The Last Word

Professional Word Processing for the Atari XL/XE

With dual 40 and 80 column display

Version 3.1

Copyright © 1992-2009
by Jonathan Halliday

Contents

The Last Word 3.1 Reference Manual

The Last Word 3.1 Reference Manual

The Last Word 3.1 Reference Manual

1   INTRODUCTION

1.1   OVERVIEW OF THE LAST WORD

Welcome to THE LAST WORD, the brand new 80 column word processor for Atari XL/XE computers, and one of the most powerful text editors ever written for the 8-bit Atari. The Last Word (LW) combines many of the innovative features found in public domain text editors like TextPro with capabilities of established commercial packages such as AtariWriter Plus and The First XLEnt Word Processor. This means LW offers:

- Full 80 column editing on-screen (switchable to 40 columns at any time)
- Horizontally scrolling editing window of up to 240 columns
- Editing of up 10 files simultaneously on a memory expanded machines
- Sophisticated keyboard macro language
- 80 column print preview
- Versatile cut and paste features
- Search and replace, including reverse search
- High speed editing, even at the top of large files
- Comprehensive on-line help system
- Feature-packed file selector/file management menu with up to 80 filenames in a scrolling window and support for SpartaDOS X directories with time/date stamps
- Support for SpartaDOS X, MyDOS, DOS 2.5 and many DOS 2 derivatives
- User-definable tab ruler
- Customizable, plain-text printer drivers
- Plain-text configuration files
- Completely redefinable keyboard layout
- All international characters visible on the screen
- Comprehensive print formatting commands
- Automatic heading levels
- Indents, hyphenation, and much more...

First written in 1999 and completely revamped ten years later to use an 80 column display, LW is one of the few word processors to be written for the Atari XL/XE in the last twenty years.

The Last Word 3.1 Reference Manual

1.2 ABOUT THE MANUAL

This manual assumes basic familiarity with the Atari screen editor and keyboard. Command keystrokes are enclosed in angle brackets ("<" and ">") which should NOT be typed in. Where two or more keys need to be pressed together, these keys are linked with the plus sign "+".

1.3 STARTING LW

Boot the computer with the LW disk in drive 1 while holding down the <Option> key. (If you don't hold down <Option> and the BASIC "Ready" prompt appears, type DOS and press <Return>). The DOS menu will appear. LW should be started with Binary Load (option L) on the DOS 2.5 menu. Press "L" and type "LW.COM", then press <Return>.

Note: unlike prior versions of the program, LW 3.1 is NOT compatible with DOS XE, nor with any DOS which uses RAM under the Operating System.

When LW loads, it looks on the default drive ("D:") for the following files, and if it finds them, loads them. If a file isn't found, default "built-in" values are used.

LW.SYS   System configuration file: sets up memory usage, keyboard buffer, keyboard redefinition and path for help and system files.

LW.CFG   Configuration file: contains editor settings, preferences, and default drive settings.

LW.FNT   Standard graphics 0 font which will be used in the editor and throughout the program.

LW.F80   Special 80 column font for 80 column editing mode.

LW.PDR   Plain-text printer driver file, which can be edited in LW.

LW.MAC   Macro file, containing automated, user-written command sequences. If a macro is defined for the "@" key, it will be run immediately. See section 6.

LW.EXT   Machine code extensions (expanded memory machines only). Contains extra program functionality such as character maps, calculators, etc.

1.3.1 LOADING LW FROM SPARTADOS X

Under SpartaDOS X, LW is launched by typing

X LW

Note: LW 3.1 is NOT compatible with versions of SpartaDOS prior to SpartaDOS X. Also, SpartaDOS X MUST be configured to use BANKED memory in order to

work with LW" ˙ H \ ] g ˙ a Y U b g ˙ mc i ˙ WU b Ð h LW for a machine with U 8 C G ˙ L ˙ less than 128K.

With SpartaDOS X, you can specify a file to edit on the command line after the program name, such as:

X LW LETTER.TXT

LW will attempt to load LETTER.TXT automatically. If the file isn't found, LW will present you with an empty file bearing the name that you typed on the command line.

1.4 BASIC OPERATION

You can begin using LW without reading this manual. If you get stuck, press the <HELP> key, then a number 1 to 9 or 0. If you don't read the manual, however, you'll be missing out on a huge amount of invaluable information.

1.4.1 THE EDIT SCREEN

To begin using LW, load the program as described above and take a moment to familiarize yourself with the editing screen. The first thing you'll notice is that there are 80 columns of text on the screen.
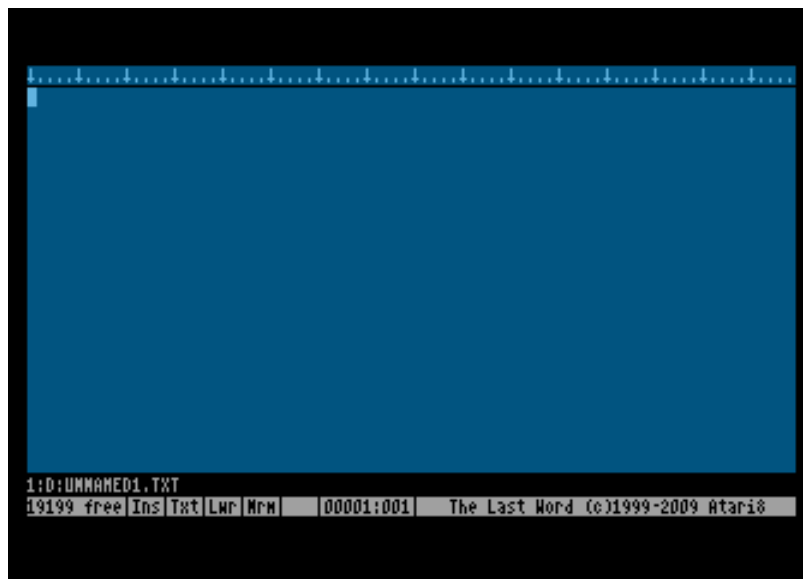
LW defaults to an 80 column display, but you can get the 40 column display back at any time (and make the program default to 40 columns). To switch to a 40 column display, press:

<SHIFT+CTRL+W>

At the lower left of the screen, you'll see:

80 Columns [Y/N]?

Press the "N" key, and for the moment, just press <Return> when you see the "Width" prompt. A 40 column display will then appear:

The Last Word 3.1 Reference Manual
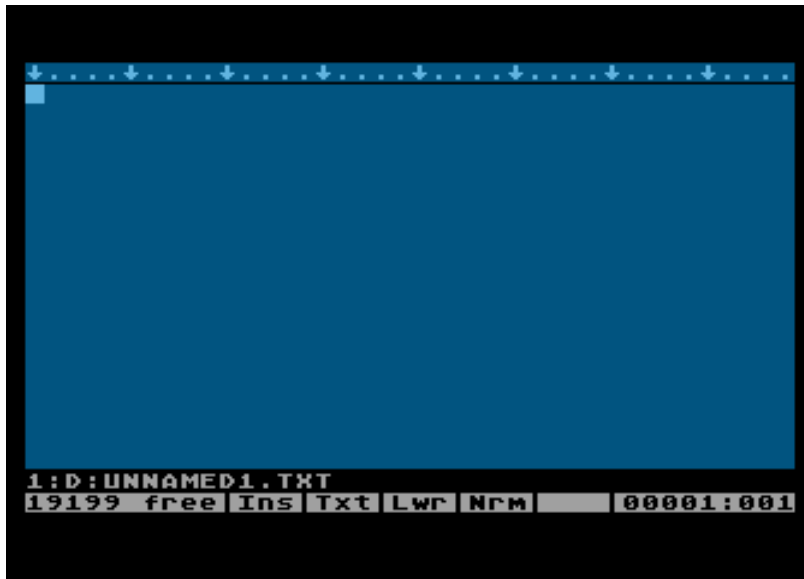
You can get the 80 column display back by performing the same procedure in reverse: type <SHIFT+CTRL+W>, type "Y" for 80 columns, then press <Return> at the "Width" prompt.

In this manual, many illustrations depict LW's 40 column display. This is merely for reasons of clarity, and operation of the program is essentially identical in 40 and 80 column modes.

On both 40 and 80 column screens, you'll see a tab ruler line along the top (which scrolls horizontally if you define a screen wider than the limits of the display) below that a 20 line editing window, and, at the foot of the screen, two lines for status information. The flashing cursor indicates the current typing position.

Until you press a key, the first line of status information will be the title and version # of the program; thereafter it will default to the name of the file currently in memory. Until you load a file or give it a name, it will be called UNNAMED.TXT.

Entering text in LW is easy: just type as you normally would, pressing <RETURN> only at the end of a paragraph and letting the program wrap words at the ends of lines. Cursor keys, <DELETE/BK SP>, and <INSERT> keys behave exactly as you would expect.

