Commodore® AMIGA®

ENHANCER SOFTWARE

FEATURING

AMIGADOS VERSION 1.3

INCLUDING

KICKSTART™ 1.3
WORKBENCH™ 1.3
EXTRAS 1.3

Disk-Based update contains FastFileSystem, improved printing, additional fonts and more capabilities For Amiga 500, 1000, and 2000 Computers.

AMIGADOS ENHANCER SOFTWARE*

Featuring Workbench™ Version 1.3
and including Extras Version 1.3
and Kickstart™ Version 1.3 (for Amiga 1000 owners)

*The Software is compatible with Version 1.2 Kickstart ROM's.

TRADEMARKS

ACKNOWLEDGEMENTS

AmigaDOS Enhancer Software

This Enhancer package provides the necessary software to upgrade your Amiga computer to the new 1.3 operating system. Included in this package are:

- Kickstart Version 1.3 diskette (for Amiga 1000 owners)
- Workbench Version 1.3 diskette
- Extras Version 1.3 diskette
- Program License Agreement

By upgrading to Version 1.3, you'll maximize the power of your Amiga computer. Version 1.3 improves upon the existing technology, expands the Amiga's capabilities, and enhances many software applications.

To make full use of Version 1.3 Software, you need to use the new Version 1.3 Kickstart. If you are an Amiga 1000 owner, this is extremely simple: just use the new 1.3 Kickstart diskette instead of your old 1.2 Kickstart disk when you boot the system.

> YOU CANNOT USE THE KICKSTART DISK WITH THE AMIGA 500 OR AMIGA 2000. Instead you may wish to replace the Kickstart ROM in your computer with the 1.3 Kickstart ROM. The 1.3 Kickstart ROM is available from an authorized Commodore Service Center. The ROM will be installed at the service center. Please request part number 314864-01.

You can use the new software without the 1.3 Kickstart ROM in place. The only application that will not be available is the reboot function of the Recoverable ramdrive.device, explained later in this manual. Also, Amiga 2000 owners who have an A2090A Hard Disk Controller Card will not be able to autoboot off their hard disk until the 1.3 Kickstart ROM is installed in their machine.

CONTENTS

ABOUT THIS MANUAL

Welcome to the new Version 1.3 AmigaDOS Enhancer Software. This manual, which assumes a familiarity with the information in the *Introduction to the Commodore-Amiga* manuals, covers both Workbench Version 1.3 (Chapters 1 through 5) and Extras Version 1.3 (Chapters 6 and 7).

The first several chapters cover all the changes made to the basic Workbench diskette. This new, updated version of the Workbench diskette improves upon many of the features included in Version 1.2. Several new programs have also been added to enhance your computing pleasure:

> SHELL — acts like a CLI but permits editing of lines and lets you backtrack through previous commands
> FixFonts — lets you update the Workbench's font directory after adding new fonts
> MORE — allows you to display ASCII text files
> CMD — redirects serial or parallel output to a file

There are several new AmigaDOS commands, and some of the original commands have been expanded upon and revised. Also, a new graphic select screen which supports extended printer graphics features has been added to Preferences.

All of this new material is covered in this manual. First, it takes you through drawer by drawer to explain any changes to the different programs. Later, for users familiar with AmigaDOS, it explores the new commands and handlers, and gives you instructions on modifying your startup sequences.

The later part of the book documents the Extras disk. The Extras window displays icons for AmigaBasic, BasicDemos, FD1.3, Tools and PCUtilities. However, the disk also includes directories which contain printer drivers, keymaps and fonts that did not fit onto the Workbench disk.

The utilities in the Tools and PCUtilities drawers are covered in chapters six and seven. Since AmigaBasic, BasicDemos, and FD1.3 pertain to the Amiga Basic programming language, which is an adaptation of Microsoft® BASIC developed especially for the Amiga, they are not covered in this manual. Full information on Amiga Basic is available in the *Amiga Basic* manual that was packaged with your computer.

1. THE WORKBENCH

The SHELL, NEWCON:, and The CLI

One of the most obvious differences between Workbench 1.2 and Workbench 1.3 is the addition of a SHELL program in the Workbench window. (Note that the Demos drawer has been removed.) Both the SHELL and the CLI (the Command Line Interface, located in the System drawer) allow you to communicate with your Amiga computer through AmigaDOS commands. Any of the programs on the Workbench disk can be run from either the SHELL or the CLI. The main difference between the two is that the SHELL provides an *upgraded* CLI environment — meaning that the SHELL can do everything that the CLI can do, and more.

Since the SHELL and the CLI share many similarities, the functions that pertain to both are explained first. Then, the additional capabilities of the SHELL are explored.

Startup Files

When the SHELL or CLI is opened, by double-clicking on its icon, the utility executes the commands in its startup file. For the SHELL that file is s:Shell-Startup, and for the CLI it is s:CLI-Startup. Both startup files are located in the Workbench's S: directory.

A common command in both startup files is the PROMPT command. The PROMPT command specifies the prompt that is present when the SHELL or CLI window is opened. When the CLI is opened, it displays a simple numerical prompt — 1>. However, the SHELL opens with a ''1.SYS:>'' prompt which will change to reflect changes to the current directory. For instance, if you were to type:

```
cd utilities
```

The prompt would change to:

```
1.Workbench 1.3:utilities>
```

You can change the prompts for the SHELL by using the PROMPT command:

PROMPT %N>	displays only the CLI number
PROMPT %N.%S>	displays the CLI number, a period, then the current directory
PROMPT %S.%N>	displays the current directory, a period, then the CLI number

If you want a different prompt to appear when you open the window, edit the s:Shell-Startup file. To do this, use the AmigaDOS ED command, the MEMACS utility, or any other text editor.

TOOL TYPES

When you select an icon and choose INFO from the Workbench menu, a window appears displaying more detailed information about the utility represented by the icon. You can control different variables which affect the SHELL and the CLI by using the Workbench menu's INFO item to change the utility's TOOL TYPES. A utility's TOOL TYPES is a group of characteristics that govern a utility's behavior. When you open the INFO window, you can change the default TOOL TYPES by using the add/delete gadgets in the TOOL TYPES box.

You can add the following TOOL TYPES to both the SHELL and the CLI icons.

WINDOW = window specification

This TOOL TYPES allows you to control the placement, size, and handler that the new SHELL or CLI will use. A full window specification must be used, for instance:

WINDOW = CON:0/10/640/100/NEWCLI

WINDOW = NEWCON:0/1/640/100/AmigaShell

The format for a window specification is "Handler:x/y/w/h/name", where:

Handler = CON: or NEWCON:
x = <# of pixels from left edge of screen to left edge of window>/
y = <# of pixels from top of screen to top of window>/
w = <width of window, in pixels>/
h = <height of window, in pixels>/
name = <title for window>

In the first example above, a CON: window would open flush with the left edge of the screen and ten pixels from the top of the screen. The window would be 640 pixels wide and 100 pixels high. NEWCLI would appear in the window's title bar.

The handlers you can choose from are CON: or NEWCON:. CON: is the standard Amiga window, while NEWCON: is a new alternative window. NEWCON: allows command line editing and has a history buffer. The SHELL is an example of a NEWCON: window. (NEWCON: is explained in Chapter 3, Other Workbench Directories.)

STACK =

This TOOL TYPES allows you to specify the initial stack size used by the CLI or SHELL. The stack size is an amount of memory reserved for a specific tool. (Default for most utilities is 4000 bytes.)

Multiple Icons

Multiple icons for both of these utilities can be created by copying any *project* TYPE icon, such as the SHELL, or even the Pointer icon in the Preferences window. (You can tell if an icon is a project icon by the TYPE description in the icon's INFO window.) Set the default tool in the INFO

window to SYS:System/CLI, and set the WINDOW= and STACK= TOOL TYPES to different values. This gives you an easy way to bring up windows in different locations.

For instance, if you wanted an additional SHELL, called Shell2, in the Utilities drawer, you would:

1) Use the COPY command to make a copy of the Shell icon*

   `COPY Sys:Shell.info to Sys:Utilities/Shell2.info`
2) Open the Utilities window
3) Check the TOOL TYPES of the Shell2 icon to make sure that the default tool is set to SYS:System/CLI
4) Change the WINDOW= and STACK= TOOL TYPES, if desired.

*Or, you could copy the icon using the Workbench menu's Duplicate item.

Additional Features of the SHELL

One of the advantages of using the SHELL is that it saves a lot of typing. With the SHELL, you can edit a command line. When using the CLI, if you make a typing error when entering a command, you have to use the backspace key and delete all the other characters until you reach the error. With the SHELL, you can use the left cursor key to move **over** the characters, correct the error, and hit RETURN. There is no need to re-type the whole command.

The SHELL has a ''history.'' This means that you can move through previous commands by using the up cursor key. Instead of re-typing the same line you executed three or four commands ago, you can recall each previous command by pressing the up cursor key. You can also search for specific commands by typing a partial command line, then pressing Shift and the up cursor key (or Control-R). For instance, type DIR and press Shift-up cursor key, and you will be returned to the last command to perform a DIR of any directory.

Once you've moved back a few commands, you can move forward with the down cursor key. This lets you retrace what you've already done, in case you've forgotten how you achieved a specific result or action. Pressing Shift and the down cursor key (or Control-B) will bring you to the bottom of the history buffer and leave you on a blank command line.

Some other ways to edit the command line are:

The Delete key	Deletes the characters at the cursor
The Backspace key	Deletes the character to the left of the cursor
Control-K	Deletes everything from the cursor forward to the end of the line
Control-U	Deletes everything from the cursor backward to the start of the line
Control-X	Deletes the entire line

Shift-Left Cursor key (or Control-A)	Moves cursor to start of line
Shift-Right Cursor key (or Control-Z)	Moves cursor to end of line
Control-W	Moves cursor to the next tab stop.

With the SHELL, you can create aliases for AmigaDOS commands. Think of an alias as a shortcut for commonly used commands. The format for creating an alias is:

ALIAS <alias> <string>

For instance, if you commonly perform a DIR of df1:, you can create an alias called ''d1'' that would perform the same operation:

```
ALIAS dl "DIR dfl:"
```

When the SHELL encounters an alias, it:

1) replaces the alias with the <string>,
2) adds the rest of the command line you have typed,
3) executes the entire command string.

If you want to be able to substitute a filename or other instruction within an alias, use square brackets ([]) to indicate where the insertion will take place. For example:

```
ALIAS ls "LIST [ ] NODATES"
```

In this case the brackets represent the filename or directory to be speci-fied. For instance:

```
ls dfl:
```

will list the directories and files on the diskette in df1: without listing any dates or times. Typing ''ls c'' will list all the files in the C: directory without any dates or times.

Some special features pertaining to aliases:

- Aliases are specific to a certain SHELL. Aliases created in one SHELL will not work in a newly created SHELL. The s:Shell-Startup file is executed automatically whenever the NEWSHELL command is used. You can place commonly used aliases in the s:Shell-Startup file so that you don't have to re-create them each time you open a SHELL. (See Chapter 5, Changing Your Startup Sequences.)
- ALIAS typed by itself lists the current aliases.
- To remove an alias, type ALIAS <alias> with no string.
- A space is not automatically placed on the end of a string. If you need a space, be sure to place one in the string.

The SHELL also supports the use of the new script protection bit. With Version 1.2, the available protection bits for a file were r (readable), w (writable), e (executable), and d (deleteable). Version 1.3 supports a few

The Workbench

additional protection bits — script, archive and pure. (The archive and pure bits are explained in Chapter 2. See the explanations of the PROTECT and RESIDENT commands.)

Use the PROTECT command to set the script bit on a file. The SHELL will then execute that file as an EXECUTE script file instead of as a command. To speed up the execution of the script file, make the EXECUTE command ''resident'', using the AmigaDOS RESIDENT command. This is usually done in the Workbench's Startup-Sequence. For the most effective use of this function, you may want to do the following:

- Make sure the S: directory is in the Amiga command search path, via the PATH command (you may want to do this in the Startup-Sequence file), and/or
- Copy the entire S: directory or frequently used script files to RAM:, and make sure RAM: is in the command path.

There are some scripts in the S: directory that can be used with the SHELL's ALIAS command to give wildcard capability to commands that don't already have it. The scripts are SPAT, which allows pattern matching for single argument commands, and DPAT, which is used with double argument commands. For example:

```
alias ren "execute s:dpat rename []"
alias pro "execute s:spat protect []"
```

gives wildcard capability to the RENAME and PROTECT commands, respectively. (See *The AmigaDOS Manual* for instructions on using wildcards. Examples of SPAT and DPAT are given in Chapter 3.)

The SHELL supports normal redirection using the < and > symbols, as well as an appended form of redirection using >>. Redirection refers to routing the input for or output from a command to a file. Note that >> does not create a file, but merely appends information to an existing file. You can also include comments in the command line by typing them after a semi-colon (;).

The System Drawer

As in Version 1.2, the System drawer contains the CLI, NoFastMem, DiskCopy, FastMemFirst, Format, SetMap, and InitPrinter. However, two new utilities, FixFonts and MergeMem, have been added, while the Say program has been moved to the Utilities drawer and IconEd to the Extras disk.

NoFastMem

Some very old software programs may not run properly when memory other than CHIP RAM is present in the Amiga system, such as the FAST RAM present in the A2000 or the additional RAM provided by an A501 memory cartridge in the A500. In this case, double-clicking on the NoFastMem icon forces the Amiga to only use resident CHIP RAM. The icon works like a toggle switch. To restore the expansion memory to the system, click on the NoFastMem icon again.

If you are working through the CLI, the expansion memory can be turned on again by sending the NoFastMem program a ''break'', either via the BREAK command or by typing Control-C. Control-C will only work if you didn't start the program with the RUN command.

FastMemFirst

This program ''rearranges'' the Amiga's memory list, affecting when memory at $C00000 will be allocated. This allows programs to use fast memory before $C00000 memory, resulting in faster system operation. Unless specifically requested to use CHIP RAM, the Amiga defaults to using $C00000 RAM, FAST RAM, then CHIP RAM. FastMemFirst changes the order of allocation to FAST RAM, $C00000 RAM, then CHIP RAM.

MergeMem

You may want to use MergeMem if you have additional RAM expansion boards added to your Amiga computer. When you double-click on this icon, MergeMem attempts to merge the MemLists (memory lists) of sequentially configured RAM boards. The RAM boards must have the same attributes as contiguous expansion RAM. Memory from separate RAM boards is usually kept in separate memory pools. MergeMem attempts to merge the separate memory pools into one large pool to allow allocation of larger memory blocks by programs. MergeMem only merges memory of the same attributes.

To run MergeMem from the CLI, simply type MERGEMEM. If it is not possible to merge the memory pools, you will receive a message listing the RAM configurations and stating that no merging is possible.

Format

Double-clicking on this icon results in a requester telling you to select Initialize from the Workbench's Disk menu. If you want to format a disk, select Initialize or use the FORMAT command, if you're familiar with the CLI. The format for this command is:

FORMAT DRIVE <drive> NAME <name> [NOICONS]
[QUICK] [FFS | NOFFS]

The NOICONS option prevents a Trashcan icon from being added to the newly formatted disk.

The QUICK option specifies that FORMAT will only format and create the root block (and track), the boot block (and track), and create the bitmap blocks. This is used when reformatting a previously formatted floppy disk.

The FFS and NOFFS options pertain to hard disks. This option overrides the MountList keywords. The FFS option marks the disk as being used with the FastFileSystem. The NOFFS option marks the disk as being used with the default file system. Since FORMAT checks to see if an alternate file system is specified in the MountList, both options are available. If the FastFileSystem is specified, then the FFS option is automatically used. If another file system is in the MountList, the NOFFS option is used. If your

hard disk is divided into partitions, you must set the DosType keyword in the MountList to the appropriate setting for the file system you are using. (There is a section on the MountList in Chapter 3.)

SetMap

Allows you to select the correct keymap for your keyboard. A keymap tells the computer which character to register for each key on the keyboard. With an A2000, your choice is usa1. The usa1 keymap is for use with the A500 and A2000, but it will also work on the A1000. Usa1 enables the new keys on the keypad.

Additional keymaps are included on the Extras disk (in the devs/keymaps directory) and can be copied over to your Workbench diskette. (See the Keymaps section of Chapter 3.) The usa0 keymap is for compatibility with some old software that made assumptions about Version 1.0 Kickstart. The usa2 keymap is for the Dvorak keyboard. The others are international keymaps for use with various European and Scandanavian keyboards.

FixFonts

Updates the .font files in the FONTS: directory. This program is used after fonts have been added to or deleted from any of the subdirectories in the FONTS: directory. FixFonts corrects all the .font files, making them accurately reflect the current contents of the subdirectories in FONTS:.

InitPrinter

Selecting this icon initializes your printer using the printer settings specified in Preferences. InitPrinter sends escape codes to the printer device which causes the printer device to read the variables from Preferences. This command can be used from either the Workbench or CLI.

The Prefs Drawer

With Version 1.3, Preferences is now in the Prefs drawer. When you open the drawer, a window appears and lets you choose from five icons:

Preferences — opens the main Preferences window

Pointer — opens the Edit Pointer window

Printer — opens the Change Printer window

Serial — opens the Change Serial window

CopyPrefs — copies the system configuration to df0:devs (this is useful for non-autobooting hard drive systems)

Even if you enter the Pointer, Printer or Serial window directly, you are still returned to the main Preferences window when you exit. You must select Save on the Preferences window in order to save any changes you made to another window. After selecting Save, Use, or Cancel, you will be returned to the Workbench screen or CLI window.

Also, you can now specify command line arguments when running Preferences from the CLI. The format is:

PREFERENCES [POINTER|PRINTER|SERIAL]

The option specifies which window to open.

There have been changes made to both the main Preferences window and the Change Printer window. The Change Serial and Edit Pointer windows remain the same as with Version 1.2.

The Main Preferences Window

The only change to the main Preferences window is that the CLI On/Off option has been removed. Also, if your system is equipped with a battery backed-up clock, the date and time are automatically updated whenever you choose the Save gadget on the main Preferences window.

The Change Printer Window

The Change Printer window remains the same, except that the Graphic Select gadget has been replaced by two new gadgets: Graphic 1 and Graphic 2. Also, the only printer driver included on the Workbench disk is Generic. All the other printer drivers are now on the Extras disk. You must use InstallPrinter, in the Utilities drawer, to copy printer drivers to the Workbench's DEVS:printers directory. Please see Chapter 4, Printer Drivers, to determine which printer driver works with your printer.

Graphic 1

The new Graphic 1 gadget is equivalent to the V1.2 Graphic Select gadget and is used to request the first of two printer graphic Preferences windows.

There is a new Print Shade selection: Gray Scale2. This shade is for printing pictures designed using the A2024 Monitor (which supports a maximum of four shades of gray).

Graphic 2

Selected by clicking the Graphic 2 gadget in the Change Printer window, this new Preferences window supports extended printer graphics features. The various gadgets are as follows:

Smoothing [ON|OFF] — attempts to smooth diagonal lines. This option is best suited for use with programs that do graphic dumps of text (like ProWrite, PageSetter, CityDesk, Publisher 1000, NotePad, etc.). This technique incurs an approximate 2:1 speed penalty on printing. Note: Floyd-Steinberg (F-S) dithering cannot be used in conjunction with Smoothing. If F-S is enabled when Smoothing is turned on, the dithering mode will automatically change to Ordered. (Dithering is explained later in this section.) Default is OFF.

