

The Last Word

Professional Word Processing For the Atari XL/XE



- Dual 40/80 Column Display
- High Speed Editing
- Up to Ten Files in Memory
- SpartaDOS X Integration
- DOS 2.5 and MyDOS Support
- Advanced Macro Feature
- Comprehensive File Manager
- Customisable Keyboard
- Flexible Print Formatting
- Spelling-Checker
- And Much, Much More...

Version 3.21

By Jonathan Halliday

For
Atari XL/XE
Computers
with 64K
or more

The Last Word

The Last Word is one of the most advanced word processors ever written for the 8-bit Atari. It was conceived in the late 1980s, written in the late 1990s, and spent nearly a decade lying unpublished in a disk box on top of a wardrobe. Then, in November 2008, the disks were rediscovered and the program enjoyed well-received publication on the Internet. But in the ten years which had elapsed since version 2.1 of the Last Word was written, the Atari world had changed beyond recognition. Applications, operating systems and hardware were all much more sophisticated than in the past. So began a development cycle which resulted in the first word processor for the Atari 8-bit to feature both 40 and 80 column displays, working at high speed without any additional hardware.

Since its initial release, The Last Word has undergone continuous development and testing, and is used by Atari enthusiasts around the world.

Version 3.1 of the program won the "Application of the year 2009" award, voted for by members of the ABBUC forum. Here is a sample of some of the other accolades LW has received:

- < *"Your word processor is the finest one ever seen on A8 machines."*
- < *"Awesome work!"*
- < *"A monumental accomplishment."*
- < *"An amazing piece of software."*
- < *"This is fantastic. Voted 'Best software of the Year!' by me."*

Credits:

The Last Word 3.21 was developed using:

- < *ATASM*
- < *WUDSN IDE for Eclipse*
- < *Atari800WinPlus*
- < *Atari800MacX*
- < *Altirra*
- < *Aspeqt*
- < *SpartaDOS X*
- < *SIO2SD*
- < *VBXE2*
- < *MyIDE*

Earlier versions were developed using the MA65 assembler for SpartaDOS, the XEDIT text editor (both available at www.atari8.co.uk) and SpartaDOS X.

Special thanks:

Paul Fisher, Konrad Kokoszkiewicz (KMK), Sebastian Bartkowicz (Candle ~~0~~6\$in), David Whyte, Claus Bucholz, Cabell Clarke, Mark Grebe

The Last Word

Professional Word Processing
for the Atari XL/XE

With dual 40 and 80 column display

Version 3.2

Written by and Copyright © Jonathan Halliday 1999-2010

Title screen and fonts by Paul Fisher.

Contents

1	INTRODUCTION	1-1
1.1	OVERVIEW OF THE LAST WORD.....	1-1
1.2	ABOUT THE MANUAL	1-1
1.3	HARDWARE REQUIREMENTS	1-1
1.4	STARTING LW	1-2
1.4.1	LOADING LW FROM SPARTADOS X.....	1-3
1.5	BASIC OPERATION.....	1-3
1.5.1	THE EDIT SCREEN	1-3
1.5.2	SAVING AND LOADING TEXT	1-5
1.5.3	TEXT BANKS.....	1-6
1.5.4	THE FILE SELECTOR	1-6
1.5.5	BASIC CONFIGURATION	1-7
1.5.6	LEAVING THE PROGRAM	1-7
2	EDITOR COMMANDS.....	2-9
2.1	CURSOR MOVEMENT	2-9
2.2	TEXT ENTRY MODES.....	2-9
2.3	INSERTING AND DELETING TEXT	2-10
2.4	MOVING AND COPYING WITH TEXT BLOCKS.....	2-11
2.5	FINDING AND REPLACING TEXT	2-12
2.5.1	SEARCHING WITH WILDCARDS.....	2-13
3	ADDITIONAL EDITOR FEATURES.....	3-14
3.1	COUNTING WORDS	3-14
3.2	EDITED TEXT INDICATOR	3-14
3.3	TABULATION.....	3-14
3.3.1	TABULATION MODES	3-14
3.4	BOOKMARKS	3-15
3.5	TEXT AND DOCUMENT MODES.....	3-15
3.6	USER OPTIONS	3-15
3.7	EDITING MULTIPLE FILES.....	3-16
3.8	HANDLING LARGE FILES.....	3-16
4	DISK OPERATIONS	4-18
4.1	DISK OPERATIONS FROM THE EDITOR	4-18
4.2	THE DISK MENU.....	4-18
4.2.1	ADDITIONAL COMMANDS	4-21
4.2.2	SUBDIRECTORY FEATURES	4-21

The Last Word 3.21 Reference Manual

5	PRINTING WITH LW	5-22
5.1	PREVIEWING TEXT	5-22
5.2	KEEPING TRACK OF PAGINATION.....	5-22
5.3	EDITOR PRINT COMMANDS.....	5-22
5.4	EMBEDDED COMMANDS.....	5-23
5.4.1	STAGE 1 COMMANDS	5-23
5.4.2	STAGE 2 COMMANDS	5-27
5.4.3	CREATING HANGING INDENTS.....	5-28
5.5	OTHER PRINT FEATURES	5-28
5.5.1	INTERNATIONAL CHARACTERS.....	5-29
5.6	CONFIGURING THE PRINT FORMATTER	5-29
6	CONFIGURING LW FOR YOUR PRINTER.....	6-30
6.1	PRINTER DRIVERS.....	6-30
6.2	CREATING A PRINTER DRIVER	6-30
6.2.1	PRINT TOGGLES	6-30
6.2.2	CONTROL STRINGS	6-31
6.2.3	INTERNATIONAL CHARACTERS.....	6-31
6.2.4	STYLES.....	6-32
7	THE SPELLING CHECKER AND OTHER EXTENSIONS	7-34
7.1	CHECKING SPELLING	7-34
7.2	OTHER EXTENSIONS.....	7-35
8	MACROS	8-36
8.1	LOADING MACROS.....	8-36
8.2	RUNNING MACROS.....	8-37
8.2.1	AUTORUN MACROS	8-37
8.3	WRITING AND EDITING MACROS	8-38
8.4	SPECIAL MACRO COMMANDS	8-38
8.4.1	DISABLING THE SCREEN FROM MACROS	8-41
8.4.2	SPECIAL CHARACTERS	8-42
8.4.3	ENTERING OTHER COMMANDS FROM MACROS.....	8-42
8.4.4	THE SPECIAL MACRO FONT	8-43
8.4.5	KEYBOARD CONVENTIONS FOR MACROS	8-43
8.5	CREATING AND EDITING MACROS	8-43
8.6	EXAMPLE MACROS	8-44
8.7	MACRO SUMMARY	8-46
9	CONFIGURING LW	9-47
9.1	CONFIGURATION OPTIONS IN THE EDITOR	9-47
9.2	.CFG CONFIGURATION FILES.....	9-48
9.2.1	THE DEFAULT DRIVE	9-49

The Last Word 3.21 Reference Manual

9.3	THE LW.SYS FILE.....	9-50
9.3.1	CONFIGURATION PROFILES USING LW.SYS	9-51
9.3.2	CONFIGURATION USING A SUPPORTED DOS	9-51
9.3.3	CONFIGURATION USING OTHER DOS PACKAGES	9-52
9.3.4	THE SEARCH PATH.....	9-53
9.3.5	THE KEYBOARD BUFFER.....	9-53
9.4	USING MULTIPLE TEXT BUFFERS.....	9-53
9.5	CUSTOM FONTS.....	9-54
9.6	CUSTOMISING THE KEYBOARD.....	9-54
9.6.1	THE KEYBOARD TABLE	9-54
9.6.2	REMAPPING COMMANDS USING MACROS.....	9-56
9.6.3	REMAPPING COMMANDS USING THE VECTOR TABLE	9-56
9.6.4	1200 XL KEYS	9-59
10	DOS PACKAGES AND LW.....	10-60
10.1	MEMORY REQUIREMENTS.....	10-60
10.2	ATARI DOS 2.5	10-60
10.3	ATARI DOS XE.....	10-60
10.4	MYDOS 4.5.....	10-61
10.5	DISK-BASED SPARTADOS.....	10-61
10.6	SPARTADOS X	10-61
10.6.1	SPARTADOS X ENVIRONMENT VARIABLES	10-62
10.6.2	SPARTADOS X MEMORY CONFIGURATIONS.....	10-63
10.6.3	SPARTADOS X DEVICE NAME MODE.....	10-63
10.6.4	ACCESSING DRIVE LETTERS FROM THE DISK MENU	10-65
11	LW COMMAND SUMMARY.....	11-66
11.1	EDITOR COMMANDS	11-66
11.2	SPECIAL KEYS.....	11-68
11.3	MACRO COMMANDS	11-69
12	PRINT FORMATTING COMMANDS	12-70
13	PROGRAMMER'S TECHNICAL NOTES	13-71
13.1	ASSEMBLY LANGUAGE ADD-INS	13-71
13.2	MEMORY USAGE	13-71
13.3	PROGRAM DESIGN.....	13-72
13.4	DEVELOPMENT AND TESTING	13-73
13.5	WHY LW CAME INTO BEING	13-73
13.6	DEVELOPMENT.....	13-76
13.7	CORRESPONDENCE.....	13-77

The Last Word 3.21 Reference Manual

1 INTRODUCTION

1.1 OVERVIEW OF THE LAST WORD

The Last Word (LW) is one of the most powerful word processors ever written for the 8-bit Atari. The program combines many of the advanced features found in the best word processors and combines them into one package, adding support for the latest disk operating systems. 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 to 10 files simultaneously on expanded memory machines
- Sophisticated keyboard macro language
- 80 column print preview
- Spelling checker (130XE only)
- Versatile and flexible Cut and Paste features
- Search and replace, including reverse search and global replace
- Consistently responsive editing, regardless of position in the text
- Comprehensive on-line help system
- Feature-packed file selector/disk menu which is the equal of many dedicated file management programs and includes up to 80 filenames in a scrolling window, support for SpartaDOS X directories with time/date stamps, batch file deletion, renaming and copying, and much more
- User-definable tab ruler
- Customizable, plain-text printer drivers
- Plain-text configuration files
- Completely redefinable keyboard layout
- Microsoft Windows[®] compatible keyboard shortcuts
- All international characters visible on the screen
- Support for SpartaDOS X, MyDOS, DOS 2.5 and many DOS 2 derivatives
- Runs with only 64K
- Comprehensive print formatting commands
- Automatic heading levels
- Indents, hyphenation, and much more...