In 80 column mode, you'll notice the editor works more slowly than in 40 column mode. However, LW still keeps up with your typing and doesn't "drop" keystrokes. This is because it has its own type-ahead keyboard buffer. Note that if you're using SpartaDOS X and the DOS keyboard buffer (KEY.COM) is turned on when you run LW, the SpartaDOS keyboard buffer will supersede LW's built in buffer. To force use of the LW keyboard buffer (which doesn't buffer auto-repeated keys, preventing the cursor from "running away"), simply ensure the SDX keyboard buffer is turned off ("KEY OFF") before running LW.

When you're ready to save your text, you can follow one of two procedures, outlined below.

1.4.2 TEXT BANKS

If you have a machine with at least 128K of RAM running SpartaDOS X, DOS 2.5 or MyDOS, LW will try to use the extra banks of memory for additional text buffers for the editing of up to ten files at once. You can switch between these banks with:

 <SHIFT+CTRL+n>

where "n" is one of the number keys, with "0" denoting the tenth bank. The program is smart enough not to overwrite any RAMdisks which are installed, so even if you have extra memory, LW may still present you with only one text banks if there are RAMdisks active.

You can see how many text banks are active by pressing:

<SHIFT+CTRL+?>

Will display:

n banks (*n*), using *[low/banked], n* reserved.

This command reports how many text buffers are available, how many free banks in total are detected on the computer, whether the program has its internal buffers in main ("Using Low") or extended ("Using banked") memory, and how many (if any) of the selected memory banks are reserved for use by extensions.

See "Setting up Multiple Text Buffers" in Chapter 8 for more information.

1.4.3 SAVING AND LOADING TEXT

To save the text in memory to disk for the first time, press <CTRL+S> Save text. A prompt will appear with a default filename. Either press <RETURN> to accept this name, or type a new one: the old one will disappear automatically. After you press <RETURN>, your text will be written to disk. If an error occurs, you'll be informed. To abort the save operation, just press <Esc>.

If you type no extender, LW will append one of your choosing before opening the file. The default extender and that defined in the supplied configuration file "LW.CFG" is ".TXT". You can change this, however, or disable it altogether by using the configuration editor.

The first time you save a file, the name you give it becomes the default for subsequent save operations. Once the file has been saved once, pressing <CTRL+S> subsequently will "silently" save the file to disk with the original name it was saved under. To save the file with a different name or to a different folder on disk, use

<SHIFT+CTRL+S> Save As

 which always brings up the **Save As>** prompt.

 When using "Save As", if LW finds a file on disk of the same name, you will be warned. The program will ask:

[Filename] **exists: Overwrite [Y/N]?**

If you type "Y", the existing file will be overwritten. Typing "N" will return you to the editor without doing anything.

To load previously written text, press:

<CTRL+L> Load text

A prompt will appear with a default drive specifier. Type the name of the file you wish to load (you needn't type a drive specifier), and press <Return>.

The Last Word 3.1 Reference Manual

1.4.4 THE FILE SELECTOR

What if you can't remember which files are on the disk? LW has a file selector which is accessible from any filename input dialogue. Just press the <Tab> key when the program is waiting for a filename, and the file selector will appear, listing all the files in the current folder on disk.
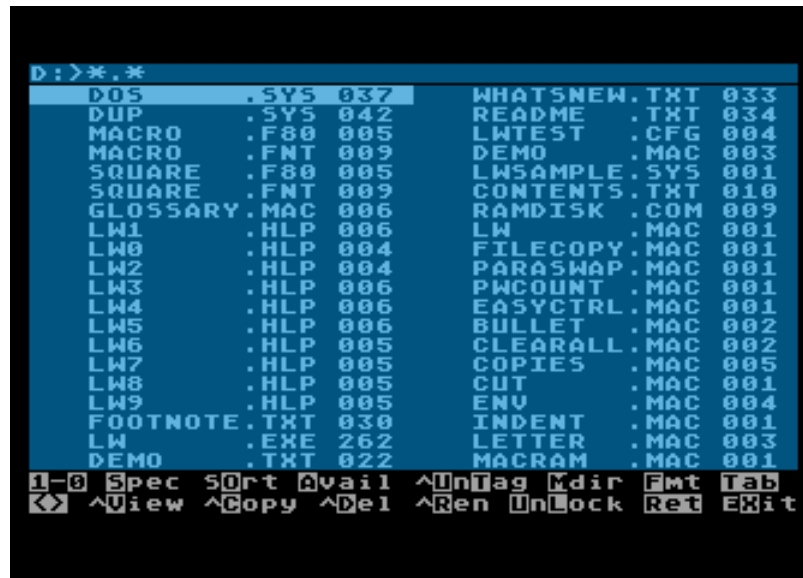
Just highlight the file you want and press <Return> to load it. See "THE DISK MENU" in Chapter 4 for more information about the file selector.

1.4.5 BASIC CONFIGURATION

You can configure LW to suit yourself. Pressing:

<SHIFT+CTRL+Q> Save configuration

will allow you to save the configuration to disk. The file will automatically be given the default extender CFG, and you should call the file LW.CFG if you want your new preferences to be available the next time you boot the program.

As described above, LW normally wraps words to the next line if they don't fit as you type. You can turn this feature off with:

<CTRL+W> Word wrap toggle

By default in LW, <RETURN> characters appear as curved arrows. You can make them invisible with:

<SHIFT+CTRL+CLR> Toggle visible returns

These are just a few of the settings saved in the configuration. For further information on configuring LW, see section 7.

1.4.6 LEAVING THE PROGRAM

To finish using LW and go to DOS, type:

<CTRL+X>   Exit to DOS

and respond "Y" to the prompt. If any text in memory hasn't yet been saved, you'll be offered the chance to save before the program shuts down.

2   EDITOR COMMANDS

LW's commands are all accessed by key combinations. Once you become familiar with LW's keystrokes, a huge number of commands becomes instantly available.

2.1   CURSOR MOVEMENT

The following commands allow rapid cursor movement around the text:

<CTRL+LEFT ARROW>	Cursor left
<CTRL+RIGHT ARROW>	Cursor right
<CTRL+UP ARROW>	Cursor up
<CTRL+DOWN ARROW>	Cursor down
<Tab>	Next tab stop (if in Over-Type Mode)
<CTRL+A>	Start of line
<CTRL+Z>	End of line
<SHIFT+LEFT ARROW>	Previous start of word
<SHIFT+RIGHT ARROW>	Start of next word
<SHIFT+UP ARROW>	Previous start of paragraph
<SHIFT+DOWN ARROW>	Start of next paragraph
<SHIFT+CTRL+UP ARROW>	Screen up
<SHIFT+CTRL+DOWN>	Screen down
<CTRL+H or START>	Top of screen, then top of file
<CTRL+E>	End of file

2.2   TEXT ENTRY MODES

These commands affect various setting in the editor:

<SHIFT+CTRL+INS>   Toggle Insert and Over-type modes. In Insert mode, text after the cursor is pushed along as you type, and closes up when you press <DELETE>. In Over-type mode, new text overwrites existing text. Notice that the operation of the <TAB> key differs depending on which mode the editor is in: in Over-type mode, the <TAB> key simply skips to the next tab stop, whereas in Insert mode, <TAB> inserts spaces up to the next tab stop.

<CAPS>   Toggle upper/lowercase.

<CTRL+CAPS>   Forced control key mode toggle. Allows entry of control codes without pressing <CTRL+ESCAPE> or <SHIFT+ESC> first. The current case is saved when you save the configuration (see later), and becomes the default next time you load the program.

<SHIFT+CAPS>   Uppercase lock.

<INVERSE>   Toggle inverse video on and off.

<CTRL+ESCAPE>   Allow subsequent control key to be entered as normal text (same as pressing <ESCAPE> in the normal Atari screen editor. Also de-selects a marked block of text.

<SHIFT+ESCAPE>   Alternative to <CTRL+ESCAPE>.

<CTRL+W>   Turn word-wrap on and off. Saved in config file.

<SHIFT+CTRL+W>   Set screen editing width. Using this command, you can select the display mode (40 or 80 characters), and type the number of characters per line you want - anything from 5 to 240. If the line length becomes longer than the physical width of the screen, the screen will become a horizontal as well as a vertical window onto your text. Setting the editor line length to the same length as printed lines means you can set tables out almost exactly as they will print. You can skip setting the number of columns by just pressing <Return>, and the value will be set to match the physical width of the screen. The screen mode is saved in the config file.

2.3 INSERTING AND DELETING TEXT

The following commands allow simple insertion and deletion of text:

<DELETE>   Delete character to left of cursor

<CTRL+INSERT>   Insert a space at the cursor

<TAB>   Insert spaces to next tab stop (if in Insert Mode)

<CTRL+DELETE>   Delete character to right of cursor

<SHIFT+DELETE>   Displays the prompt:

        Delete   ord, ine, entence, aragraph?

        Respond by pressing the highlighted letter, or <ESCAPE> to cancel. Pressing <RETURN> defaults to DELETE LINE. Deleted text will fill up the paste buffer from the beginning. Paste the text back into the main buffer with <CTRL+P> or <SHIFT+INSERT>.

