==================
CEDRIC VERSION 2.1
By M.J.RANDALL
==================

INTRODUCTION

CEDRIC is a screen-oriented text editor specifically designed for software development, rather than word-processing. That is not to say however that CEDRIC cannot be used to prepare text - this manual was written using CEDRIC together with the text processor JUST. CEDRIC runs under the FLEX operating system on a 6809-based micro-computer. The user may verify for himself that the design goals of speed and flexibility have been met.

Speed has been achieved in three ways. The program has been written in assembly language - it occupies only 23 sectors on disc so it loads quickly. Text is kept memory resident and is efficiently stored - for example the editor source program (35 pages of well-documented code, taking 112 sectors on disc) leaves 14400 bytes free in the text when memory to $BFFF is available; the text buffer holds about 165 sectors altogether. Once an editing operation (most require only a single keystroke) is complete internally, the screen "repainting" is aborted if another operation is pending; thus avoiding continual "repainting" delays.

Flexibility is achieved through the comprehensive selection of single-keystroke editing operations and the menu commands; together with a configuration overlay feature that allows the program to be easily configured to the particular terminal.

CEDRIC is a single-mode editor. Printable ASCII characters entered from the keyboard are inserted in the text at the cursor position. Certain control or "meta" characters defined at configuration cause appropriate editing operations to occur. Text can at any time be inserted at the cursor position from selected disc files. Part or all of the text can be saved to disc with a freely selected file name. Part,of the text may be "cut" and is stored in a "paste buffer" from where it can be re-inserted at any place (or even several places). Search target and replacement words can be defined (a "word" being any printable string not including a space).

The text is to be seen displayed as through a window whose width and length is determined on configuration. The width is (at most) the number of characters displayable per line, while the length is (at most) one less than the number of lines displayable on the terminal. The bottom line is reserved for certain status information:- (i) A decimal count of the free space available in the text buffer. (ii) The current search target word. (iii) The current replacement word.(But only if the screen width is greater than 60 characters).

There is no restriction (apart from available memory!) on the length of lines. If a line longer than the width of the screen is within the window, moving the cursor right along the line will eventually cause the display window to move to the right past the begining of the line. Characters will disappear to the left of the window as more appear at the right.

Searches can be performed forwards or backwards in the text. Replacement can take place either word by word or globally.

A very useful decimal counter function is available. The counter can be set to zero, incremented, or inserted in the text, each with a single keystroke.

A "macro" feature is available whereby a sequence of up to 40 keystokes (including most of the editing operations) may be defined as a macro and be recalled at any time by a single keystroke. This allows a complicated editing sequence to be easily repeated. In addition most single-key operators can be repeated automatically a chosen number of times. Two "permanent" macros can be defined at configuration.

RUN-TIME MEMORY MAP

```
            ___________________
MEMEND =>  |                   |
           |                   |
           |                   |
           |                   |
           |                   |
           |                   |
           |                   |
           |        TEXT       |
           |       BUFFER      |
           |                   |
           |                   |
           |                   |
           |                   |
           |                   |
           |                   |
           | ----------------- |
           |       PASTE       |
           |       BUFFER      |
           | ----------------- |
           |                   |
           |                   |
           |      PROGRAM      |
           |                   |
           |                   |
           |                   |
           | ----------------- |
           |   CONFIGURATION   |
           |       TABLES      |
     $100  | ----------------- |
           |        STACK      |
           |___________________|
```

CALLING CEDRIC

The version of the editor on the distribution disc is named "CED.CMD". First read the section on configuration of CEDRIC to check whether this version suits your terminal. If not, edit "CED-CNFG.TXT" according to the instructions within it. After assembling this file append the .BIN file to "CED.CMD" to form a re-configured version. We will assume this is copied to your system disc and named "CED.CMD".

To run CEDRIC type "CED" following the FLEX prompt (+++). The MENU should be displayed:-

```
           SCREEN EDITOR (C) M.J.RANDALL 1985

   T--TOP              I--INPUT            Y--REPLACE WORD
   B--BOTTOM           W--WRITE            C--COPY PART
   P--REPLACEMENT      !--ERASE ALL        X--EXIT TO FLEX
   M--SEARCH TARGET    V--VIEW TEXT        L--LOOK AT TABS
   N--NEW TABS         R--REPEAT           S--START MACRO
   E--END MACRO        U--PAGE UP          D--PAGE DOWN
   tab-TO NEXT WORD
```

An input text file may be specified on the command line if desired, in which case the file is loaded and displayed instead of the menu. The menu can be displayed at any time by typing the <MU> key [ESC] twice. Commands are performed by typing one of the option keys listed. Once you are familiar with the menu.. commands may be performed without displaying it by typing just one <MU> followed by the appropriate option key.