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 HARDWARE REQUIREMENTS

LW will run on most stock XL/XE Ataris. The minimum requirements are:

- < 64K of memory
- < Floppy disk drive
- < DOS 2.5 (supplied on the distribution disk)

The Last Word 3.21 Reference Manual

However, for enhanced performance, the following specification is recommended:

- < 128K . 1088K of RAM
- < SIO2SD, SIO2PC, MyIDE, IDEa or other high-speed disk storage
- < SpartaDOS X (version 4.42 or above recommended, 128K required)
- < RAM disk (optional)
- < High quality LCD or CRT monitor
- < Parallel printer

Although LW will function on a 64K Atari, only single file editing is allowed and the cut and paste and macro buffers are extremely small. On expanded memory machines, LW will consume up to 256K for its own use and allow the loading of extensions, complex macros, and a large paste buffer.

1.4 STARTING LW

Boot the computer with the LW disk in drive 1 while holding down the <Option> key. (If ^ [~ Á â [] q c Á @[| á Á â [, } Á Ł U] c ã [} N Á æ} á Á c @^ Á Ó Œ Ù Q Ô Á %Ü ^ æ å press <Return>). The DOS menu will appear. LW should be started with Binary Load Ç [] c ã [} Á Š D Á [} Á c @^ Á Ö U Û Á G È L W Á X E + } È ^ Á È Á @ Ú ! } ^ Á • Á % Š + Á Á æ } á Á d }

Note: LW 3.x is NOT compatible with DOS XE, or with any DOS which uses RAM under the Operating System.

When LW loads, it looks on the default drive Ç %Ö K using DOS 2.5, MyDOS, or the DOS logged drive if using SpartaDOS X) 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 8.
LW.EXT	Machine code extensions (expanded memory machines only). Contains extra program functionality such as character maps, calculators, etc.

The Last Word 3.21 Reference Manual

1.4.1 LOADING LW FROM SPARTADOS X

Under SpartaDOS X, LW is launched by typing

X LW.EXE

Note: LW 3.x 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. LW will not work with U8CG's 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.EXE LETTER.TXT

LW will attempt to load LETTER.TXT automatically. If the file isn't found, LW will start

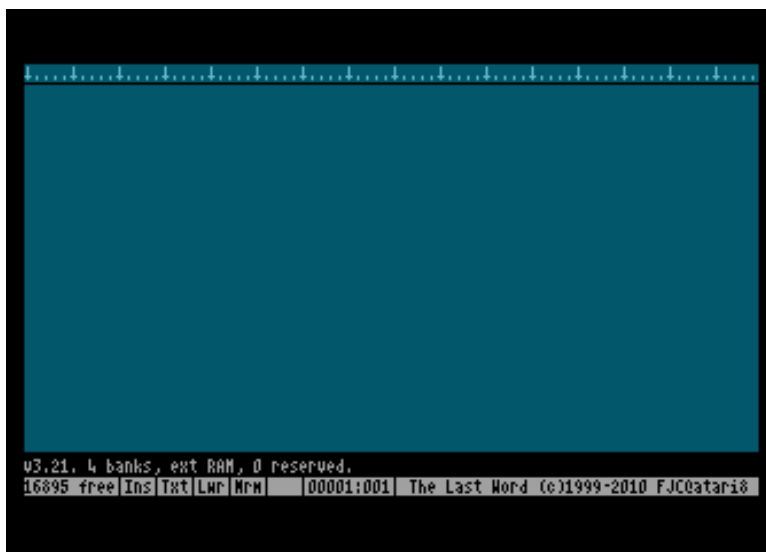
reading the command line. This means any device names on the command line will be read as is. See section 10.6.2 for a list of SDX device names in LW.

1.5 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.

1.5.1 THE EDIT SCREEN

To begin using LW, load the program as described above and take a moment to familiarize yourself with the editing screen. LW will display 80 columns of text on the screen.



The Last Word 3.21 Reference Manual

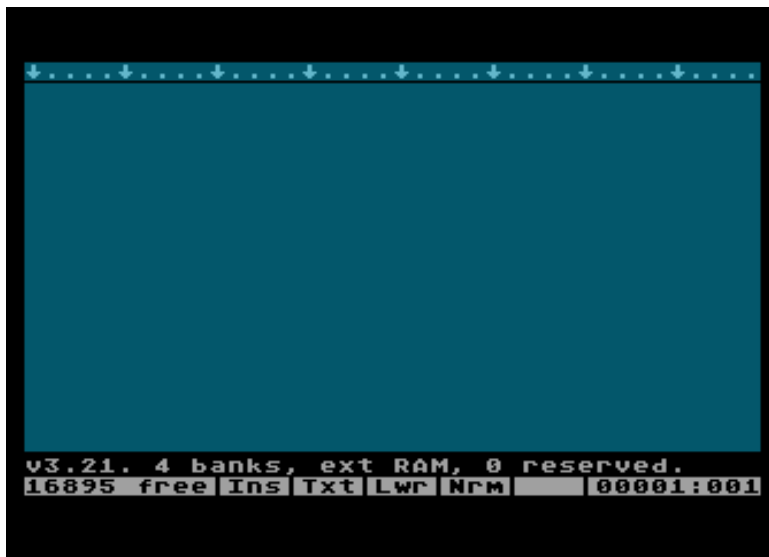
LW defaults to an 80 column display, but you can get the 40 column display back at any time (and make the program default to 40 columns). To switch to a 40 column display, press:

<Shift+Ctrl+W>

At the lower left of the screen, y[~ q | | Á • ^ ^ K

80 Columns **[Y/N]?**

Ú! ^ • • Á c @^ Á %P + Á \ ^ ^ Ê Á æ} á Á ~ [! Á c @^ Á { [{ ^ } c Ê Á b ~ • c Á] ! ^ • Á prompt. A 40 column display will then appear:



You can get the 80 column display back by performing the same procedure in reverse: type <Shift+Ctrl+Y N Ê Á c ^] ^ Á %ÿ + Á ~ [! Á Ì € Á & [| ~ { } • Ê Á c @^ } Á] ! ^ • • Á prompt.

In this manual, many illustrations á ^] ã & c Á Š Y q • Á l € Á & [| ~ { } Á á ã •] | æ^ Ê Á V @ 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 (the message line) will show the version # of the program, the number of text banks, whether the internal buffers are in low or banked memory, and the number of banks reserved for extensions. Thereafter, the message line will default to the name of the file currently in memory (preceded by the bank number if there is more than one buffer set up). Until you load a file or save the contents of the text buffer under a different name, the default filename will be UNNAMED.TXT (followed by the bank number if there is more than one text bank).

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.

The Last Word 3.21 Reference Manual

Cursor keys, <DELETE/BK SP>, and <INSERT> keys behave exactly as you would expect.

You might notice that 80 column mode is a natural result of the 80 column display being rendered entirely as a bit-mapped graphical display. However, LW still uses 80 keystrokes. This is because it has its own type-ahead keyboard buffer. Note that if you repeat keys when you run LW, the SpartaDOS keyboard buffer will supersede the LW keyboard buffer. This version is more compatible with LW that previous incarnations of the SDX keyboard buffer, and is highly recommended.

BCH9 . ' H \ Y ' ` U h Y g h ' j Y f g] c b ' c Z ' G d U f h U 8 C G ' L ' \ U g ' U

1.5.2 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 %Ø Q Š Ò Ò Ý V + Á statement in the configuration file.

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 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.

If you save a file on disk of the same name, you will be warned. The program will ask:

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

Q ~ Á ^ [~ Á c ^] ^ Á %Ø Ý + Ê Á c @ ^ , Á ã c ã • ç Æ Á V Á -] ä | } ^ Á , %Ø | + | Á Ä ä ^ Á [Á ç 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, and press <Return>.

The Last Word 3.21 Reference Manual

BCH9 . ' H \ Y ' X f] j Y ' g d Y W] Z] Y f ' k \] W \ ' U d d Y U f g ' U h ' h \ Y ' í
other file load prompts in the program) reflects the drive currently logged by
@K Ð g ' 8] g _ ' A Y b i ž ' U b X ' B C H ' h \ Y ' X Y Z s D O S s e e s i t f ' W i f f Y b h ` n
The device name will disappear when you start typing, but will be prepended to
the filename you enter unless you specify a different device ID.

1.5.3 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>

, @ ^ ! ^ Á % } + Á ã • Á [} ^ Á [~ Á c @ ^ Á } ~ { à ^ ! Á \ ^ ^ • g f a n i s ã c @ Á % € + Á á ^
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 bank if there are RAMdisks present.

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

<Shift+Ctrl+?>

which will display:

n banks, [*low/banked*] RAM, *n* reserved.

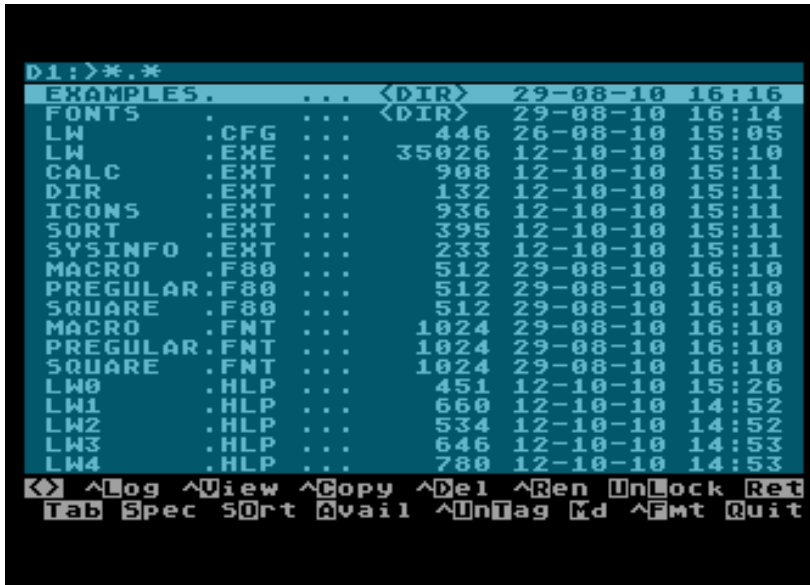
This command reports how many text buffers are available, whether the program has
ã c • Á ã } c ^ ! } æ | Á à ~ ~ ^ ! • Á ã } Á { æ ã } Á Ç % Š [, Á Ü Æ T + D Á [! Á ^ ç c ^
how many (if any) of the selected memory banks are reserved for use by extensions.