<SHIFT+INSERT>   Insert previously deleted text

<CTRL+P>   Paste, or insert previously deleted text (same as above)

<CTRL+CLEAR>   Erase all text

<SHIFT+CLEAR>   Erase all text (same as above)

2.4 MOVING AND COPYING WITH TEXT BLOCKS

The following commands allow blocks of text to be marked, then moved, copied or deleted:

<CTRL+M>   Mark or highlight block. Before a block can be copied, moved or deleted, it must be marked. Use this command to define the starting point of your block. Subsequently, as you move the cursor, the text between the marked beginning and the cursor position will be inverted. You can also mark the end of a block, then cursor back to the beginning. Several other block commands only work once a block has been defined in this way. To un-mark a highlighted block of text, press <CTRL+ESCAPE>.

<CTRL+C>   Cut block. Use this command once a block has been marked as outlined above. The marked text will be copied from the main buffer to the paste buffer, providing the block is not too large. Note that any text already in the paste buffer will be overwritten. The text will then be erased from the main buffer, and block mode is cancelled. You can paste text back with the Paste command.

<CTRL+O>   Copy block. This copies text to the paste buffer exactly like the Cut option, but the text also remains in the main buffer, still highlighted.

 <DELETE>   Delete block. This deletes a marked block without copying it to the paste buffer. Because text deleted this way is irretrievable, you are first asked for confirmation. Note that the block to be deleted may be of any length, regardless of paste buffer size.

<SHIFT+CTRL+I> Write block to a file. Supply a filename at the prompt and the block - which may be of any length - will be written to disk. The file will have the extension "BLK" unless you supply a different one. This option, along with the merge command, allows for the transfer of large blocks of text between different files.

<CTRL+I>   Insert, or merge, file. Allows a file to be inserted into the middle of the text in memory. The filename you type will have the usual text file extender appended to it unless you supply another. If the file you attempt to insert exceeds in size the available space, the text will remain unchanged.

<CTRL+N>   Displays size of file, cursor position, and the number of words and bytes in the file (or the block if any text is marked)

<CTRL+Y>   Convert block to lowercase

<SHIFT+CTRL+Y> Convert block to uppercase

<CTRL+[>   Un-invert text in block

<CTRL+]>   Invert text in block

2.5 FINDING AND REPLACING TEXT

LW has extensive search features which work both forwards and backwards through the text. Searches can be either case sensitive or insensitive. Search and replace operations can be performed either individually or on the whole file, with or without confirmation.

<SHIFT+CTRL+F> Define find string. This option allows you to type in the text you wish to search for (up to 30 characters).

<CTRL+F> Find string. This will move the cursor to the next occurrence of the previously defined string.

<CTRL+U> Upwards find string. Searches backwards for the previously defined string.

<CTRL+R> Replace string. Once a string has been "found" with <CTRL+F> or <CTRL+U>, this command will change it to the "replace" string.

<SHIFT+CTRL+R> Define replace string

<CTRL+G> Global search and replace. Allows you to type a search string and a replace string, then attempts to replace each occurrence of the search string with the replace string. Unless the command is run from a macro, the first time the string is found, a menu will appear, asking if you wish to

Change, All, To End, Skip?

Press the highlighted letter of the option you want, or <ESCAPE> to cancel. "Change" replaces the string and moves to the next occurrence. "All" will change all occurrences of the string throughout the entire document, looping around to the start of the document when it reaches the end. "To End" does a global replace, but without looping around to the top of the document. "Skip" simply ignores the text and moves on to the next occurrence of the string.

LW always returns you to the original point in the document after a global replace operation. When a global replace is in progress, the display is not updated to show each replacement. However, you can cancel the operation at any time with the Break key.

Whether or not search/replace operations are case sensitive is one option set with the <SHIFT+CTRL+U> set options command. If "Match Case" is false ("N"), LW will not differentiate between upper and lower case when searching. You can also deselect the use of wildcards using this command, allowing for literal searching for the inverse "?" character.

2.5.1 SEARCHING WITH WILDCARDS

In find strings, the inverse question mark (▮) will match any character, just as in DOS filenames:

Find>TH▮S▮

will match both "THESE" and "THOSE". Wildcards in replace strings leave the relevant characters in the text unchanged, so:

Find>(▮)

Change to>(▮.)

will place a dot after any single, unknown parenthesized character.

Search strings may be surrounded by spaces to ensure that only whole words are matched. In the case of words followed by punctuation symbols, a macro to perform multiple search/replaces through the text could be written. See macros (section 6).

Note that in order to search/replace the inverse question mark literally, you must turn wildcard searching off with the "Set Options" command (see above).

3   ADDITIONAL EDITOR FEATURES

LW includes many features to aid in the editing of text, such as place markers, pagination guides, and tabulation. The range of facilities available makes LW one of the most complete word processors for the Atari.

3.1   COUNTING WORDS

LW's fast word count will instantly tell you how many words are in the current document.

<CTRL+N>   Will display the position of the cursor in the file and the number of bytes in the file (in the form n of n). It also displays the number of words in the file and a byte count (if text is marked, it will display the number of words and bytes in the marked text).

Unlike many other word processors, LW's word count only counts actual text and not embedded printer commands. Anything typed in reverse video is ignored by the word count. Unfortunately this does mean that header/footer definitions and filename arguments are still counted, since these are typed in normal video, so you will need to allow for this when counting words.

3.2   EDITED TEXT INDICATOR

If text in any LW memory bank has been changed without being saved, an asterisk will appear to the left of the filename on the message line. This is to remind you to save any vulnerable work. The reminder will vanish once your text has been saved.

3.3   TABULATION

LW's tab ruler can be set up with your own tab stops, which can then be saved with the configuration file. These are the commands for editing the tab ruler:

<SHIFT+TAB>   Set tab at cursor position.

<CTRL+TAB>   Clear tab at cursor position.

<SHIFT+CTRL+TAB>Reset default tab stops.

<SHIFT+CTRL+E> Erase ALL tab stops.

3.3.1   TABULATION MODES

In insert mode, the <TAB> key will insert as many spaces as necessary to get to the next tab stop. In over-type mode, <TAB> will just skip over existing text and on to the next tab stop.

3.4 BOOKMARKS

LW has a system of invisible markers which make navigating your text simplicity itself. If you're working on a section of text which you want to leave but will need to return to later, mark it with a place marker.

<CTRL+B>   Set bookmark at cursor position. Asks for which bookmark (1-4) to set.

<SHIFT+CTRL+G> Go to bookmark. Asks for number of the bookmark to find. Providing the marker has been set, and doesn't reside in text which has been deleted, the cursor will jump to the position of the relevant marker.

Bookmarks are saved with the file if you're working in document mode.

3.5 TEXT AND DOCUMENT MODES

LW can save files is two different formats: Text files (.TXT) and Document files (.DOC). While TXT files are plain text files, DOC files contain the tab line and any place markers which have been set. You won't see the header information in LW because it is always hidden, and LW can sense on loading whether a file is a DOC or TXT file regardless of its file extension. If you want to use LW to edit source code files for compilers, etc, you should always save files as plain text (TXT) files.

When you save a file, LW will include the DOC header information only if the program is in "Doc" mode. You can tell by looking at the status bar which mode the program is in, and you can toggle between Doc and Text mode with the *Editor Options* command (see below).

3.6 USER OPTIONS

Several options and toggles are accessed using:

<SHIFT+CTRL+U> User Options

This command presents a list of options which are either switched on or off. The current state of the item appears to the right of the prompt.

Option	Meaning	Default
Match case [N] (Y/N)?	Differentiate between upper and lowercase characters when searching	Off
Warnings [Y] (Y/N)?	Display warning before abandoning file edits or overwriting an existing file	On
Wildcards [Y] (Y/N)?	Use "?" as a wildcard character in search and replace operations	On
False spaces [N] (Y/N)?	Display false spaces in the editor	On
Doc mode [N] (Y/N)?	Operate in document mode	Off

 To leave an option as it is and step on to the next, press <Return>. Press <Y> to switch the option ON, <N> to switch it off, and <Esc> to return to the editor at any point.

3.7 EDITING MULTIPLE FILES

On expanded memory machines, LW allows you to edit several files at once. Setting up LW for your memory configuration is explained later in Configuring LW (section 8). Using DOS 2.5, MyDOS or SpartaDOS X without a custom LW.SYS file which sets up expanded memory, LW will use any RAM banks not in use by RAMdisks. You can control which and how many banks LW uses by creating a suitable LW.SYS configuration file. This step is essential when using a DOS not directly supported by LW: with an unsupported DOS, LW will work out how much memory is attached to the machine, but won't use any banks unless told to do so. This way, you can set up a RAMdisk to use certain banks, and tell LW to use the rest. The supplied SYS files incorporate various sample memory set-ups. To use them, rename the config file you want to use to LW.SYS and reload LW from DOS. If you use one of the supported DOSes, however, LW does all the work for you.

You can access extended text banks with:

<SHIFT+CTRL+n> Select memory bank

where <n> is a number from 1 to 9, or 0, which denotes 10. Note that banks beyond 5 can only be accessed when LW is configured for machines expanded to 192K and beyond (see section 7: Configuring LW). Bank 1 (main memory) is ALWAYS the main bank, so you can see that a maximum of 9 banks of expanded RAM can be made

available. Each bank has the same 16K capacity and its own set of place markers and its own filename. You can cut and paste between banks with ease, and by keeping all the files of a large document in separate banks and by using the include bank print commands from the main file, you can keep track of pagination as if you were editing a single, contiguous file.

