X16 Edit user manual

January 29, 2022

1 Introduction

X16 Edit is a simple text editor written in 65C02 assembly especially for the Commander X16 retro computer.

The look and feel of the program is inspired by GNU Nano, but there are, naturally, many differences.

One primary design goal is to support editing of large text files. A lot of care has been put into the design of the internal memory model to make this possible.

The Commander X16 was devised by David Murray a.k.a. the 8-Bit Guy. For more information on the platform, go to www.commanderx16.com.

2 Getting help

X16 Edit is controlled by keyboard shortcuts. The most frequently used shortcuts are always displayed at the bottom of the screen.

There is also a built-in help screen, which is displayed when you press Ctrl+G. The help screen lists all keyboard shortcuts with a short description of what they do.

All keyboard shortcuts are also listed in section 3.3 of this manual.

3 Basic usage

3.1 Entering text

X16 Edit is a modeless editor. As soon as it is started, everything you type on the keyboard is inserted into the text buffer.

By default, line breaks are not made automatically. There is no limit to the length of a line, other than the available memory. If the current line does not fit on the screen, it is scrolled horizontally.

There is, however, an optional automatic word wrap mode. Read more about that in section 4.3.

3.2 Moving the cursor

The cursor is primarily moved by the standard arrow keys.

You may move to start of line with the Home key, and to end of line with the End key or Shift+Home.

The PgUp and PgDn keys are supported. Alternatively you may press Ctrl+Y for page up or Ctrl+V for page down.

Finally, it is also possible to move the cursor to a specific line number with the go to line feature (Ctrl+L).

Both the Backspace and the Delete keys have their standard meaning.

3.3 Commands

Commands are selected with keyboard shortcuts. They may be entered in the following three ways:

- Press and hold down the Ctrl key or the left Alt key at the same time as you press a command key. This is the preferred way of selecting commands.
- Alternatively you may press and release the Esc key. The program is now ready to receive a command key, and a message stating this is displayed in the status bar. Press a command key, without holding down Ctrl, or Esc to abort. This option is mostly made as a backup, in case the Ctrl+key sequence does not work.
- Finally, some commands are available via an optional function key.

Table 1: List of keyboard shortcuts

Ctrl or Esc	F-key	Description
G	F1	Display built-in help screen
X	F2	Exit from the program
O	F3	Write text buffer to file
R	F5	Open and read a file into the buffer
N	—	Create new text buffer
J	F4	Justify text buffer
Y	F7	Page up
V	F8	Page down
K	—	Cut current line and save it to clipboard
C	—	Copy current line to clipboard
U	—	Paste (uncut) all content from clipboard
DEL	—	Delets current line, no copy to clipboard
W	F6	Search and find string in buffer (case sensitive)
S	—	Replace string (case sensitive)
L	—	Goto line number
A	—	Toggle auto indent on/off

Z	—	Toggle word wrap on/off
E	—	Change charset
I	—	Invoke DOS command
D	—	Set file storage device number, default is 8
T	—	Cycle through text colors
B	—	Cycle through background colors
M	—	Show memory usage (1 block=251 bytes)
space	—	Insert non-breaking space

The tab stop width is set by first pressing and releasing Esc and then one of the digits 1–9.

The command key bindings are user configurable, see section 6.

3.4 User interface

The user interface is inspired by GNU Nano, and should be mostly self-explanatory. It consists of the following main parts:

- The title bar
- The status bar
- The shortcut list
- The editor area

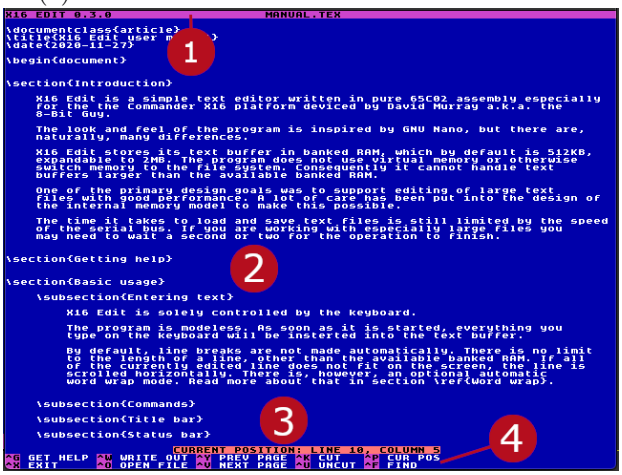
The *title bar* is displayed on the first row of the screen. You find the program name and version to the left. The current file name is shown at the center. If the text buffer has never been saved to file, the string "NEW BUFFER" is shown instead of a file name. At the right edge, the letters "MOD" are shown if the text buffer has been modified since last saved to file.