Ù ^ ^ Á % W • ã } * Á T ~ | c ã] | ^ Á V ^ ç c Á Ó ~ ~ ^ ! • + Á ã } Á Ô @ æ] c ^ ! Á ì Á ~

1.5.4 THE FILE SELECTOR

Y @ æ c Á ã ~ Á ^ [~ Á & æ } q c Á ! ^ { ^ { à ^ L W Á h a s @ f i l e s e l e c t o ã w h i c h i s Á æ ! ^ Á [} Á c @
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.

The Last Word 3.21 Reference Manual



Just highlight the file you want and press <Return> c [Á | [æå Á ã c È Ù \$ ¡ ¤ ¤ % W + P Ò Á Ö in Chapter 4 for more information about the file selector.

1.5.5 BASIC CONFIGURATION

You can configure LW to suit yourself. Pressing:

<Shift+Ctrl+O> 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 file. For further information on configuring LW, see section 9.

1.5.6 LEAVING THE PROGRAM

To finish using LW and go to DOS, type:

<Shift+Ctrl+Q> Quit

From the editor and respond "Y" to the prompt.

The Last Word 3.21 Reference Manual

Ø! [{ Á c @ ^ Á ~ ã | ^ Á • ^ | ^ & c [! Ê Á • ã {] | ^ Á] ! ^ • • Á % Û + Á c [Á ^ ¢ ã c
Q ~ Á æ } ^ Á c ^ ¢ c Á ã } Á { ^ { [! ^ Á @ æ • } q c Á ^ ^ c Á à ^ ^ } Á • æ ç ^ á Ê Á ^ [^
before the program shuts down.

The Last Word 3.21 Reference Manual

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>	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+Ctrl+CAPS>	%oQ } c ^ ! } æc ã [] æ Á Š [& \ + Á { [á ^ È Á V @ã • Á { [alpha control characters (Ctrl+A through Ctrl+Z) without first pressing <Shift+ESC>. You need to turn off International Lock mode again by pressing <Shift+Ctrl+CAPS> in order to access functions such as Save <Ctrl+S> and Load <Ctrl+L>. International lock mode is designed to make it easier for

The Last Word 3.21 Reference Manual

	European users to type long strings of characters using the international character set.
<Shift+CAPS>	Uppercase lock.
<INVERSE>	Toggle inverse video on and off.
<ESCAPE>	Cancel block marking if active, or enter control character. Also cancels string input and abandons operations.
<Ctrl+ESCAPE>	Enter control character in editor or input dialogue. Does NOT cancel string input or abandon operation.
<Shift+ESCAPE>	Same as <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 (closes up text to the right of the cursor in insert mode), or marked block.
<Ctrl+INSERT>	Insert a space at the cursor (pushes existing text along to the right).
<TAB>	Insert spaces to next tab stop if in Insert Mode. If the editor is in overwrite mode, the cursor is advanced to the next tab stop without inserting any spaces into the text.
<Ctrl+DELETE>	Delete character under the cursor (closes up text to the right of the cursor when in insert mode).
<Shift+DELETE>	Displays the prompt:

Delete **Word, **L**ine, **S**entence, **P**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.

The Last Word 3.21 Reference Manual

	Paste the text back into the main buffer with <Ctrl+V> or <Shift+INSERT>.
<Shift+INSERT>	Insert previously deleted text
<Ctrl+V>	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. %oT ! \ + Á ã • Á å ã •] æ ^ ^ å Á [} Á c @ while marking is active. 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+M> again or <Ctrl+ESCAPE>.
<Ctrl+X>	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+C>	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.

The Last Word 3.21 Reference Manual

<Ctrl+N>	Displays the cursor position, size of the file in bytes, 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+]>	Invert text in block
<Ctrl+[>	Un-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.
<Shift+Ctrl+R>	Define replace 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.
<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, **A**ll, **T**o **E**nd, **S**kip?

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

The Last Word 3.21 Reference Manual

display is not updated to show each replacement. However, a counter displays a running total of the number of replacements made, and 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> key. You can also deselect the use of wildcards. You can also deselect the use of wildcards by pressing the <Shift+Ctrl> key. You can also deselect the use of wildcards by pressing the <Shift+Ctrl> key.

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 8).

Note that in order to search/replace the inverse question mark literally, you must turn

3 ADDITIONAL EDITOR FEATURES

LW includes many features to aid in the editing of text, such as bookmarks, 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, or if the last character in the text buffer is deleted.

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.

The Last Word 3.21 Reference Manual

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 æ! ^ Á • æç ^ á Á , ã c @Á c @^ Á document mode Á ^ [~ q | ^ Á , [| \ ã }

3.5 TEXT AND DOCUMENT MODES

LW can save files in 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] | æ& ^ Á { æ! \ ^ | • Á , @ã & @Á @æç ^ Á à ^ ^ } Á • ^ c È ÁW [~ Á , [} q c Á 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 ã • Á ã } Á %Ö [& + Á { [á ^ È Á Ý [~ Á & æ} Á c ^ | | Á à ^ Á | [[\ ã } * Á æ c Á in, and you can toggle between Doc and Text mode with the *User 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)?	W• ^ Á %N + Á æ• Á æÁ , ã á &æhdá A replace operations	On
Padding [N] (Y/N)?	Display false spaces in the editor	On
Doc mode [N] (Y/N)?	Operate in document mode	Off

The Last Word 3.21 Reference Manual

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 9). 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 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 9: 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), all be very small (only about 1K each).

3.8 HANDLING LARGE FILES

Although the largest text buffer LW can provide is around 18K when using a machine

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

When you load a file into a textc 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 %0S even if the file would otherwise have fit into the buffer).

The Last Word 3.21 Reference Manual

To protect against accidental obliteration of the original file on disk when only the first

Instead, `<Ctrl+U>` will always be warned before overwriting an existing file. Beyond that,

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 use the `<Ctrl+L>` (or `<Ctrl+V>`) key combination (see the `LOAD` command in the `HELP` file). For example:

Load>REPORT.DOC,C

This simply loads the next segment of the file, providing that the filename given is 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

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 `<Ctrl+S>` switch also works with file copying on the disk). The `<Ctrl+S>` 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, 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 in a single bank. It is an 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 `SPLIT.MAC` on the distribution disk which will perform precisely this task.

The Last Word 3.21 Reference Manual

4 DISK OPERATIONS

LW allows full manipulation of files and directories, and has support for many different DOS packages. The disk 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 <B| ^ æ\ N Á \ ^ ^ È Á Q ~ Ú + [Á • Á ä] c & | @ Á æ ~ Á c ^ @ ^ Á Á % @ Á ~ ã | the text will be displayed in paged rather than scrolling format.

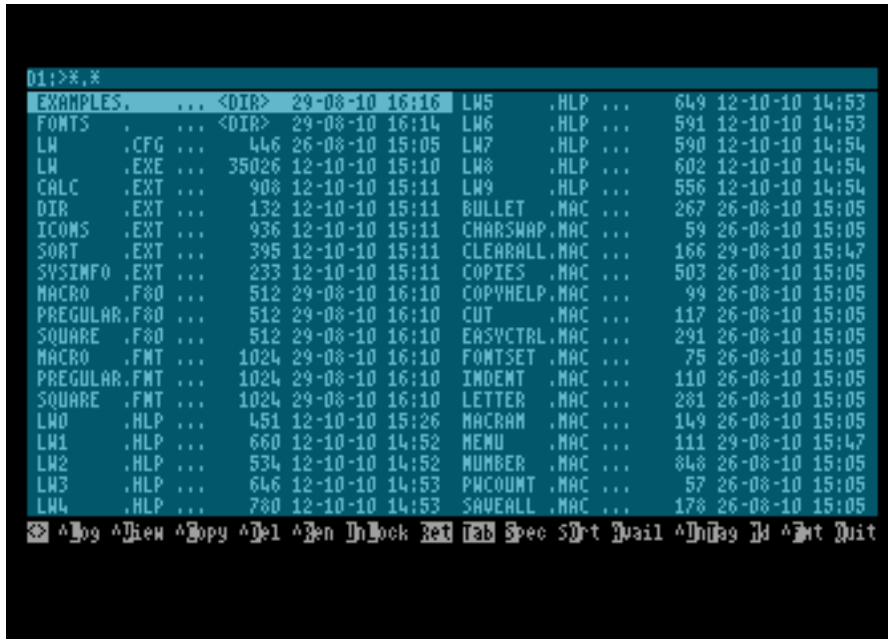
4.2 THE DISK MENU

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

<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 drive, path and file mask before calling the disk menu.

The Last Word 3.21 Reference Manual



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

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.

^Log Log drive. Follow with 1-9 for drives 1-9, 0 for no drive number
 Ç %Ö K + D Ê Á: [through O: of using SpãrãDOS kernel device names (see section 10.6.3).

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 fã | ^ Ê Á ^ [~ q | | Á à ^ Á ! ^ c ~ ! } ^ á Á c [Á c @ ^ Á [! ä * ä filename replacing the originally displayed filename).

Ret Select the filename under the highlight bar.

View View the file under the selector bar. Same as view from the editor without t@^ Á %Ö • , ä c & @

Note: You can also obtain a paged listing (same as from the ^ á ä c [! Á ~ Á * Á c @ @D:Á [! ^ • • ä } * Á Ł

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.

The Last Word 3.21 Reference Manual

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. Press <Ctrl+R> to rename all tagged 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!

The destination filename for Copy can be `^ Á c æ \ ^ Á c , [Á • , ã c & @ ^ • K Á % & æ ~ • ^ Á c @ ^ Á • [~ | & ^ Á ~ ã | ^ Á c [Á æ]] ^ } á ^ á Á c [Á c @` will bypass the check for already existing destination file(s), unless warnings have already been disabled in the editor options (<Shift+Ctrl+U>). In the unlikely event you need to combine both switches, simply end the destination file spec with a comma, `~ [| | [, ^ á Á à ^ Á à [c @ Á • , ã c & @ ^ • Á Ç ^ È * È Á % Ò È Æ Þ + D È`

Press <Ctrl+C> to copy all tagged files. Wildcards are fully supported. For example, you could tag all files on the disk and then press <Ctrl+E Ò N Á Á c [Á & [] ^ Á c @ All files on the % Ö G K E È Ó Æ S + destination drive will have the .BAK extender. To copy a selection of files to drive 8 without being warned if the destination files already exist, tag the files you want to copy, press <Ctrl+C>, then type:

D8:*.*,N

or even:

D8:;N

The files will be copied with their original names, overwriting any files on drive 8 with the same names.

Md Create a new directory in the current directory, providing the DOS used supports subdirectories.

Esc Exit the directory menu.

UnLock Lock or unlock the highlighted file. Note that U and L should be pressed *without* the <CONTROL> key.