If your Atari XL/XE has no extended RAM (or you chose not to use extended memory), you'll only have one 16K text bank and the paste, macro, and disk directory buffers will all be very small (only about 1K each).

3.8 HANDLING LARGE FILES

Although the largest text buffer LW can provide is around 19K when using a machine with extended memory (as well additional banks which are fixed at 16K in size), it's still possible to handle much larger files by splitting them across banks. Text banks can therefore hold separate files, different segments of the same large file, or a mixture thereof. Even if you're using a machine with no extended memory and only one 16K bank, it's still possible to edit larger files.

When you load a file into a text bank, the message "Linked Load" will display if the file didn't fit completely into memory. The buffer will contain as much of the file as would fit, together with 255 bytes of free space for editing (Note: because of the way LW works, any file longer than the total buffer size minus 255 bytes will be classed as a "Linked Load", even if the file would otherwise have fit into the buffer).

To protect against accidental obliteration of the original file on disk when only the first segment has been loaded, linked segments will not "Auto" save with <CTRL+S>. Instead, <CTRL+S> ALWAYS brings up the "Save As" prompt, as does <SHIFT+CTRL+S>. Thus the user will always be warned before overwriting an existing file. Beyond that, it's up to the user to ensure that segments are saved in the correct order.

So – having loaded the first part of a segmented file, we can proceed in one of two ways:

1. Edit the first segment, save it under a new name, then repeat the process until all segments of the original file have been loaded, edited, and appended to a new file.

2. Load all segments of the original file into separate banks, edit them simultaneously, then save the segments in order, either to a new file or overwriting the original.

In either case, the procedure for LOADING the second and subsequent segments of a linked file is to follow the filename with the "/C" switch (without quotes). For example:

Load>REPORT.DOC/C

This simply loads the next segment of the file, providing (obviously) that the filename given is always that of the original file. The name of the "Last File Loaded" can be obtained on the input line with <CTRL+L>, which is useful shortcut when loading successive segments of the same file. The final segment of a file, when loaded, will not

display the "Linked Load" message and thereafter the "/C" switch will simply cause an end of file error.

The "/A" switch should be appended to the filename when saving all but the first segment of a linked file. For example:

Save As>D:REPORT.DOC/A

This will cause the contents of the buffer to be appended to the file on disk (the "/A" switch also works with file copying on the disk menu). As a time saver, the "/A" switch will automatically be appended to filenames when saving all but the first segment of a linked file. And to assist further with the saving sequence of segmented files, the filename on the message line is followed by the number of the loaded segment. For example:

2:D:THESIS.TXT[2]

This denotes bank 2, containing the second segment of THESIS.TXT. In this case, even if the contents of buffer 2 are saved under a new name (without the "/A") switch, the [2] suffix will remain until another file is loaded into that bank or the text buffer is cleared.

Although the linked file feature of LW offers a partial solution to the problem of editing large files, the best way to organize your files is to keep them small enough to fit comfortably into the text buffers. With the include file feature of the print formatter, it's easy and good practice to keep files below 16K and link them together when printing. The main reason for including segmented file support in LW was to enable the conversion of large, unmanageable files into smaller ones. There's a macro called SPLIT.MAC on the distribution disk which will perform precisely this task.

4   DISK OPERATIONS

LW allows full manipulation of files and directories, and has support for many different DOS packages. The mini DOS menu allows viewing, loading, deleting, renaming and copying of files at the touch of a key. The menu displays a paged window, showing up to 80 filenames at a time. Files can be previewed on screen just as they appear in the editor without being loaded into memory.

4.1   DISK OPERATIONS FROM THE EDITOR

In addition to the <CTRL+L>oad, <CTRL+S>ave and <SHIFT+CTRL+S>ave As... commands, the following file handling features are available from the editor:

<CTRL+I>   Insert, or merge, file. Allows a file to be inserted into the middle of the text in memory. The filename you type will have the usual text file extender appended to it unless you supply another. If the file you attempt to insert exceeds in size the available space, the text will remain unchanged.

<SHIFT+CTRL+I>   Write block to a file. Supply a filename at the prompt and the block - which may be of any length - will be written to disk. The file will have the extension "BLK" unless you supply a different one. This option, along with the merge command, allows for the transfer of large blocks of text between different files.

<CTRL+J>   View file. From the editor, this allows you to enter a filename and view the file in a scrolling window on the screen, complete with word-wrap. Pause the listing with <CTRL+1> or by holding down one of the three console keys. Viewing can be aborted at any time with the <Break> key. If you include the "/P" switch after the filename, the text will be displayed in paged rather than scrolling format.

4.2   THE DISK MENU

The functions of the disk menu are accessed by pressing the highlighted keys on the menu at the foot of the screen. The highlight bar is moved with the cursor keys, pressed either with or without <CTRL>. When you reach the limits of the screen in any direction, the display will "page" to the next screen of files.

<CTRL+D>   Call up the disk menu from the editor screen. The program will read in the current directory and display up to 80 filenames (if in 80 column mode) on the screen.

<SHIFT+CTRL+H> As above, but allows the user to specify the directory file mask before calling the disk menu.

The following options are available on the menu, selected by pressing the highlighted letter:

Spec   Set the directory search mask. Use this to narrow or expand the criteria for the directory search.

If the disk menu was called with <CTRL+D> from the editor, <Return> will load the file under the cursor. If the disk menu was called with <Tab> while entering a filename on the input line, <Return> will return the highlighted filename for use in the current load or save operation (if you were currently saving or outputting a file, you'll be returned to the original input line with the selected filename replacing the originally displayed filename).

View   View the file under the selector bar. Same as view from the editor without the "/P" switch.

Note: You can also obtain a paged listing (same as from the editor using the "/P" switch) by pressing <CTRL+V>.

^Del   Delete the file under the selector bar. If the deletion is successful, the filename is removed from the list. Press <CTRL+D> to delete all tagged files.

^Ren   Asks for a new name and renames the highlighted file. The entry in the list is changed to the new name. Wildcards are supported. Type <CTRL+R> to rename the tagged selection of files..

^Copy   Asks for the name of the new file into which you want to copy the contents of the highlighted file. You can type a new drive number, add a subdirectory path if your DOS supports them, and include wildcards. If you want to make a copy of the file under the same name but on a different drive, type the drive identifier, then "*.*". Files of any length may be copied, even those which won't fit into the LW editor. NOTE: The copy operation utilizes the unused part of the current text bank as a buffer. The more unused memory

there is, the faster the copy operation will be, so you will want to be in the bank with plenty of unused memory before you copy anything. A completely full bank actually has 1 spare byte, so copy will still work with it, albeit agonizingly slowly!

Press <CTRL+C> to copy tagged files. Wildcards are fully supported. For example, you could tag all files on the disk and then press <CTRL+C> to copy them to "D2:*.BAK". All files on the destination drive will have the .BAK extender.

Mdir	Create a new directory in the current directory, providing DOS used supports subdirectories.
UnLock	Lock or unlock the highlighted file.
^UnTag	Tag/Untag the highlighted file with <T>. <CTRL+T> will tag all files, while <CTRL+U> will untag all files.
Format	Format the disk. You'll be asked for confirmation first.
EXit	Leave the program and go to DOS.
1-0	Catalogue drive. 1-9 denote the corresponding drive number, and 0 denotes an unnumbered drive ("D:"). This is important if you want to open MyDOS subdirectories.
SOrt	This option will present a menu asking whether to sort the directory by name, extender, date, size, or none. Any other key will leave the setting, which is saved in the configuration, unaltered. "None" will turn off the sorting function.
	Toggle short/long directories (SpartaDOS X only). Pressing <Tab> will switch between DOS 2.5 directory listings and the full SpartaDOS listings.
Avail	Under DOS 2.5, displays the number of free sectors remaining on the disk. With SpartaDOS X, this option displays the number of free bytes on the disk.
Esc	Exit the directory menu.

4.2.1 ADDITIONAL COMMANDS

There are several additional commands not listed on the disk menu.

<CTRL+H>	Cursor home. Moves cursor to the first filename in the directory.
<CTRL+E>	Moves the cursor to the last filename in the directory.
<SHIFT+CTRL+UP ARROW>	Moves the cursor up by a screenful of filenames
<SHIFT+CTRL+ DOWN>	Moves the cursor down by a screenful of filenames

4.2.2 SUBDIRECTORY FEATURES

The following options only work with DOSes which support subdirectories.