MENU COMMANDS

T--TOP The cursor is moved to the top of the text and the text, if any, will be seen through a window at the top left of the text.

B--BOTTOM The cursor is moved to the end of the bottom line of the text, if any.

I--INPUT The name of a text file will be requested. Type the name of an existing file (the default extension is .TXT and the default drive is the current working drive) followed by a carriage return (CR). The contents of the file will be inserted at the cursor position. Control characters other than CR will be ignored. The FLEX space compression convention for text files is observed. If you wish to abort the command type "#" instead of the filename. This also applies to W--WRITE and C--COPY.

W--WRITE The name of a text file will be requested. The original input file name will be displayed, if you wish use the same name just type "=" follwed by a carriage return. If a file of the same name and extension already exists on that drive it will be renamed with the extension .BAK (an existing .BAK file will in this case be deleted first). The whole text will be written to a new file of the selected name, with FLEX space compression.

C--COPY PART The marked text will be stored as a disc file but retained in the text. The name of the file will be requested as for W--WRITE.

!--ERASE ALL The whole text will be erased, freeing the text buffer. You can now edit another file without re-loading the editor. Any contents of the paste buffer and the macro buffer will be preserved, as will the target and replacement word.

M--SEARCH TARGET You will be asked to define a target word. Spaces and control characters will be ignored but backspace will be honoured as in FLEX. A carriage return terminates the definition. Up to 20 characters are allowed in the target.

P--REPLACEMENT You will be asked to define a replacement word as in M--SEARCH TARGET.

Y--REPLACE WORD Assumes you are at the target word, which is replaced by the replacement word. (See also <SR> and <GR>.)

X--EXIT TO FLEX Returns to FLEX. In case you have forgotten to save any wanted text you will be asked if you are sure. Type "Y" to return to FLEX. All text in memory will be lost.

tab-TO NEXT WORD The cursor goes to the beginning of the next word.

D--PAGE DOWN The display window moves forward one screen.

U--PAGE UP The display window moves back one screen.

S--START MACRO Starts the definition of a "macro". Subsequent key-strokes are remembered in a 40-character MACRO BUFFER until the command <MU>E is encountered. The command itself does not affect the text, but the following keys perform their normal function as they are being entered in the macro buffer. It is as if the editor is "learning" as it goes.

E--END MACRO Ends the macro sequence definition, but does not affect the text.

R--REPEAT COMMAND A (decimal) repeat count will be asked for. Type a number followed by a carriage return. The command to be repeated will be requested. Type the desired operation key. The <MU> key is invalid and will be ignored.

L--LOOK AT TABS The tab positions will be displayed on the status line.

N--NEW TABS You will be prompted to define a new set of tab positions (up to 5).

V--VIEW TEXT Repaints the screen. This is useful to force a repaint without displaying the menu. Because screen updating can be aborted by new keyboard input it is possible for the screen to "lie" if the last operation did not completely repaint it. If you are suspicious <MU>V will force a complete repaint.

Any other character typed will return the text display.

SINGLE-KEY OPERATIONS

As the actual control character corresponding to a given operation can be re-defined by the user by re-configuring the editor to suit his own preferences or to take advantage of a key-pad, function keys etc, we will refer to the operators with the notation <..> and provide the distributed control character thus:- [^L] (for control L) or [1C] (the ascii code in hexadecimal).

(i) CURSOR MOVEMENT OPERATIONS

<CB> [^L] CURSOR BACK The cursor goes back one position. If it is at the beginning of a line it goes to the end of the previous line.

<CF> [^R] CURSOR FORWARD The cursor goes forward one position. If at the end of the line it has the effect of inserting a space - the cursor moves into the white space at the end of a line.

<CU> [^U] CURSOR UP The cursor moves up one line. If at the top line already the screen scrolls down by half a screen if possible so that the cursor line is centred.

<CD> [^D] CURSOR DOWN The cursor moves down one line. If already at the bottom line the screen scrolls up by half a screen if possible so that the cursor line is centred.

<LB> [^B] LINE BACK The cursor goes back to the beginning of the line. If already at the beginning it goes back to the beginning of the previous line.

<LF> [^F] LINE FORWARD The cursor goes forward to the beginning of the next line.

<LE> [^J] LINE END The cursor goes forward to the end of the line. If already at the end it goes to the end of the next line. (^J is usually available as the "linefeed" key.)

(ii) ERASE OPERATIONS

<EC> [^E] ERASE CHARACTER The character under the cursor is erased from the text.