^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. `ÿ [~ q | | Á à ^ Á æ • \ ^ á Á ~ [| Á & [] ~ ã | { æ c ã [`

The Last Word 3.21 Reference Manual

- Quit** Leave the program and go to DOS.
- 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.
- 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.
- Tab** When using SpartaDOS X, press <Tab> to toggle between short and long format directory listings.
- In version 3.21 of LW, the directory format is automatically and temporarily changed to short (DOS 2.5 style) when listing a directory on a patched hard drive (Hn:) when the program is used on an emulator.

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 ARROW> Moves the cursor down by a screenful of filenames

4.2.2 SUBDIRECTORY FEATURES

The following options only work with DOSes which support subdirectories.

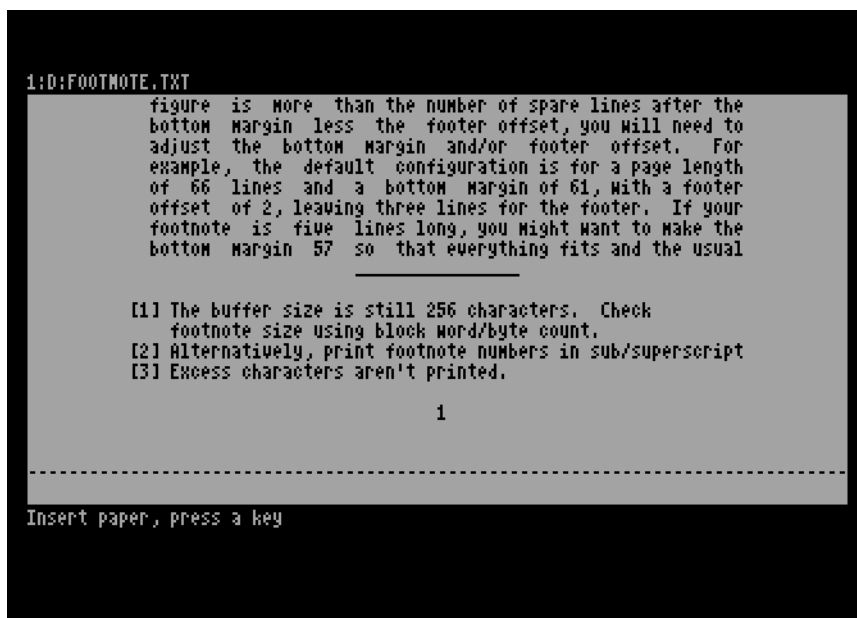
- Return** or > Catalogue the highlighted directory. Only used with Subdirectory oriented DOSes.
- < Go back up one level towards the root directory. DOS-specific, as above.

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

<Shift+Ctrl+P> 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

< 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 <Shift+Ctrl+P> 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>.

The Last Word 3.21 Reference Manual

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.
- Only Stage 2 formatting commands may appear inside header and footer strings. Stage 1 formatting commands are not allowed.
- Formatting commands must not contain extraneous spaces.

Here are some examples of print formatting commands:

l20<Return>

Sets the left margin to 20.

l20r60hello<Return>

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

f#Page #<Return>

Defines a running footer which prints the current page number, centred on the line.

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

The Last Word 3.21 Reference Manual

print your footer (if any), which may be up to 8 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:

fFooter<Return>

Y ã | | Á] ! ã } c Á c @ ^ Á ! ~ } } ã } * Á ~ [[c ^ ! Á %ø [[c ^ ! + Á
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 **f** 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 **f** 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 **f**<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 file which links to other documents will still be in the edit buffer when printing is completed. 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 **f**ooter command. The optional number specifies the offset from the top of the page, and the default value is 2.

l

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

l

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

l

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

The Last Word 3.21 Reference Manual

fj	Justify fully. All following text will be aligned flush with both the left and right margins.
k	Add-in #1
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.
q	Add-in #2
r <n>	Set Right margin. This is the rightmost column in which text will print. Default is 70.
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.

The Last Word 3.21 Reference Manual

- <n>** Right paragraph indent. All following text up the next <Return> will be indented by <n> spaces from the right margin.
- l<n>** Left header/footer margin, default 10. This works like the **l** 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.
- r<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.
- l<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.

The Last Word 3.21 Reference Manual

& 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 (e.g. `^ Á æ!` ^ Á %oC [* * | ^ • + Á ã } Á c @^ Á • ^ } • ^ Á c @æc Á c @^ Á ~ ã ! feature on, and the next occurrence turns it off again). A handy way to enter these commands which saves pressing the inverse key two times is to enter them in conjunction with the <Select> key.

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 be 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. You can precede the edge right command with a centre command to create centred and edged-right text on the same line. See *Centre Line*.














i Toggle italics on or off (see *double-strike*).

s<n> Toggle print style. <n> is 1-5 (also in reverse video). This sends one of 5 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). You would turn on print style #1 with `<s><1>`, and then turn it off with `<s><1>` again.

u Toggle underline on or off (see *double-strike*).

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.

The Last Word 3.21 Reference Manual

 <n>	Send printable code. Works like <i>output ASCII</i> , 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.

The Last Word 3.21 Reference Manual

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 9, Configuring LW, for more information.

The Last Word 3.21 Reference Manual

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 5 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.x printer drivers, which are now 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[1-5] ON!OFF	n,n,n...	Styles 1 . 5 on and off

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:

The Last Word 3.21 Reference Manual

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. The only exceptions are the %Q} c ^ ! } æc ã [] æ| Á U Þ U Ø Ø + Á & [á ^ Á • ^ ~ ^ } & ^ • È Á V @ ^ • ^ Á 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

V @ã • Á ã • Á C Ec æ! ã q • Á } [! { æ| Á Ò } á Á [-LW,ŠcunlesÁy@æ! æ&c ^ ! Á , æ} c Á c @^ Á Ô Û Š Ø Á • ^ ~ ^ } & ^ Á c [Á à ^ Á æ} ^ c @ã } * Á [c @^ ! Á c 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-• ^] æ! æc ^ á Á | ã • c Á [~ Á c @^ Á ^ • & æ] ^ ^ Pri} c ^ ! + Á ã } Á ^ [~ ! If you don't want an initialization string sent to the printer, • ã {] | ^ Á ! ^ { [ç ^ Á c @^ Á %Q P Q V + Á | ã } ^ Á ~ ! [{ Á c @^ Á | ã

6.2.3 INTERNATIONAL CHARACTERS

LW ã } & | ~ á ^ • Á G J Á •] ^ & ã æ| Á %Q} c ^ ! } æc ã [] <Ctrl+Á & @æ! æ&c ^ ! to <Ctrl+Z>, <Ctrl + comma>, <Ctrl + full stop> and <Ctrl+ semi colon>. When using a ~ [] c Á , @ã & @Á ! ^ á ^ ~ ã } ^ • Á c @^ • ^ Á & @æ! æ&c ^ ! • Á æ} á Á æ& & 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.

The Last Word 3.21 Reference Manual

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 five print style toggles for printer features not covered by the italic, double-strike and underline commands:

STYLE1 ON 27,52

STYLE1 OFF 27,53

The above example defines style 1 as an italics on/off toggle, although this is not useful since we already have an italic on/off command.

Þ [c ^ Á c @æc Á c @^ Á %• c ^ | ^ + Á ~ [! } 3.21 of LW and printer } Á & @æ} * ^ á Á ã }
drivers from previous 3.x versions *will not work with version 3.21 if they include style definitions*. Styles in older printer drivers need to be re-written to conform to the new system. The reason the system was changed was to enable the print formatter to selectively turn off printer character styling before printing header and footer lines. In prior versions of LW, the formatter made no attempt to keep header/footer and body text character styling separate from one another, with the result that character styling in the headers could inadvertently cross over to the body text and vice-versa. This long-standing problem is a compelling reason to upgrade to the latest version of LW.

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.

The Last Word 3.21 Reference 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.

The Last Word 3.21 Reference Manual

7 THE SPELLING CHECKER AND OTHER EXTENSIONS

Users running LW on a machine with at least 128KB have access to extensions supplied with the program. Extensions are files ending in *.EXT, and are small executable files invoked with <Ctrl+;> (Control and the semi-colon key). Extensions will only work if you have allocated space for them, and this space is taken from the main text bank when LW is using extended RAM. Therefore, as a minimum requirement, LW.SYS should contain the following lines:

```
BANKED ON
BANKS 1,2,3,4,5,6,7,8,9
EXTPAGES <n>
```

where <n> is a number between 1 and 7 (7 being the maximum size allowed). SpartaDOS X users have the option of leaving out the BANKED ON and BANKS <list> lines, since if DOS has sufficient free extended banks, the program will automatically use them.

Many extensions will run in 1KB of space, the one exception being SPELL.EXT, which requires 16KB banks. An example LW.SYS designed to facilitate the use of SPELL.EXT would be:

```
BANKED ON
BANKS 1,2,3,4,5,6,7,8,9,10,11,12
EXTPAGES 7
RESERVE 2
```

Note that reserved banks are taken from the pool of RAM specified with the BANKS command. Therefore (assuming you have sufficient RAM), if you want ten text banks and two reserved banks, you should specify banks 1 to 12 (two of these banks will be reserved, and one will automatically be allocated for buffers, leaving nine extended banks plus the main bank, i.e. ten text banks).

LW 3.21 is supplied with the following extensions:

SPELL.EXT	Spelling checker
SORT.EXT	Line sorting utility
CALC.EXT	Calculator
DIR.EXT	Utility to import disk directories into the editor
ICONS.EXT	Icon menu system
INFO.EXT	Utility to manage multiple text banks, obtain word counts, etc.

All extensions will run in 4 EXTPAGES, except SPELL.EXT, which requires 7.

7.1 CHECKING SPELLING

To use the spelling checker, you must have dictionary files (LWA.DCT, LWZ.DCT) in a folder accessible by the LW path. One might, for example, place the files in D3:>LW>SPELL> when using SpartaDOS X with a hard disk. The LW path would then need to contain the following entries:

```
D3:>LW;D3:>LW>SPELL
```

The Last Word 3.21 Reference Manual

Placing the dictionary folder last in the path is sensible, especially if creating
à ã & c ã [] æ! ã ^ • Á ~ ! [{ Á • & ! æc & @É Á • ã } & ^ Á ã ~ Á æÁ à ã & c ã []
& ! ^ æc ^ à Á ~ • ã last entry.@^ Á] æc @q •

Naturally one can define the path in LW.SYS, or using an environment variable if using SDX:

```
SET LWPATH=D3:>LW;D3:>LW>SPELL
```

If the LWSDXDEV environment variable is set to 1, the above might read:

```
SET LWPATH=C:>LW;C:>SPELL
```

So, with the system set up, to spell check a file, simply load it into the editor, press
Ł Ő c ! | É L N Ê Á c ^] ^ Á % ù Ú Ò Š Š + Ê Á æ } à Á] ! ^ • • Á Ł Ü ^ c ~ ! } N Á Ç }
are also accessible via the LW path, i.e. stored in the LW folder using the example
above). The spelling checker will immediately begin to check the document,
highlighting misspelled words, offering the chance to correct them or add them to the
dictionary, or ignore them for the duration of the spelling check.