Return or **>**	Catalogue the highlighted directory. Only used with Subdirectory oriented DOSes.
<	Go back up one level towards the root directory. DOS-specific, as above.
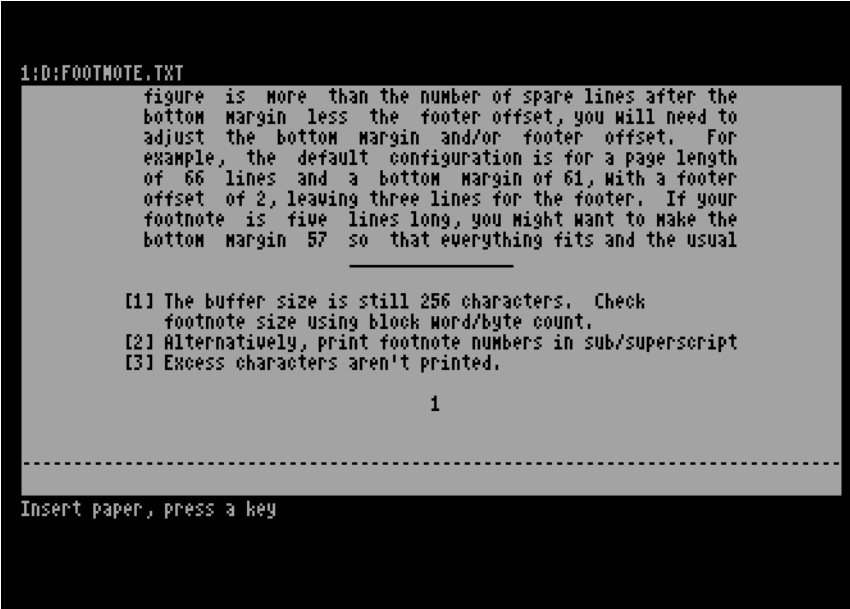
5   PRINTING WITH LW

LW's print processor is one of the most comprehensive of any Atari word processor. Useful features abound to make life easier when printing complex documents.

5.1   PREVIEWING TEXT

<CTRL+V>   Preview printed pages. Text is sent to a 20 line, 80 column window almost exactly as it will appear when printed. Page breaks appear as rows of dashes, and page wait is active if selected. Pause the output with <CTRL+1> or by holding down <SELECT> or <OPTION>. Pressing <START> at any time will skip to the next page, and <BREAK> will return you to the editor.

5.2   KEEPING TRACK OF PAGINATION

<CTRL+?>   "Where's the cursor on the printed page?" This is an innovation also seen on TextPro, and simply prints the page and line number of the text under the cursor.

5.3   EDITOR PRINT COMMANDS

<SHIFT+CTRL+P> Print text. This brings up a menu of 3 choices. Pressing "P" will send text straight to the printer. Choosing "S" previews text in exactly the same way as <CTRL+V> from the editor. "D" allows you to type a filename and send formatted output to disk or another device (default filename extender is "PRN"). Note that the preview screen always becomes active when printing documents. Output can be abandoned by pressing <BREAK>.

All the above commands will read any included files (see later) unless you comment out include statements. This means that you can always know exactly where you are on the printed page, even in documents made up of many different files.

5.4 EMBEDDED COMMANDS

LW has a wealth of print formatting commands which will allow you to tailor your printed output exactly to your needs. Formatting commands follow these simple rules:

- Formatting commands consist of 1 or 2 letter alphanumeric symbols, entered in **reverse video**, often followed by numeric or textual arguments.
- Formatting commands may be in either upper or lowercase.
- **Numeric arguments of formatting commands are entered in reverse video.**
- String arguments (footer lines, header lines and filenames) are entered in normal video.
- Stage 1 formatting commands, either singly or grouped together, must be the first things on a line. They may be optionally terminated with a <Return>. Note: if a string of Stage 1 formatting commands end in <Return>, a blank line will NOT be output in the printed document.
- Formatting commands must not contain extraneous spaces.

Here are some examples of print formatting commands:

`l20`<Return>

Sets the left margin to 20.

`l20r60`hello<Return>

Sets the left margin to 20, the right to 60, then prints "hello." 20 spaces from the left of the page.

`fc`Page `#`<Return>

Defines a running footer which prints the current page number.

5.4.1 STAGE 1 COMMANDS

The following commands, entered as inverse characters in upper or lower case, affect the size and layout of the page. Generally, they should be the first things on a line. Where numeric arguments are required (n), these are entered, also in reverse video, directly after the command. Several commands may be placed together consecutively on a line. Commands may be followed by a <RETURN> (which will NOT print).

`a`<n>   First page to print. `a2` will start output at page 2. Default is 1.

`b`<n>   Set bottom margin, default 61. This is measured in lines from the top of the page, and is the last line on which body text will print. With a page length of 66, a bottom margin of 61 will print 5 blank lines at the foot of each page. Ensure you leave enough lines to

print your footer (if any), which may be up to 3 lines long. If the footer doesn't fit, it won't print.

f<[n]text>   Define running footer to be printed at the bottom of each page. <n> is an OPTIONAL offset, in lines, from the bottom page margin, and should be typed in inverse video immediately before the text of the footer. For example:

f3Footer<Return>

Will print the running footer "Footer" 3 lines below the last line of the printed page. Actual footer text should be in normal video, except where Stage 2 formatting commands appear (Stage 1 commands cannot appear in headers or footers), and must end with a <Return>. Use the **#** symbol to print the page number. A footer or header can consist of up to 8 lines, each terminated by a <Return>. These lines must each be preceded by the   symbol and must be defined on consecutive lines. If the footer is redefined elsewhere in the text, the lines already defined are discarded. To get rid of a footer, just include <Return> in your text.

g<n>   Get text bank. Should be on a line on its own, followed by <Return>. The contents of the text bank will be read and printed in place of the command.

g<fspec>   Get file from disk. This command should be on a line on its own, terminated by a <Return>. The contents of the file will be read and printed in place of the command. This works very quickly, even when reading a file from disk, because a double-buffering system is used to eliminate slow single-byte read commands.

Any formatting commands in the included files will be carried out. The advantage that this method has over the link commands of many word processors is that the same file will be in the edit buffer after printing. You can have a main file with include statements and using the "where's the cursor?" command and print preview, always see the correct pagination. Note that due to memory constraints on buffering this command is NOT nestable, i.e. an included file may NOT in turn include another, although an included BANK may include a FILE.

h<n/text>   This defines a running header to be printed at the top of each page, and works identically to the ooter command. The optional number specifies the offset from the top of the page, and the default value is 2.

jl   Justify text left. All text following this command will be aligned with the left margin.

jr   Justify text right. All text following this command will be aligned with the right margin.

jc   Justify centre. All following text will be centred on the page.

jf   Justify fully. All following text will be aligned flush with both the left and right margins.

l<n>   Set the left margin. The default is 10.

m<n>   Margin outdent by <n> chars, as in this line. This outdents the next line of text. Subsequent lines revert to the normal margin. The line is properly lengthened to fill the extra space. This paragraph uses a paragraph indent and a margin outdent on the first line, creating a hanging indent. NOTE: To aid in alignment, the outdented part of the line will be unaffected by full justification. Also, an outdented line cannot be centred or flushed right.

n<n>   New page. The optional argument will make the command begin a new page only if fewer than <n> lines remain on the current page.

p<n>   Page length. This is the overall length of the page, including the top and bottom margins. Default is 66.

r<n>   Set Right margin. This is the rightmost column in which text will print. Default is 70.

s<n>   Print style. <n> is 0-9. This sends one of 10 non-printing control sequences to the printer. These sequences are set up in the printer driver editor and can each consist of any codes you like, up to 7 bytes each. Handy for selecting fonts or print styles not supported by print style directives (see later).

t<n>   Set top margin, default 5. This sets the number of blank lines which will print at the top of each page. Leave enough lines for your running header, if you've set one up.

v<fspec>   Verbose include file. This sends the named file to the printer regardless of its contents. The file could be a printable bit-image, enabling you to include graphics in your document (this won't show on the preview screen, however). If you include a graphic, ensure you adjust the page length and bottom margin accordingly.

w<n>   Turn page wait on **1**, or off **0**. The default may be either, depending on the configuration. Used for single sheet printing, it will pause and wait for a keystroke at the end of each page. This also works during print preview. Press escape at the prompt to abandon the print/preview operation. Note that the key press is NEVER taken from a macro. This is so that page prompts won't steal subsequent macro keystrokes and knock a macro out of step when printing is finished.

y<n>   Line spacing, default of **1** means no blank lines between each line of text. **2** will print in double-spacing.

z<n>   Last page to print. Stops printing at page <n>.

><n>   Left paragraph indent. All following text up to the next <Return> will be indented by <n> spaces from the left margin.

<<n>   Right paragraph indent. All following text up the next <Return> will be indented by <n> spaces from the right margin.

[<n>   Left header/footer margin, default 10. This works like the command, but sets the margin for the headers and footers, which don't obey the normal left margin. The reason for this is in case the left and right margins are altered within the text. If these alterations crossed a page boundary, headers and footers which shared those margins might not be properly aligned.

]<n>   Right header/footer margin, default 70. As above, but for the right margin.

?<n>   Set starting page number, default is **1**. To begin numbering a document with a page number of 3, set <n> to **3**.