Left Offset — horizontally offsets the printed picture. This option effectively sets up a left margin. The offset can be entered in increments of tenths of an inch. The Center option (below) disables Left Offset. Default is 0.0 inches.

Center [ON|OFF] — horizontally centers the printed picture. This option overrides the Left Offset value. Default is OFF.

Density [1 through 7] - selects the graphics print density. The lower the density, the faster the print time (on those printers with multiple densities). Refer to the specifications of the V1.3 Printer Drivers for a description of densities supported by each individual printer (included in Chapter 4). Default is Density 1.

Color Correct R/G/B — tries to better match the screen colors to the colors produced by the printer. This option selectively tries to match all shades of either red (R), green (G), or blue (B) from the screen to the printer. Without color correction, the printer device can print all 4,096 colors displayed by the Amiga on a color printer. However, as color correction is applied, the total number of printed colors is reduced to a low of 3,172 (308 shades are lost per color selected).

To see the effect of this option, print out a picture with solid red, green, and blue shades twice; first with the Color Correct option on, and then with it off. Now compare the two pictures with the screen display. The shades of the color corrected picture should more closely represent the colors displayed on your monitor. Default is no color correction (R, G, and B are not selected).

Dithering [Ordered|Halftone|F-S] - sets the dithering mode. Dithering refers to the printing of dots of different colors in such a way that they are so small and close together that the eye perceives them as one color. This enables you to produce printouts with various shades while the printer only uses four colors. Default is ORDERED. Note: when a Shade of Black and White is selected (from the Graphic 1 screen), changing the dither method has no effect on the printout.

Ordered — Color intensities on the printer are formed using an ordered dither method. Ordered dithering produces shades on the printer using an ordered pattern of dots.

Halftone — Color intensities on the printer are formed using a halftone dither method. This technique is similar to the one used in newspapers and comic books. It works best on high density printers (greater than 150 dots per inch).

F-S — Color intensities on the printer are formed using the Floyd-Steinberg error distribution method. This option incurs an approximate 2:1 speed penalty when printing. Note: Selecting this option automatically turns off Smoothing as they can not be used in conjunction with each other. F-S works best on high density printers (greater than 150 dots per inch).

Scaling [Fraction|Integer] — selects the scaling method. Default is FRACTION.

Fraction — Normal scaling is performed.

Integer — Every dot on the screen is guaranteed to appear as an even number of dots on the printout in both the x and y

dimensions. For example, if the source picture is 320 x 200, the printed picture width will be either 320, 640 or 960 dots wide, etc. The height will be 200, 400 or 600 dots high, etc. This option should be selected when trying to print a picture that contains thin vertical and horizontal lines (like a grid).

The actual size of the printed picture will be the requested size scaled up or down to the nearest multiple of the width and height of the picture. Integer scaling completely overrides the aspect feature of the printer device. Thus it is possible to get a slightly distorted (non-aspect ratio correct) picture. This option is also useful for printing out bit-image text (as in Notepad), since the fonts will not be distorted due to fractional scaling.

Width Limit — limits the width of the printed picture. The width can be limited in tenths of an inch, pixels or a multiplication factor. Default is 0. (See Limits, below.)

Height Limit — limits the height of the printed picture. The height can be limited in tenths of an inch, pixels or a multiplication factor. Default is 0. (See Limits, below.)

<- Limits [Ignore|Bounded|Absolute|Pixels|Multiply] — determines how the Width and Height limits are interpreted. Default is IGNORE.

Ignore — The limits are to be ignored. (This option is the default and is included to remain compatible with pre-V1.3 software.) The printed picture's size is the size requested by the application, bounded by:

width = (right margin - left margin + 1) / characters per inch
height = lines per page / lines per inch

Bounded — The printed picture's size is bounded by Width Limit and Height Limit. For example, if the printed picture should be no bigger than 4.0 x 5.0 inches (but it could be smaller), set MaxWidth to 40, MaxHeight to 50, and select Bounded. This option is provided so that the text settings (margins, lines per page, etc.) need not be changed every time a graphic print is made.

Absolute — Width Limit and Height Limit are interpreted as absolute values. The printed picture's size is no longer bounded. Rather, it is the absolute size specified. For example, if the printed picture should be exactly 4.0 x 5.0 inches, set MaxWidth to 40, Width Limit to 50, and select Absolute. This completely overrides the aspect feature of the printer device. Thus it is possible to get a very distorted (non-aspect ratio correct) picture.

There may be an instance when you want your printout to be a specific width or height, and you want the aspect ratio to be correct so that the picture is not distorted. In this case, set the Width Limit or Height Limit to the desired dimension, and set the other Limit to zero. When one dimension is set to zero, the

aspect ratio corrects only that dimension. For example, if Width Limit is set to 40 and Height Limit to 0, then the printed picture will be 4.0 inches wide and as tall as necessary in order to be aspect ratio correct. If both of the dimensions are zero, the printed picture will be the printer's maximum dots wide and as tall as necessary in order to be aspect ratio correct.

Pixels — The Width Limit and Height Limit are interpreted as pixels, instead of tenths of an inch. Otherwise, the same rules for the Absolute option apply.

Multiply — Width Limit and Height Limit are used to multiply the source picture's width and height. For instance, if you specified a Width Limit of 2 and a Height Limit of 4, the printed picture will be two times the source picture's width (in pixels) and four times the source picture's height. Thus if the source picture were 320 x 200, the printed picture would be 640 (320 x 2) pixels wide and 800 (200 x 4) pixels high. The same rules for Absolute scaling (Width Limit = 0 and Height Limit = 0) apply. In addition, the benefits inherent to Integer scaling are valid here.

CopyPrefs

Double-clicking on this icon brings up a window with the message ''Copying devs:system-configuration to DF0:DEVS.'' This is for users who have assigned DEVS: to somewhere other than the devs directory of the boot disk, for instance, users with hard drive systems that are not autobooting and who must boot off a floppy disk (i.e., A2090 users). If you have changed Preferences on your hard disk and want to update your Workbench boot disk, use CopyPrefs. CopyPrefs will copy the system-configuration file from the hard disk onto your Workbench disk. The proper disk must be in df0: when CopyPrefs is started. CopyPrefs is a script file which you can edit to alter the COPY command, such as from RAD: to DH0: when using a recoverable ramdrive.

The Utilities Drawer

The Utilities Drawer on Workbench Version 1.3 has more than tripled its size since Version 1.2 was released. Say and GraphicDump have been moved out of the System drawer and into Utilities, along with the familiar Notepad, Calculator and Clock. In addition, five new programs have been moved in — ClockPtr, CMD, InstallPrinter, MORE, and PrintFiles.

- ClockPtr — turns the pointer into a digital clock
- CMD — redirects serial or parallel output to a file
- InstallPrinter — copies printer drivers from the Extras disk onto the Workbench disk
- MORE — allows you to view ASCII text files in a Workbench window
- PrintFiles — lets you print multiple files

ClockPtr

The ClockPtr program changes your Workbench pointer into a digital clock whenever the Workbench backdrop screen is active. To run ClockPtr from the Workbench, double-click on its icon. Move the pointer out of the Utilities window, click the mouse, and it becomes a clock. ClockPtr displays the hour and minute. To display minute and seconds, move the pointer to the top left corner of the Workbench screen, and an interval timer will be displayed. To display the date, move the pointer to the left side of the Workbench screen.

Close ClockPtr by double-clicking on its icon or by typing Control-C if started from the CLI. If ClockPtr is RUN from the CLI (using the Amiga-DOS RUN command), use the BREAK command to close it.

Use the DATE or SETCLOCK command to set the time (See Chapter 2).

CMD

CMD allows you to redirect either serial or parallel device CMD__ WRITEs to a designated file, allowing you to reroute your printer output to a file. (A CMD__WRITE is an internal Amiga function which controls the computer's behavior. When you use a software program or a CLI command to send information to your printer, the Amiga is informed of your decision via a CMD__WRITE.)

To run CMD from the CLI, the format is:

CMD <devicename> <filename> [OPT s|m|n]

<devicename> is either ''serial'' or ''parallel'', and <filename> is the name of the file to which the redirected output should be sent. Note that CMD does not accept ''PAR:'' or ''SER:'' as device names.

The CMD options are as follows:

s Skip any short initial write (usually a reset if redirecting a screen dump)*

m Install for MULTIPLE files until a BREAK command or Control-C is typed

n Enable NOTIFY (helpful progress messages)

*On a screen dump, the first CMD__WRITE is usually a printer reset. The printer device then delays long enough for the reset to complete to prevent the loss of subsequent output data. When CMD is used to redirect the output to a file, this delay is lost. If your printer driver outputs a reset at the start of a dump, use the s option to keep the initial CMD__WRITE out of the file.

You can also use CMD from the Workbench. You specify the device name, filename, and options through the icon's TOOL TYPES. Select the CMD icon, then choose INFO from the Workbench menu. CMD recognizes the following parameters in the TOOL TYPES field of the CMD icon:

DEVICE   Set to either parallel or serial; default is parallel.

FILE   The name of the output file; default is ram:CMD__file.

SKIP   Set to TRUE to skip any short initial write; default
                is FALSE — does not skip initial write.

MULTIPLE   Set to TRUE to install for multiple files; default is
                FALSE — does not install.

NOTIFY   Set to TRUE to receive progress messages; de-
                fault is FALSE — no messages are sent.

Double-clicking on the CMD icon implements redirection as specified in
the TOOL TYPES.

InstallPrinter

InstallPrinter is a program that lets you copy a printer driver from the
Extras disk to your DEVS:printer directory on the Workbench disk with-
out having to use any AmigaDOS commands or open a SHELL or CLI.
Double-click on the InstallPrinter icon, and a window appears displaying
a list of the drivers included on the Extras disk. Type the name of the
appropriate driver at the prompt, and hit RETURN. The printer driver file
will be automatically copied to the DEVS:printers directory of the Work-
bench disk, and the name of the driver will appear in the Change Printer
window of Preferences. You can use wildcards to copy more than one
driver. To quit InstallPrinter without copying a driver, hit RETURN.

MORE

MORE is a utility for displaying ASCII text files. There are two ways to run
MORE from the Workbench. If the text file has an icon attached to it,
select the MORE icon, then **while holding down the Shift key,**
double-click on the text file icon. If there is no icon for the text file, simply
double-click on the MORE icon. A prompt will appear asking you for the
name of the text file. Be sure to specify the complete path.

To run MORE from the CLI, the format is:

MORE <filename>

When you specify the file to be displayed, be sure to enter the complete
path, including the disk name (or drive number), directory, and file. If you
don't specify a file, MORE will display a short explanation and a prompt
asking you to enter the filename.

When you run MORE, a message such as '' – More (48%) –'' will
appear at the bottom of the window. This indicates the percentage of the
file viewed so far. To move through the display, use the following key
sequences:

<SPACE>	displays the next page
<BACKSPACE>	displays the previous page*
<RETURN>	displays the next line
<	displays the first page*
>	displays the last page*
%n	displays approximately n% into the file*
CTRL + L	refreshes the window
/<string>	MORE will perform a case sensitive search for the string of characters speci-fied after the /

The Workbench   1-13

. \<string\>	MORE will perform a non-case sensitive search for the specified string
n	finds the next occurrence of the string
h	help (displays a list similar to this one)
q	quit (or Control-C)
E	edit using editor set in ENV: EDITOR

*MORE also accepts input from a PIPE. Since standard input from the Pipe-Handler is of unknown length, the commands marked with an asterisk are disabled when the MORE input is from a PIPE. (See Chapter 3, Other Workbench Directories.)

A case-sensitive search means that MORE looks for the string exactly as it is entered. If you type the string in capital letters, MORE will only look for occurrences of the string that appear in capital letters. A non-case sensitive search means that whether the string is entered in upper or lower case letters does not make a difference.

When you reach the last page of the display, the message at the bottom of the screen will read ''– Less –''.

When using MORE from the CLI or SHELL, you can bring up an editor to use on the file you are viewing (type Shift-E), if the EDITOR environment variable is defined. The EDITOR variable should have the complete path to the editor specified, i.e. Extras:Tools/MEMACS or C:ED. (For more information on using environment variables, see the RAM-Handler section of this chapter and the GETENV and SETENV commands in Chapter 2.)

NOTE: If you accidentally press the Escape key while using MORE, you will be interrupted. To continue, press either the "q" or the "h" key.

PrintFiles

With PrintFiles you can copy files to your printer using either the Workbench or the CLI. PrintFiles also accepts multiple filenames, so you can designate a string of files to be printed. If PrintFiles cannot find or open one of the files, it will skip it and go on to the next one.

To run PrintFiles from the CLI, the format is:

PRINTFILES [-f] \<filename\> [[-f] \<filename\>] [[-f] \<filename\>] . . .

The -f flag turns on the form feed mode which places a form feed between subsequent files and at the end of the file(s). To specify the form feed mode from the Workbench, use the Workbench menu's INFO item to bring up the PrintFiles INFO window. Then add "FLAGS = formfeed" to the icon's TOOL TYPES.

To use PrintFiles from the Workbench:

1) Select the icon of the first file you want to print.
2) Hold down the SHIFT key and select the icons of any additional files you want to print.
3) Hold down the SHIFT key and double-click on the PrintFiles icon.

The Workbench

GraphicDump

GraphicDump sends a dump of the frontmost Intuition screen to the printer about ten seconds after double-clicking on the icon. (The ten second delay allows you to rearrange your screens so that the frontmost screen is the one you want to print.) There has been a change made to GraphicDump since the 1.2 Version; you can now specify the size of the resulting printout. To invoke GraphicDump through the CLI, the format is:

GraphicDump [TINY|SMALL|MEDIUM|LARGE|xdots:ydots]

There are five options that determine the size of the printout:

TINY   The width of the resulting printout is about 1/4 the total width allowed by the printer. The height is such that the screen's original aspect ratio is maintained.

SMALL   The width of the resulting printout is about 1/2 the total width allowed by the printer. The height is such that the screen's original aspect ratio is maintained.

MEDIUM   The width of the resulting printout is about 3/4 the total width allowed by the printer. The height is such that the screen's original aspect ratio is maintained.

LARGE   The width of the resulting printout is the full width allowed by the printer. The height is such that the screen's original aspect ratio is maintained. (When you run GraphicDump from the Workbench, LARGE is the default size of the printout.)

xdots:ydots Use xdots as the absolute width in dots; use ydots as the absolute height in dots. This allows the user to specify any dimensions within the confines of the printer.

In order for GraphicDump to recognize the TINY, SMALL, MEDIUM or LARGE options, the Limits setting in Preferences must be set to Ignore (the default option). If a different option is chosen in Limits, the size of the resulting printout will be determined by the specified option. The following examples assume that Limits is set to Ignore.

1) To produce a printout of the frontmost screen that is about one-half the total width allowed by the printer, you would type:

1> GRAPHICDUMP SMALL

2) To specify the exact dimensions of the printout, for instance, if you wanted it to be 640 pixels by 200 pixels, you'd type:

1> GRAPHICDUMP 640:200

The Clock

When you open the Clock, a 170 x 90 pixel analog clock will appear. When the Clock is in Analog mode, you can use its sizing gadget to shrink or enlarge it. This is the only mode which will let you change the size of the Clock.

If you prefer a digital display, you now have a choice: Digital 1 or Digital 2. Digital 1 displays the clock in a rectangular window with two lines — one for the time and one for the date, if desired. Digital 2 is a smaller window that appears in the Title Bar so that it doesn't obstruct anything on the screen. When the Date option is turned on, the time and date are alternately displayed in the window.

The Alarm works exactly as it did before, except for one minor detail. Now when the Alarm window is present, the clock itself remains functional. With Version 1.2, the clock would freeze when the alarm requester was displayed. Remember though, you cannot close the Clock when the Alarm is set.

If you start the Clock from the CLI, you can use command line arguments to specify which type of Clock you want. The format is:

CLOCK [ANALOG|DIGITAL1|DIGITAL2] [=<x>,<y> [, <width>, <height>]] [12HOUR|24HOUR] [SECONDS] [DATE]

The options are as follows:

ANALOG, DIGITAL1, or DIGITAL2 — corresponds to the Clock's Type menu, described above; default is ANALOG.

=<x>,<y> — allows you to determine the placement of the clock. The number assigned to <x> is equal to the number of horizontal pixels from the left side of the screen, while <y> is equal to the number of pixels from the top. The <width>,<height> option pertains to the size of the Analog clock, in pixels. If this option is given for a Digital clock, it is ignored. When you type this option, be sure there is no space between the word ANALOG and the equals sign. Default is =10,15.

12HOUR or 24HOUR — choose either a 12 hour (AM/PM) clock or the military 24 hour clock; default is 12 hour.

SECONDS — displays the seconds; default is for the seconds option to be off.

DATE — displays the date; default is for the date option to be off.

For example:

```
1> Run Clock ANALOG=10,20,600,160 12HOUR DATE
```

will display an analog clock at screen position 10, 20, 600 pixels wide by 160 pixels high, in 12-hour mode (AM/PM indicator), with the date.

The Workbench

The Two RAM Disks

There are two RAM disks on the new Workbench. The standard RAM-Handler (RAM:), for which a disk icon automatically appears on the Workbench screen, and the new recoverable ramdrive.device (RAD:).

The RAM-Handler

The RAM-Handler, located in the Workbench's L: directory, is the standard RAM: disk that has been with the Amiga all along. This handler appears in the upper right corner of the Workbench screen as the RAM DISK icon. The Amiga recognizes this ''disk'' as RAM:, just as it recognizes a disk in your internal drive as df0:. RAM: changes size depending on what is stored in it. Therefore, it is always 100% full. As you add files it expands, and as you delete files it shrinks.

All the popular bugs in the RAM-Handler were fixed in Version 1.3. The most obvious differences are:

- The RAM-Handler is now known as ''RAM DISK''.
- It can be renamed using the Workbench menu's RENAME item.

During the Workbench's startup-sequence, several directories are created in RAM: then assigned the logical device names of T:, ENV:, and CLIPS:. RAM:'s T: directory is used for scripts, while CLIPS: is a new, alternate directory for storing clips from the clipboard.device. You can leave these directories in RAM: or assign them elsewhere if you desire.

ENV: is for storing environment variables created with the SETENV command. You use SETENV to assign a ''value'' to a variable. This is not necessarily a mathematical or numerical value, rather it is a variable string. For instance, if you want to use the Editor option of the MORE program, and you want to use MEMACS as the editing tool, type:

```
SETENV EDITOR "Extras 1.3:Tools/MEMACS"
```

In this case the name and path to the editing program is the variable string. If you were to look in the ENV: directory after using SETENV, you would find a file called EDITOR. Currently, MORE and the IF command use environment variables. Other commands will use them in the future.

If you create a variable with SETENV and later want to check to see the string that you've assigned to it, use the GETENV command. Simply type GETENV followed by the name of the variable, and the system will respond with the value, or string, assigned to that name.

The Recoverable Ramdrive.device

A recoverable ramdrive allows you to access the contents of the ramdrive.device after rebooting your machine. Even if you reboot with something other than the Workbench disk (for instance, a word processing program), the recoverable ramdrive will still be present and will still hold its contents. The contents are not lost until you turn the Amiga's power off or invoke the REMRAD command.

