 1 from the boundary. Otherwise the display will cover the whole text buffer. These display boundaries remain constant until one of the following:

1. The current line moves to the first line displayed, last line displayed or out of the displayed text segment.
2. A move command is executed.
3. A read command is executed.

The above three things have in common the possibility of destroying the display's internal text pointers. Essentially, because the text lines are chained, the most important pointer is that to the first line displayed. A check for the above actions is initiated upon every entrance to the display processor, DISPLA.

Because DISPLA uses the EDITS routine UTRA to unpack text characters DISPLA must save and restore UTRA's pointers associated with EDITS. Upon display boundary initializaton a similar saving and restoring process is done with the EDITS routine, FIND which returns the pointer to the first displayed line buffer.

The main text display loop lies in the seven locations between NEXCHR, IOCHK1 and DISPNT-1.

```
NEXCHR
IOCHK1, TSF          / A TSF or KSF is put here to check on the state
                     / of TTY after every character is displayed
  SKP                / TTY is still waiting
  JMP DISEND         / The TELETYPE IO wait is finished and the TTY
                     / is ready to be read.
  CIF CDF 00         / Fetch another character of the text with a
                     / routine in FIELD 0
  JMS PUTRA1         / UTRA routine unpacks and returns next text
                     / character
  JMS CRTPCH         / This acts like a TLS, CLA sequence to display
                     / character and advance scope position. If the
                     / character is a carriage return we jump to
                     / NEXLIN which starts the display of the next line
```

```
JMP NEXCHR        / Onto the next character
```

The scope display control is for the most part concentrated in a set of routines at the end of the DEDIT.PA file under the collective name CRTROT. The motivation, and hopefully the consequence, of this package was to give the process of CRT scope text display the simplicity, if not the persistence, of teletype output. The CRT routines associated with DEDIT.PA do not correspond identically to actions of a teletype. For closer correspondence see CRTPS8.PA, ZIP8.PA by the same author.

All operations of scope displays with text generation have several necessary actions to perform.

These include: gross scope positioning, character size & intensity setting, character generation & positioning, and text formatting-eg. end of line, page special characters. Note that with asynchronous hardware the additional problem of checking for completion is added.

In DEDIT's CRTROT these actions are accomplished by the following routines.

a. Hardware completion checking-CRWAIT returns when the previous character has been completed. Gross scope positioning is usually fast enough not to require checking.

b. Gross scope positioning-XPUT, YPUT, DETFOR; All routines wait till previous character is done.

    XPUT - sets X position
    YPUT - sets Y position
  CRTFOR - sets both X & Y position

c. character size & intensity- CRTSET (uses special hardware)
CRTSET-sets character X & Y scale (3 bits) sets character intensity (4 bits)

(NOTE: with accompanying software character generation only the intensity is set)

d. Character generation and positioning-CRTPCH

CRTPCH - i) checks for and simulates tabs (calculation only, no movement yet)

ii) checks for carriage return and transfers to NEXLIN if found

iii) spaces right one character width (calculation only)

iv) checks on right margin and moves scope as follows;

1. in bounds-wait for completion-move right
2. out of Bounds-

a) ACO SWITCH = 0-truncate line and go to NEXLIN
b) ACO SWITCH = 1-display CR,LF, indent, Move right.

e. Text formatting-CRTPCH, CETLIN
CRTPCH- see d
CRTLIN- simulates carriage return, line feed

NOTE: bottom of scope is checked for after NEXLIN. This check should be possibly be moved to MARCHK

IV. HINTS & KINKS WITH EDITS

Typing while the current line is the last line introduces considerable flicker because the display is reinitialized every key stroke. This often occurs when one is appending text. This trouble can be eliminated by inserting an end of page comment line say "//" and then inserting text before it. One can always delete this later if one chooses.

There is a trade off between the number of lines displayed and the flicker One can change the number of lines displayed and/or the character size when one specifies the input/output files.