@<n>   Page select. <n> is the number of pages to skip during printing, and defaults to 0. Use this command with a parameter of **1** to print only the odd pages in a document, **2** to print every third page, etc. To print the even pages, set page select to **1** and use **a2** to start printing at page 2 Useful for creating multi-pass double-column documents or pages where the headers and footers are offset for binding purposes. You can print the odd pages with blocked right footers, then set up blocked left footers and print the even numbered pages.

!<n>   Set heading level. <n> can range from **1**-**9**. This prints an automatic section heading in place of the command. You can follow the command with a space and a line of text for a title.

Say you structured your text as follows (with your body text between these headings):

!1 TRANSPORT
!2 BUSES
!2 TRAINS
!1 AMENITIES
!2 LIBRARIES
!2 LEISURE
!3 SWIMMING
!3 OTHER SPORTS

The printout will be:

1 TRANSPORT
1.1 BUSES
1.2 TRAINS
2 AMENITIES
2.1 LIBRARIES
2.2 LEISURE
2.2.1 SWIMMING
2.2.2 OTHER SPORTS

The print formatter will work out the section headings when you print the text, so you don't need to renumber the headings whenever you reorganize the document.

> Reset heading levels. This character simply resets all heading levels to their initial values of 1. Allows you to use more than one sequence of headings in a document.

5.4.2 STAGE 2 COMMANDS

The following commands can appear anywhere on a line, even in headers and footers, and affect individual lines of text or characters. Some take parameters, but most don't. A handy way to enter these commands which saves pressing the inverse key two times is to enter them in conjunction with <Select>.

#
Print page number. Embed in header and footer lines to print the current page number.

c
Centre line. Following text on the line is centred. This command can be used to centre header/footer text or any individual lines. The centred line should end in a <Return>. This command need not be first character on the line - you can have text blocked left, centred and edged right all on the same line. NOTE: This command is NOT that same as centre justify, which works on ALL following text. If you centre or edge right individual lines in paragraphs justified by the Stage 1 justify command, justification will suppressed on that line.

d
Toggle double strike on or off. Block any text you want printed in boldface in <d> characters, i.e. **d**this is bold**d**. This feature is set up by the printer driver editor. Your printer may not support boldface, however.

e
Edge right. Forces subsequent text on the line up against the right margin. See Centre Line.

i
Toggle italics on or off.

u
Toggle underline on or off.

o<n>
Output ASCII char. This outputs the ASCII code <n>. The character is NOT counted as a printable character, so it won't affect the formatting or word-wrap. Handy for sending any control codes to the printer which aren't covered by the printer driver.

x<n>
Send printable code. Works like **o**utput ASCII, but the character is counted as printed matter by the formatter and appears on the preview screen as a question mark. Handy for printing any international characters not supported by the printer driver.

-
Soft hyphen (dash). Insert in the middle of especially long words. When these words won't fit onto a line during printing, the word will be broken where the soft hyphen is embedded, and a hyphen

printed at the end of the line. If the word fits onto the line, no hyphen is printed.

█   Hard hyphen (underscore). Normal hyphens between words allow the line to be split at that point. Use a hard hyphen instead to prevent this happening.

▌   Hard space (can also be an inverse space). Use hard spaces between words to force them to always be printed on the same line. A quick way to enter a hard space is with <SHIFT+CTRL+SPACE>.

❨   Ignore to closing brace ❩. Everything up to the next inverse closing brace is ignored by the print processor.

❩   Comment line: everything until the next <Return> is ignored by the print processor.

⬆   Toggle superscript on or off

⬇   Toggle subscript on or off

⬅   Add-in #3

➡   Add-in #4

*   Add-in #5

%   Add-in #6

5.4.3 CREATING HANGING INDENTS

It's easy to create hanging indents using LW's paragraph indent and margin release commands. Say you wanted to indent the next paragraph by 15 columns, but have the first line flush with the original left margin. Just include the line:

>15m15<Return>

5.5 OTHER PRINT FEATURES

Extraneous spaces following the end of a line not terminated by a <Return> are suppressed at the beginning of the next line. This means sentences with two or more spaces following the full stop will not leave extra spaces at the start of the next line, should the line break occur directly after the full stop.

Missing arguments and illegal commands will produce error messages and halt printing.

5.5.1 INTERNATIONAL CHARACTERS

LW supports the Atari international character set with printed output which directly matches the preview display. Characters with ASCII codes from 0-26, and codes 96 and 123, can be re-defined so they send the actual codes to the printer which correspond to the foreign characters in the Atari international character set. You can set up any characters you like, but unless they correspond to the standard international set, they won't be represented correctly on the preview screen. This feature is set up with the printer driver editor (see PRINTER DRIVERS).

5.6 CONFIGURING THE PRINT FORMATTER

The print formatter defaults for the following margins can be set in the CFG configuration file:

LEFT/RIGHT MARGIN
LEFT/RIGHT HEADER/FOOTER MARGIN
TOP/BOTTOM MARGINS
HEADER/FOOTER OFFSETS

See Section 8, Configuring LW, for more information.

6   CONFIGURING LW FOR YOUR PRINTER

You can customize LW's print styling commands to suit any kind of printer. Toggles can be set up for italics, bold, underlining, superscript and subscript, and up to 10 further styling commands can be defined for any purpose you can think of.

6.1   PRINTER DRIVERS

LW uses printer driver files (with the extension "PDR") to configure itself for various printers. At run-time, LW will attempt to load LW.PDR, so you can have the settings in this file available every time you run the program. If LW.PDR can't be found, LW uses its own default printer driver, which supports no special formatting and will send documents to the printer completely "clean". You can load printer drivers at any time during an editing session with:

<SHIFT+CTRL+D> Load printer driver. Just type a filename as usual - ".PDR" will be appended if you supply no extender.

Printer driver files translate the styling commands for italics, underline, boldface, etc., as well as international characters, into codes specific to your printer.

6.2   CREATING A PRINTER DRIVER

Although previous versions of LW used the LWPD.COM printer driver editor, that program is no longer necessary when editing LW 3.1 printer drivers, which are now stored as plain text files. Several printer drivers are supplied with the program, but it's easy to write your own if you know the escape codes for your printer.

Printer driver commands must be placed one per line in the printer driver file and terminated with <Return>. The following statements are available:

Statement	Arguments	Comments
UNDERLINE ON¦OFF	n,n,n...	Underline on and off
ITALIC ON¦OFF	n,n,n...	Italics on and off
BOLD ON¦OFF	n,n,n...	Bold on and off
SUPERSCRIPT ON¦OFF	n,n,n...	Superscript on and off
SUBSCRIPT ON¦OFF	n,n,n...	Subscript on and off
INTERNATIONAL ON¦OFF	n,n,n...	International character set on and off
CRLF	n,n,n...	Carriage return/linefeed codes
INIT	n,n,n...	Printer initialisation codes
CODE	char,n	Assign international character code
STYLE	<1 - 10>,n,n,n...	Assign styles 1 - 10

Let's step through creating a printer driver for an EPSON compatible STAR LC-10 printer (not that you need to, since an EPSON driver is supplied, but it will do as an example).

6.2.1 PRINT TOGGLES

Start LW with an empty text file. In your printer manual, find the codes for "ITALICS ON". For the EPSON compatible example, the sequence is 27, 52. In the editor, type:

ITALICS ON 27,52 <Return>

Italic off would be 27,53, so on the next line needs to be:

ITALICS OFF 27,53 <Return>

All 6 pairs of ON/OFF toggles (underline, italic, bold, superscript, subscript and international font) work the same way. In a document, the first in a file will turn italics on, the second off, the third on again, and so on (see initialization string later). The only exception are the "International ON¦OFF" code sequences. These are automatically sent before and after any international characters in the document. This means you can select any font built-into the printer, without affecting the rest of your text when printing non-international characters.

6.2.2 CONTROL STRINGS

The CRLF statement allows you to set up a code string for the carriage return/linefeed sequence you want to send to the printer at the end of every line. For an EPSON compatible printer, you would type:

CRLF 13,10

This will sent an ASCII carriage return and linefeed sequence to the printer at the end of each line. The ICD Printer Connection sends the line feeds for you, however, so most of the time you can just have:

CRLF 155

This is Atari's normal End of Line character and is the default in LW, so unless you want the CRLF sequence to be anything other than 155, you needn't include a CRLF instruction in your printer driver.

The INIT code is just sent to the printer at the top of every document you print. You might want to send a printer reset code every time you print. To do so, type:

INIT

Follow this with a space and a comma-separated list of the escape codes for "Reset Printer" in your printer manual. If you don't want an initialization string sent to the printer, simply remove the "INIT" line from the printer driver file.

6.2.3 INTERNATIONAL CHARACTERS

LW includes 29 special "International" characters. These are the characters <CTRL+A> to <CTRL+Z>, <CTRL + comma>, <CTRL + full stop> and <CTRL+ semi colon>. When using a font which redefines these characters and accented or "International" characters, it's important to be able to coax the same accented

character out of the printer. However, EPSON international codes and ATASCII codes are rarely the same, so using the CODE statement, you can assign an ASCII code to any of the 29 international character codes.

CODE 1,129

Will sent ASCII code 129 to the printer every time <CTRL+A> is encountered in the document. You need to set up the codes for all 29 characters in the same way.