Pressing <Break> at any time (or <Esc> at a prompt) will abort the check.

The spelling checker is supplied with very basic (English) dictionary files. The intention
is that users build dictionaries which reflect their personal vocabulary. Each dictionary
file (representing one letter of the alphabet) can be a maximum of 16KB in size. This
should be enough for a total of 30-40,000 words. Word lists are stored with the first
letter omitted, and in alphabetical order. A blank line represents the initial letter on its
[, } Á Ç ~ [! Á ^ ç æ {] easy to import word lists from other sources, simply by
ensuring they are less than 16KB in size, alphabetically sorted, and lacking their initial
letter.

Q c q • Á @[] ^ à Á c @æc Á • ~]] [! c Á , ã | | Á à ^ Á æ à à ^ à Á ~ [! Á ã } c
now, international characters are supported, but not as the first letters of words.

Note that the spelling checker (like many of the other extensions) ties its functionality
to <Ctrl+T> until you run a different extension. Therefore, any time you want to spell
check a file, just press <Ctrl+T>.

7.2 OTHER EXTENSIONS

Other extensions work in much the same way as the spelling checker. They are all
loaded with <Ctrl+>, and some automatically execute.

- < ICONS.EXT adds an icon menu to LW, accessed with <Ctrl+T>.
- < CALC.EXT accepts numeric expressions (with parenthesis) and displays the results on the message line
- < INFO.EXT displays information about all text files being edited
- < SORT.EXT sorts marked lines of text into order (use <Shift+Ctrl+T> to reverse the sort order)
- < DIR.EXT simply loads the specified directory listing into the editor at the insertion point.

8 MACROS

LWq • Á T æ&! [Á ~ æ&ã | ã c ^ Á ã • Á æ { [-] 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 } ^ , Á & [{ { æ } á • Á à ^ Á & [{ à ã } ã } * Á ^ ç ã • c ã } * Á ~ ^ æ c ~ | ^ • Á [~ Á c 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.

LWq • Á { æ&! [Á & [{ { æ } á • Á æ! ^ Á! ^ æ | | ^ Á æ Á • ~] ^! • ^ c Á [~ Á c @^ 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 with no spaces or EOL characters in between. 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.

T æ&! [Á ~ ã | ^ • Á & æ } Á à ^ Á ~] Á c [Á | S Á ã } Á | ^ } è the@^ } Á ~ • ã } * | ã { ã c Á ã • Á F S È Á T æ&! [• Á & æ } Á %& @æã } + Á [c @^! Á { æ&! [Á ~ ã | ^ the target macro file.

8.1 LOADING MACROS

Macros are loaded with the <Shift+Ctrl+M> Load Macros command. If the file contains æÁ { æ&! [Á æ c c æ& @^ á ^ Á! c È Á c @^ • Á %B æ ~ Á c [@æ! ^ æ& c Á { æ&! [Á , ã | | Á à after the macro file hæ • Á | [æ á ^ á È Á ~ } | ^ + Á Á ^ , [ã ç & @æã æ Á c @^ Á %B @^ Á { æ&! [filename on the input line.

Y @^ } Á c @^ Á { æ&! [Á | [æ á Á & [{ { æ } á Á ã • Á ã • • ~ ^ á Á ~! [{ Á , ã c @æ pre-select a á ã ~ ~ ^! ^ } c Á { æ&! [Á c [Á! ~ } Á ã } • eselect a macro ~ Á c @^ Á %B + Á command later in this chapter for details.

When LW first starts, it looks for the macro file LW.MAC æ } á Á c! ã ^ • Á c [Á! ~ } Á c @^ Á %B ã ~ Á [} ^ Á ^ ç ã • c } • È Á (Væ&!) • [Á %B æ ~ Á c [@æ! ^ æ& c Á { æ&! [Á , ã | | Á à after the macro file hæ • Á | [æ á ^ á È Á ~ } | ^ + Á Á ^ , [ã ç & @æã æ Á c @^ Á %B @^ Á { æ&! [filename on the input line. In 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.

The Last Word 3.21 Reference Manual

8.2 RUNNING MACROS

There are several ways to execute macros:

- < Pressing <Ctrl+Q>, 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.

Command keystrokes (those keystrokes which normally execute editor commands such as Load and Save when not preceded by <Shift+Esc>) 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 [ç ^ | | ã á ^ Á c @^ Á %Š [æ á + Á & [{ { æ } á È Á c p r o p r i e t a r y c l o s e Á c @^ Á | [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 <Start> while typing the command keystroke.

The basic rules of macro execution are as follows:

- < Press <Ctrl+Q> then <key> to run any macro attached to a non-inverse æ|] @æ} ~ { ^ | ã & Á & @æ! æ& c ^ | Á Ç c @æc q • Á æ} ^ Á & @æ! æ& c
- < Press <Option+key> to run any macro attached to any character (inverse or } [! { æ| D Á , @ã & @Á ã • } q c Á æÁ & [{ { æ } á Á \ ^ ^ • c ! [\ ^ Ú! ^ • • Á Ł \ ^ ^ • c ! [\ ^ N Á [} Á ã c • Á [, } Á c [Á! ~ } Á æÁ { æ& ! command keystrokes (the macro supersedes the built-in command)

The <Start> key serves two functions:

<Start> Pressed and released on its own will run the <#> macro, should one exist. This is another throwback to TextPro. However, in LW 3.2, 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 %• c [| ^ } + Á à ^ Á æÁ { æ& ! [Á á ^ ~ ã } ã c ã [] È

%Ò Ý Œ T Ú Š Ò Ù È T Œ Ô + Á [] Á c @^ Á á ã • c ! ã à ~ c ã [] Á á ã • \ Á] ! [ç ã for your own needs.

8.2.1 AUTORUN MACROS

Š Y Á @æ• Á c , [Á \ ã } á • Á [~ Á %œ ~ c [| ~ } + Á { æ& ! [• K Á V @^ Á Ù c æ V @^ Á Ù c æ! c ~] Á { æ& ! [Á ã • Á æc c æ& @^ á Á c [Á c @s:Á %O + Á \ ^ ^ Á c @^ ! ^ ~ [! + Á È Á c @^ Á %Q Á à ^ Á á ^ ~ ã } ^ á Á ã } Á Š Y È T Œ Ô È

The second kind of autorun macro is the autoexec macro, which is should be defined [] Á c @^ Á %B + Á \ ^ ^ È Á V @ã • Á { æ& ! [Á ã • Á! ~ } Á , @^ } Á æÁ { æ& ! 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

The Last Word 3.21 Reference Manual

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:

- < Ù] ^ & ã ~ T Á Á • @, ^ ã % Æ @ Á ã { { ^ á ã æ c ^ | ^ Á æ ~ c ^ ! Á c @ ^ Á { æ & ! [Á ~
- < 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).
- < Ù] ^ & ã ~ ^ + Á Á • @, ^ ã % Æ @ Á ã { { ^ á ã æ c ^ | ^ Á æ ~ c ^ ! Á c @ ^ Á { æ & ! [Á ~ <Shift+Ctrl+M> command.

8.3 WRITING AND EDITING MACROS

Ó ^ & æ ~ • ^ Á { æ & ! [• Á c ^ } á Á c [Á & [} c æ ã } Á | [c • Á [~ Á & [} c ! [| Á & 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 be& æ ~ • ^ Á c @ ^ ! ^ q • Á æ Á { æ & ! [Á ~ 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 @æç ^ } q c Á æ | ! ^ æ á ^ Á á [] ^ Á • [Ë

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

8.4 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

The Last Word 3.21 Reference Manual

<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>branch 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+V> 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:

Key Code	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>	Go to Bookmark	If bookmark NOT FOUND
<Select+Ctrl+C>	Macro Confirm	Q ~ Á ˇ • ^ Á ^ •
<Select+Ctrl+A>	T æ & ! [Á %oO	If user enters null string
<Select+Ctrl+Z><n>	Select bank	If text bank not found
<Ctrl+L>	Load Text	Q ~ Á %oS ã } \ ^ á Á

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>branch 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>branch 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