Just like a hard disk or some other expansion device, the ramdrive must be MOUNTed before use. A sample MountList entry (called RAD:) is

included in the Workbench's DEVS: directory MountList file. A sample CLI session performing these operations is described below:

```
1> TYPE devs/mountlist
```

Just to make sure the entry for RAD: is in there. The default entry has the high cylinder set at 21. For comparison purposes, a standard Amiga disk has 79 cylinders. If you wish to enlarge the size of the device, increase the number for the high cylinder entry.

```
1> MOUNT rad:
```

Tells the Amiga that the recoverable ramdrive.device is in use. To check that you've mounted RAD correctly, use the INFO command.

```
1> RELABEL DRIVE Rad: NAME MyRad
```

You can name the device anything you choose except RAM. You don't want to confuse it with the Workbench's other RAM disk.

```
1> DISKCHANGE Rad:
```

Informs the Workbench of the name change made with RELABEL.

If you have not upgraded your computer to Kickstart Version 1.3, either by using the new 1.3 Kickstart disk in your A1000 or by having the new 1.3 Kickstart ROM installed in your A500 or A2000, you will need to MOUNT RAD: each time you reboot.

If you want to remove RAD: without turning the entire system off, you can use the REMRAD command. Simply type REMRAD at a CLI prompt, and the RAD: device will shrink. If you reboot your machine, RAD: will not recover.

When using Kickstart Version 1.3, it is possible to reboot directly from RAD: without having to insert a Workbench, or other bootable disk, into the disk drive. (See Chapter 5, Changing Your Startup Sequences.)

2. AMIGADOS AND WORKBENCH'S C DIRECTORY

This chapter outlines any changes or additions made to the commands in the Workbench's C: directory. There are also several new commands documented in this section. For more information on AmigaDOS, consult *The AmigaDOS Manual* published by Bantam Books and available from many computer dealers and bookstores.

There are sixty-four commands in the Workbench's C: Directory:

ADDBUFFERS	EDIT	JOIN	RESIDENT
ASK	ELSE	LAB	RUN
ASSIGN	ENDCLI	LIST	SEARCH
AVAIL	ENDIF	LOADWB	SETCLOCK
BINDDRIVERS	ENDSKIP	LOCK	SETDATE
BREAK	EVAL	MAKEDIR	SETENV
CD	EXECUTE	MOUNT	SETPATCH
CHANGETASKPRI	FAILAT	NEWCLI	SKIP
COPY	FAULT	NEWSHELL	SORT
DATE	FF	PATH	STACK
DELETE	FILENOTE	PROMPT	STATUS
DIR	GETENV	PROTECT	TYPE
DISKCHANGE	ICONX	QUIT	VERSION
DISKDOCTOR	IF	RELABEL	WAIT
ECHO	INFO	REMRAD	WHICH
ED	INSTALL	RENAME	WHY

Many of these commands are only used by programmers and more advanced users. However, many ''average'' users find AmigaDOS very valuable for organizing the information contained on their disks. For this reason, any new or revised commands will be documented here so that all users can have access to this information.

Unchanged Commands

These commands have not been changed with the release of Version 1.3 and are fully documented in *The AmigaDOS Manual*.

BREAK	To set attention flags in the specified process.
CD	To set, change, or list the current directory.
ED	To edit text files (a screen editor).
EDIT	To edit text files by processing the source file sequentially (a line editor).
ELSE	To have an alternative in a conditional in a script file.
ENDIF	To terminate an IF block in a script file.
FAILAT	To set the failure condition of a script file.
FAULT	To print the specified error message(s).
LAB	To specify a label in a script file.
MAKEDIR	To create a new directory.
QUIT	To exit from a script file with a specified error code.
RELABEL	To change the volume name of a disk.
RENAME	To change the name of a file or directory.
SORT	To sort small files.

STACK   To display or set the stacksize of the current CLI.
WAIT   To cause AmigaDOS to wait a specified amount of time.
WHY   To print an error message to explain why the previous command failed.

New or Revised Commands

Many AmigaDOS commands have been revised since the release of Version 1.2, and others are not fully documented in the available reference material. These commands are listed here using the standard AmigaDOS outline: Format, Template, Purpose, Path, and Specification. Some of the text conventions you should be familiar with are:

COMMAND   AmigaDOS commands appear in capital letters to distinguish them from the rest of the text.

< >   Angled brackets enclose arguments that you need to specify. For instance, <filename> means that you must enter a specific filename along with the AmigaDOS command.

[ ]   Square brackets enclose options; options will be accepted by AmigaDOS but are not required.

{ }   Braces enclose items that can be repeated any number of times (or not at all). For example, {<args>} means that a number of arguments can be given but are not required.

. . .   Three dots indicate a series that can be continued.

| A vertical bar is used to separate a list of options of which you can choose one. For instance, [OPT R|S|RS] means that you can choose the R option, the S option or both (RS) options.

<argument>/A The **argument** must be specified in order for the command to work.

<option>/K The option **keyword** must be specified, even if other parameters are also given. For instance, one of the options of the DIR command is [OPT A|I|AI|D]. The Template is "OPT/K". This means that you must type the word OPT when you specify the chosen option. Either OPT A, OPT I, OPT AI, or OPT D. You cannot simply type A, I, AI or D alone.

<option>/S The option works as a **switch**. You must type the name of the option in order to specify that option. For instance, the DIR command also has an [ALL] option. You must type the word ALL if you want to use that option.

Several of the command explanations refer to a command's *condition flag*. A condition flag is set when a command fails in some way. A condition flag of 5 represents a warning and means that although some sort of error occurred, it was not serious enough to terminate the process. Subsequent commands will be executed. This is further explained in the section on the FAILAT command in *The AmigaDOS Manual*.

ADDBUFFERS

Format: ADDBUFFERS <drive>: <nn>

Template: ADDBUFFERS ''DRIVE/A,BUFFERS/A''

Purpose: To command the file system to add cache buffers.

Path: C:ADDBUFFERS

Specification:

ADDBUFFERS adds <nn> buffers to the list of sector caches for <drive>. When using the standard file system, additional buffers can make disk access significantly faster. However, each additional buffer reduces memory by approximately 500 bytes. As a general rule, only add 25-30 buffers. Disk access time does not really increase by adding more than 25 or 30; you would also be using up valuable memory.

When using the FastFileSystem with a hard drive, adding additional buffers always speeds up disk access. (For more information on the FastFileSystem, see Appendix A.) The recommended number of additional buffers varies depending on how much extra memory is available.

Example:

```
1> ADDBUFFERS df1: 25
```

Adds 25 buffers to the sector caches for disk drive df1:.

ASK

Format: ASK <prompt>

Template: ASK ''PROMPT/A''

Purpose: To obtain user input when executing a script file.

Path: C:ASK

Specification:

ASK writes the <prompt> string to the current output stream, then waits for the user's keyboard input. Valid responses are Y (yes), N (no), and RETURN (same as no). ASK then sets the condition flag to 5 (=WARN) if a YES response was typed, and 0 if a NO response was typed. This command is normally only useful in script files.

See also: IF, EXECUTE

ASSIGN

Format: ASSIGN [[<name>:] <dir>] [LIST] [EXISTS] [REMOVE]

Template: ASSIGN ''NAME,DIR,LIST/S,EXISTS/S, REMOVE/S''

Purpose: To assign a logical device name to a file system directory.

Path: C:ASSIGN

Specification:

With Version 1.3, ASSIGN will now print device names with more than 3 letters properly. If the LIST keyword is given along with a logical device name, ASSIGN will search the ASSIGN list for that name and remove it from the list.

If the EXISTS keyword is given along with a logical device name, AS-SIGN will search the ASSIGN list for that name, and display the device and the directory assigned to the device. If the device is not found, the condition flag is set to WARN.

The REMOVE option disconnects a volume or device from the list of mounted devices. It does not free up resources; it merely removes the name from the list. THIS OPTION IS PRIMARILY USED DURING SOFTWARE DEVELOPMENT. CARELESS USE OF THIS OPTION MAY CAUSE A GURU FAILURE.

Examples:

```
1> ASSIGN fonts: EXISTS
FONTS: Workbench 1.3:fonts

1> ASSIGN fonts: LIST
Volumes:
RAM Disk [Mounted]

Workbench 1.3 [Mounted]

Directories:
CLIPS:          RAM DISK:clipboards
ENV:            RAM DISK:env
T:              RAM DISK:t
S:              Workbench 1.3:s
L:              Workbench 1.3:l
C:              Workbench 1.3:c
DEVS:           Workbench 1.3:devs
LIBS:           Workbench 1.3:libs
SYS:            Workbench 1.3:sys

Devices:
SPEAK NEWCON DF2 DF0 PRT
PAR SER RAW CON RAM
```

Note that the assignment ''FONTS: Workbench 1.3:fonts'' has been removed from the ASSIGN list.

AVAIL

Format: AVAIL [CHIP|FAST|TOTAL]

Template: AVAIL ''CHIP/S,FAST/S,TOTAL/S''

Purpose: To report the amount of available CHIP and FAST memory.

Path: C:AVAIL

Specification:

The AVAIL command gives a summary of the system RAM memory, both CHIP and FAST. For each memory type, AVAIL reports the total amount, how much is available, how much is currently in use, and the largest contiguous memory block not yet allocated.

By using the CHIP, FAST and/or TOTAL options, AVAIL will display only the number of free bytes of CHIP, FAST or TOTAL RAM available, instead of the complete summary. This value can be used for comparisons in scripts.

Examples:

```
1> AVAIL
Type       Available    In-Use    Maximum    Largest
chip       233592       282272     515864      76792
fast       341384       182896     524280     197360
total      574976       465168    1040144     197360

1> AVAIL CHIP
233592
```

See Also: EXECUTE, IF

BINDDRIVERS

Format: BINDDRIVERS

Template: BINDDRIVERS

Purpose: To bind device drivers to hardware.

Path: C:BINDDRIVERS

Specification:

BINDDRIVERS is normally part of a startup sequence. It is used to bind device drivers found in the SYS:Expansion directory to add-on hardware that has been automatically configured by the expansion library. What this means, is that if drivers for expansion hardware are in the Expansion drawer, the hardware will be configured automatically when the system is booted.

CHANGETASKPRI

Format: CHANGETASKPRI <priority> [<process>]

Template: CHANGETASKPRI ''PRIORITY/A,PROCESS/K''

Purpose: To change the priority of processes started from the CLI.

Path: C:CHANGETASKPRI

Specification:

Since the Amiga is a multitasking machine, it uses priority numbers to determine which tasks it should attend to and in which order. Normally, most tasks have a priority of 0, and the time and instruction cycles of the CPU are divided among them. CHANGETASKPRI changes the priority of the specified CLI process. (If no process is specified, the current CLI process is assumed.) Any tasks started from <process> inherit its priority.

The range of acceptable values for <priority> is -128 to 127. However, you should not enter values above +5 to avoid disrupting important system tasks.

Process numbers of CLIs can be determined by using the STATUS command.

Example:

```
1> CHANGETASKPRI 5
```

The priority of the current CLI task is changed to 5. Any tasks started from this CLI will also have a priority of 5. They will have priority over any other user tasks created without using CHANGETASKPRI (those tasks will have a priority of 0).

See also: STATUS

COPY

Format: COPY [[FROM] <name>] [TO] <name> [ALL] [QUIET]
[BUF|BUFFER = <nn>][CLONE] [DATE] [COM]
[NOPRO]

Template: COPY ''FROM,TO/A,ALL/S,QUIET/S,BUF = BUFFER/K,
CLONE/S,DATE/S, COM/S, NOPRO/S''

Purpose: To copy a file or directory.

Path: C:COPY

Specification:

You can use COPY to copy multiple files to a device. Either use the ALL option to copy an entire directory, or use pattern matching to copy files that share a common set of characters or symbols.

For instance, you can use pattern matching to copy all the files of a directory by specifying the directory name as the pattern. (For a full explanation of patterns see *The AmigaDOS Manual*. Note: The 31 character restriction on the use of wildcards has been removed.)

The BUF= option is used to set the number of 512 byte buffers used during the copy. (Default is 100K or 200 buffers.) It is often useful to limit the number of buffers when copying to the RAM: disk.

The CLONE option tells COPY to copy the date, comments and protection bits of the original to the destination file. The DATE and COM options direct COPY to copy the creation date and the comments of the original file. By default, COPY retains the protection bits of the original file when it makes a copy. This may be overridden by using the NOPRO option.

Formerly, if a directory specified as the destination did not exist, COPY did not create one. With Version 1.3, COPY will now create a new destination directory. You can also use a pair of quotes ('' '') to refer to the current directory when specifying a destination.

DATE

Format: DATE [<date>] [<time>] [TO = VER <filename>]

Template: DATE ''DATE,TIME,TO = VER/K''

Purpose: To display or set the system date and/or time.

Path: C:DATE

Specification:

DATE now allows either a leading zero or a single digit in the date specification. Formerly you had to supply the leading zero.

DELETE

Format: DELETE <name>* [ALL] [Q|QUIET]

Template: DELETE ''NAME(S),ALL/S,Q = QUIET/S''

Purpose: To delete up to 10 files or directories.

Path: C:DELETE

Specification:

You can use pattern matching to delete files. If you are using pattern matching, the 31 character restriction on the use of wildcards has been removed. (For a full explanation of patterns, see *The AmigaDOS Manual*.)

DIR

Format: DIR [<name>] [OPT A|I|AI|D] [ALL] [DIRS] [INTER] [FILES]

Template: DIR ''NAME,OPT/K,ALL/S,DIRS/S,INTER/S,FILES/S''

Purpose: To display a sorted list of the files in a directory.

Path: C:DIR

Specification:

ALL, DIRS, and INTER are now synonyms for the A, D, and I options, respectively. In addition, there is a COMMAND= option in interactive mode (invoked through OPT I or INTER) which allows almost any AmigaDOS command to be executed during the interactive directory list.

When you want to issue a general command in interactive mode, type C (or COM) at the question mark prompt. DIR will ask you for the command. Type the desired command, then hit RETURN. The command will be executed and DIR will continue. You can also combine the C and the command on one line, by putting the command in quotes following the C. For instance, C ''type prefs.info hex''.

The FILES option displays only the files in a directory; subdirectories are not displayed.

DISKCHANGE

Format: DISKCHANGE <drive>:

Template: DISKCHANGE ''DRIVE/A''

Purpose: To inform the Amiga that you have changed a disk in a 5 1/4″ disk drive.

Path: C:DISKCHANGE

Specification:

You must use the DISKCHANGE command when you are using 5 1/4″ floppy disk drives with your Amiga system. Whenever you change the disks in the 5 1/4″ drive, you must use DISKCHANGE to inform the system of the switch.

This command is also useful to inform Workbench of any name changes made to diskettes with the RELABEL command.

Example:

If a requester appears and asks you to insert a new disk into your 5 1/4″ drive, known as df2:, you must insert the disk, then type:

```
1> DISKCHANGE df2:
```

AmigaDOS will then recognize the new disk, and you can proceed.

DISKDOCTOR

Format: DISKDOCTOR <drive>:

Template: DISKDOCTOR ''DRIVE/A''

Purpose: To attempt to repair a corrupt disk.

Path: C:DISKDOCTOR

Specification:

DISKDOCTOR attempts to repair a corrupt floppy disk enough to allow you to retrieve files from it and copy them onto a good disk. If AmigaDOS detects a corrupt disk, it displays a requester stating that the disk could not be validated. By using DISKDOCTOR, you can try to restore the disk's file structure. After running DISKDOCTOR, you should copy the restored files to another disk, then reformat the corrupt disk.

You can use DISKDOCTOR on both the standard file system and the FastFileSystem. However, to use DISKDOCTOR with the FastFileSystem, YOU MUST MAKE SURE THAT THE DOSTYPE KEYWORD IN THE MOUNTLIST IS SET TO 0X444F5301. Do NOT use DISKDOCTOR on a FastFileSystem partition if the DosType Keyword is not set correctly. (See Appendix A for more information on the FastFileSystem.)

With Version 1.3, DISKDOCTOR checks for enough memory before starting operations and changes the boot block to type DOS.

Example:

If you receive a message stating that ''Volume Workbench is not validated'' or ''Error validating disk/Disk is unreadable'', you can use DISKDOCTOR to retrieve the disk's files. For instance, if the corrupt disk is in df1, type:

```
1> DISKDOCTOR df1:
```

AmigaDOS will ask you to insert the disk to be corrected and press RETURN. DISKDOCTOR then reads each cylinder of the disk. If it finds an error, it displays ''Hard error Track xx, Surface xx''. As each file and directory is replaced, the filename is displayed on the screen. When DISKDOCTOR is finished, it says ''Now copy files required to a new disk and reformat this disk.''

ECHO

Format: ECHO <string> [NOLINE] [FIRST <nn>] [LEN <nn>]

Template: ECHO ''STRING,NOLINE/S,FIRST/S,LEN/S''

Purpose: To print a string.

Path: C:ECHO

Specification:

Version 1.3 has introduced the NOLINE option. When the NOLINE option is specified, ECHO does not automatically output a new line after printing the string.

The FIRST and LEN options allow the echoing of a substring. FIRST <nn> indicates the character position to begin the echo; LEN <nn> indicates the number of characters of the substring to echo, beginning with the first character.

If the FIRST option is omitted and only the LEN keyword is given, the substring printed will consist of the rightmost <nn> characters of the main string. For instance, if your string is 20 characters long and you specify LEN <4>, the 17th, 18th, 19th, and 20th characters of the string will be echoed.

ENDCLI

Format: ENDCLI

Template: ENDCLI

Purpose: To end an interactive CLI process or a SHELL process.

Path: C:ENDCLI

Specification:

ENDCLI can be used to end either a CLI process or a SHELL process.

ENDSKIP

Format: ENDSKIP

Template: ENDSKIP

Purpose: To terminate a SKIP block in a script file.

Path: C:ENDSKIP

Specification:

When an ENDSKIP is encountered during a SKIP, execution of the script file resumes at the line following the ENDSKIP. (The condition flag is set to WARN.)

EVAL

Format: EVAL <value1> <operation> <value2> [TO <file>]
            [LFORMAT = <string>]

Template: EVAL "VALUE1,OPERATION,VALUE2,TO/K,LFORMAT/K"

Purpose: To provide a means of evaluating simple expressions.

Path: C:EVAL

Specification:

The EVAL command is used to evaluate, and to print the answer of, simple (1 and 2 argument) integer expressions. <value1> and <value2> may be in decimal, hexadecimal, or octal numbers. Decimal numbers are the default. Hexadecimal numbers are indicated by either a leading OX or #X. Octal numbers are indicated by either a leading 0 or a leading # (followed by other digits). Alphabetical characters are indicated by a leading single quote (').

The supported operations and their corresponding symbols are as follows:

addition	+	not	~
subtraction	−	left shift	<<
multiplication	*	right shift	>>
division	/	negation	−
mod	mod	exclusive or	xor
and	&	bitwise equivalence	eqv
or	|		

The output format defaults to decimal; however, through use of the LFORMAT keyword, you may select another format. The LFORMAT keyword specifies the formatting string used to print the answer. You may use %X (hexadecimal), %O (octal), %N (decimal), or %C (character). Note that the %X and %O options require a ''number of digits'' specification, (i.e., %X8 gives 8 digits of hex output.)