The STAR LC-10 manual has various character sets, selectable from software or DIP switches. We're interested in IBM character set #2, since it contains most of the Atari international character set in the codes 128-255. Normally these print as italicised versions of normal characters, so you will want to select the IBM set #2 with the DIP switches. There is a main bank of 12 DIP switches on the LC-10. To get the characters we want out of the printer, set switch 1-6 (Printer Mode) to ON (Standard), and 1-7 to OFF (Graphics). The other switches can be set according to your preferences. There is one character still missing from the printer's character set ("u" with an acute accent) which has to be coaxed out by software. The country-specific character set we need for the "u" acute can't be selected with the DIP switches, so the printer driver needs to send out the appropriate codes before printing international characters. We can do this with the INTERNATIONAL statement:

INTERNATIONAL ON 27,82,12

INTERNATIONAL OFF 27,82,0

Every time an international character is sent to the printer, ASCII 27,82,12 will first be sent to the printer, selecting IBM character set #2. Once the international character has been sent, the original character set will be selected with 27,82,0.

6.2.4 STYLES

You can also set create up to ten print styles:

STYLE 1,27,45,8

STYLE 2,27,83,2

The first number is the style ID, and the rest are the characters which should be sent to the printer when the style is selected in the document with   <n>.

Save your printer driver with <CTRL+S> and call it LW.PDR if you want it to load as the default every time you start LW.

The procedure for most EPSON compatibles should be very similar to the method outlined above, although unfortunately I haven't had access to such equipment while writing this manual.

I used to use a Canon BJ-200ex bubble-jet printer set up in Epson emulation mode with an ICD Printer Connection, and the EPSON.PDR printer driver worked perfectly with the Canon once the DIP switches were set correctly.

I've supplied the EPSON printer driver along with drivers for all the Atari printers. Although I don't own an Atari printer, I was able to figure out the codes they use by making AtariWriter Plus think I had one hooked up, then printing to disk and studying the output. Note that not all Atari printers support features like italics and boldface. I trust these drivers work well with the actual equipment.

Some work may be required to coax international characters out of your printer. However, even if your printer doesn't support them, there are plenty of utilities for downloading fonts to printers. Just download a font which emulates the Atari international character set, set up a printer driver, and you're good to go. Being able to print international characters without fuss was one of the key reasons that LW was written in the first place. I wanted a word processor which had them visible on the screen and didn't require special commands in the middle of a document.

7   MACROS

LW's Macro facility is among the most powerful of any Atari 8-bit word processor. Macros allow you to automate frequent tasks, redefine the keyboard layout, call up passages of text with a single keystroke, create interactive menu systems, and construct entirely new commands by combining existing features of the program. What's more, macros can now be attached to standard command keystrokes, so they can work like built-in features of the program. Macros can even disable screen updates and prompts while they work, so all you see is the end result.

LW's macro commands are really a superset of the commands found in the Public Domain word processor TextPro, and any users familiar with the way macros work in that program will have little difficulty in understanding how they are implemented in LW. Macros are written as plain text files (NOT .DOC document files) containing one macro definition after another. A macro is defined as follows:

\<Macro ID\>▄\<Macro Definition\>

Where Macro ID is the keystroke the macro is called with, the equate symbol is an inverse equals sign, and Macro Definition is simply the string of characters and commands the macro issues. Macros are NOT terminated with \<Return\>. The end of a macro definition is merely marked by another definition following it or by the end of the macro file.

Macro files can be up to 4K in length when using "Banked" memory, otherwise the size limit is 1K. Macros can "chain" other macro files, and even selectively run macros in the target macro file.

7.1   LOADING MACROS

Macros are loaded with the \<Shift+Ctrl+M\> Load Macros command. If the file contains a macro attached to the "&" character, this "autoexec" macro will be run immediately after the macro file has loaded, unless you add the "/N" switch after the macro filename on the input line.

When the macro load command is issued from within a running macro, it's possible to pre-select a different macro to run instead of the "&" macro. See the pre-select macro command later in this chapter for details.

When LW first starts, it looks for the macro file LW.MAC and tries to run the "@" macro if one exists. This "start-up" macro is only ever run when the program first loads LW.MAC. SpartaDOS X users can disable the start-up macro from the command line or even select a different macro in LW.MAC to execute at start-up. Users of other DOS packages can disable autoexec and start-up macros by holding down the \<Option\> key during the loading process.

7.2   RUNNING MACROS

Macros are executed in one of three ways:

- Pressing <ESCAPE>, then the key the macro is attached to.

- Holding down <Option> then pressing the keystroke the macro is attached to.

- Pressing the key combination the macro is attached to.

An important change has been implemented in LW 3.1 regarding macros. In response to requests from several users to be able to enter <CTRL+KEY> characters into the editor without pressing <CTRL+Escape> first, keystrokes are now scanned against macro definitions BEFORE they are scanned against the list of internal commands. This means you can now write a macro which totally supersedes a built-in command. You can, for example, attach a macro to the <CTRL+L> keystroke, and this macro will override the "Load" command. Note that the load command will be completely lost unless you implement a way of calling the <CTRL+L> command in another way. The only way you can now override a macro which replaces an internal command and get the internal command back from the keyboard is by holding down <Option> while typing the command keystroke.

There is one other special macro keystroke:

<Start>   Pressed on its own will run the <#> macro, should one exist. This is another throwback to TextPro. However, in LW, use of the Start key has been augmented. Holding down <Start> while pressing another key combination will by-pass macro scanning. This means you can selectively run the underlying built-in command, even if its keystroke has been "stolen" by a macro definition.

The way <Escape> triggers macros is one of the reasons <CTRL+ESCAPE> or <SHIFT+ESCAPE> is necessary to enter control characters in the editor and input dialogues. Another reason is that I wanted to make <ESCAPE> an abort key at all other times.

7.2.1   AUTORUN MACROS

LW has two kinds of "autorun" macros: The Startup macro and the Autoexec macro. The Startup macro is attached to the "@" key and is only run when LW first loads: therefore, the "&" needs to be defined in LW.MAC.

The second kind of autorun macro is the autoexec macro, which is should be defined on the "&" key. This macro is run when a macro is loaded during an editing session with the <SHIFT+CTRL+M> command. Note that the autoexec macro will not be run in LW.MAC when it is loaded at startup (the startup macro is run instead); if LW.MAC is re-loaded later in an editing session, however, the autoexec macro (&) will be run instead of the startup macro.

You can disable the startup macro (@) in one of several ways:

- Specify the "/M" switch on its own on the SpartaDOS X command line
- Hold down <Option> while LW is loading

You can disable the autoexec macro (&) as follows:

- Hold down <Option> when loading a macro with <SHIFT+CTRL+M> (press <Option> after pressing <Return> and hold it down until the macro file has loaded.
- Specify the "/M" switch immediately after the macro filename when issuing the <SHIFT+CTRL+M> command.

7.3 WRITING AND EDITING MACROS

Because macros tend to contain lots of control codes, it's best to load in the special macro editing font before you start. It substitutes special alphabetic characters for the control codes and makes macros much more readable. Load the macro font with:

<Shift+Ctrl+N> then type MACRO and press <Return>

The above will work in either 80 or 40 column modes because there's a macro font for each mode: MACRO.F80 for 80 column mode and MACRO.FNT for 40 column mode. You may find it much preferable to switch to 40 column mode when editing macros simply because the 40 column font is easier to read and accuracy is important when writing macros.

Switch to 40 column mode with:

<Shift+Ctrl+W>

You may need to go through the procedure to load the 40 column macro font if you haven't already done so.

All the examples here are illustrated by screen shots of the macros in 40 column mode with the MACRO.FNT character set loaded.

7.4 SPECIAL MACRO COMMANDS

As with TextPro, the special macro commands are entered as INVERSE <CTRL+KEY> characters, and are only available from within macros. You type these with <CTRL+ESCAPE> followed by either <INVERSE>, <CTRL+KEY>, then <INVERSE> again, or <SELECT+CTRL+KEY>, which is another TextPro feature which has been emulated to make the typing of the odd inverse character less of a hassle.

<Select+CTRL+A> Ask for Input String. This command obtains string input from the user on the command line. Follow the command with message, ending in <RETURN>, just as you would with the macro print message command. This message becomes the input prompt.

Any following text appears as the default contents of the input field. To actually get input from the user, you MUST include the <CTRL+L>ine Input macro command as normal. The macro will then pause, allowing the text entry until <RETURN> is pressed. The input is sent to the paste buffer and overwrites its contents, even if a null string was entered. If the <CTRL+B>ranch Macro is used before the Ask command, the branch will occur if the input string was EMPTY.

The input string can be pasted into the document in the same way as any normal paste operation. It can also be inserted into a filename/search/replace input line with the <CTRL+B> command (see below).

The Ask for Input command has virtually unlimited scope for macro development, allowing the creation of truly interactive, professional looking applications. Just remember that the contents of the paste buffer are lost when this command is used.