<EW> [^W] ERASE WORD Erases from the cursor to the end of the word.

<EL> [^X] ERASE LINE Erases from the cursor to the end of the line.

<DL> [^H] DELETE LEFT The character to the left of the cursor is deleted from the text.

(iii) SEARCH OPERATIONS

<SF> [^C] SEARCH FORWARD As for search back but searches right (forward) for the target word. If the target is not found the cursor will be at the end of the text.

<SB> [^Z] SEARCH BACK The text will be searched to the left (back) for the next occurrence of the target word, and the window centered there with the cursor on its first character. If the target is not found the cursor will be at the beginning of the text.

<SR> [^N] SEARCH+REPLACE Searches forward (right) for the next occurrence of the target word and replaces it with the replacement word.

<GR> [^A] GLOBAL REPLACE Repeats <SR> until the end of the text is reached.

(iv) COUNTER OPERATIONS

<ZC> [^0] ZERO COUNTER Sets the decimal counter to zero "0". The text is not affected.

<IC> [^V] INCREMENT COUNTER Adds one to the decimal counter. The text is not affected.

<PC> [^Q] PUT COUNTER Inserts the current value of the decimal counter at the cursor position. Leading zeroes are suppressed. When the counter is zero a single "0" is inserted.

(v) MACRO OPERATIONS

<PM> [^K] PERFORM MACRO Causes the defined macro sequence to be performed as though you had entered the key-strokes individually.Certain keys are invalid (for fairly obvious reasons) - <PM> and <MU>. You will be given the opportunity to either abort the rest of the macro or continue, with the illegal operation ignored.

<Ml> [^T] MACRO 1 is called.

<M2> [^G] MACRO 2 is called. These "permanent" macros may be defined by the user on configuration - see the CED.CNFG listing.

(vi) OTHER OPERATIONS

<TB> [^I) TAB Spaces will be inserted to bring the cursor to the next tab position. If past the last tab a single space will be inserted.

<PH> [^C] POINT HERE The present cursor position will be remembered. Subsequent movement of the cursor to the right (forward) will cause the text between the present position and the subsequent position to be "marked" (by displaying it in reverse video if the terminal permits). Subsequent movement of the cursor to the left (back) of the present position will cancel the MARK function. Marking is used to define text to

be copied to disc or cut to the paste buffer.

<CT> [^Y] CUT TEXT The marked text will be cut from the text and saved in the paste buffer. If the marked text is too long for the paste buffer you will be so advised and asked if you want to erase it anyway.

<PT> [^P] PASTE TEXT The contents of the paste buffer will be re-inserted in the text at the present cursor position.

<CC> [^S] CHANGE CASE The case of the cursor character will be changed (lower => upper, upper => lower) and the cursor moved one place to the right (forward).

<MU> [esc] MENU Precedes a menu command. If <MU> is typed again the menu will be displayed. If a legal menu option is typed the appropriate function will be performed, otherwise the text will be re-displayed.

(vii) ORDINARY CHARACTERS

When an ordinary character is typed it will be entered in the text at the cursor position and the cursor and the characters following it on the line will be moved right. To speed text entry the screen is not completely repainted unless the window has been moved; this happens only at the right of very long lines.

CONFIGURATION

The information CEDRIC needs to communicate correctly with the video terminal is contained in configuration tables located from $100 to $1CD. While this data can be altered by using a utility such as FIX, it is better to assemble a small overlay program and append it to the end of the distributed file CED.CMD. This will give the user complete documentation of his reconfigured version.

There follows an assembler listing of the file CED-CNFG.TXT which reproduces the configuration of CED.CMD as distributed. It contains comments that fully explain the purpose of each section of the configuration tables.

The single-key operators are best left as they are until the user is familiar with the operation of CEDRIC. Ascii control characters are used as they are generated by most keyboards. If your terminal has a seperate keypad that generates either control characters or "meta" characters ($81-$FF) you may wish to change the operator assignments to take advantage of it and to achieve a logical layout of the operator keys.

The minimum complement of sendable characters and receivable video control strings needed for CEDRIC are:-

(1) 29 control characters $01..$1F or $81..$FF that may be sent from the terminal. This number may be reduced by using special printable characters at the expense of their not being available for insertion in the text from the keyboard.

(2) A direct-cursor-addressing control string that takes the form:- string, row (or column) byte, column (or row) byte.

(3) A cursor-left string.

(4) A cursor-up string.

(5) A cursor-down string.

(6) An erase-to-end-of-line string.

(7) A TLHC-and-clear-screen string.