The *status bar* is the line third from the bottom of the screen. All messages from the program are displayed in the status bar. Press any key to hide a message. If the program needs to prompt you for input, the prompt is also shown in the status bar. Press Enter to confirm input or Esc to abort the operation.

The last two lines at the bottom of the screen contain the *shortcut list*. The most frequently used commands are shown here.

The *editor area* covers all lines between the title bar and the status bar.

Figure 1: Main elements of the user interface, (1) title bar, (2) editor area, (3) status bar, and (4) shortcut list.



3.5 Non-breaking space

By default, the Commander X16 interprets Shift+space as a non-breaking space.

To prevent typing errors, the editor will, however, insert a normal space character even if the Shift key is held down. Non-breaking spaces are rarely used when you edit plain text files. And it is quite easy to type them by mistake, especially if the character immediately before or after the space requires the Shift key.

If you actually want to insert a non-breaking space you may type Ctrl+space.

3.6 Text buffer size

X16 Edit stores its text buffer in banked RAM, which by default is 512 kB, expandable to 2 MB.

The program does not use virtual memory or otherwise switch memory to the file system. Consequently, it cannot handle text buffers larger than the available banked RAM.

You may get the available free memory by pressing Ctrl+M. The available memory is reported as number of blocks free. One block may hold at most 251 characters.

If you have used all available memory, the editor will display a memory full message in the status bar. Further insertion of characters is ignored.

4 Features

4.1 Tab stops

The default tab stop width is four spaces. You may change the width by pressing and releasing the Esc key followed by one of the digits 1–9. The selected digit indicates the tab stop width.

The tab key works by inserting blank spaces until reaching the next tab stop.

4.2 Auto indent

Use the auto indent feature to keep the level of indentation when line breaks are inserted manually or automatically by the word wrap feature.

Auto indent is turned off when the editor starts. To toggle the feature on or off, press Ctrl+A.

4.3 Word wrap and text justification

By default, automatic word wrap is turned off. If you type a line longer than the width of the screen, the line is scrolled horizontally.

Automatic word wrap is toggled on or off with Ctrl+Z. When turned on, you are prompted for the column where to wrap. The feature works in a simplified way. When you reach the right margin, the editor breaks the line after the previous blank space. If there is no blank space on the line, the line break is inserted at the right margin. If you delete characters from a line or if you insert characters at the beginning of a line, the line breaks are not recalculated.

There is, however, a command to justify the whole text buffer (Ctrl+J). The justify command breaks the text buffer into paragraphs, and recalculates the word wrap using the line length set by the the word wrap function (default 80). The word wrap function need not be enabled to run the justify function.

The justify command interprets the start of a new paragraph the same way as GNU Nano.

When auto indent is turned off a new paragraph is considered to begin

- if two or more consequtive line breaks are found, or
- if a line starts with one or more blank spaces.

If auto indent is turned on, a new paragraph is considered to begin

- if two or more consequtive line breaks are found,
- if a line contains only blank space characters, or
- if the level of indent is different from the previous line.

4.4 Cut, copy and paste

X16 Edit supports the traditional cut, copy, and paste features.

The copy (Ctrl+P) and cut (Ctrl+K) commands copy all of the current line to the clipboard. It is not possible to select a part of a line. The lines you copy or cut are placed in the clipboard in the order they were copied or cut.

The clipboard may hold a maximum of 3 kB of data. If you reach that limit, the program will inform you.

Pasting or uncutting (Ctrl+U) will insert all content in the clipboard at the position of the cursor. The clipboard is cleared upon the first cut or copy after the clipboard content was pasted into the document, which is the same behavior as GNU Nano.

4.5 Search and replace

X16 Edit also supports search (Ctrl+W) and replace (Ctrl+S).

Both search and replace are case sensitive. Search starts from the cursor position and is only forward looking.

When searching for a string, the editor moves the cursor to the start of the next occurrence. If the string is not found a message is displayed in the status bar.

When replacing a string, you are given the option to only replace the next occurrence or all subsequent occurrences.

4.6 Supported character sets

X16 Edit supports the three character sets of the Commander X16:

- PETSCII upper case/graphics. This is the default mode of both the Commander X16 and the C64.
- PETSCII upper/lower case. This is the same mode as is available on the C64.
- ISO character set. This mode is new, and there is no corresponding mode supported by Commodore 8 bit computers. Text is encoded according to ISO-8859-15, making it easier to transfer files to and from modern computers.

On startup, X16 Edit detects the current character set. If the detection is successful, it continues using that character set.