When using the LFORMAT keyword, you can specify that a new line should be printed by including a *N in your string.

EVAL can be used in scripts to act as a counter for loops. In that case, the TO option, which sends the output of EVAL to a file, is very useful.

Examples:

```
1> EVAL 4 * -5
-20

1> EVAL 0x4f / 010 LFORMAT = ' 'The answer is %X4*N' '
The answer is 9
```

Example in a script:

```
.key loop/a
;demo a loop using eval and skip
.bra {
.ket }
echo >env:loop {loop}
lab start
echo "Loop #" noline
type env:loop
eval <env:loop >NIL: to=t:qwe{$$} value2=1 op=- ?
type >env:loop t:qwe{$$}
IF val $loop GT 0
skip start back
endif
echo "done"
```

EXECUTE

Format: EXECUTE <command file> <arguments>

Template: EXECUTE ''COMMANDFILE,ARGUMENTS''

Purpose: To execute a command file with argument substitution.

Path: C:EXECUTE

Specification:

EXECUTE now uses the logical name T: if it is ASSIGNed; otherwise it uses the :T directory.

Since script files call on the EXECUTE command, it is an excellent candidate for being made resident when using the SHELL.

The current CLI number can be referenced by the characters <$$>. This is useful in creating unique temporary files, logical assignments, and PIPE names.

See Also: ASSIGN, and the SHELL-SEG section of Chapter 3.

FF

Format: FF [-0] [-N]

Template: FF ''-0/S,-N/S''

Purpose: To speed up text on the Amiga.

Path: C:FF

Specification:

FF (FastFonts) is a program written by Charlie Heath, of Microsmiths, Inc. FF speeds up the display of text on the Amiga. To turn on the FastText routines, use the -0 option. To turn off the FastText routines, use the -N option. FF can also be used to replace the default system font with a plain 8 × 8 pixel font. Use the command: FF <fontname>.

FF is used by permission of Microsmiths, Inc.

FILENOTE

Format: FILENOTE [FILE] <filename> COMMENT <comment>

Template: FILENOTE ''FILE/A,COMMENT/K''

Purpose: To attach a comment to a file.

Path: C:FILENOTE

Specification:

The length of <comment> is limited to 79 characters.

GETENV

Format: GETENV <name>

Template: GETENV ''NAME/A''

Purpose: To get the value of an environment variable.

Path: C:GETENV

Specification:

GETENV is used to get the value of an environment variable. Environment variables are stored in the ENV: handler. (Currently, this is simulated by using the RAM: disk.)

You can use many AmigaDOS commands with the ENV: handler, such as DIR, LIST, ASSIGN, etc. This gives users great flexibility in working with environment variables.

See also: SETENV

ICONX

Format:　ICONX

Template:　ICONX

Purpose:　To execute an AmigaDOS script file from Workbench.

Path:　C:ICONX

Specification:

ICONX allows you to execute a script file of AmigaDOS commands from the Workbench. ICONX changes the current directory to the directory containing the project icon before executing the script. An input/output window for the script file is opened on the Workbench screen. You can use the icon's TOOL TYPES to specify the size of the window (WIN-DOW =). The DELAY = TOOL TYPES will add a short delay after the execution of the file is complete to allow time for reading the output. If a 0 is specified for DELAY = , ICONX waits for a Control-C before exiting.

To use ICONX, create a script file (an ASCII file containing AmigaDOS commands), and attach a project icon to it. (Notepad can be used for this purpose, as long as you do not use styles and multiple fonts in your document). Using the Workbench INFO menu item, change the default tool of the icon to c:ICONX. Add the WINDOW = and DELAY = TOOL TYPES specifications if you choose. Then save the changed info. To execute the script, double–click on the icon.

You can work through the Workbench and use extended selection to pass Workbench files to the script. These files appear to the script as keywords. To use this facility, the .key keyword must be at the start of the script. In this case, the AmigaDOS EXECUTE command is used to execute the script file. Adding a .key ' " ' or .<space> allows the use of commands that require input in an ICONX script. (See *The AmigaDOS Manual* for more information on executing script files.)

See Also:　EXECUTE

IF

Format:　IF [NOT] [WARN] [ERROR] [FAIL] [<string> EQ|GT|GE
　　　　　<string>] [VAL] [EXISTS <file>]

Template:　IF ''NOT/S,WARN/S,ERROR/S,FAIL/S,,EQ/K,GT/K, GE/K,
　　　　　VAL/S, EXISTS/K''

Purpose:　To handle conditional operations in script files.

Path:　C:IF

Specification:

In an EXECUTE script file, IF carries out all subsequent commands until an ENDIF or ELSE command is found.

IF now supports the GT (greater than) and GE (greater than or equal to) comparisons. Normally, the comparisons are performed as string com-

parisons. However, if the VAL option is specified, the comparison is a numeric comparison.

The IF command can evaluate environment variables. To specify an environment variable, the variable's name is prefaced with a $ character.

NOTE: You can use NOT GE for LT and NOT GT for LE.

For more information on the IF command, see *The AmigaDOS Manual*.

INFO

Format: INFO [<device>]

Template: INFO ''DEVICE''

Path: C:INFO

Purpose: To give information about the file system(s).

Specification:

INFO's output now adjusts to allow longer volume names. INFO reads reserved blocks from the environment vector of the specified device.

The DEVICE option allows INFO to provide information on just one device or volume.

INSTALL

Format: INSTALL DRIVE <DF0|DF1|DF2|DF3>: [NOBOOT] [CHECK]

Template: INSTALL ''DRIVE/A,NOBOOT/S,CHECK/S''

Path: C:INSTALL

Purpose: To handle the boot block of a formatted disk.

Specification:

INSTALL now clears memory used to build boot block. The NOBOOT option makes the disk a DOS disk, but not bootable. The CHECK option checks against valid boot code. INSTALL reports if a disk is bootable, or not, and whether standard Commodore-Amiga boot code is present on the disk. The condition flag is set to 0 if the boot code is standard (or the disk isn't bootable), 5 (=WARN) otherwise.

NOTE: INSTALL's NOBOOT option will write the boot block on a non-DOS disk.

JOIN

Format: JOIN <name> <name>* AS|TO <name>

Template: JOIN ''NAME(S),TO = AS/K''

Purpose: To concatenate up to 15 files to make a new file.

Path: C:JOIN

Specification:

TO is now a synonym for AS.

LIST

Format: LIST [<dir|pattern>] [P|PAT <pattern>] [KEYS]
[DATES] [NODATES] [TO <name>] [SUB <substring>]
[SINCE <date>] [UPTO <date>] [QUICK] [BLOCK]
[NOHEAD] [FILES] [DIRS] [LFORMAT <string>]

Template: LIST ''DIR,P = PAT/K,KEYS/S,DATES/S,NODATES/S,
TO/K, SUB/K, SINCE/K, UPTO/K, QUICK/S, BLOCK/S,
NOHEAD/S, FILES/S, DIRS/S, LFORMAT/K''

Purpose: To list specified information about directories and files.

Path: C:LIST

Specification:

LIST now displays the new protection bits: s (script), p (pure — used with the RESIDENT command), and a (archive).

When searching for a pattern, you can specify a search string within a directory. For example, to search for a pattern within the C: directory, you could type LIST C:l#?. This would result in a display of all the files in the C: directory that start with the letter l.

The LIST options are as follows:

QUICK   extra spaces are not output after each filename

BLOCK   LIST displays file sizes in blocks, rather than bytes

NOHEAD   suppresses the printing of the header information

FILES   limits LIST to displaying files only (no directories)

DIRS   limits LIST to displaying directories only (no files)

LFORMAT modifies the output of LIST and can be used as a quick method of generating script files. When LFORMAT is specified, the QUICK and NOHEAD options are automatically selected. When using LFORMAT you must specify an ''output format specification'' string; this string is incorporated into the script file. Also, if you want the output to be saved, you must redirect it to a file by using the > command.

The format for the output format specification string (hereafter referred to as ''string'') is LFORMAT = ''string''. To include the output of LIST in this string, you can specify a %S substitution. The path and filename can also be part of this string. The number of occurrences of the %S determines the type of output to be included in the string, as follows.

Number of %S	Output
1	filename only
2	path, filename
3	path, filename, path
4	path, filename, path, filename

Example:

```
1> LIST >RAM:qwe #? LFORMAT="protect %S -d"
```

A new script file ''qwe'' is created in RAM:. The contents will include a list of all the files in the current directory. When "qwe" is executed, it will remove the delete protection from each file. For instance, the contents may look like this:

```
protect Expansion.info -d
protect Trashcan -d
protect .info -d
protect c -d
protect Clock.info -d
protect Prefs -d
protect Clock -d
protect System -d
protect L -d
protect Shell -d
protect devs -d
protect S -d
protect Shell.info -d
protect T -d
protect fonts -d
protect libs -d
protect Empty -d
protect Utilities.info -d
protect Disk.info -d
protect Prefs.info -d
protect System.info -d
protect Empty.info -d
protect Trashcan.info -d
protect Utilities -d
protect Expansion -d
```

LOADWB

Format: LOADWB [DELAY] [-DEBUG]

Template: LOADWB ''DELAY/S,-DEBUG/S''

Purpose: To start Workbench.

Path: C:LOADWB

Specification:

The LOADWB command is used to start the Workbench. Normally, this is only done when booting, by placing the LOADWB command in the startup-sequence file.

Workbench snapshots the current paths in effect when the LOADWB command is executed. It uses these paths for each CLI (or SHELL) started from Workbench.

If the DELAY option is specified, LOADWB waits three seconds before exiting. This allows disk activity to stop before execution continues.

The -DEBUG option tells Workbench to bring up a hidden Workbench menu — DEBUG. This menu contains the menu items Debug and FlushLibs. The Debug item breaks you into ROMWACK, the Amiga's ROM-resident debugger, so you must have a 9600 baud serial terminal attached if you intend to use this command! FlushLibs causes Workbench to attempt to allocate as much memory as possible, so that any libraries, devices, fonts, etc., resident in memory but not currently in use will be expunged. (Assuming that the particular library or device allows itself to be expunged.)

Both the DELAY and -DEBUG options cannot be used at the same time. You can only choose one or the other.

LOCK

Format: LOCK <drive>: [ON|OFF] [<passkey>]

Template: LOCK ''DRIVE/A,ON/S,OFF/S,PASSKEY''

Purpose: To set the write protect status of a hard disk using the FastFileSystem.

Path: C:LOCK

Specification:

The LOCK command sets or unsets the write protect status of a hard disk drive or partition mounted using the FastFileSystem. The LOCK remains on until the system is rebooted or until the LOCK is turned off with the LOCK OFF command. An optional 4 character passkey may be specified. If the passkey is used to lock a hard disk partition, the same passkey must be specified to unlock the partition.

See Also: The section on the FastFileSystem in Appendix A.

MOUNT

Format: MOUNT <device> [FROM <file>]

Template: MOUNT ''DEVICE/A,FROM/K''

Purpose: To create an AmigaDOS device node.

Path: C:MOUNT

Specification:

MOUNT is used to create an AmigaDOS device node. In other words, MOUNT informs the Amiga that an additional device has been added to the system. When the MOUNT command is issued, MOUNT looks in the DEVS:MountList file (or the optional FROM file) for the parameters of the device that is being mounted.

See Also: The section on MountLists in Chapter 3.

NEWCLI

Format: NEWCLI [<window specification>] [FROM <filename>]

Template: NEWCLI ''WINDOW,FROM/K''

Purpose: To start a new interactive CLI with a new window.

Path: C:NEWCLI

Specification:

NEWCLI uses the default startup file, s:CLI-Startup unless another file is specified with the FROM option.

If you want to invoke a SHELL, instead of a CLI, use the NEWSHELL command.

NEWSHELL

Format: NEWSHELL [<window specification>] [FROM <filename>]

Template: NEWSHELL ''WINDOW,FROM/K''

Path: C:NEWSHELL

Purpose: To start a new interactive SHELL with a NEWCON window.

Specification:

NEWSHELL invokes a new, interactive SHELL, using the NEWCON: window handler. For NEWSHELL to work properly, NEWCON: must be mounted first and the SHELL-SEG must be made resident. (This is normally done in 1.3's default startup-sequence.)

If the SHELL-SEG has not been made resident, NEWSHELL will invoke a new CLI window instead of a new SHELL window. If NEWCON: has not been mounted, NEWSHELL will use a CON: window.

NEWSHELL uses the default startup file s:Shell-Startup, unless a FROM filename is specified.

PATH

Format: PATH [SHOW] [ADD <directory>*] [RESET] [QUIET]

Template: PATH ''SHOW/S,ADD,RESET/S,QUIET/S''

Purpose: To control the PATH list that the CLI uses to search for commands.

Path: C:PATH

Specification:

The PATH command lets you see, add, or change the search path that AmigaDOS follows when looking for a program to execute. Enter the PATH command alone, or with the SHOW option, and the current path will be displayed.

Use PATH with the ADD option to specify directory names to be added to the current PATH. You can add up to ten directories with one PATH ADD command; names of the directories must be separated by at least one space, and the ADD keyword is optional.

To replace the existing search path with a completely new one, use PATH RESET, followed by the names of the directories. The existing search path, except for the current directory and SYS:C, is erased and the new one is substituted.

When the QUIET flag is specified, PATH searches the paths quietly (when SHOWing) — without displaying requesters for unmounted volumes. If a volume is unmounted, only the volume name will be displayed; PATH will not list the directories.

PROMPT

Format: PROMPT <prompt>

Template: PROMPT ''PROMPT''

Path: C:PROMPT

Purpose: To change the prompt string of the current SHELL.

Specification:

Prompt can print the current directory automatically when used with the SHELL. For example:

    %N>   displays only the CLI number
    %N.%S>   displays the CLI number, a period, then the current directory
    %S.%N>   displays the current directory, a period, then the CLI number

See Also: The SHELL section of Chapter 1, and the SHELL-SEG section of Chapter 3.

PROTECT

Format: PROTECT [FILE] <filename> [FLAGS] < + - status bits>
[ADD] [SUB]

Template: PROTECT ''FILE/A,FLAGS,ADD/S,SUB/S''

Purpose: To change the protection bits of a file.

Path: PROTECT

Specification:

PROTECT can now change the new protection bits, script (s), pure (p), and archive (a). In addition, PROTECT can be used to add or subtract protection bits to or from a file, rather than just setting specific bits. The keywords ADD and SUB, as well as the special flags + and −, are used for this purpose. Use the LIST command to display the protection bits of files.

Examples:

1> PROTECT c:ED + rw

adds the protection bits r (readable) and w (writable) to the file ''ED'' in the C: directory.

1> PROTECT l:filename −e

deletes the e (executable) protection bit from the file ''filename'' in the L: directory.

See Also: LIST

REMRAD

Format: REMRAD

Template: REMRAD

Purpose: To remove the recoverable ramdrive.device.

Path: C:REMRAD

Specification:

If you want to remove the recoverable ramdrive.device from the system, but you do not want to turn the system's power off, you can use the REMRAD command. REMRAD commands the ramdrive.device to delete all of its files. The device also decreases its size and becomes very small. The next time the Amiga is rebooted, the ramdrive.device is removed from the system.

RESIDENT

Format: RESIDENT <Resident Name> <File> [REMOVE] [ADD]
                   [REPLACE] [PURE] [SYSTEM]

Template: RESIDENT "NAME,FILE,REMOVE/S,ADD/S,REPLACE/S,
                   PURE/S,SYSTEM/S''

Purpose: To load and add commands to the resident list.

Path: C:RESIDENT

Specification:

RESIDENT is used to load commands and add them to the resident list
maintained by the SHELL. This allows the command to be executed
without it having to be reloaded each time, resulting in faster execution
and reduced memory usage when multitasking. This is only available
when using the SHELL.

Only certain commands can be made resident. The command should be
both reentrant, meaning that it can be used by two or more independent
programs at the same time, and re-executable. Commands that have
these characteristics have the pure bit set in the protection field. Many of
the commands in the C: directory, as well as the MORE command in
Utilities, are PURE commands and can be made resident. If a command
does not have its pure bit, it probably can't be made resident safely.

The REPLACE option is the default option and does not need to be
explicitly stated. If RESIDENT is invoked with no options, or with the
REPLACE option, it lists the programs on the resident list. If a <resident
name> is specified and RESIDENT finds a program with that name
already on the list, it will attempt to replace the command. The replace-
ment will succeed only if the already resident command is not in use. If no
<resident name> is specified (i.e., just a filename is specified) RESI-
DENT will use just the filename portion as the name on the resident list.

Note: The full pathname to the file must be used.

If the SYSTEM option is specified, the command will be added to the
system portion of the resident list. Any commands added to the resident
list with the SYSTEM option cannot be removed. To list SYSTEM files on
the RESIDENT list, you must specify the SYSTEM option.

The PURE option forces RESIDENT to load commands which are not
marked as pure (i.e., they do not have their pure bit set), and can be used
to experimentally test the pureness of other commands and programs.
PLEASE USE THIS OPTION WITH CAUTION. Remember that in order
for a command to be made RESIDENT, it must be both reentrant and re-
executable. Although it is unlikely, some of your programs may be pure
enough to be fully reentrant and usable by more than one process at the
same time. Other programs may not be fully reentrant but may be pure
enough to be re-executable. Such commands can be made RESIDENT,
but you must be extremely careful to only use the command in one
process at a time.

If you are going to experiment with the PURE option, be sure to boot with a standard, write-protected Workbench disk, one that has never been altered in any way. Do not use directories on or assign directories to a hard disk. Open a SHELL window and CD to RAM: or RAD:. While you are experimenting, make sure that any files you read or write are in RAM:, RAD: or on a backup or freshly formatted floppy disk. Then make the program you are testing RESIDENT, by typing:

> 1> RESIDENT filename PURE

To test if a program is re-executable, run the program from the SHELL, use as many of the program's functions as you can, then exit the program. Run the program a second time, calling it up with different command line options, if possible. It is possible that the program will crash at this point, meaning that it is definitely not re-executable. If the program works, examine all of the things that the program usually remembers during a session, for instance, command line options, checked menu selections, search strings, names of last loaded or saved files, etc. If the program remembers things from the previous session, it is not re-executable. If the program settings are fresh, keep trying as many options as you can to make sure that they still work properly. If the program appears to be working well, it can probably be used as a resident command by **one process at a time.**

To test if a program is fully reentrant, open a **second** SHELL, CD to RAM:, and try running the resident program from both SHELLS **at the same time**. Test all the program options, always checking both running programs to make sure that the options selected in one SHELL are not affecting the program in the other SHELL. If the program loads or saves files, thoroughly test the load/save options in both programs at the same time. Check the loaded/saved files to make sure that they are not incomplete or corrupt. At this point you may feel confident that you can safely use the program as a resident command. But, if you do, be alert for file problems and improper behavior.

See Also: PROTECT and the SHELL-SEG section of Chapter 3.

RUN

Format: RUN <command> [+ <comment>]

Template: RUN ''COMMAND''

Purpose: To execute commands as background processes.