STOP PRESS

See READ-ME.TXT for instructions on using the program CONFIGUR to configure CEDRIC easily!

```
                                   OPT     PAG


CEDRIC CONFIGURATION                    9-24-85 TSC ASSEMBLER   PAGE    1
OVERLAY


                     ************************************************
                     * CEDRIC CONFIGURATION OVERLAY
                     * FILE "CED-CNFG.TXT"
                     * THIS EXAMPLE RE-PRODUCES THE CONFIGURATION
                     * AS ORIGINALLY DISTRIBUTED
                     * M.J.RANDALL 1985
                     ************************************************
                     * TO RECONFIGURE CEDRIC RENAME CED.CMD TO
                     * CED.BIN THEN ASSEMBLE CED-CNFG.TXT THEN
                     * APPEND IT TO CED.BIN:-
                     * "+++APPEND CED.BIN,CED-CNFG.BIN,CED.CMD"
                     * TRY "+++1.CED" TO MAKE SURE THAT IT WORKS
                     * IF IT DOES THEN COPY CED.CMD TO YOUR
                     * SYSTEM DISC. IF NOT YOU SHOULD RE-READ
                     * YOUR TERMINAL DOCUMENTATION AND CHECK
                     * YOUR VERSION OF CED-CNFG. IF ALL ELSE FAILS
                     * GET IN TOUCH WITH THE AUTHOR
                     ************************************************


  0100                       ORG    $0100

                     ************************************************
                     *         OPERATOR CHARACTER TABLE
                     ************************************************

                     * ANY 8-BIT CHARACTER AVAILABLE FROM THE
                     * KEYBOARD CAN BE USED - ASCII CONTROL
                     * CHARACTERS ARE AVAILABLE ON MOST TERMINALS.
                     * IF A KEYPAD IS AVAILABLE THAT GENERATES
                     * SINGLE CONTROL OR "META" CHARACTERS, THESE
                     * CAN BE CONFIGURED TO MAKE THE LAYOUT
                     * CONVENIENT OR LOGICAL.
                     * MAKE SURE YOUR FLEX I/O ROUTINES DONIT
                     * STRIP PARITY IF USING "META" CHARACTERS
                     * (THEY HAVE BIT 7 MSB SET).
                     * ANY PRINTABLE CHARACTER COULD BE USED
                     * AS A COMMAND BUT THEN IT WOULD NOT BE
                     * AVAILABLE FOR INSERTION IN THE TEXT FROM
                     * THE KEYBOARD!

  0100 12            CF      FCB    $12       ^R
  0101 0C            CB      FCB    $0C       ^L
  0102 04            CD      FCB    $04       ^D
  0103 15            CU      FCB    $15       ^U
  0104 06            LF      FCB    $06       ^F
  0105 02            LB      FCB    $02       ^B
  0106 05            EC      FCB    $05       ^E
  0107 18            EL      FCB    $18       ^X
  0108 13            CC      FCB    $13       ^S
  0109 1B            MU      FCB    $1B       ESC
  010A 08            DL      FCB    $08       ^H BACKSPACE
  010B 09            TB      FCB    $09       ^I TAB
```

```
  010C 01            GR      FCB    $01       ^A
  010D 17            EW      FCB    $17       ^W
  010E 0E            SR      FCB    $0E       ^N
  010F 10            PT      FCB    $10       ^P
  0110 19            CT      FCB    $19       ^Y
  0111 03            SF      FCB    $03       ^C
  0112 1A            SB      FCB    $1A       ^Z
  0113 1C            PH      FCB    $1C       ^SHIFT-L ON SOME KEYBDS
  0114 0F            ZC      FCB    $0F       ^O
  0115 16            IC      FCB    $16       ^V
  0116 11            PC      FCB    $11       ^Q
  0117 14            M1      FCB    $14       ^T
  0118 0B            PM      FCB    $0B       ^K
  0119 07            M2      FCB    $07       ^G
  011A 00                    FCB    0         SPARE
  011B 0A            LE      FCB    $0A       ^J LINEFEED
  011C 20            SPACE   FCB    $20
  011D 0D            RETURN  FCB    $0D
  011E 00 00                 FCB    $0,$0     SPARE

                     ************************************************
                     *        CONFIGURABLE "CONSTANTS"
                     ************************************************

  0120 0800          PASLEN  FDB    $800      PASTE BUFFER LENGTH