If the current character set for some reason cannot be detected, the program defaults to ISO mode.

During the operation of the program, it is possible to change the character set. Press Ctrl+E to cycle through the options.

4.7 Line break encoding

The selected character set mode determines how the editor encodes line breaks when writing the text buffer to file.

In both PETSCII modes, it uses a single CR to indicate line breaks. This is the closest you get to a standard for Commodore 8 bit computers. This is also the setup most likely to work with applications written for Commodore computers.

In ISO mode it uses a single LF to mark line breaks. This is the standard used today by Linux and MacOS. Following this standard makes it easier to transfer text files to or from modern computers.

Internally the editor uses a single LF as line break marker. On reading a file it converts all occurrences of CR to LF. When writing the text buffer back to file, the line breaks are converted to CR if in PETSCII mode.

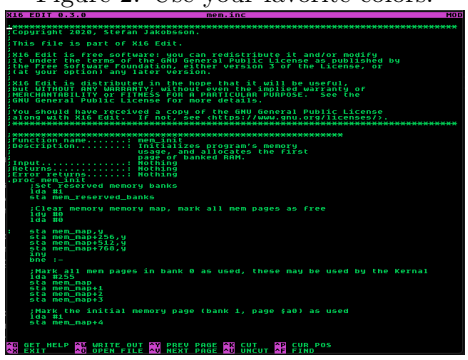
In the event you want to save a PETSCII file with LF line breaks or an ISO file with CR line breaks, all is not lost. Use the preferred character set mode while editing the file. Before saving, switch to ISO mode for LF line breaks or PETSCII mode for CR line breaks.

4.8 Background and text color

Both the background and the text color may be changed while using the editor. The program runs in 16 color text mode, so there are 16 background and 16 text colors to choose from.

Press Ctrl+T to cycle through background color options. Press Ctrl+B to cycle through text color options.

Figure 2: Use your favorite colors!



5 File handling

5.1 Reading and writing files

X16 Edit file handling is centered around functions for reading a file into the text buffer and writing the text buffer to a file.

Press Ctrl+R to read from a file or Ctrl+O to write to a file. The file name may be typed in at the prompt that is displayed. A file name may be just the name of the file or a complete CBM DOS path.

If you are in ISO mode, it's recommended to type file names in upper case. Otherwise file names will not be readable when listed in PETSCII upper case/graphics mode.

5.2 File browser

At the prompt where a file name is typed in you may alternatively press Ctrl+T to show the built-in file browser. The file browser lists the content of the current directory. To select a file, first highlight it with the up or down arrow keys, and then press Enter.

If the selected item is a directory, it will be made the new current directory, and its content will be displayed in the file browser.

The file browser shows at most 50 files or directories on one page. If not all items fit on one page the listing is ended with "— MORE —". In case the items are spread over several pages, you may go to the next page with Ctrl+V and back to a previous page with Ctrl+Y.

If there are no more items to show the listing is ended with "— END —".

5.3 Disk commands

X16 Edit lets you send commands to the disk. To invoke a command, press Ctrl+I, and then enter the command at the prompt.

Any valid command may be sent. You find a list of commands at X16 DOS. Some of the most useful commands are:

- "C:dst=src", copy src file to dst file
- "R:dst=src", rename src file to dst file
- "S:filename", delete filename
- "CD:dirname", change current directory
- "RD:dirname", remove directory

Please be careful though. There is nothing stopping you from deleting files or even formatting the disk.

5.4 Change device number

By default the file handling functions use device #8. The device number may be changed by pressing Ctrl+D.

6 User-configurable key bindings

The shortcut key bindings used in X16 Edit are user-configurable.

On startup, the program reads custom key bindings from the file X16EDITRC in the root folder of the SD card. If that file is not available, the default key bindings are used.

The X16EDITRC file has a very simple format. It is only a stream of bytes, one for every shortcut, with no metadata.

Each key is represented by the value returned by the Kernal function GETIN when pressed without Shift, Ctrl, Alt, or any other modifier key held down.

The extra keys suppurted by the editor are represented by the following values:

- $15 = Delete
- $16 = End
- $17 = PgUp
- $18 = PgDn
- $1a = Insert

The values in X16EDITRC are bound to shortcuts in a fixed order, the same order as the shortcuts appear in section 3.3.

If X16EDITRC holds fewer values than there are shortcuts, the editor will use default bindings for the remaining ones. If there are more values in the file than there are bindings, the excess is ignored.

To bind the first three shortcuts – Show help, Exit, and Write Out – to other keys, the file could for example begin with the following three bytes:

```
$72, $81, $83 ;ASCII values of H, Q, and S
```

To make it a bit easier to setup X16EDITRC, you may use the provided tool for this purpuse (X16EDIT-KEYBINDINGS.PRG).