Path: C:RUN

Specification:

RUN may now be used to launch background processes which do not prevent the closing of the CLI in which the process was started. To do this, redirect the output of RUN to NIL: by using the > symbol. However, if for some reason the program you are running gets a lock on ''*'' (the current window), you still will not be able to close that window.

Commands that are RUN may come from the commands stored on the resident list. For speed, resident commands are checked before commands in the command path.

When used to start a new CLI, RUN sets things up so the new CLI will use a default startup file: s:Shell-Startup, if the Shell is in operation, or s:CLI-Startup, if the Shell is not operative.

Example:

```
1> RUN >NIL: sample program
```

redirects the output of the sample program to the NIL: device.

SEARCH

Format: SEARCH FROM <name><pattern> [SEARCH] <string>
[ALL] [NONUM] [QUIET] [QUICK] [FILE]

Template: SEARCH ''FROM,SEARCH/A,ALL/S,NONUM/S,
QUIET/S,QUICK/S,FILE/S''

Purpose: To look for the specified text string in the specified files.

Path: C:SEARCH

Specification:

Search now returns a 0 if the object is found, and a 5 (=WARN) otherwise. This makes it more useful in scripts. Search stops searching when a Control-C is typed.

The options are as follows:

NONUM   SEARCH does not print line numbers with the strings

QUIET   SEARCH searches ''quietly''; no output is displayed

QUICK   SEARCH uses a more compact output format

FILE   SEARCH will look for a file by the specified name, rather than for a string in the file

NOTE: The 31 character restriction on the use of wildcards has been removed.

SETCLOCK

Format: SETCLOCK LOAD|SAVE|RESET

Template: SETCLOCK ''LOAD/S,SAVE/S,RESET/S''

Purpose: To set or read the real time clock.

Path: C:SETCLOCK

Specification:

This command is used to set the time and date of the real time clock from the current system time (the SAVE option), or to set the current system time from the real time clock (the LOAD option). When the SAVE option in Preferences is used, Preferences sets both the real time clock and the current system time. The RESET option is used to completely reset the clock. This may be necessary if a wild program turns the clock off or sets the test bit of the clock.

SETDATE

Format: SETDATE <file> <date> <time>

Template: SETDATE ''FILE/A,DATE,TIME''

Purpose: To change the date and time of a file or directory.

Path: C:SETDATE

Specification:

The input for SETDATE has been relaxed, so that the output of the DATE command may be used as input. SETDATE also does not need a leading 0 when specifying the date. SETDATE <file> does a 'touch'—it changes the date/time of the file to the current system date/time.

See Also: DATE

SETENV

Format: SETENV <variable name> <variable string>

Template: SETENV ''NAME/A,STRING''

Purpose: To set an environment variable.

Path: C:SETENV

Specification:

SETENV is used to set or remove the value of an environment variable. Environment variables are stored in the ENV: handler (currently, this is the RAM: disk).

To remove an environment definition, use SETENV <variable name>. The variable will remain in ENV:, but it will be empty.

Examples:

```
1> SETENV Editor "Extras 1.3:Tools/MEMACS"
```

creates the environment variable "EDITOR" which can be used with the MORE utility. This specifies the editor as being MEMACS which is located in the Tools drawer of the Extras disk.

```
1> SETENV Editor C:ED
```

same as above, only the editor specified is the AmigaDOS Editor ED.

See Also: GETENV and the Section on the ENV: handler in Chapter 1.

SETPATCH

Format: SETPATCH [R]

Template: SETPATCH "R/S"

Purpose: To make ROM patches in Version 1.2 and Version 1.3 Kickstarts.

Path: C:SETPATCH

Specification:

Both Version 1.2 and Version 1.3 Kickstart ROMs have a few known bugs that are fixed by running SETPATCH. SETPATCH must be run in the first line of the startup-sequence file, as is done in the startup-sequence file of Version 1.3 Workbench. SETPATCH patches Display-Alert(), the 68000 math exception vectors, the graphic library Delete-Layers() routine, and the AllocEntry() Exec function.

The R option is used to protect the recoverable ramdrive.device on systems with 1 megabyte of CHIP RAM. This is necessary since Version 1.3 Kickstart is designed to work with 512K of CHIP RAM.

See Also: Chapter 5, Changing Your Startup Sequences

SKIP

Format: SKIP <label> [BACK]

Template: SKIP "LABEL,BACK/S"

Purpose: To skip ahead when executing command files.

Path: C:SKIP

Specification:

When the BACK option is used, SKIP starts searching for the specified label at the beginning of the file. Without the BACK option, SKIP starts the search at the current line of the file. This allows backwards SKIPs. You can only SKIP back as far as the last EXECUTE statement. If there are no EXECUTE statements in a script, you can SKIP back to the beginning of the file.

STATUS

Format: STATUS <process> [FULL] [TCB] [CLI|ALL] [COM-MAND]

Template: STATUS "PROCESS,FULL/S,TCB/S,CLI=ALL/S,COM-MAND/K"

Purpose: To list information about the CLI/SHELL processes.

Path: C:STATUS

Specification:

STATUS now supports negative priorities. With the new COMMAND option, you can tell STATUS to search for a command. STATUS then scans the CLI list looking for that command. If the command is found, the CLI number is printed, and the condition flag is set to 0. Otherwise the flag is set to 5 (=WARN).

This is useful in script files. For example, to send a BREAK to the process executing the ClockPtr command, the sequence is:

```
1> STATUS >ram:qwe COMMAND=ClockPtr
   BREAK <ram:qwe >NIL: ?
```

See Also: BREAK

TYPE

Format: TYPE <from> [TO <name>] [OPT H=HEX| N=NUMBER]

Template: TYPE "FROM/A,TO/S,OPT/K,HEX/S,NUMBER/S"

Purpose: To type a text file.

Path: C:TYPE

Specification:

TYPE now checks to see if a destination file exists and complains if it does, unless the TO option is specified explicitly.

The OPT H and OPT N options are now available by the HEX and NUMBER keywords, respectively.

VERSION

Format: VERSION <library name|device name> [<version #>]
             [<revision #>] [<unit #>]

Template: VERSION ''NAME,VERSION,REVISION,UNIT''

Purpose: To find the version and revision numbers of a Workbench
             disk.

Path: C:VERSION

Specification:

VERSION is used to find the version and revision number of a library,
device, or the Workbench disk. VERSION can also test for a specific ver-
sion/revision, and set the condition flag if the version/revision is greater.

VERSION with no <library name|device name> argument prints the
Kickstart version number and the Workbench version number. If a <li-
brary name|device name> is specified, version attempts to open the
library or device and read the version information.

When a <version #> (and possibly a <revision #>) is specified, VER-
SION will set the condition flag to 0 if the version (and revision) numbers
of the Workbench disk, library, or device are greater than or equal to the
specified values. Otherwise, the flag is set to 5 (=WARN). (If a revision
number is not specified, no comparison on the revision number is
performed.)

In the case of using VERSION on the Workbench disk itself, the first value
is the Kickstart version; the second is the Workbench version. You cannot
use the VERSION command with a path name to check on the version of
a device.

The <unit #> option allows you to specify a unit number other than 0.
This is occasionally necessary for accessing multi-unit devices.

See Also: IF, EXECUTE

WAIT

Format: WAIT <n> [SEC|SECS] [MIN|MINS] [UNTIL <time>]

Template: WAIT '',SEC=SECS/S, MIN=MINS/S,UNTIL/K''

Purpose: To wait for the specified time.

Path: C:WAIT

Specification:

WAIT now allows a leading zero when specifying the wait period.

WHICH

Format: WHICH <filename> [NORES] [RES]

Template: WHICH ''FILE/A,NORES/S,RES/S''

Purpose: To search the command path for a particular command.

Path: C:WHICH

Specification:

WHICH lets you find a particular command by returning the location of that command. Normally, WHICH searches the resident list, the current directory, the command path(s), and the C: directory. If the file is not found, the condition flag is set to WARN.

If the NORES option is specified, the resident list will not be searched. If the RES option is specified, only the resident list will be searched.

Examples:

```
1> WHICH avail
C: avail
1>WHICH C:
RAMWB:C
```

Formats of Unchanged Commands

BREAK

Format: BREAK <process> [ALL|C|D|E|F]

Template: BREAK ''PROCESS/A,ALL/S,C/S,D/S,E/S,F/S''

Purpose: To set attention flags in the specified process.

Path: C:BREAK

CD

Format: CD <directory>

Template: CD ''DIR''

Purpose: To set, change, or list the current directory.

Path: C:CD

ED

Format: ED [FROM] <filename> [SIZE <n>]

Template: ED ''FROM/A,SIZE''

Purpose: To edit text files (a screen editor).

Path: C:ED

EDIT

Format: EDIT [FROM] <filename> [[TO] <filename>] [WITH <filename>] [VER <filename>] [OPT <option>]

Template: EDIT ''FROM/A,TO,WITH/K,VER/K,OPT/K''

Purpose: To edit text files by processing the source file sequentially (a line editor).

Path: C:EDIT

ELSE

Format: ELSE

Template: ELSE

Purpose: To have an alternative in a conditional in a script file.

Path: C:ELSE

ENDIF

Format: ENDIF

Template: ENDIF

Purpose: To terminate an IF block in a script file.

Path: C:ENDIF

FAILAT

Format: FAILAT <n>

Template: FAILAT ''RCLIM''

Purpose: To set the failure condition of a script file.

Path: C:FAILAT

FAULT

Format: FAULT <error number>

Template: FAULT '',,,,,,,,,,''

Purpose: To print the specified error messages(s).

Path: C:FAULT

LAB

Format: LAB <string>

Template: LAB ''STRING''

Purpose: To specify a label in a script file.

Path: C:LAB

MAKEDIR

Format: MAKEDIR <dir>

Template: MAKEDIR ''DIR/A''

Purpose: To create a new directory.

Path: C:MAKEDIR

QUIT

Format: QUIT <return code>

Template: QUIT ''RC'

Purpose: To exit from a script file with a specified error code.

Path: C:QUIT

RELABEL

Format: RELABEL DRIVE <drive>: NAME <name>

Template: RELABEL ''DRIVE/A,NAME/A''

Purpose: To change the volume name of a disk.

Path: C:RELABEL

RENAME

Format: RENAME [FROM] <name> [TO|AS] <name>

Template: RENAME "FROM/A,TO=AS/A"

Purpose: To change the name of a file or directory.

Path: C:RENAME

SORT

Format: SORT [FROM] <filename> [TO] <filename> [COL-START <n>]

Template: SORT "FROM/A,TO/A,COLSTART/K"

Purpose: To alphabetically sort each line of a small file.

Path: C:SORT

STACK

Format: STACK [<n>]

Template: STACK "SIZE"

Purpose: To display or set the stack size of the current CLI.

Path: C:STACK

WHY

Format: WHY

Template: WHY

Purpose: To print an error message to explain why the previous command failed.

Path: C:WHY

3. OTHER WORKBENCH DIRECTORIES

In addition to the AmigaDOS commands explained in Chapter 2, there are many other programs and files included on the Workbench disk to help you maximize your computing and programming efforts. For instance, there are script files in the S: directory that let you add wildcard capabilities to many of the AmigaDOS commands. The MountList file in the DEVS: directory informs the Amiga of the characteristics of peripherals and expansion devices you may add to your system.

This chapter covers files included in the Workbench's DEVS:, L:, S: and LIBS: directories. Some of the files are new, while others have been revised with the release of Version 1.3. Some prior knowledge of the Amiga is assumed. Novice users may want to consult additional reference materials in order to fully comprehend these tools. Seasoned users of the Amiga will want to read this chapter thoroughly for both new and revised information.

Devices

The DEVS: directory contains .device files, such as the ramdrive.device, the printer.device, the serial.device, etc. Many of these files correspond to actual physical devices, such as peripherals attached to the Amiga's ports (i.e., serial.device controls communications through the serial port while the parallel.device controls the flow of information through the parallel port, usually to a printer). This section does not explore all of these files. They are covered in other reference works, such as the *ROM Kernel Manuals* published by Addison-Wesley. This section does cover the MountList, which many users need to become familiar with when they install expansion devices in their Amiga, and a new assignment called CLIPS: which is used by the clipboard.device.

The MountList

Found in the DEVS: directory, the MountList contains the descriptions of devices that are to be mounted with the AmigaDOS MOUNT command. You may need a MountList entry for a device, handler, or file system. When you add a new device to your Amiga system, such as a hard disk or even some external disk drives, you must make the Amiga aware of the existence of the device. To do this you use the MOUNT command. But the MOUNT command must read a MountList entry in order to determine the characteristics of the device.

Several sample MountList entries are already included in the MountList file in the DEVS: directory. Some of these can be used as they are, but it is always a good idea to double-check the file to make sure it accurately corresponds with your device.

A MountList entry consists of a number of keywords describing the device, handler, or file system, as well as values for those keywords. Some keywords may only apply to a file system or a handler. If a keyword is omitted, a default value is used. You should always check the default value in case it is not appropriate for whatever you are mounting.

There are certain rules for creating a MountList entry:

- Each entry in the MountList must start with the name of the device.
- Keywords are followed by an equals sign (=).
- Keywords must be separated by a semi-colon or by placing them on their own, individual line.
- Comments are allowed in standard C style (i.e., comments start with /* and end with */).
- Each entry must end with the # symbol, on a line of its own.

The Keywords supported by MOUNT are as follows:

KEYWORD	FUNCTION
Handler =	A handler entry (i.e., Handler = L:Newcon-Handler)
FileSystem =	A file system entry (i.e., FileSystem = L:Fast-FileSystem)
Device =	A device entry (i.e., Device = ramdrive.device)
Priority =	The priority of the process; 5 is good for handlers, 10 for file systems
Unit =	The unit number of the device
Flags =	Flags for OpenDevice (usually 0)
Surfaces =	The number of surfaces
BlocksPerTrack =	The number of blocks per track
Reserved =	The number of blocks for the boot block; should be 2
PreAlloc =	The number of blocks reserved from the end of a partition; used with a few hard drives that store information in the last few blocks of a drive. This is usually set to 0 and probably will not need to be changed. Please refer to the documentation packaged with your hard drive and hard drive controller.
Interleave =	Interleave value; varies with the device
LowCyl =	Starting cylinder to use
HighCyl =	Ending cylinder to use
Stacksize =	Amount of stack to allocate to the process
Buffers =	Number of initial cache buffers
BufMemType =	Memory type used for buffers; (0 and 1 = Any, 2 and 3 = CHIP, 4 and 5 = FAST)
Mount =	If a positive value, MOUNT loads the device or handler immediately rather than waiting for first access
MaxTransfer =	The maximum number of blocks transferred; used with the FastFileSystem
Mask =	Address Mask to specify memory range that DMA transfers can use; used with the FastFileSystem
GlobVec =	A global vector for the process; -1 is no Global Vector (for C and assembler programs), 0 sets up a private GV; if the keyword is absent, the shared Global Vector is used

Startup =	A string passed to the device, handler, or file system on startup as a BPTR to a BSTR
BootPri =	A value which sets the boot priority of a bootable and mountable device, such as the recoverable ramdrive.device. This value can range from -129 to 127. By convention, -129 indicates that the device is not bootable and is not automatically mounted; this value should be included for a recoverable ramdrive used with the FastFileSystem.
DosType =	Indicates the type of file system. If the FastFileSystem is used, DosType should be set to 0x444F5301. Otherwise, the DosType should be 0x444F5300. Or, you could simply omit it altogether. It is possible that other values may be used in the future.

Sample MountList entries are included in the MountList file. Usually if you need to create a new MountList, you will be given instructions in the documentation that accompanies the device you are mounting. There are also several MountList examples in this chapter accompanying the descriptions of the various handlers.

The Clipboard.device

One change that occurred with Version 1.3 which affects the clipboard .device is the advent of the CLIPS: assignment. CLIPS: allows you to specify an alternate directory for storing clips from the clipboard.device. The Workbench's startup-sequence assigns CLIPS: to RAM:clipboards; however, you can re-assign it if you wish.

Keymaps

Keymaps is a subdirectory of DEVS: (devs/keymaps). As mentioned in the section on SetMap in Chapter 1, the Workbench's keymaps directory only contains the usa1 keymap. International keymaps are available in the devs/keymaps directory of the Extras disk:

KEYMAP FILE	KEYBOARD IT REPRESENTS
cdn	French Canadian
ch1	Swiss French
ch2	Swiss German
d	German
dk	Danish
e	Spanish
f	French
gb	British
i	Italian
is	Icelandic
n	Norwegian
s	Swedish
usa2	Dvorak

To use an international keymap:

1) Copy the keymap file to the DEVS:keymaps directory. For example:

   COPY "Extras 1.3:devs/keymaps/d" to devs:keymaps

2) Use the SetMap program (in the System drawer) to inform the system of the change.

   SETMAP d

If you want to use a different keymap on a regular basis, copy the file to DEVS:keymaps and change the SetMap assignment in the Workbench's startup-sequence by using a text editor such as ED or MEMACS.

Fonts

The Workbench's FONTS: directory contains the standard Amiga fonts: Diamond, Emerald, Garnet, Opal, Ruby, Sapphire, and Topaz (the default font). A quick way to see the different fonts is to open a Notepad window, type a test sentence, and then go through the font menu selecting the different fonts. Each time you select a font, the text in the Notepad window will change.

The Extras disk also has a fonts directory which contains three new fonts: Courier, Helvetica and Times. To use the new fonts, you must either:

1) Copy one or all of them to your Workbench disk using the Amiga-DOS COPY command. For instance,

   COPY "Extras 1.3:fonts/Times" to "Workbench 1.3:fonts"

   After copying any fonts to your Workbench disk, you should use the FixFonts program (in the System drawer) to update the FONTS: directory. Or,

2) ASSIGN FONTS: to the fonts directory of the Extras disk:

   ASSIGN FONTS: "Extras 1.3:fonts"

Now if you were to open a Notepad window, you could choose Times (if you followed example 1) or from Helvetica, Courier and Times (if you followed the second example). These fonts can be used with various software programs, such as many word processors, which let you add or change fonts. When using these new fonts, please keep in mind that Helvetica and Times are both proportional fonts, while Courier is a fixed width font.

Handlers

Several new handlers are located in the Workbench's L: directory. Most handlers are treated as if they are actual physical devices and are referred to in the same format as a disk drive. For instance, RAM: signifies the RAM-Handler which emulates an actual disk. SPEAK: represents the Speak-Handler which provides speech output for the Amiga. Handlers even control the appearance of the windows on your display (CON: and NEWCON:).

The information in this section may be slightly repetitive of material covered elsewhere in this manual. For instance, the characteristics of the Newcon-Handler were mentioned in the SHELL section. However, for your convenience, full explanations of the new handlers will be given here.

Aux-Handler

The Aux-Handler provides unbuffered serial input and output. It is basically a console handler that uses the serial port rather than the Amiga screen and keyboard.

The MountList entry is:

```
AUX:    Handler = L:Aux-Handler
        Stacksize = 1000
        Priority = 5
#
```

A sample entry is already in the MountList file.

You can use Aux-Handler to use another terminal with your computer. For example:

```
1>MOUNT AUX:
1>NEWCLI AUX:
```

WARNING: The Aux-Handler allows you to issue multiple NEWSHELL/ NEWCLI commands. However, keyboard data from the aux unit will be distributed across the multiple CLIs. To end one of two SHELLs assigned to AUX:, you must type:

```
1>E N D C L I<space> <RETURN>
```

You must put a space between each of the letters and after the final "I".