<Select+CTRL+B> Branch to macro. Use this macro to create a branching condition. Follow with a macro identifier prior to the following macro commands, and the branch will according to the following conditions:

Command	Branch
<SHIFT+CTRL+M> Load Macros	To target macro in loaded macro file
<CTRL+F> Find String	If string is NOT FOUND
<SHIFT+CTRL+G> Goto bookmark	If bookmark NOT FOUND
<Select+CTRL+C> Macro Confirm	If user responds "N"
<Select+CTRL+A> Macro "Ask"	If user enters null string
<Select+CTRL+Z><n> Select bank	If text bank not found
<CTRL+L> Load Text	If "Linked Load"

See the <CTRL+Z> command for more conditional branching commands.

<Select+CTRL+C> Confirm (Y/N). Follow with a message terminated with <RETURN>. LW will print the message, followed by a question mark, then "(Y/N):". The user responds with the appropriate key. "Y" will allow the macro to continue. "N" will terminate all macros (even if the macro running is nested), or, if a branch macro is pending after a <CTRL+B>ranch command, that macro will be run. NOTE: Before the introduction of "macro conditionals" in later TextPro versions, that program's <equivalent "Y/N" command, <SELECT+CTRL+A>sk, always attempted to run the "&" macro if "N" was pressed. LW does NOT include this feature: use the <CTRL+B>ranch to pre-select a macro to run if "N" is pressed.

<Select+CTRL+J> Macro menu. Follow with a line of text terminated with <RETURN>. The text should be in the form of some kind of small menu. This message will be printed, then the program will run the macro attached to the next key pressed.

The macro menu has been augmented somewhat since version 2.1 of LW. Since the macro menu command is by definition the last command in a macro, it's now possible to add two extra parameters after the menu text. On a new line, type all the valid keystrokes for the menu options, ending in <Return>. Then, on the next line, type the ID characters for the macros you want each keystroke to run (in the same order), ending the line with <Return>.

For example, you could say:

#▊<INV CTRL+J>▊**oad,** ▊**ave,** ▊**rint**<Return>
LSP<Return>
▊▊▊<Return>

The above macro will display:

▊**oad,** ▊**ave,** ▊**rint**

when <Start> is pressed, and wait for a keystroke. Pressing <L> will run the macro attached to the inverse L character, <S> the macro on the inverse S character, and <P> will attempt to run the inverse P macro. This way, you can keep all your macros off commonly used keystrokes, but still invoke them using simple keystrokes from within menus.

Note: It's still possible to omit these two extra lines of information, and the menu will simply run the macro attached to whichever key the user presses.

<Select+CTRL+K> Get key. Simply waits for a key, then continues the macro.

<Select+CTRL+L> Accept line. In either the editor or an input dialogue, this character will pause the macro and allow user keyboard input until <RETURN> is pressed or the macro is stopped with <BREAK>. Note that many features of the editor, including the icon bar, are disabled in Accept Line mode. If you run another macro whilst in accept line mode, the current macro will be abandoned and accept line mode terminated. NOTE: TextPro input mode always works in OVERTYPE mode - LW accept line mode works IN WHATEVER MODE THE EDITOR IS IN AT THE TIME. Also, the colon delimiters of TextPro are NOT supported by LW.

The Accept Input command no longer filters out cursor movement other than left/right as did Version 1.0. Only a few commands - mostly those requiring input - are now disabled during macro input mode. This change was implemented to allow interactive macros far greater scope. A macro can now, for example, pause while the user marks a block of text, then, when return is pressed, operate on the defined block.

<Select+CTRL+V> Print message. Follow with text terminated by <RETURN>. LW will print the message, which will clear on the next keystroke.

<Select+CTRL+X> Execute macro. Follow with a macro identifier. LW will attempt to execute the macro in the form of a SUBROUTINE or PROCEDURE. This means when the executed macro terminates, the calling or parent macro will resume from the next instruction following the execute command. Macros calls can be nested in this way up to a depth of 128.

<Select+CTRL+Z> Set toggles and test flags. Follow with one of the characters below:

U	Put the keyboard into uppercase.
L	Put keyboard into lowercase.
I	Set insert mode.
O	Set overtype mode.
R	Set reverse video mode.
N	Set normal video mode.
1-9 or 0 (0=10)	Select the appropriate text bank when multiple banks are set up. Bank 1 is always the MAIN (unextended) bank, and 2-10 correspond to banks of extended memory.
B	Select the text bank the program was in when the macro was started.
H	Hide the screen display.
U	Turn the screen on again.
X	Clear "Edited Text" ("*") flag.

These parameters should be in normal video and each setting requires a separate <CTRL+Z>.

As well as setting flags, the <CTRL+Z> command can also test certain conditions:

M	Test for block marking.
S	Test if any text is already selected.
C	Test for file edits since the text in memory was last saved.
A	Test if file has already been saved.

You would precede these tests with a macro branch command. The branch will occur if the above conditions are FALSE.

7.4.1 DISABLING THE SCREEN FROM MACROS

You can use the:

<SELECT+CTRL+Z> set toggles command

to turn the display on and off from within a macro. Follow with an "H" to "hide" any screen updates, and a "V" to make them visible. Nothing is actually printed to the screen when the display is disabled - the display will immediately change to reflect any changes made by the macro when it is switched on again. This is a great way to make macros run as if they were built-in commands, bypassing the prompts that usually

whizz by on the command line. Certain commands - such as print, view, spool, disk menu and the four macro commands which display prompt information - re-enable the display automatically.

7.4.2 SPECIAL CHARACTERS

<CTRL+P>   Entered in a filename dialogue will enter the device, path and name of the current file.

<CTRL+N>   Entered in a filename dialogue will enter name of current file without device.

<CTRL+B>   When pressed in ANY input dialogue (unless preceded by <CTRL+ESCAPE>), <CTRL+B> will place the contents of the paste buffer (or as much of it as will fit) into the input line. Using this method, if you previously captured text with the Ask for Input command, it can be transferred into any LW command which requires input. Similarly, you could cut text from the document and feed it into an LW command. Note that if the input dialogue is associated with a filing operation, the string will appear in uppercase.

<CTRL+L>   Will insert the name of the last file loaded in any bank (see "Linked Files").

In order to make these new commands as flexible as possible, the device/path/name and path/name variables are now accessible from ANY input dialogue. Precede them with <CTRL+ESCAPE> to type them literally.

7.4.3 ENTERING OTHER COMMANDS FROM MACROS

While <ESCAPE> is used to start macros from the editor, from within a macro, <CTRL+X> does this job. This means that FROM MACROS, <ESCAPE> PERFORMS ITS USUAL JOB OF PRECEDING CONTROL CHARACTERS. If you want to enter any command code from within a macro as part of your text rather than as a command, just precede it with an <ESCAPE> character in the macro. Many LW commands are attached to <SHIFT+CTRL> key combinations. Obviously these have no ASCII equivalents, so how are these commands denoted in macros? Simple - from a macro, just think <INVERSE CTRL> instead of <SHIFT+CTRL>. So, to enter the<SHIFT+CTRL+F>ind string command from within a macro, you would type an:<INVERSE CTRL+F>instead, or:<SELECT+CTRL+F>This is why the special macro commands use only those characters that relate to illegal <CTRL+SHIFT> key presses.

When a macro is running, the only keys read from the keyboard are get key commands, confirm commands, text entered during accept line mode, and characters pressed during printing when page wait is on. The "Press a key" prompt after a file view/print operation requires a keystroke from the active macro to clear it and return to the editor.

7.4.4 THE SPECIAL MACRO FONT

The font MACRO.FNT/MACRO.F80 font on the distribution disk can be loaded by typing:

<SHIFT+CTRL+N> New font, typing MACRO <RETURN>. This works in both 80 and 40 column modes. These fonts define all the control keys as special, heavy characters instead of international characters in order to make editing macros a little easier.

7.4.5 KEYBOARD CONVENTIONS FOR MACROS

Understanding how the keys in LW work may seem complex at first, so before we step through some example macros, let's recap:

- <CTRL+ESCAPE> or <SHIFT+ESC> allows you to enter control codes into the editor or into an input dialogue, as <ESCAPE> on its own normally does in BASIC, etc. To get the escape code itself (which appears in LW as a curved downward pointing arrow) in the text, press <CTRL+ESCAPE> or <SHIFT+ESCAPE> twice.

- You can also press <SHIFT+CTRL+ESC> to put the Escape character directly into the text.

- <CTRL/ SHIFT+ESCAPE> pressed when text is marked will unmark the text, as will <BREAK> or any text typed.

- The <ESCAPE> symbol in a macro duplicates <CTRL/SHIFT+ESCAPE> typed at the keyboard. To make a macro put the <ESCAPE> symbol into the editor, include two consecutive <ESCAPE> symbols in the macro.

- <ESCAPE> pressed at the keyboard runs macros from the editor.

- <ESCAPE> pressed outside the editor aborts the current operation.

- In macros, special macro commands and <SHIFT+CTRL> commands are entered as INVERSE <CTRL+KEY> COMMANDS.

The best way to consolidate our understanding of macros is with a couple of examples. Studying the macros supplied on the distribution disk will also help you to understand the LW macro language.

7.5 CREATING AND EDITING MACROS