7 Application Interface

7.1 Program Entry points

X16 Edit has a small Application Interface (API) that consists of the three entry points described in more detail below.

The entry points may be used by other programs to start X16 Edit, and to control how it's configured on startup.

7.2 Default entry point

The default entry point starts the editor with default options and with an empty new text buffer.

The call address is $080D in the RAM version and $C000 in the ROM version of the program. The default entry point is called in the RAM version when started with the RUN command.

Before calling the entry point in the ROM version of the program, you need to set the following parameters.

Table 2: List of parameters

Register	Description
X	First RAM bank used by the program
Y	Last RAM bank used by the program

Those parameters are ignored in the RAM version of the program, and defaults to using all available banked RAM, banks 1–255.

The purpose of setting the first and last RAM banks, is to reserve parts of banked RAM for other purposes.

7.3 Load file entry point

On startup, the load file entry point loads a specified text file from the SD Card.

The call address is $0810 in the RAM version and $C003 in the ROM version of the program. Before calling the entry point, you need to set the following parameters.

Table 3: List of parameters

Register	Address	Description
X		First RAM bank used by the program
Y		Last RAM bank used by the program
r0	$02–03	Pointer to file name
r1L	$04	File name length, or 0=no file

If the specified file does not exist, the editor will display display an error message. If the file name length is 0, the program will not try to load any text file on startup.

The first and last RAM bank settings control what parts of banked RAM is used by the program. This option may be used to reserve parts of banked RAM for other purposes.

7.4 Load file with options entry point

On startup, the load file with options entry point applies the specified editor options, and then loads a text file from the SD Card.

The call address is $0813 in the RAM version and $C006 in the ROM version of the program. Before calling the entry point, you need to set the following parameters.

Table 4: List of parameters

Register	Address	Bits	Description
X			First RAM bank used by the program
Y			Last RAM bank used by the program
r0	$02–03		Pointer to file name
r1L	$04		File name length, or 0=no file
r1H	$05	0	Auto indent on/off
r1H	$05	1	Word wrap on/off
r1H	$05	2–7	Unused
r2L	$06		Tab width (1..9)
r2H	$07		Word wrap position (10..250)
r3L	$08		Current device number (8..30)
r3H	$09	0–3	Text color
r3H	$09	4–7	Background color
r4L	$0A	0–3	Header text color
r4L	$0A	4–7	Header background color
r4H	$0B	0–3	Status bar text color
r4H	$0B	4–7	Status bar background color

Parameters out of range are ignored silently, and the default values are used instead.

Color settings are ignored if both the text and background color is 0 ("black on black").

If the specified file does not exist, the editor will display an error message. If the file name length is 0, the program will not try to load any text file on startup.

The first and last RAM bank settings control what parts of banked RAM is used by the program. This option may be used to reserve parts of banked RAM for other purposes.

7.5 Code samples

Below are some code samples showing how to use the program entry points. The samples are made to work with the RAM version of the program. More information on how to use the entry points in the ROM version is found in the ROM notes document.

First, code that invokes the load file entry point.

```
ldx #1      ;First RAM bank used by the editor
ldy #255    ;Last RAM bank used by the editor
lda #<fname ;Pointer to file name, LSB
sta $02     ;r0L
lda #>fname ;Pointer to file name, MSB
sta $03     ;r0H
lda #4      ;File name length
sta $04     ;r1L
jsr $080A   ;Load file entry point
rts
fname: .byt "test"
```

Secondly, code that invokes the load file with options entry point.

```
ldx #1          ;First RAM bank used by the editor
ldy #255        ;Last RAM bank used by the editor
lda #<fname     ;Pointer to file name, LSB
sta $02         ;r0L
lda #>fname     ;Pointer to file name, MSB
sta $03         ;r0H
lda #4          ;File name length
sta $04         ;r1L
lda #1
sta $05         ;r1H, auto indent on, word wrap off
lda #4
sta $06         ;r2L, tab width
lda #80
sta $07         ;r2H, word wrap position
lda #8
sta $08         ;r3L, device number
lda #1|11<<4    ;Text white, background light green
sta $09         ;r3H, screen color
lda #7|0<<4     ;Text yellow, background black
sta $0A         ;r4L, header color
lda #0          ;Ignore, use default color
sta $0B         ;r4H, status bar color
jsr $0813
rts
fname: .byt "test"
```

8 License

modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or, at your option, any later version. For detailed terms see license file distributed with the program. Also available from www.gnu.org/licenses.