The character size determines the maximum number of lines that can be displayed on the CRT SCOPE. With variable character size hardware we are able, with EDITS, to modify the size of the characters along with the number of lines displayed to achieve a great variety in the amount of text displayed. One could integrate a variable size character with software in EDITS if one defined LAPCX and LAPCY to be calls to set the character X scale and Y scale respectively. Our standard Edits character size was chosen as a compromise between the fact that one must view the text for long periods of time and the desire to have as many characters on a line as possible. For certain applications such as scanning listings for errors etc. for short periods, we will put up with smaller characters and increased flicker.

The PS/8 8k Programming System User's Guide (pg 4-9) fails to mention that one of the options open to the user after the search character is typed is to insert text after the search character, before continuing the search. To end this insertion one can type control G twice which causes the remainder of the line to be typed out. Carriage return deletes the remainder of the line to the right.

EXAMPLE:

The line was:
ALPHA BETA DELTA
*S
ALPHA BETA GAMMA DELTA

a. the user typed (space) and the computer typed out ALPHA
b. the user typed control form and the Editor typed out BETA
c. the user typed GAMMA (space) inserting text after ALPHA BETA.
d. the user typed (control G)(control G) and the computer typed the rest of the line -DELTA (CR)

The new line is:
ALPHA BETA GAMMA DELTA

A Short Description of Programs

EDITS.SV - Save file for Edits (software character generator version)

SEDSYM.PA - Assembly redefinitions for EDITS (software character generator version)

DEDIT.PA - Display processor portion of EDITS

EDITS.PA - Editing portion of EDITS - Essentially the same as EDIT.002

DISCHR.PA - Software character generator - 6 x 4 - modified DECUS 8-23

ZIP8.PA - a) PDP10 remote terminal package
b) Runs with DECTAP.PA and TYPEDT.PA
c) Allows the PDP8/I system to act as a remote PDP10 terminal through the Teletype console
d) Allows PS/8 Dectape/PDP10 text transmission in both directions
e) Displays PDP10 output on the CRT scope (if turned on)

DECTAP.PA - Modified DEC TC01 Dectape routines able to run in any field unlike the original

TYPEDT.PA - Text display processor for ZIP8

CRTPS8.PA - PS/8 device handler for a hardware character generator

PIPS.PA - a) Modified version of PIP with /P option for CRT
b) /P=nn is to page output to the CRT screen nn lines at a time because the CRTPS8 handler has no internal paging capability.

CRTODT.PA- a) Displays a movable segment of core - its contents and various other locations refered to by and through these contents.

CRTDIG.PA- Still another CRT character generator formatting routine - used by CRTODT

ODUMP.SB - FORTRAN compatible SABR routine to facilitate the use of the CRTPS8 scope display under FORTRAN's PS/8 I/O
b) Fakes full buffer forcing an immediate call to the handler
c) Closing and opening make for a lot of tape motion and time under Dectape systems.

Displays using only the VC8/I

The following are in SABR and are FORTRAN compatible

DISPS.SB - a) CALL DISPS (IX,IY,IZ,N) where
IX = array of x coordinates (integer)
IY - array of y coordinates
N - number of point (x,y) pairs to be displayed

b) Very fast 1000 points with little or no flicker (P7 phospher)

DISPS2.SB -a) CALL DISPS (IX, IY, IZ, N, IDELX)
b) If IDELX is negative same as DISPS otherwise
IX is the first x position and subsequent points are equally spaced to the right by the amount IDELX
c) For some uses it save space

DISPER.SB- a) CALL DISPS (IX, IY, IBAR, IZ, IDELX, IFBAR, N)
b) If IFBAR is zero same as DISPS2 otherwise
IBAR is an array of vertical error bars centered around the y value of the points.

AXIS.SB- a) CALL AXIS (XO, YO, N) where XO, YO is the origin of a full screen set of x - y axes and N is the dot spacing used in the lines

AXIST.SB- a) CALL AXIST (XO, YO, XTIC, YTIC)"
b) Displays a full screen x - y axes with
XO - x origin
YO - y origin
XTIC - spacing of tick markings on x axis
YTIC - spacing of tick markings on y axis