The Last Word 3.21 Reference Manual

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 | æ• c Á & [{ { æ} á Á ã } Á æÁ { æ& ! [Ê Á ã c q • Á } [, Á] [• • ã 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> , ,  <Return>  
 <Return>  
 <Return>
```

The above macro will display:

, , 

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 K Á Q c q • Á • c ã | | Á] [• • ã à | ^ Á c [Á [{ ã c Á c @ ^ • ^ Á c 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.

The Last Word 3.21 Reference Manual

<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.
K	Start block marking
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	Ô ^ æ! Á %Ô à ã c ^ à Á V ^ ç c + Á Ç %E + D Á

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

As well as setting flags, the <Select+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.

8.4.1 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

The Last Word 3.21 Reference Manual

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. The display is automatically re-enabled when the macro finishes executing if it has not already been turned back on.

8.4.2 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+V> When pressed in ANY input dialogue (unless preceded by <Ctrl+ESCAPE>), <Ctrl+V> 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.
- NOTE: <Ctrl+B> to insert the contents of the paste buffer has been dropped in this version. Use <Ctrl+V> instead.
- <Ctrl+L> Will insert the name of the last file loaded in any bank (see %oŠ ā } \ ^ á Á Ø ã | ^ • + D Ę

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.

8.4.3 ENTERING OTHER COMMANDS FROM MACROS

While <Ctrl+Q> 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

The Last Word 3.21 Reference Manual

view/print operation requires a keystroke from the active macro to clear it and return to the editor.

8.4.4 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.

8.4.5 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> does in the text editor. 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.

NOTE: In version 3.2, you can press <ESCAPE> on its own twice in succession in the editor to enter the escape character into the text.

- You can also press <Shift+Ctrl+ESC> to put the Escape character directly into the text.
- <Ctrl or Shift+ESCAPE> pressed when text is marked will unmark the text, as will <BREAK>, <ESCAPE> or any text typed.
- The <ESCAPE> symbol in a macro duplicates <Ctrl or 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.
- <Ctrl+Q> pressed at the keyboard runs macros from the editor.
- 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.

8.5 CREATING AND EDITING MACROS

Ó ^ & æ ~ • ^ Á { æ & ! [• Á c ^ } á Á c [Á & [] c æ ã } Á | [c • Á [~ Á & [] c ! [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>

V @ ^ Á æ à [ç ^ Á , ã | | Á , [| \ Á ã } Á ^ ã c @ ^ ! Á ì € Á [! Á | for Á & [| ~ { } each mode: MACRO.F80 for 80 column mode and MACRO.FNT for 40 column mode.

The Last Word 3.21 Reference Manual

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

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

8.6 EXAMPLE MACROS

The best way to illustrate the creation of macros is with some useful examples.

DUAL FONT LOADER

When loading fonts into LW, only the font relevant to the current display mode is normally loaded, i.e. if LW is in 80 column mode and you specify a font to load, only the 80 column (.F80) font is loaded, while the 40 column font remains unchanged. You

What if you wanted to load both the 40 and 80 column fonts at the same time? We can write a macro which will function as a new command for loading both types of font.

Y ^ q | | Á] ~ c Á c @<Option+A> keystroke, so create an empty file and type the | ^ c c . The ^ type *Shift+Esc>, <Select+=> to get the assignment character. Now we can type the commands which actually make up the macro.

The first thing we want to do is capture the name of the font set to load. We do this using the Macro Ask command. So we type <Shift+Esc>, <Select+Ctrl+A> to get the macro ask character. This is followed by the input prompt we wish to appear on the screen. Type:

Font Set

Press <Return> to end the string. Next, we need to stop the macro issuing keystrokes until the user has a chance to type the name of the font set and press <Return>. This is done with the macro input command, so we type:

<Shift+Esc>, <Select+Ctrl+L>

Now press <Return> - this adds the <Return> which the macro will issue to terminate c @ ^ Á ~ • ^ | q • Á ã }] ~ c Á Ç , @ ^ } Á c @ ^ Á ~ • ^ | Á] | ^ • • ^ • Á Ł Ü ^ c ~ | } N input, that <Return> is not issued to the input line: it merely tells the macro we have ~ ã } ã • @ ^ á Á ^ } c ^ | ã } * Á c ^ ç c É Á V @ æ c q • Á , @ ^ Á , ^ Á @ æ ç ^ Á c [Á ã • terminate text input to the macro ask command).

The Last Word 3.21 Reference Manual

The string captured by the macro ask command is stored in the paste buffer because it allows the resulting text to be easily inserted into the document. We can also place the contents of the paste buffer back into the input line of another command, using the <Ctrl+V> (Paste) command. We then type <Shift+Esc>, <Select+Ctrl+N>. We then type <Shift+Esc>, <Ctrl+V>, followed by .FNT, and finally a <Return>. This simply loads the 80 column font. To load the corresponding 80 column font, we type the same line again, except

So the next character we need in the macro is <Shift+Esc>, <Select+Ctrl+N>. We then type <Shift+Esc>, <Ctrl+V>, followed by .FNT, and finally a <Return>. This simply loads the 80 column font. To load the corresponding 80 column font, we type the same line again, except

A screenshot of the completed macro is shown below.



you press <Option+A> and type . for example . T. The macro will simply terminate).

We can refine this macro further by hiding screen updates. By placing the <Select+Ctrl+Z> set options command in the macro, followed by H immediately before the first <Select+Ctrl+F> Load Font command, we can turn off the screen at that point. We could place <Select+Ctrl+Z> followed by V to turn it on at the end of the macro, re-enabled when a macro ends or is prematurely terminated by an error or the <Esc> or <Break> keys.

The revised macro looks like this:



In operation, the macro is now indistinguishable from a built-in command.

TRANSPOSE CHARACTERS

The Last Word 3.21 Reference Manual

LW doesn't have a command to transpose mistyped characters, but we can create this command using a macro.

Note: this macro is included in the LW.MAC macro supplied on the distribution disk, along with macros to transpose words and paragraphs.

We'll write the transpose adjacent characters macro first.

We'll put this macro on <ESCAPE> <Ctrl+T> for transpose. To make entry of this macro easier, first type <Ctrl+CAPS> to go into "control" mode. This feature disables LW's commands, enabling you to type control keys without preceding them with <Ctrl+ESCAPE>. If you need to make corrections, type <Ctrl+CAPS> again to turn off control mode.

Now, with an empty editor as before, type <Ctrl+T>.

Now type <SELECT+EQUALS SIGN>.

Type <Ctrl+M>. This means we are about to mark a block.

Now type <Ctrl+RIGHT ARROW>. When the macro runs, this will define the character under the cursor as a marked block. Next, type <Ctrl+C>. This is the Cut Marked Text command, and will send that character to the paste buffer.

Now type <Ctrl+RIGHT ARROW> again to move the cursor over the next character. Next, type <Ctrl+P>. This will paste the original character to the right of the character that followed it. Finally, type <Ctrl+LEFT ARROW> to move the cursor back to its original position.

We'll finish with a message for a neat effect.

Type <SELECT+Ctrl+V> (the "print message" command) then type "Characters Transposed", and end with <RETURN>.

You'll now need to take the editor out of control mode by typing <Ctrl+CAPS>. Save the macro, then load it as described previously. Now when you press <ESCAPE>, then <Ctrl+T>, then character under the cursor will swapped with the one to its right, and a message to that effect will be displayed.

8.7 MACRO SUMMARY

As you can see, the scope of the macro language is governed only by your imagination. If you think of something LW doesn't do, chances are it's possible to construct the feature you want using a macro. And LW is fast enough to make your macros execute seamlessly, as if they were built-in features of the program. You can have your address or other frequently used text passages attached to a macro, or have a macro which merges in sections of text from disk at the cursor position. Check the supplied macro files to get an idea of the diverse applications of macros.

9 CONFIGURING LW

You can configure LW so that it always loads with the settings you prefer. You can even load different configurations part way through an editing session. You can set up everything from screen colour to additional banks of RAM for text.

LW supports two kinds of configuration files. LW.SYS is loaded when the program first starts and contains information about the memory configuration, keyboard buffer and keyboard redefinition. LW.SYS is read once when the program starts and any settings it contains cannot be changed once the program is loaded.

9.1 CONFIGURATION OPTIONS IN THE EDITOR

The following commands toggle or set up various LW features, and these settings are all saved in the configuration file. They can all be set up from the configuration program, with the exception of the tab ruler. Not all the options in the configuration program can be altered from the editor during an editing session: only those options which are likely to need changing once the program is up and running are available.

<Ctrl+W>	Toggle word-wrap
<Shift+Ctrl+W>	Set screen resolution (40 or 80 column) and the maximum line length (5-240 characters)
<Ctrl+TAB>	Clear tab stop at current column
<Shift+TAB>	Set tab stop at current column
<Shift+Ctrl+E>	Erase all tab stops
<Shift+Ctrl+TAB>	Reset default tab stops
<Shift+Ctrl+INS>	Toggle Insert/Over-type modes
<CAPS>	Toggle Upper/Lower case
<Shift+Ctrl+U>	User options

Several settings from the disk menu are also saved in the config file:

<S>pec	Set the directory filename mask
<1-0>	Set the current drive #
<Tab>	Directory style

Use the following commands to load and save different configurations during an editing session:

<Ctrl+O>	Load config
<Shift+Ctrl+O>	Save config

Unless you supply your own filename extender, ".CFG" will be appended during both save and load.

One other command from the editor is:

<Shift+Ctrl+N>	Install/Load alternative character set
----------------	--

The character set information isn't saved in the config file. To make a character set of your choice load a run-time, rename it "LW.FNT" and put it on your LW disk.

The Last Word 3.21 Reference Manual

9.2 .CFG CONFIGURATION FILES

CFG files contain user preferences such as screen colours, screen width and resolution, default drive number, filespecs, etc. Many of these settings can be changed via commands in the editor, and the current configuration can be saved as a CFG file at any time with the <Shift+Ctrl+O> command.

The configuration file is always used at startup. This way, you can keep all your most commonly used settings in LW.CFG, and load different configurations during an editing session with <Ctrl+O>.

CFG files are plain text files consisting of lines ending in <Return>. Each line takes the form of a keyword, followed by a space, and then one or more numeric or textual arguments separated by commas or spaces.

The full list of CFG file keywords is shown below:

Numeric Settings	Argument	Comments	Default
TMARGIN	0-255	Set default top printed margin	5
BMARGIN	0-255	Default bottom margin	61
LMARGIN	0-255	Set left margin	10
RMARGIN	0-255	Set right margin	70
PAGELEN	0-255	Set page length	66
HFLEFTMARG	0-255	Set left header/footer margin	10
HFRIGHTMARG	0-255	Set right header/footer margin	70
SPACING	0-255	Set line spacing	1
HEADOFF	0-255	Set header offset	2
FOOTOFF	0-255	Set footer offset	2
EOLCHAR	0 or 219	Set end-of-line character (internal code)	219
PADCHAR	0 or 125	Set false space character (internal code)	0
TEXTCOL	0-255	Set text luminance	10
SCREENCOL	0-255	Set editor screen colour	148
PROMPTCOL	0-255	Set message line text luminance	10
BORDERCOL	0-255	Set border colour	0
BARCOL	0-255	Set progress bar colour	184
KEYDELAY	0-255	Set initial key delay	30
KEYREPEAT	0-255	Set key repeat rate	3
TABWIDTH	0-255	Set default tab width	5
FILESORT	0-4	Set file sort type: 0 = no sort 1 = sort by name 2 = sort by extender 3 = sort by date/time 4 = sort by size	0
PAGEWIDTH	5-240	Set editor column width	

The Last Word 3.21 Reference Manual

Flags (ON/OFF)	Argument	Comments	Default
80COLUMNS	ON OFF	Set 80 / 40 column mode	ON
PAGEWAIT	ON OFF	Set page wait mode during printing	OFF
WORDWRAP	ON OFF	Word wrap on/off	ON
INSERT	ON OFF	Insert mode on/off	ON
CASESENS	ON OFF	Case sensitivity (search and replace)	OFF
CAPSLOCK	ON OFF	Caps lock on/off	OFF
KEYCLICK	ON OFF	Set key click noise on/off	ON
SIONOISE	ON OFF	Set I/O noise on/off	ON
WILDCARDS	ON OFF	Set wildcards in search/replace on/off	ON
ATTRACT	ON OFF	Enable/Disable OS colour cycling	ON
DOCMODE	ON OFF	Set document/text mode	OFF
SDXDIR	ON OFF	Set long directory style (SDX only)	OFF
Text Settings	Argument	Comments	Default
FILEEXT	<EXT>	Set default text mode extender. Should be entered WITHOUT leading period.	TXT
DRIVE	Dn:	Set default drive ID	D:
FILESPEC	<filemask.ext>	Set default file mask for disk menu	*.*

A good way to understand CFG files is to load the LW.CFG file into the editor. You might also save the current configuration with <Shift+Ctrl+O>, giving it the name TEST.CFG to see how any settings you have altered in the editor are reflected in the CFG file. Ther^ q • Á } [c @ã } * Á c [Á • c [] Á ^ [~ Á | [æå ã } * Á æÁ Ô Ø Õ Á ~ manually, although you should take care not to introduce syntax errors into the file. Some settings . like the default print margins . require you to edit the CFG file by hand, since there is no command in LW to changed them.

Note: Prior to version 3.0, LW configuration files were binary files and were edited using the supplied & [] ~ ã * ~ ! æc ã [] Á ^ á ã c [! È Á Þ [, È Á @[, ^ ç ^ ! È Á Š files and the configuration editor is no longer required. Note also that configuration (.CFG) files from earlier versions of LW are totally incompatible with version 3.0. P [, ^ ç ^ ! È Á & [] ~ ã * ~ ! æc ã [] Á ~ ã | ^ • Á ~ ! those of version Á æ! ^ Á %~] 3.1 and 3.2.

9.2.1 THE DEFAULT DRIVE

The DRIVE instruction in a configuration file sets the default drive in LW. This is the device ID which is added to any filenames you type without Dn: at the front. It is also used as the drive number when cataloguing files using the disk menu.

DRIVE D1:

The above line simply sets the default drive to drive 1.

The Last Word 3.21 Reference Manual

9.3 THE LW.SYS FILE

The LW.SYS file is read once when LW first starts up. The information in the file is used to set~] Á Š Y q • Á { ^ { [| ^ Á & [} ~ ā * ~ | æ c ā [} Á æ} á Á [c @^ | Á • ^ c once the program has finished setting itself up. LW.SYS must be written as a plain text file in the editor. Most of the time, the memory configuration commands in LW.SYS will be unnecessary, since the settings it affects are usually automatically configured. However, the ^ Á { æ ^ Á à ^ Á [& & æ • ā [} • Á , @^ } Á ā c q • Á á ^ • ā | æ à | ^ Á c [Á Š Y È Ù Ÿ Ù Á ā • Á æ | • [Á } ^ & ^ • • æ | ^ Á ā ~ Á ^ [~ in key bar @ Á c [Á { æ } ~ æ | ^ buffer or remap the keyboard, or define the LW search path on non-SpartaDOS X systems.

LW.SYS may contain the following instructions:

Instruction	Arguments	Comments	Default
BANKED	ON!OFF	Turn banked memory usage on or off	ON (with supported DOSes)
BANKS	} È } È } È } ò	Specify banks to be used Ç ~ [{ Á Š Y q • Á ā }	Depends on available memory
RESERVE	n	Reserve # banks for extensions	0
EXTPAGES	n	Reserve # pages for extension code	0
BUFFER	ON!OFF	Turn internal keyboard buffer on or off	ON (unless SDX buffer installed)
PROGBAR	ON!OFF	Turn the file I/O progress bar on or off	ON
PATH	Dn:<path>	Set LW search path	none
KEY	code, ATASCII	0Ec c æ & @AT&S&C&I&+ Á & % \ ^ ^ Á & [á ^ (á % á c d á	none
COMMAND	Command #, ATASCII code	Assign keycode to internal command (advanced)	none
LOADCFG	[Dn:][path]filename[.ext]	Specify config file to load when LW first starts	LW.CFG
LOADPDR	[Dn:][path]filename[.ext]	Specify printer driver to load when LW first starts	LW.PDR
LOADEXT	[Dn:][path]filename[.ext]	Specify extension to load when LW first starts	LW.EXT
LOADMAC	[Dn:][path]filename[.ext]	Specify macro file to load when LW first starts	LW.MAC
LOADFNT	[Dn:][path]filename[.ext]	Specify 40 column font to load when LW first starts	LW.FNT
LOADF80	[Dn:][path]filename[.ext]	Specify 80 column font to load when LW first starts	LW.F80
DOCEXT	<ext>	Define the default file extension for files saved in Document mode.	LWD

The Last Word 3.21 Reference Manual

9.3.1 CONFIGURATION PROFILES USING LW.SYS

As of LW 3.2, the facility to specify the default files other than LW.* are loaded at start-up means that different LW.SYS files can be used to select different configurations when LW is first run. Using SpartaDOS X, this feature becomes even more powerful, since the location and name of LW.SYS can be specified on the command line with the /S switch. See the section on SpartaDOS X for more information.

9.3.2 CONFIGURATION USING A SUPPORTED DOS

When using LW with a supported DOS (DOS 2.5, MyDOS, and SpartaDOS X), the program will automatically detect any extended memory on the machine and avoid those banks used by DOS/RAMdisks (providing the default RAMdisk drivers are used with DOS 2.5 and MyDOS). While manual configuration of the memory banking

• & @^ { ^ Á • @[~ | á Á à ^ Á ~ } } ^ & • • æ! ^ Á ~ } á ^ | Á c @^ • ^ Á & ã | & override the default settings.

Using a supported DOS, LW will scan the hardware to establish how many extended memory banks are installed in the system, then subtract from the resulting list those banks used by DOS (the only exception is SpartaDOS X, which helpfully provides its own built-in list of unused banks). The result is a list of the *free* banks on the system, rather than a list of all the banks installed on the machine. You may specify a selection of banks from this list as follows:

BANKS 1,2,3,4

Under DOS 2.5, MyDOS or SDX, this line will tell LW to use the first four banks from the list of *free* banks the program generated when it first initialized (the banks are ordered numerically according to the PORTB banking value, and are numbered beginning at 1D È Á Q ~ Á c @^ | ^ Á æ! ^ } q c Á ^ } [~ * @Á ~ ! ^ ^ Á à æ} \ • Á c [% Ò Æ P S Û + È Á æ • Á { æ } ^ Á æ Á à æ} \ ^ á Á Ü Æ T È Á Š Y q • Á { æ & ! [Á à ~ ~ ^ | Á assuming there were at least four free banks on the target machine . LW would allocate one bank for the macro/paste/directory buffer (which it always does when using extended memory, before allocating any other banks as text buffers), three banks for extra text buffers, and it would enlarge the main text buffer to 19K.

Note K Á Y @^ } Á ~ • ã } * Á à æ} \ ^ á Á Ü Æ T È Á Š Y q • Á { æ & ! [Á à ~ ~ ^ | Á 6.5K, and the disk directory buffer provides room for a maximum of 255 files.

The RESERVE instruction is intended to allocate banks of extended memory for use à ^ Á { æ & @ã } ^ Á & [á ^ Á ^ ç c ^ } • ã [] • È Á Q c q • Á ^ } ç ã • æ * ^ á Á c @ proof-readers and table of contents generators will be written as extensions for LW. The extensions load at \$3300 and must first have space allocated for them with the EXTPAGES statement, which merely allocates the specified number of 256 byte memory blocks (to a maximum of 12) above \$3300 to house the executable code of extension programs (this memory is taken from the 19K main text buffer: extensions are only available when LW is using BANKED memory).

EXTPAGES 4
RESERVE 1

The Last Word 3.21 Reference Manual

The above statements in LW.SYS will reserve four pages of RAM from main memory at \$3300 for extension code, and a single bank of extended RAM for use by extensions which require it. If there is insufficient extended memory, or if LW is not using banked memory, these instructions will have no effect.

Another example:

```
BANKED ON  
BANKS 1,2,3,4,5,6  
RESERVE 1  
EXTPAGES 8
```

On a 320K machine running DOS 2.5 with a 64K RAMdisk installed, the LW.SYS file above will configure LW with four extended text buffers, and one bank for the macro/paste/directory buffers, reserving a single bank (actually the last in the list) for use by extensions. Since LW is using banked memory, the main text buffer would be 19K. However, the EXTPAGES statement has reserved 8 pages (2K) at \$3300 for use by extension executable code, reducing the size of the main text buffer to 17K.

You can disable banked memory use altogether by including the following line in LW.SYS:

```
BANKED OFF
```

This will cause LW to ignore any BANK, EXTPAGES or RESERVED statements in LW.SYS. The main (and only) text buffer will be 16K in size and the paste, macro and directory buffers will each be only 1K long. This is the same configuration LW will adopt when run on a standard 64K XL/XE machine.

9.3.3 CONFIGURATION USING OTHER DOS PACKAGES

When LW is loaded on a system using an unsupported DOS (anything other than SpartaDOS X and DOS 2.5 or MyDOS with the standard RAMdisk drivers), it makes no assumptions about whether the operating system is using extended memory for RAMdisks or other purposes. Therefore the default behaviour of the program under these circumstances is NOT to use banked memory. To use banked memory when running LW under an unsupported DOS, you MUST create a custom LW.SYS file containing the following line:

```
BANKED ON
```

```
  ù ã } & ^ Á Š Y Á & æ} q c Á á ã ~ ~ ^ | ^ } c ã æc ^ Á à ^ c , ^ ^ } Á ˇ • ^ á Á æ} á Á ~ |  
unsupported DOS, the internal list it builds when it first starts up is of ALL the extended  
memory banks present on the system. Therefore, if you wish to configure LW to use  
only specific banks of extended memory while co-existing with any installed RAMdisks  
or other parts of DOS residing in extended RAM, you must first know which banks of  
RAM are used by the operating system and specify only UNUSED banks after the  
%oÓ œP S ù + Á • c æc ^ { ^ } c È
```

Examples:

```
BANKED ON  
BANKS 1,2,3,4
```

The Last Word 3.21 Reference Manual

The above LW.SYS file used with an unsupported DOS on a 128K machine will cause LW to allocate three extended text banks, a 19K main text bank, and a single extended bank for its macro/paste/directory buffer.

**BANKED ON
BANKS 5,6,7,8,9,10,11,12
RESERVE 1**

In the above example, assuming LW is running on a 320K machine under an unsupported DOS which is using the lowest four banks of extended memory as a RAMdisk, LW will allocate one bank for its macro/paste/directory buffer, six banks for extended text buffers, and reserve a single bank for use by extensions. It will do this while *avoiding* the lowest four banks of extended RAM (those in use by the operating

Clearly when using unsupported RAMdisks with LW, careful planning is required.

However, when using an unsupported DOS with no RAMdisks or other components

needs (up to a maximum of 16), yielding up c [Á c ^ } Á c ^ ¢ c Á à æ } \ • É Á Q ~ Á c @ ^ | ~ fitted to the system, LW will simply use as many as it can find.

9.3.4 THE SEARCH PATH

The s^ æ! & @Á] æc @Á æ | | [, • Á ^ [~ Á c [Á æ & & ^ • • Á Š Y q • Á & [} ~ ã * from specified drives/paths on the system without typing the pathnames every time you load the files.

PATH D8::D1:>LW

This line in a SYS file will cause LW to first search the current directory of drive 8 ~ [| | [, ^ á Á à ^ Á c @ ^ Á ~ [| á ^ ; Á % Š Y + Á [} parallel drives and Á , @ ^ } Á | config files, when no other path has been supplied. Note that if the file is not found by searching the specified paths, the default drive will ALWAYS be searched last. Note æ | • [Á c @ æ c Á Š Y q • Á á ^ ~ æ ~ | c Á á ! ã ç ^ Á ã • Á } [c Á } ^ y & ^ • • æ ! ã % Ö K + D É Á Š Y q • Á á ^ ~ æ ~ | c Á á ! ã ç ^ Á •] ^ & ã ~ ã ^ ! Á ã • Á c @ ^ Á • æ prepended to all filenames for which no device identifier has been explicitly provided. If for any reason you want to ensure that the default DOS drive is searched, inc | ~ á ^ Á % Ö K + Á (without quotes) as an entry in the path.

9.3.5 THE KEYBOARD BUFFER

V @ ^ Á ~ ã ~ c @ Á \ ã } á Á [~ Á ã } • c ! ~ & c ã [} Á ã } Á Š Y È Ù ÿ Ù Á ã • Á c @ (OFF). This simply turns the keyboard buffer on or off. The keyboard buffer will always default to on, unless it detects that another keyboard buffer (such as the SpartaDOS X keyboard buffer) is active.

9.4 USING MULTIPLE TEXT BUFFERS

Multiple buffers can allow the loading of up to 160K of text in memory at any one time. The files are always kept separate (unlike AtariWriter Plus), but may be linked together when printed by using include bank commands. You can also append text banks when saving using the /A switch on the command line. By having "include bank #"

The Last Word 3.21 Reference Manual

commands in the main bank (bank #1), you can keep track of pagination with the <Ctrl+?> command, or preview the whole document with <Ctrl+V> without once having to access a file.

From the main program, banks are selected with:

<Shift+Ctrl+n> Select bank command

where <n> is any one of the number keys. <1> always calls up the main (unexpanded) bank, while the other 9 numbers can be set up any way you wish. From macros, these same numbers follow the <SELECT+Ctrl+Z> Settings command.

9.5 CUSTOM FONTS

Several alternative character sets are supplied on the disk. Many fonts are supplied in extenders, while 40 column fonts are in standard Atari format and have the extender %o. Depending on whether the editor is in 40 or 80 column mode, leaving the font extender off the filename will loading of 40 or 80 column versions of a font by specifying the extender on the input line. The MACRO.FNT/MACRO.F80 font is most suited to editing macros, since the control characters are specially emboldened and easy to distinguish from alphanumeric characters in the 80 column version. The other fonts provide many different styles and weights, and all provide full international characters.

9.6 CUSTOMISING THE KEYBOARD

LW allows you to customise the keyboard in two ways: by using macros, and by using a custom keyboard layout (in the LW.SYS file). Redefining the keyboard using the macro is the best way to reassign keystrokes, while a keyboard definition file allows you to totally remap the keyboard (to create a DVORAK layout, for example).

9.6.1 THE KEYBOARD TABLE

KEY statements, which should appear in the LW.SYS file.

KEY n,n

The KEY statement takes two numeric arguments. The first number is an index to the hardware scan code table. The scan code table is 256 bytes long, and is divided into four groups of 64 bytes. The first 64 bytes represent normal keys (without Shift or Control), the next 64 bytes are the Shifted characters, and the next 64 bytes are Control characters. The final 64 bytes represent keys pressed while both Shift and Control are held down together (<Shift+Ctrl> keys in this manual). LW

The table on the following page shows the entire default key mapping of LW. The keys are shown in the left column, and the grey columns represent the code offsets for the normal, shifted, control and Shift+Ctrl characters. When writing complex macros, this table is a useful means of finding out what character to use in a macro to . for example . feed <Ctrl+6> to the editor.

The Last Word 3.21 Reference Manual

Key		Normal		Shift		Ctrl		Shift+Ctrl
i	0	108	64	76	128	12	192	200
j	1	106	65	74	129	10	193	200
;	2	59	66	58	130	123	194	200
F1	3	28	67	8	131	19	195	200
F2	4	29	68	5	132	12	196	200
k	5	107	69	75	133	11	197	200
+	6	43	70	92	134	30	198	200
*	7	42	71	94	135	31	199	200
o	8	111	72	79	136	15	200	143
Invalid	9	200	73	200	137	200	201	200
p	10	112	74	80	138	16	202	144
u	11	117	75	85	139	21	203	149
Return	12	155	76	155	140	155	204	155
i	13	105	77	73	141	9	205	137
-	14	45	78	95	142	28	206	223
=	15	61	79	124	143	29	207	252
v	16	118	80	86	144	22	208	200
Help	17	200	81	200	145	200	209	200
c	18	99	82	67	146	3	210	200
F3	19	30	83	1	147	4	211	200
F4	20	31	84	26	148	22	212	200
b	21	98	85	66	149	2	213	200
x	22	120	86	88	150	24	214	200
z	23	122	87	90	151	26	215	200
4	24	52	88	36	152	180	216	164
Invalid	25	200	89	200	153	200	217	200
3	26	51	90	35	154	5	218	163
6	27	54	91	38	155	182	219	166
Esc	28	27	92	27	156	27	220	200
5	29	53	93	37	157	181	221	165
2	30	50	94	34	158	253	222	162
1	31	49	95	33	159	200	223	161
,	32	44	96	91	160	0	224	128
Space	33	32	97	32	161	32	225	160
.	34	46	98	93	162	96	226	224
n	35	110	99	78	163	14	227	142
Invalid	36	200	100	200	164	200	228	200
m	37	109	101	77	165	13	229	141
/	38	47	102	63	166	224	230	221
Inverse	39	129	103	129	167	129	231	129
r	40	114	104	82	168	18	232	146
Invalid	41	200	105	200	169	200	233	200
e	42	101	106	69	170	5	234	133
y	43	121	107	89	171	25	235	153
Tab	44	127	108	159	172	158	236	220
t	45	116	109	84	173	20	237	148
w	46	119	110	87	174	23	238	151
q	47	113	111	81	175	17	239	145
9	48	57	112	40	176	185	240	168
Invalid	49	200	113	200	177	200	241	200
0	50	48	114	41	178	176	242	169
7	51	55	115	39	179	183	243	167
Backspace	52	126	116	156	180	254	244	222
8	53	56	117	64	181	184	245	192
<	54	60	118	125	182	125	246	219
>	55	62	119	157	183	255	247	222
f	56	102	120	70	184	6	248	134
h	57	104	121	72	185	8	249	136
d	58	100	122	68	186	4	250	132
Invalid	59	200	123	200	187	200	251	200
Caps	60	130	124	131	188	132	252	132
g	61	103	125	71	189	7	253	135
s	62	115	126	83	190	19	254	147
a	63	97	127	65	191	1	255	129

The Last Word 3.21 Reference Manual

By working out where in the table the key combination you want to redefine resides, you can totally remap the keyboard. For example, you could have the following line in LW.SYS:

KEY 10,97

This `^ [~ Á | ^ á ^ ~ ã } ^ Á c @ ^ Á Ł Ú N Á \ ^ ^ Á • [Á c @ æ c Á , @ ^ } Á ^ [~ Á] ; Á] ; [á ~ & ^ á È Á Þ [c ^ Á c @ æ c Á c @ ã • Á { ^ æ } • Á c @ æ c Á] ; ^ • • ã } * Á Ł Ú N everywhere in the program. The most useful application for this would be to produce things like DVORAK keyboard layouts.`

Something to consider when redefining the keyboard in this way is that for every key `^ [~ Á | ^ á ^ ~ ã } ^ È Á ^ [~ Á } ^ ^ á Á c [Á | ^ á ^ ~ ã } ^ Á ã c • Á % & [{] | ^ { ^ Á } * Á c @ ^ Á • æ { ^ Á & [á ^ Á Ç æ • Á , accepts any more) } ^ Á & [á ^ Á ^ [~ Á & æ } Þ [c ^ Á c @ æ c Á c @ ^ Á F G € € Ý Š q • Á ~ ~ } & LW, and you can assign the commands of your choice to the twelve possible key combinations.`

9.6.2 REMAPPING COMMANDS USING MACROS

The second way of redefining the keyboard takes advantage of `LWq • Á { æ & ! [Á & æ] æ à ã | ã c` and has the advantage that each key will still produce the expected character. Using `{ æ & ! [• È Á @ [, ^ ç ^ ! È Á ã c q • Á] [• • ã à | ^ Á c [Á c ! æ } • | æ c ^ Á æ Á & [` before it gets processed by the program.

Looking again at our Windows cut and paste shortcuts, consider the following macro `Ç , ^ q ç ^ Á æ | ; ^ æ á ^ Á | [æ á ^ á Á c @ ^ Á T Ç È Õ Ü U È Ø Þ V : Á & @ æ ! æ & c ^ ! Á • ^`



There are five macros in this file. The first is mapped to <Ctrl+X>, and simply issues <Ctrl+C> whenever <Ctrl+X> is pressed in the editor. The second issues <Ctrl+P> when <Ctrl+V> is pressed and so on. The last two macros assign keys to the two functions whose original keystrokes have now been reassigned.

9.6.3 REMAPPING COMMANDS USING THE VECTOR TABLE

With version 3.21 of LW comes the facility to remap the keyboard by changing the `ã } c ^ ! } æ | Á & [{ { æ } á Á c æ à | ^ È Á V @ ã • Á ã • Á æ & & [{] | ã • @ ^ á Á ç ã a` LW.SYS. The command statement takes the form:

COMMAND <command number>,<key code>

The Last Word 3.21 Reference Manual

the numbered internal command in the range 0 to 100, with that command.

10 1 1 0 0 4 e 127.p88 6-16r15 .0ne Te2577411.33 e 127.p88 44 e35]TJ E 0 2 02.a356047. 5 0 0

Command Number	Function	Default Key Code
0	Cursor up	28
1	Cursor down	29
2	Cursor left	30
3	Cursor right	31
4	Page up	223
5	Page down	252
6	Macro toggles	154
7	Cut selected text	24
8	Set tab stop	159
9	Clear tab stop	158
10	Change found string	18