  0122 50            WIDTH   FCB    80        SCREEN WIDTH
  0123 17            LINES   FCB    23        ONE LESS THAN DISPLAYABLE

                     ************************************************
                     * TERMINAL CONTROL SEQUENCES ETC
                     ************************************************
                     * IT IS ASSUMED THAT DIRECT CURSOR ADDRESSING
                     * IS ACHIEVED BY SENDING A SEQUENCE OF BYTES
                     * FOLLOWED BY TWO BYTES THAT SPECIFY THE ROW
                     * AND COLUMN IN ANY ORDER. ROW OR COLUMN MAY
                     * HAVE A CONSTANT OFFSET ADDED TO THE BYTE

  0124 00            XFIRST  FCB    0
                     * THIS BYTE IS NON-ZERO IF THE Y (ROW)
                     * CO-ORDINATE IS SENT FIRST

  0125 00            CUROFS  FCB    0
                     * THIS IS THE OFFSET TO BE ADDED.
                     * CUROFS,CUROFS IS THE T.L.H.C.

  0126 00            LFFLG   FCB    0
                     * THIS BYTE IS ZERO IF THE TERMINAL AUTOMATICALLY
                     * PUTS THE NEXT CHARACTER AFTER THE LAST DISPLAYABLE
                     * ON A LINE AT THE BEGINNING OF THE NEXT LINE.
                     * THAT IS NO CR/LF IS NECESSARY.
```

```
0127 00            RVFLG   FCB    0
                   * THIS BYTE IS ZERO IF THE TERMINAL DISPLAYS AN
                   * INCOMING CHARACTER WITH BIT 7 SET IN REVERSE
```

```
                   * VIDEO, BUT NON-ZERO IF REVERSE VIDEO NEEDS
                   * A CONTROL SEQUENCE

                   * CONTROL SEQUENCES - THESE ARE TERMINATED BY 0
                   * THUS A NULL (0) CANNOT BE INCLUDED IN A SEQUENCE
                   * 8 BYTES MAXIMUM INCLUDING TERMINATOR

0128 0B 00 00 00   CURADD  FCB    $B,0,0,0  CURSOR ADDRESS SEQUENCE
012C 00 00 00 00           FCB    0,0,0,0
0130 0C 00 00 00   CURLFT  FCB    $C,0,0,0  CURSOR LEFT
0134 00 00 00 00           FCB    0,0,0,0
0138 06 00 00 00   ERALIN  FCB    $6,0,0,0  ERASE TO END OF LINE
013C 06 00 00 00           FCB    0,0,0,0
0140 05 00 00 00   HOMERA  FCB    $5,0,0,0  CURSOR TO T.L.H.C AND
0144 00 00 00 00           FCB    0,0,0,0   CLEAR SCREEN
0148 14 00 00 00   HOMCUR  FCB    $14,0,0,0 CURSOR TO T.L.H.C.
014C 00 00 00 00           FCB    0,0,0,0
0150 00 00 00 00   RV-ON   FCB    0,0,0,0   REVERSE VIDEO ON
0154 00 00 00 00           FCB    0,0,0,0
0158 00 00 00 00   RV-OFF  FCB    0,0,0,0   REVERSE VIDEO OFF
015C 00 00 00 00           FCB    0,0,0,0
0160 15 00 00 00   CUR-UP  FCB    $15,0,0,0 CURSOR UP
0164 00 00 00 00           FCB    0,0,0,0
0168 04 00 00 00   CUR-DN  FCB    $4,0,0,0  CURSOR DOWN
016C 00 00 00 00           FCB    0,0,0,0

             0170  LAST    EQU    *         IT MUST BE $0170


                   ************************************************
                   *       ADVANCED CONFIGURATION
                   ************************************************

                   * PRE-CONFIGURED MACROS

0170 00            MACRO1  FCB      0       COUNT OF MACRO BYTES
0171 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0 UP TO 40 CAN BE USED
0175 00 00 00 00
0179 00 00
017B 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0
017F 00 00 00 00
0183 00 00
0185 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0
0189 00 00 00 00
018D 00 00
018F 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0
0193 00 00 00 00
0197 00 00

0199 00            MACRO2  FCB    0         COUNT OF MACRO BYTES
```

```
      019A 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0 UP TO 40 CAN BE USED
      019E 00 00 00 00
      01A2 00 00
```

```
      01A4 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0
      01A8 00 00 00 00
      01AC 00 00
      01AE 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0
      01B2 00 00 00 00
      01B6 00 00
      01B8 00 00 00 00           FCB    0,0,0,0,0,0,0,0,0,0
      01BC 00 00 00 00
      01C0 00 00


                         * DEFAULT TABS

      01C2 0008 000F     TABS    FDB    8,15,24,40,0 5 AVAILABLE TAB POSITIONS
      01C6 0018 0028
      01CA 0000
      01CC 0000                  FDB    0         GUARD WORD

                                 END

0 ERROR(S) DETECTED


+++
```