If you want to copy a file over the serial port, type:

```
1>COPY AUX: to RAM:testfile
```

Newcon-Handler

The Newcon-Handler acts as a replacement for the CON: handler, and is referred to in use by the name NEWCON:. Like the other new handlers, it must be mounted before use. This is generally done by a command in the Startup-Sequence file. (Workbench Version 1.3 does include such a command in its startup sequence.)

The entry in the MountList file is:

```
NEWCON:   Handler = L:Newcon-Handler
          Priority = 5
          StackSize = 1000
#
```

A NEWCON: window allows users much more flexibility when entering commands. Unlike a CON: window which only permits ''straight'' typing, a NEWCON: window allows the user to manipulate the text. The

SHELL is an example of a NEWCON: window in action. A list of a NEWCON: window's capabilities follows.

1) Command line editing

The left and right cursor keys are used to move through the command line. The Backspace key deletes the character to the left of the cursor; the Delete key deletes the character underneath the cursor.

Some other editing commands are:

Control-K   Deletes everything from the cursor forward to the end of the line.

Control-U   Deletes everything from the cursor backward to the start of the line.

Control-X   Deletes the entire line.

Control-W   Moves the cursor to the next tab stop.

Shift-Cursor Left (or Control-A) brings the cursor to the start of the line. Shift-Cursor Right (or Control-Z) brings the cursor to the end of the line.

2) History buffer

NEWCON: has a 2K circular history buffer. Previous commands can be retrieved by using the up cursor key. (The down cursor key moves you down through the history buffer; this is useful if you go past the command you are looking for).

Shift-Cursor Up (or Control-R) searches back through the history buffer for the last command that matches a partially typed string. Shift-Cursor Down (or Control-B) brings you to the bottom of the history buffer.

3) Control characters

Control characters are displayed by NEWCON: as reversed characters. The characters are read as Control characters, and can be used in files just as before.

Pipe-Handler

PIPE: is an AmigaDOS handler written by Matt Dillon.

The Pipe-Handler is an I/O mechanism used to provide input/output communication between programs. It essentially creates an interprocess communication channel. When the pipe is written to, up to 4K bytes of data are buffered before the writing process is blocked. After you write to a PIPE:, other processes can read the data.

Like the other handlers, it must be mounted. The MountList entry for PIPE: is as follows:

```
PIPE:    Handler = L:Pipe-Handler
         Stacksize = 6000
         Priority = 5
         GlobVec = -1

    #
```

PIPE: may be used from other programs, like a word processor (as a filename during a save operation) or a terminal program (as a capture buffer filename). You can use any rendezvous name you wish. PIPE: uses a 4K internal buffer per name, but its optimal situation is one in which there is a pending read and a pending write. In this case it copies directly from one process to the other, rather than going through its internal buffer. This is a true PIPE:, therefore, the source and destination processes must be distinct (i.e. not the same process) in order to prevent lockout situations.

The buffer is transparent. This means that data written, no matter how little it is, is immediately available to be read by the other process.

The PIPE: device can be useful when you're using two application programs and want to transfer huge amounts of data from one (write) to the other (read) without using a temporary file in RAM: or on disk. Assuming the application does not attempt a Seek(), you simply specify 'PIPE:name' and it looks like an ordinary file to the application.

For those terminal programs which do not use asynchronous writes, you can fix the jerkiness in CAPTURE by capturing to a PIPE: and having another COPY command running from the PIPE: to a file.

You can also copy information from one PIPE: to another. For example:

CLI window 1:	COPY hugefile PIPE:a
CLI window 2:	COPY PIPE:a PIPE:b
CLI window 3:	COPY PIPE:b PIPE:c
CLI window 4:	COPY PIPE:c PIPE:d
CLI window 5:	COPY PIPE:d PIPE:e
CLI window 6:	wordcount PIPE:e (or something similar)

Speak-Handler

The Speak-Handler provides speech output for the Amiga. With SPEAK: you can literally have the Amiga ''read'' the contents of a file to you.

Just as most of the other handlers, SPEAK: must be mounted before use. The MountList entry is:

```
SPEAK:  Handler = L:Speak-Handler
        Stacksize = 4000
        Priority = 5
        GlobVec = -1
    #
```

In addition to the MountList entry, SPEAK: also requires the narrator.device and translator.library. They must be in DEVS: and LIBS: respectively. (These files are included on Workbench Version 1.3 in the appropriate directories.)

The format for using SPEAK: is:

```
SPEAK:OPT/K
```

After the OPT keyword, the following options may be used. The options must be separated by a slash (/) and there should be no space between the colon and OPT:

p###	pitch (where ### is from 65-320)
s###	speed (where ### is from 30-400)
m	male voice
f	female voice
r	robot voice
n	natural voice
o0	do not allow these options in the input stream
o1	allow these options in the input stream
a0	turn off direct phoneme mode
a1	turn on direct phoneme mode (do not use translator.library)
d0	break up sentences on punctuation alone
d1	break up sentences on punctuation, RETURN, and LINEFEED

SPEAK: may be used from other programs, like a word processor (as a filename during a save operation) or a terminal program (as a capture buffer filename) to get spoken output.

For example, to listen to the contents of your startup-sequence file, type:

1> MOUNT SPEAK:
1> COPY s:startup-sequence to SPEAK:OPT/f/s160

The contents of the startup-sequence will be read in a female voice at a moderate speed.

SHELL-SEG

SHELL-SEG is the program which controls the new SHELL. SHELL-SEG is not executed directly. Instead, the RESIDENT command is used to invoke it, under the name CLI. Once started, each new SHELL started either from the NEWSHELL command or the SHELL icon will be a SHELL CLI, rather than a normal CLI. Normally the SHELL-SEG is invoked in the startup sequence.

The SHELL provides many additional features that a standard CLI does not. Those features were explored in the SHELL section of Chapter 1, such as resident commands, aliases, the prompt features, and the use of scripts.

Scripts

There are some new script files in the Workbench's S: directory. These files can be used with other commands to expedite your programs:

SPAT — uses the LIST command to add pattern matching to **single** argument commands. This can be used with the SHELL's alias function. For example, to display all of the script files that begin with the letter ''s'' in the Workbench's S: directory, you could type:

1> SPAT MORE s:start#?

A script file would be generated, similar to the one below:

```
more "s:SPAT"
more "s:Startup-Sequence.hd"
more "s:Shell-Startup"
more "s:Startup-Sequence"
more "s:StartupII"
```

SPAT would then execute this script file and display each of the files consecutively.

DPAT — uses the LIST command to add pattern matching to **double** argument commands. This can be used with the SHELL's alias function. The following example assumes that you have a series of files called Chapter1, Chapter2, Chapter3, etc., in the root directory of the disk in df0:. If you wanted to rename these files to a new directory called Book, you'd type:

```
1> MAKEDIR df0:BOOK
1> DPAT RENAME df0:Chap#? df0:book
```

A script file is created to rename each of the files starting with Chap. DPAT will then execute that file and rename each of the Chap files to the new BOOK directory.

PCD — similar to the AmigaDOS CD command, but it remembers the last directory. For example, typing:

```
1>PCD RAM:
1>PCD
```

will return you to the starting directory.

Libraries

There are new IEEE libraries included in the LIBS: directory of the Version 1.3 Workbench disk: mathieeedoubtrans.library and mathieee-doubbas.library. These new libraries can take advantage of a 68881 coprocessor, if available. They can also use a peripheral math chip; however, the chip must be supplied with the appropriate software to inform the system of the chip's presence and to handle the saving and restoring of the chip's status during task switching.

4. PRINTER DRIVERS

Version 1.3 supports more printer drivers than Version 1.2; however, all the drivers, except Generic, have been put on the Extras disk. This leaves more room on the Workbench disk, while still supplying the Amiga user with a wide range of alternatives when it comes to choosing a printer.

A printer driver acts as a translator for the computer. The Amiga has one way of encoding information, but a printer may have a different way of deciphering that code. The printer driver takes the information from the Amiga and reworks it so that the printer interprets it correctly.

Printer drivers are located in the devs/printers directory of the Extras disk. To move a driver to your Workbench disk, either use the InstallPrinter utility (explained in Chapter 1) or the AmigaDOS COPY command as illustrated below:

```
COPY "Extras 1.3:devs/printers/<name>" to
devs:printers
```

After the file is copied, the driver will now appear in the list of supported drivers that is displayed in the Change Printer window in Preferences.

The various printer drivers included with Version 1.3 are:

Alphacom AlphaPro 101	EpsonXOld	Imagewriter II
Brother HR-15XL	Generic*	NEC Pinwriter
CalComp ColorMaster	Howtek Pixelmaster	Okidata 92
CalComp ColorMaster2	HP DeskJet	Okidata 293I
Canon PJ-1080A	HP LaserJet (includes	Okimate 20
CBM MPS1000	HP LaserJet Plus and	Quadram QuadJet
Diablo 630	HP LaserJet II	Qume LetterPro 20
Diablo Advantage D25	compatible)	Toshiba P351C
Diablo C-150	HP PaintJet	Toshiba P351SX
Epson Q	HP ThinkJet	Xerox 4020
Epson X		

*Included on the Workbench disk.

Specifications about each of these drivers are listed later in this chapter. Also included is information on the Seiko 5300 and 5300a and the Tektronix 4693D and 4696 printer drivers. While these drivers are not included on disk, they are compatible with the Amiga and are available from the manufacturer or in the listings topic of the Amiga Developers (amiga.dev) Conference on BIX, Byte's Electronic Information Exchange.

Before listing the drivers, some of the general information applicable to graphic printing is explained. Before proceeding with this section of the manual, please read through the section on Preferences, in Chapter 1, for details on color correction, smoothing, multiple densities, and changing the size of the printout.

Graphic Printing

The Amiga is world-famous for its graphic capabilities. However, if you can't get the graphics off the screen and onto paper, you are limiting your applications. Whether it's a company newsletter, engineering drawings, or simply a home-made greeting card, if you can't print out what's on the screen, you can't distribute your work. The Amiga is capable of supporting several printers — from a basic daisy-wheel printer that only prints black-and-white text to a sophisticated, thermal-transfer color printer.

Speed

The drivers on Version 1.3 are approximately five to 30 times faster than with Version 1.2. The speed increase is dependent on the type of printer you are using, the number of colors you are printing (black-and-white, gray-scale or color), the amount of white-space in the picture, the aspect of the picture (vertical or horizontal), and the actual printer driver. Printing speed is now more closely related to the printer or the software program as opposed to the computer itself.

Color Printing

HAM Pictures — A HAM (hold-and-modify) picture is one where anywhere from 32 colors to the Amiga's full-range of 4,096 colors are displayed. Previously, you could only print a HAM picture starting at an x position of 0 (corresponding to the top, left corner of the paper). Now, you can start printing at any valid x position. It is also possible to print inverted HAM pictures.

Below are some things to keep in mind when performing screen dumps.

For better *quality* screen dumps:

- On most printers, friction fed paper tends to produce better looking graphic dumps than tractor fed paper. You wind up with less horizontal banding.
- Densities which use more than one pass should only be used for black-and-white screen dumps. (Density is controlled by the Density setting in Preferences. For the densities supported by your printer, see the list of printer drivers later in this chapter.) If you use a multiple-pass density for a gray-scale or color-shade dump, the output may be muddy or dark. Multiple-pass color dumps also dirty the printer ribbon (i.e., the yellow will become contaminated with other colors on the ribbon.)

For *faster* screen dumps:
- The lower the density the faster the print out.
- Horizontal dumps are much faster than vertical dumps.
- If you are dumping a one bitplane image (two colors), select the black-and-white mode in Preferences. This is much faster than a gray-scale or a color dump.
- Turning on the Smoothing option doubles the printing time. Use the Smoothing option for the final copy only.

- Be careful which dithering mode you choose. F-S dithering doubles the printing time, while Ordered and Half-tone dithering cause no increase in printing time.
- If you are dumping a screen that in low-resolution mode (320 x 200 or 400 pixels) displays more than 16 colors (4 bitplanes) or in high-resolution mode (640 x 200 or 400 pixels) displays more than 4 colors (2 bitplanes), you can speed up the dump by moving the screen to the back of the display ONCE PRINTING HAS STARTED. This is easily done by pressing the left Amiga key and the N key simultaneously.

Supported Printers

Below is a list of the printer drivers available for use with the Amiga computers. Most of these drivers are included on the Extras disk. The few drivers that are not on Extras are available from the printer manufacturer or from the Amiga Developers (amiga.dev) Conference on BIX.

You'll notice that many of the drivers support several densities. Density refers to the number of dots per inch that are used to make the print. The higher the number of dots, the smaller the dots will be and the clearer the picture. However, the higher the density, the slower the printing. So, in the case of multiple densities, it is up to the individual user to decide whether they prefer faster printing or a better quality print.

When applicable, there are diagrams of standard switch settings for many of the printers.

Alphacom___AlphaPro___101

- Daisy-wheel printer; prints text only.
- Switch setting:
  MODE A: ON (bidirectional print)
  MODE B: OFF (no line feed after carriage return)
  Tested Interface Card: 1011PC (IBM PC)

Brother___HR-15XL

- Daisy-wheel printer; prints text only.
- Switch settings:

```
┌─────────────────┐          ┌─────────────────┐
│ x x x x x x x x │  ON      │ x x x x x x x x │
└─────────────────┘  OFF     └─────────────────┘
  1 2 3 4 5 6 7 8              1 2 3 4 5 6 7 8
        SW1                          SW2
```
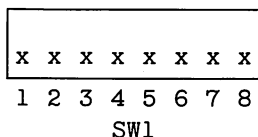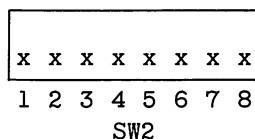
CalComp—ColorMaster

- Thermal transfer b&w/color printer; prints text and graphics.
- One density is supported — 203x200 dpi. Select a density of 1.
- This is a dual printer driver. Set the Paper Size setting in Preferences to Narrow Tractor for use with the ColorMaster printer. Set it to Wide Tractor for use with the ColorView-5912 printer (for a paper size of 11 x 17 inches).
- There are no DIP switches.

CalComp—ColorMaster2

- Thermal transfer b&w/color printer; prints text and graphics.
- One density is supported — 203x200 dpi. Select a density of 1.
- This is a dual printer driver. Set the Paper Size setting in Preferences to Narrow Tractor for use with the ColorMaster printer. Set it to Wide Tractor for use with the ColorView-5912 printer (for a paper size of 11 x 17 inches).
- This driver is the same as the Calcomp—ColorMaster driver EXCEPT that this one is approximately 2 times faster during color dumps and requires a large amount of memory (up to 1,272,003 bytes for a full 8 x 10 inch [1600 x 2000 dot] color dump). Typically full-size, color dumps are 1600 x 1149 dots and require 730,767 bytes.

  Memory requirements for the ColorView-5912 are up to 2,572,803 bytes for a full 10 x 16 inch (2048 x 3200 dot) color dump. Typically full-size color dumps are 2048 x 2155 dots and require 1,732,623 bytes.

  When doing a non-color printout, memory requirements are one-third of what is needed for a comparable color dump.
- There are no DIP switches.

Canon—PJ-1080A

- Ink jet b&w/color printer; prints text and graphics.
- One density is supported — 83x84 dpi. Select a density of 1.
- Switch settings:

Canon PJ-1080A

```
┌─────────────┐   ON
│ x x x x x x │   OFF
└─────────────┘
  1 2 3 4 5 6
```
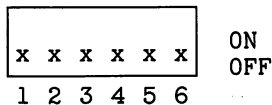
CBM—MPS1000 (Also for use with IBM 5152 Compatible Printers)

- Dot matrix non-color printer; prints text and graphics.
- Density

Density	XDPI	YDPI	XYDPI	Comments
1	120	72	8640	
2	120	144	17280	Performs two passes
3	240	72	17280	See Note 2
4	120	216	25920	Performs three passes; See Note 2
5	240	144	34560	Performs two passes; See Note 2
6	240	216	51840	Performs three passes; See Note 2
7				Same as a Density of 6

- Switch settings:

CBM MPS1000

```
┌─────────────────────┐
│ x                   │  ON
│   x x x x x x x     │  OFF
└─────────────────────┘
  1 2 3 4 5 6 7 8
```
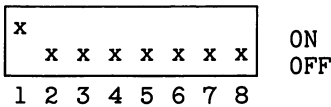
Canon BJ-130 (with Control Capsule 48/XL — IBM Proprinter compatible)

```
┌───────────────────────────┐         ┌───────────────────────────┐
│ x x x x x x x x x x x     │  OFF    │ x x x x x x x x x x x     │
│                           │  ON     │                           │
└───────────────────────────┘         └───────────────────────────┘
  1 2 3 4 5 6 7 8 9 10                   1 2 3 4 5 6 7 8 9 10
          SW1                                   SW2
```
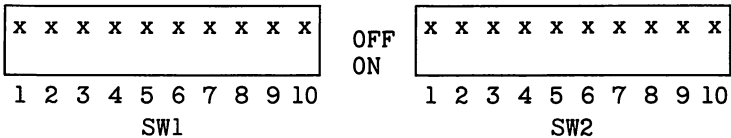
NOTE: If you own a Commodore MPS1250 printer, see the entry for EpsonX. They are compatible.

Diablo—630

- Daisy-wheel printer; prints text only.
- Switch settings:

```
┌─────────────────────┐  ON    ┌─────────────────────┐
│ x x x x x x x x     │        │ x x x x x x x x     │
│                     │  OFF   │                     │
└─────────────────────┘        └─────────────────────┘
  1 2 3 4 5 6 7 8                1 2 3 4 5 6 7 8
        SW1                            SW2
```
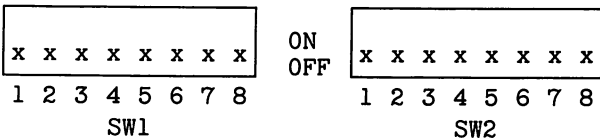
Notes are on page 4-18

Diablo___Advantage___D25

- Daisy-wheel printer; prints text only.
- Switch settings:
  Parallel Interface is recommended.

```
  OFF ON          OFF  ON
1 | x |          1 | x |
2 | x |          2 | x |
3 | x |          3 | x |
4 | x |          4 | x |
5 | x |          5 | x |
6 | x |          6 | x |
7 | x |          7 | x |
8 | x |          8 | x |
  LEFT             RIGHT
```
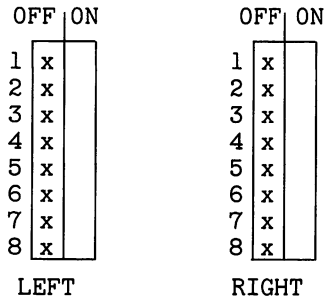
The Left switch is set for xON/xOFF handshaking and no line feed after carriage return.

Diablo___C-150

- Ink jet b&w/color printer; prints text and graphics.
- Line feeds are always equal to four dots (limitation of the printer).
- One density is supported — 120x120 dpi. Select a density of 1.
- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 8.5 inches (for wide roll paper) will be used.
- Switch settings:

```
      x x x
x x         x x x    ON
                     OFF
1 2 3 4 5 6 7 8
      SW1
```
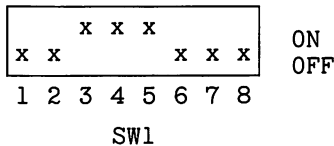
EpsonQ (24-pin printer compatible with the Epson ''Q'' series)

- Dot matrix b&w/color printer; prints text and graphics.
- Drives all EpsonQ (LQ1500, LQ2500, etc.) compatible printers.

• Density	XDPI	YDPI	XYDPI	Comments
1	90	180	16200	
2	120	180	21600	
3	180	180	32400	
4	360	180	64800	See Note 2
5	Same as a density of 4			
6	Same as a density of 4			
7	Same as a density of 4			

Notes are on page 4-18

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.6 inches (for wide carriage printers) will be used.
- If the Paper Type setting in Preferences is Single, only 16 of the 24 pins are used. A Paper Type of Fanfold uses all 24 pins. The Single option is useful for those printers which have a weak power supply and cannot drive all 24 pins continuously. If during a single pass of the print head you notice that the top two-thirds of the graphics are darker than the bottom one-third, then you'll probably need to drop down to 16 pins.

EpsonX (CBM—MPS1250 and 8/9-pin Epson ''X'' series compatible)

- Dot matrix b&w/color printer; prints text and graphics.
- Drives all EpsonX (EX/FX/JX/LX/MX/RX, etc.) compatible printers.
- If the Paper Type setting in Preferences is Single, the line feed will be the number of vertical dots printed, minus one-third of a dot. (See Note 1.)

- Density

Density	XDPI	YDPI	XYDPI	Comments
1	120	72	8640	
2	120	144	17280	Performs two passes
3	240	72	17280	See Note 2
4	120	216	25920	Performs three passes
5	240	144	34560	Performs two passes; See Note 2
6	240	216	51840	Performs three passes; See Note 2
7				Same as a density of 6

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.6 inches (for wide carriage printers) will be used.
- Switch settings:

Commodore MPS1250 Printer

```
 _____        _____
| x   x          |  ON  | x   x          |
|   x   x x x x x |  OFF |   x   x x x x x |
 ----------------        ----------------
 1 2 3 4 5 6 7 8        1 2 3 4 5 6 7 8
Serial/Parallel Interface Pack    Printer Internal Switch
```
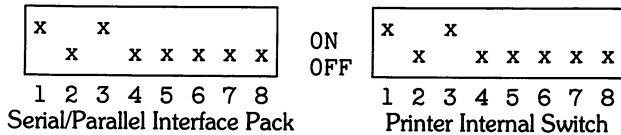
Notes are on page 4-18

Epson EX-1000 Printer

```
          x x x           x x x x x x x x
x x x x x         ON       x x x x x x x x
                  OFF     
1 2 3 4 5 6 7 8           1 2 3 4 5 6 7 8
     SW2                       SW2
```

Epson FX-80 Printer

```
x x                        x    x x x
    x x    ON       x x x      x
           OFF      
1 2 3 4             1 2 3 4 5 6 7 8
  SW1                    SW2
```

EpsonXOld (Older 8/9-pin Epson ''X'' printer compatible)

- Use this driver if the EpsonX driver doesn't work properly in graphics mode on your EpsonX compatible printer.
- Dot matrix b&w printer; prints text and graphics.
- Drives all very old EpsonX (EX/FX/JX/LX/MX/RX, etc.) compatible printers as well as the Star Micronics Gemini 10-X.

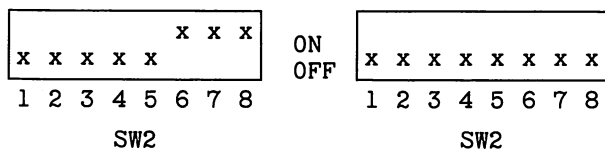| Density	XDPI	YDPI	XYDPI	Comments
1	60	72	4320	
2	120	72	8640	Double speed; See Note 2
3	120	72	8640	
4	240	72	17280	
5	120	72	8640	For use on old Star printers
6	240	72	17280	For use on old Star printers; See Note 2
7	240	72	17280	Same as a Density of 4; See Note 2

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.6 inches (for wide carriage printers) will be used.
- Switch settings for Star Micronics Gemini 10-X printer:
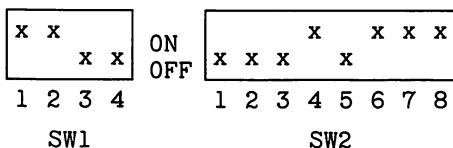
```
x x x x x x x x           x x
                  ON          x x
                  OFF     
1 2 3 4 5 6 7 8           1 2 3 4
 SW1 (internal)           SW2 (external)
```
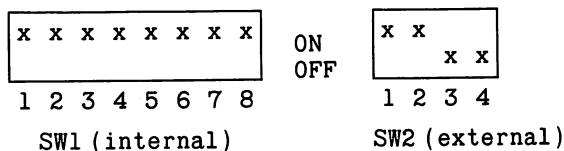
Notes are on page 4-18

Generic

- Text only printer; makes few assumptions about the printer's capabilities.

Howtek—Pixelmaster

- Plastic ink jet b&w/color printer; prints text and graphics.
- Density XDPI YDPI XYDPI
  - 1   80   80   6400
  - 2   120   120   14400
  - 3   160   160   25600
  - 4   240   240   57600
  - 5   Same as a density of 4
  - 6   Same as a density of 4
  - 7   Same as a density of 4
- Maximum print area is 8.0 x 10.0 inches.
- There are no DIP switches.

HP—DeskJet

- Ink jet b&w printer; prints text and graphics.
- Density XDPI YDPI XYDPI
  - 1   75   75   5625
  - 2   100   100   10000
  - 3   150   150   22500
  - 4   300   300   90000
  - 5   Same as a density of 4
  - 6   Same as a density of 4
  - 7   Same as a density of 4
- Maximum print area is 8.0 x 10.0 inches.
- Switch settings:

```
┌─────────────────────┐  ON  ┌─────────────────────┐
│ x x x x x x x x     │      │ x x x x x x x x     │
│                     │  OFF │                     │
└─────────────────────┘      └─────────────────────┘
  1 2 3 4 5 6 7 8              1 2 3 4 5 6 7 8

        SW1                          SW2
```
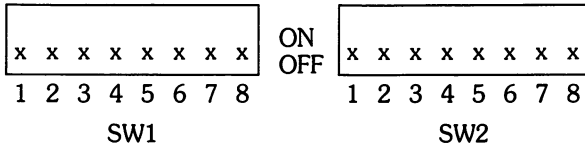
HP—LaserJet (LaserJet Plus/LaserJet II compatible)

- Laser engine b&w printer; prints text and graphics.
- Density XDPI YDPI XYDPI
  - 1   75   75   5625
  - 2   100   100   10000
  - 3   150   150   22500
  - 4   300   300   90000
  - 5   Same as a density of 4
  - 6   Same as a density of 4
  - 7   Same as a density of 4
- Maximum print area is 8.0 x 10.0 inches.
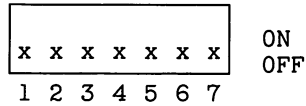- There are no DIP switches.

HP—PaintJet

- Ink jet b&w/color printer; prints text and graphics.
- One density is supported — 180x180 dpi. Select a density of 1.
- Switch settings:

```
┌─────────────────┐  ON
│ x  x  x  x  x  x  x │  OFF
└─────────────────┘
  1  2  3  4  5  6  7
```

HP—ThinkJet

- Ink jet b&w printer; prints text and graphics.
- Density

Density	XDPI	YDPI	XYDPI
1	96	96	9216
2	192	96	18432
3	Same as a density of 2		
4	Same as a density of 2		
5	Same as a density of 2		
6	Same as a density of 2		
7	Same as a density of 2		

- Switch settings:

```
┌─────────────────────┐  ON
│    x           x          │
│ x    x  x  x      x  x │  OFF
└─────────────────────┘
  1  2  3  4  5  6  7  8
```
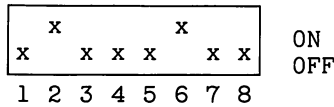
Imagewriter II (Imagewriter compatible)

- Dot matrix b&w/color printer; prints text and graphics.
- Density

Density	XDPI	YDPI	XYDPI	Comments
1	80	72	5760	
2	120	72	8640	
3	144	72	10368	
4	160	72	11520	
5	120	144	17280	Performs two passes
6	144	144	20736	Performs two passes
7	160	144	23040	Performs two passes

- Switch settings:

Imagewriter:

```
┌─────────────────────┐  ON    ┌──────────────┐
│ x  x  x  x        x  x │        │            x    │
│             x  x          │  OFF   │ x  x  x        │
└─────────────────────┘        └──────────────┘
  1  2  3  4  5  6  7  8                  1  2  3  4
        SW1                                SW2
```

Imagewriter II:

```
┌─────────────────┐          ┌─────────────────┐
│ x  x  x  x  x    x  x │  ON  │      x  x        │
│              x        │  OFF │ x  x        x  x │
└─────────────────┘          └─────────────────┘
  1  2  3  4  5  6  7  8        1  2  3  4  5  6
        SW1                          SW2
```
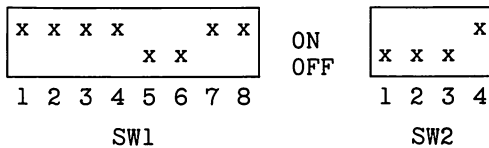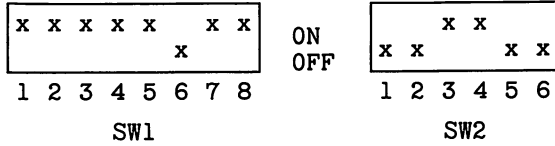
Nec—Pinwriter (24-wire Pinwriter compatible (P5/P6/P7/P9/P2200))

- Dot matrix b&w/color printer; prints text and graphics.
- Drives all Nec 24-wire Pinwriter compatible printers.
- Density XDPI   YDPI   XYDPI   Comments

Density	XDPI	YDPI	XYDPI	Comments
1	90	180	16200	
2	120	180	21600	
3	180	180	32400	
4	120	360	43200	Performs two passes
5	180	360	64800	Performs two passes
6	360	180	64800	
7	360	360	129600	Performs two passes

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.6 inches (for wide carriage printers) will be used.
- Switch settings for NEC Pinwriter P9XL:

```
┌──────────────────────┐
│ x  x  x  x  x  x  x  x │  ON
│                       │  OFF
└──────────────────────┘
  1  2  3  4  5  6  7  8
          SW1
```

```
┌──────────────────────┐
│             x         │  ON
│ x  x  x  x     x  x  x │  OFF
└──────────────────────┘
  1  2  3  4  5  6  7  8
          SW2
```

```
┌──────────────────────┐
│                x      │  ON
│ x  x  x  x  x  x     x │  OFF
└──────────────────────┘
  1  2  3  4  5  6  7  8
          SW3
```
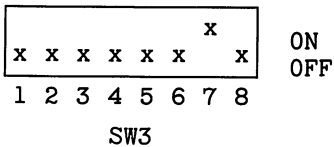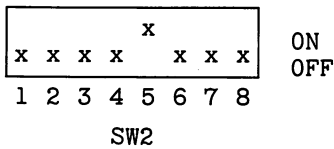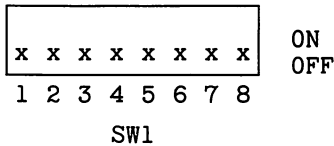
Okidata—92

- Dot matrix non-color printer; prints text and graphics.
- Line feeds are always 7/72 of an inch (limitation of printer in graphics mode).
- One density is supported — 72x72 dpi. Select a density of 1.

Okidata—293I

- Dot matrix b&w/color printer; prints text and graphics.
- Drives 292 or 293 using the IBM interface module.
- If the Paper Type setting in Preferences is set to Single, line feeds are equal to the number of vertical dots printed less one-half of a dot. (See Note 3.)
-

Density	XDPI	YDPI	XYDPI	Comments
1	120	144	17280	
2	240	144	34560	
3	120	288	34560	Performs two passes
4	240	288	69120	Performs two passes
5	Same as a density of 4			
6	Same as a density of 4			
7	Same as a density of 4			

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.6 inches (for wide carriage printers) will be used.
- Switch settings:

Interface: ML-292/293 Personality Module

SP1 | * * | Set the jumper

SP4 B * | * * | A Set the jumper in the position closest to A

B | * * | * A Set the jumper in the position closest to B

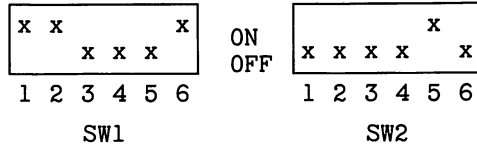*Notes are on page 4-18*

Printer Drivers

Okimate-20

- Thermal transfer b&w/color printer; prints text and graphics.
- Line feeds are equal to an even number of dots printed, (i.e., if three dots were printed, four dots will be advanced).
- One density is supported — 120x144 dpi. Select a density of 1.
- Switch settings:

Parallel Plug'n Print Kit

```
              x
  x  x  x  x     x    ON
                      OFF
  1  2  3  4  5  6
```
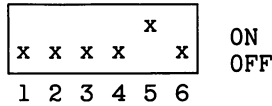
NOTE: Switch 5 on some models controls the white space between the lines of a graphic dump.

Serial Plug'n Print Kit

```
  x  x           x         ON        x
        x  x  x                   x  x  x  x     x
                           OFF
  1  2  3  4  5  6                1  2  3  4  5  6
        SW1                             SW2
```

NOTE: The SW1 settings specify a baud rate of 9600, xON/xOFF handshaking, 8 bits, and no parity. On some models, Switch 5 of SW2 controls the white space between the lines of a graphic dump.

Quadram—QuadJet

- Ink jet b&w/color printer; prints text and graphics.
- One density is supported — 83x84 dpi. Select a density of 1.
- There are no DIP switches.

Qume—LetterPro—20

- Daisy-wheel printer; prints text only.
- Switch settings:

```
                    x                       x  x
  x  x  x  x  x  x     x    ON       x          x  x
                           OFF
  1  2  3  4  5  6  7  8             1  2  3  4  5
        SWA                               SWB
```
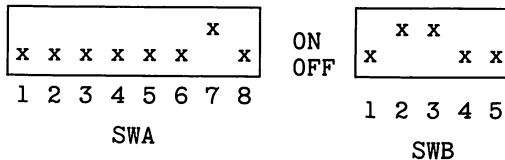
Seiko___5300

THIS DRIVER IS NOT ON THE WORKBENCH OR EXTRAS
DISK. IT IS AVAILABLE FROM SEIKO OR IN THE AMIGA.DEV
CONFERENCE ON BIX.

- Thermal transfer b&w/color printer; prints graphics only.
- Density

Density	XDPI	YDPI	XYDPI	Comments
1	152	152	23104	Drives CH-5301 printer
2	203	203	41209	Drives CH-5312 printer
3	240	240	57600	Drives CH-5303 printer
4	Same as a Density of 3			
5	Same as a Density of 3			
6	Same as a Density of 3			
7	Same as a Density of 3			

- There are no DIP switches.

Seiko___5300a

THIS DRIVER IS NOT ON THE WORKBENCH OR EXTRAS
DISK. IT IS AVAILABLE FROM SEIKO OR IN THE AMIGA.DEV
CONFERENCE ON BIX.

- Thermal transfer b&w/color printer; prints graphics only.
- Density

Density	XDPI	YDPI	XYDPI	Comments
1	152	152	23104	Drives CH-5301 printer
2	203	203	41209	Drives CH-5312 printer
3	240	240	57600	Drives CH-5303 printer
4	Same as a Density of 3			
5	Same as a Density of 3			
6	Same as a Density of 3			
7	Same as a Density of 3			

- This driver is the same as the Seiko___5300 driver EXCEPT
  that it is approximately two times faster during color dumps
  and it requires a large amount of memory (up to 1,564,569
  bytes for a full 8 x 10 inch [1927 x 2173 dot] color dump).
  Typically full-size, color dumps are 1927 x 1248 dots and
  require 898,569 bytes of memory. When doing a non-color
  printout, the memory requirements are one-third of what is
  needed for a comparable color dump.
- There are no DIP switches.

Tektronix—4693D

THIS DRIVER IS NOT ON THE WORKBENCH OR EXTRAS DISK. IT IS AVAILABLE FROM TEKTRONIX OR IN THE AMIGA-.DEV CONFERENCE ON BIX .

- Thermal transfer b&w/color printer; prints graphics only.
- One density is supported—300x300 dpi. Select a density of 1.
- Due to the way the printer images a picture, none of the printer Preferences options affect the printout with the following exceptions:
    a)Aspect — Horizontal, Vertical
    b)Shade — B&W, Grey-Scale, Color
As a result of this only full-size pictures can be printed.
- The following keypad menu options may be set from the printer's keypad. For normal prints set the option to the setting specified below:
3b—Color Adjustment—set to "do not adjust"
3d—Video Color Correction—set to "do not adjust"
5—Background Color Exchange—set to "print colors as received"
- Once a picture has been printed, additional copies may be printed without resending by using the printer's keypad.

Tektronix—4696

THIS DRIVER IS NOT ON THE WORKBENCH OR EXTRAS DISK. IT IS AVAILABLE FROM TEKTRONIX OR IN THE AMIGA-.DEV CONFERENCE ON BIX.

- Ink jet b&w/color printer; prints text and graphics.
- Line feeds are always four dots (limitation of the printer).
- It supports three densities:
    121 x 120 dpi   Select a Density of 1; outputs all colors in one pass
    242 x 120 dpi   B&W; Select a density of 2; performs a double-pass on black
    242 x 120 dpi   Color; Select a density of 3; performs a double pass on all colors
Selecting a density of 2 or higher doesn't give you true 242 dpi resolution, since the printer only supports 121 x dots per inch. Instead, selecting a density of 2 or higher tells the printer to go into its double pass mode. In this mode, it outputs a line of dots at 121 dpi, then it outputs the line again —shifted to the right by 1/242 of an inch. This produces much more vibrant colors and gives the illusion of a higher resolution. One drawback to this method, however, is that large areas of solid colors (red, green, and blue, specifically) tend to oversaturate the paper with ink.

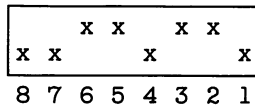The densities 1 through 3 correspond to the printer's graphics printing modes 1 through 3, respectively.

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 9.0 inches (for wide carriage printers) will be used.
- Switch settings:

```
      OFF |ON
    1 | x |
    2 |   | x
    3 |   | x
    4 |   | x
```

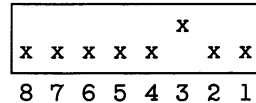Toshiba__P351C (24-pin Toshiba compatible)

- Dot matrix b&w/color printer; prints text and graphics.
- Drives all Toshiba__P351C compatible printers.
- Density

Density	XDPI	YDPI	XYDPI
1	180	180	32400
2	360	180	64800
3	Same as a density of 2		
4	Same as a density of 2		
5	Same as a density of 2		
6	Same as a density of 2		
7	Same as a density of 2		

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.5 inches (for wide carriage printers) will be used.
- Switch settings:

```
  |   x  x    x  x   |       |               x   |
  | x x       x      x |  ON  | x  x  x  x  x     x  x |
                        OFF
    8 7 6 5 4 3 2 1            8 7 6 5 4 3 2 1
          P1                         P2

  | x x x x x x x x |     ON  | x  x              |
                             | x  x              x  x  x  x  x |
                        OFF
    8 7 6 5 4 3 2 1            8 7 6 5 4 3 2 1
          P3                         P4
```

Toshiba_P351SX (24-pin Toshiba compatible)

- Dot matrix b&w/color printer; prints text and graphics.
- Drives all Toshiba_P351SX (321SL, 321SLC, 341SL) compatible printers.
- Density   XDPI   YDPI   XYDPI   Comments

Density	XDPI	YDPI	XYDPI	Comments
1	180	180	32400	
2	360	180	64800	
3	180	360	64800	Performs two passes
4	360	360	129600	Performs two passes
5	Same as a density of 4			
6	Same as a density of 4			
7	Same as a density of 4			

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 13.5 inches (for wide carriage printers) will be used.
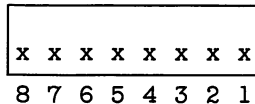- Switch settings:

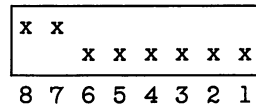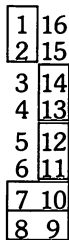 If using the serial interface, set the jumpers in the following positions: 1 & 2; 7 & 10; 8 & 9; 11 & 12; 13 & 14. See the diagram below:

```
 1 |16
[2]|15
 3 |14|
 4 |13|
 5 |12
 6 |11|
[7 10]
[8  9]
```

Xerox_4020

- Ink jet b&w/color printer; prints text and graphics.
- Line feeds are always equal to 4 dots (limitation of the printer).
- This driver is IDENTICAL to the Diablo_C-150 driver EXCEPT that it outputs all black dots TWICE. This is a special feature of this printer and produces much more solid, dark black shades. Please note that some printing time overhead results from this feature; if you don't want it, use the Diablo_C-150 driver.
- Two densities are supported:

  121x120 dpi   Select a density of 1
  242x240 dpi   Select a density of 2

Selecting a density of 2 or higher really doesn't give you true 240 dpi resolution since the Xerox—4020 only has 121 x dots per inch. Instead this mode tells the printer to go into it's pseudo 240 dpi mode. Here, it outputs a line of dots at 121 dpi; moves the paper up 1/240 of an inch and outputs the line again (shifted to the right by 1/240 of an inch). This produces much more vibrant colors and gives the illusion of more resolution. One drawback is that large areas of solid colors (red, green, and blue specifically) tend to oversaturate the paper with ink.

- If the Paper Size setting in Preferences is Wide Tractor, a maximum print width of 9.0 inches (for wide roll paper) will be used.
- Switch settings:

```
OFF |ON          OFF |ON
 1 | x |          1 | x  |
 2 | x |          2 |   x|
 3 | x |          3 | x  |
 4 |_x_|          4 | x  |
                  5 | x  |
   LEFT           6 | x  |
                  7 |_x__|

                  RIGHT
```

NOTES:

1. This is only true when 72 YDPI is selected. This option is useful if you notice tiny white, horizontal stripes in your printout.

2. In this mode the printer cannot print two consecutive dots in a row. It is recommended that you only use this density for black-and-white, shade dumps.

3. This is only true when 144 YDPI is selected. This option is useful if you notice tiny white, horizontal stripes in your printout.

5. CHANGING YOUR STARTUP SEQUENCES

When you boot your Amiga computer with the Workbench disk, Amiga-DOS executes the file called Startup-Sequence in the Workbench's S: directory:

```
1   Addbuffers df0: 10
2   c:SetPatch >NIL: ;patch system functions
3   cd c:
4   echo ''A500/A2000 Workbench disk. Release 1.3 version 34.19*N''
5   Sys:System/FastMemFirst ; move C00000 memory to last in list
6   BindDrivers
7   SetClock load ;load system time from real time clock (A1000 owners should
8                 ;replace the SetClock load with Date)
9   FF >NIL: -0 ;speed up Text
10  resident CLI L:Shell-Seg SYSTEM pure add; activate Shell
11  resident c:Execute pure
12  mount newcon:
13  ;
14  failat 11
15  run execute s:StartupII ;This lets resident be used for rest of script
16  wait >NIL: 5 mins ;wait for StartupII to complete (will signal when done)
17  ;
18  SYS:System/SetMap usa1 ;Activate the ()/* on keypad
19  path ram: c: sys:utilities sys:system s: sys:prefs add ;set path for Workbench
20  LoadWB delay
21  endcli >NIL:
```

Workbench's Startup-Sequence file

To make the most of the speed of resident commands, most of the 1.3 Startup-Sequence has been moved to a second file named StartupII. This file is run by the initial startup-sequence after the Shell-Seg is made resident:

```
1   resident c:Resident pure
2   resident c:List pure ;pre-load LIST and CD
3   resident c:CD pure
4   resident c:Mount pure ;the next 3 are loaded for speed during startup
5   resident c:Assign pure
6   resident c:Makedir pure
7   ;make IF, ENDIF, ELSE, SKIP, ENDSKIP, and ECHO resident if
8   ;you use scripts much, and can afford the ram.
9   ;also make Failat, WAIT, and ENDCLI resident if you use IconX a lot
10  makedir ram:t
11  assign T: ram:t ;set up T: directory for scripts
12  makedir ram:env ; set up ENV: directory
13  assign ENV: ram:env
14  makedir ram:clipboards ;set up CLIPS: assign
15  assign CLIPS: ram:clipboards
16  mount speak: ;just mounting doesn't take much ram at all
17  mount aux:
18  mount pipe:
19  resident Mount remove ;if you have enough ram, keep these resident
20  resident Assign remove ;by removing these lines
21  resident Makedir remove
22  ;
23  break 1 C ;signal to other process its ok to finish
```

Workbench's StartupII file

In addition, there are new s:CLI-Startup and s:Shell-Startup files which are the respective startup files executed by each NEWCLI or NEW-SHELL (unless an alternate ''FROM'' script is provided in the NEWCLI or NEWSHELL command line).

All of these S: directory startup files may be edited with any text editor, such as ED or MEMACS, to create your own customized startup sequence. This allows you to set up a preferred working environment or to take advantage of additional hardware or RAM. Sample customizations of the startup files include:

Startup-Sequence
  Opening an initial SHELL
  Setting up additional paths and logical names
  Mounting hard disk partitions
  Setting up a RAD: (Recoverable ramdrive.device)
  Transferring control to a RAD:, Workbench, or a hard drive

StartupII
  Relabeling RAM: for programs that can't handle spaces in a volume name (default volume name is ''RAM DISK'')
  Making more commands resident
  Setting environment variables such as ENV:Editor (used by MORE)

Shell-Startup
  Setting up personal command aliases

This chapter takes you through several examples of modifying these startup sequence files, including instructions on rebooting from the Recoverable ramdrive.device. Before attempting to alter any of the startup scripts, you should make a backup copy of your Workbench disk. Also, remember that the system generally looks for scripts in the S: directory. If you assign S: to a different device (such as a hard disk or ramdrive) during or after your startup, you must also place your modified scripts in that S: directory. We suggest that you keep backup copies of all of your modified startup and MountList files on a floppy disk.

The following examples describe some simple changes that can be made to startup scripts.

Opening an Initial Shell

Add the following lines to the end of the Startup-Sequence, before line 22 `"endcli >NIL:"`.

```
cd SYS: ; You may prefer cd RAM:
NewShell newcon:0/0/640/200/AmigaShell
```

Setting up Additional Paths and Logical Names

Paths and logical assignments to other floppy disks can be added in the Startup-Sequence or StartupII file. For instance, if you always boot with the Extras disk in df1:, you may want to add a path there:

```
path "Extras 1.3:" add
```

If you always boot with a C source, Includes, and Lib disk in df1: (here called MySrc), you might find it convenient to set logical names for the directories. If your other scripts always reference these logical names, you will only have to change these logical assignments when your system or configuration changes.

```
assign SRC: MySrc:
assign INCLUDE: MySrc:include
assign LIB: MySrc:Lib
```

Mounting Hard Disk Partitions

After BindDrivers in the Startup-Sequence, you may wish to mount additional hard disk partitions, assign all system directories to hard disk directories, and add paths to personal directories on your hard disk partitions. One flexible method of accomplishing this is to put all of these additional commands in a script called hd-startup in the S: directory of your first partition, then add a line to Execute dh0:s/hd-startup after the BindDrivers line in the Startup-Sequence.

Before executing the hd-startup file, CD to RAM: in case EXECUTE needs to make a temporary file. If you use your boot disk on more than one Amiga, these lines (added after BindDrivers) will only execute hd-startup when you boot on your hard drive system. (Depending on your configuration, you may have to change dh0: to a different unit name.)

```
cd ram:
failat 30
assign >NIL: dh0: exists
if not warn
  execute dh0:s/hd-startup
endif
failat 10
```

Relabeling RAM:

Edit the StartupII file, and add the following line after "`Assign CLIPS:
ram:clipboards`" (line 15):

```
relabel drive RAM: name RAMDISK
```

Making Additional Commands Resident

Most of the C: directory commands and the 1.3 MORE program can be made resident (as shown by the ''p'' or pure bit displayed when you LIST these commands). Resident commands are extremely fast, reduce memory usage when multitasking, and generally make SHELL work more pleasant, especially on a floppy-based system.

One suggested method for making more commands resident is to create a separate script called ''rescom'' similar to the following example:

```
resident c:dir pure
resident c:copy pure
resident c:delete pure
resident c:ed pure
resident sys:Utilities/More pure
```

Place the script in your S: directory, then edit s:StartupII adding this line near the end:

```
run >NIL: execute s:RESCOM
```

Because there is one resident list for the whole system, you can RUN the EXECUTE of your rescom script, allowing your StartupII to complete while the commands are being made resident.

Setting Environment Variables

The 1.3 MORE program, when started from the CLI or SHELL, now allows you to edit the file you are looking at, using any editor that can be passed a filename as a single command line argument. MORE gets the name of your preferred editor from the environment variable EDITOR. If you wish to use this feature of MORE, you should add a line to StartupII to SETENV the EDITOR environment variable to the path and name of the text editor you use. Add the SETENV line after the StartupII line that assigns ENV: to RAM:ENV (line 13).

```
SETENV EDITOR C:ed
```

Adding Personal Command Aliases to s:Shell-Startup

Using a text editor, add alias commands to the Shell-Startup.

Examples: `alias ns newshell`
`alias es endshell`
`alias ex execute`
`alias ls list`
`alias prefs sys:prefs/Preferences`

Mounting a Recoverable Ramdrive

The MountList file in the DEVS: directory of Workbench 1.3 contains an entry for RAD:, a recoverable ramdrive which can be rebooted from when using Version 1.3 Kickstart. The size of RAD: may be modified by editing the HighCyl value of the RAD: entry in the DEVS:MountList file. Note that when you change the size (HighCyl) of RAD: in the MountList, you must perform a cold powerup before attempting to mount RAD: at the new size.

On a 1 MB Amiga with no hard drive, a small RAD: can be used to hold your S: directory and some common C: directory commands. If you have more than 2 MB of RAM, you may want to try a floppy-sized 880K RAD: (HighCyl = 79). The Workbench disk can be diskcopied to an 880K ramdrive for a super-fast, recoverable Workbench-in-RAM which can be rebooted from with 1.3 Kickstart. Hard drive owners might wish to set up a RAD: that just contains the minimum number of files and directories required to transfer control to a hard drive.

Ramdrive Workbench—For Amigas with more than 2 MB of RAM

A ramdrive Workbench is extremely useful if you have a Megabyte of RAM to spare and no hard drive. On a cold powerup, these changes will mount an 880K RAD: and DISKCOPY your boot disk to it. This takes about one-fourth the time of a floppy-to-floppy DISKCOPY and is not repeated when you reboot. All system directories are assigned to RAD: so that the rest of the Startup-Sequence and StartupII are executed very quickly and df0: is left totally free for use as a work drive.

In conjunction with Version 1.3 Kickstart, you can reboot from the resulting RAD:. After making the following changes, turn your Amiga off for at least 30 seconds. Turn it back on, and boot with your modified Workbench. Note that when the DISKCOPY of df0: to RAD: is complete, a requester will flash twice. This is normal.

1. Edit the devs/MountList file and change the HighCyl value of RAD: to HighCyl = 79 (same HighCyl as an 880K floppy).

2. Insert the following lines in your Startup-Sequence, between the FastMemFirst and BindDrivers lines.

failat 30
assign >NIL: RAD: exists
if warn
  echo "Mounting RAD:"
  mount RAD:

```
    if not exists RAD:c
      echo "Backing up df0: to RAD:"
      SYS:System/diskcopy <NIL: df0: to RAD: name "RAMWB"
    endif
  endif
  RAD:c/failat 10

  RAD:c/cd RAD:c

  RAD:c/echo "Transferring control to RAD:... " NOLINE
  assign c: RAD:c
  assign s: RAD:s
  assign l: RAD:l
  assign libs: RAD:libs
  assign devs: RAD:devs
  assign fonts: RAD:fonts
  assign sys: RAD:
  echo "Done"
```

Minimal Ramdrive for Transferring Control to an A2090 Hard Disk Controller and Hard Drive

A small rebootable ramdrive can be used to transfer control to a hard drive. This allows users with non-autoboot hard disk controllers (A2090s) to reboot without placing a Workbench in df0: The ramdrive must contain your devs/system-configuration file, s:startup-sequence file, and any necessary files and directories to get through BindDrivers and the assignment of control to a hard drive partition containing Workbench.

This startup requires an initial partition containing, at minimum, the files and directories required for executing the hd-startup. The hd-startup will mount your additional partitions and assign system logical directories to the partition(s) containing the Workbench directories. Also listed are some additional files and directories you will need in this initial partition if you get an A2090A autoboot controller (you may want to put them there now or at least leave room for them). Note that these files are just enough to transfer control to another (probably FFS) partition containing all of the Workbench directories and commands. If you plan to run off the initial partition, you might as well copy all of Workbench there.

1. Make sure your initial hard disk partition contains the following directories and commands:
   ```
   s (dir)
     startup-sequence              (autoboot)
     hd-startup
   c (dir)
     cd
     assign
     echo
     mount
     BindDrivers                   (autoboot)
     AddBuffers                    (autoboot)
   ```

```
SetPatch                            (autoboot)
devs (dir)
   MountList
   system-configuration             (autoboot)
libs (dir)
   icon.library                     (autoboot)
l (dir)
   disk-validator
   FastFileSystem                   (if mounting FFS partitions)
System (dir)
   FastMemFirst                     (autoboot)
Expansion (dir)
   other non-autoboot drivers       (autoboot)
```

2. Copy your A2090 hard disk driver and icon (HdDisk and HdDisk.
   info) to two places: the Expansion drawer of your Workbench disk,
   and the Expansion drawer of the hard disk partition which will be your
   SYS: (the one containing all of the Workbench directories). In the
   following examples, that partition is FF0: which is assigned the name
   FASTWB:

3. Edit the devs/MountList file for RAD:, changing the HighCyl value
   from 21 to 7. Add MountList entries for any additional partitions you
   have.

4. Add the following lines to the Startup-Sequence immediately after the
   Binddrivers line (line 6). You may have to change the reference to
   dh0: to a different initial partition unit, depending on your
   configuration.

```
failat 30
assign >NIL: INIT: dh0:
if not warn
   cd INIT:
   execute INIT:s/hd-startup
endif
failat 10

; Now you should be running off your hard drive Workbench
; Create the RAD: for rebooting
assign >NIL: RAD: exists
if warn
   echo ''Mounting RAD:...'' NOLINE
   mount RAD:
   if not exists RAD:c
      echo ''setting up...'' NOLINE
      relabel drive RAD: name RamDrive
      makedir RAD:c
      copy c:AddBuffers|assign|BindDrivers|cd|copy|echo RAD:c quiet
      rad:c/copy c:endif|execute|failat|if|makedir|SetPatch RAD:c quiet
      rad:c/makedir RAD:devs
      rad:c/copy devs:system-configuration RAD:devs quiet
      rad:c/makedir RAD:L
```

```
       rad:c/copy L:disk-validator RAD:L quiet
       rad:c/makedir RAD:Libs
       rad:c/copy Libs:icon.library RAD:Libs quiet
       rad:c/makedir RAD:System
       rad:c/copy SYS:System/FastMemFirst RAD:System quiet
       rad:c/makedir RAD:s
       rad:c/copy s:startup-sequence RAD:s quiet
       rad:c/makedir rad:Expansion
       rad:c/copy sys:Expansion rad:Expansion quiet
     endif
   echo ''Done''
   endif
```

5. Create your hd-startup file and store it in the S directory of your initial
   hard disk partition (here referenced by the INIT: name assigned in the
   startup lines above). The hd-startup will mount your additional parti-
   tions, transfer control to them, and set up your hard disk paths and
   logical assignments. Below is a sample hd-startup file. This can be
   modified to correspond to your hard drive system.

```
   ; Assign system directories needed to mount rest of partitions
   INIT:c/cd INIT:c
   assign DEVS: INIT:Devs
   assign L:    INIT:L
   ; In this example, user has two FFS partitions, FF0: and FF1:
   ; FF0: contains all of the Workbench directories. We'll assign the
   ; logical FASTWB: to the partition containing the Workbench directories,
   ; and H: to the user's work partition to make script changes easier.
   mount FF0:
   mount FF1:
   assign FASTWB: FF0:
   assign H: FF1:
   ; assign all system directories to FASTWB:
   assign c:       FASTWB:c
   assign SYS:     FASTWB:
   assign DEVS:    FASTWB:Devs
   assign LIBS:    FASTWB:Libs
   assign L:       FASTWB:L
   assign S:       FASTWB:s
   assign C:       FASTWB:c
   assign Fonts:   FASTWB:fonts
```

6. Now copy your modified MountList to the devs directory of your initial
   partition and to the devs directory of your Workbench partition. Copy
   your modified Startup-Sequence to the S directory of both your initial
   partition and your Workbench partition. (It will be copied from the
   Workbench partition to RAD:). Turn off your machine, wait 30 sec-
   onds, then reboot with your modified Workbench.

Small Recoverable Ramdrive for a 1 MB Amiga

This small ramdrive is not suitable for rebooting because it cannot hold all of the commands and handlers required during startup. It is somewhat useful for freeing up df0: because it holds some commonly used commands, the s directory, and the Expansion directory. With this RAD:, you will also be able to start a NEWSHELL from the command line or from the icon in the ramdrive without having to insert the Workbench.

With this modification, the startup takes a couple of minutes when you power on, but is much quicker when you reboot.

1. Open up the Workbench disk icon, then open the Prefs drawer. Click once on the Pointer icon and select INFO from the Workbench menu. Change the Default Tool specification from SYS:Prefs/Preferences to Preferences. Press RETURN, then click on SAVE to save the change. Now repeat this procedure with the Printer and Serial icons.

2. Make the following changes in the devs/MountList file:
   - In the NEWCON: entry, add this line: Mount = 1
   - In the RAD: entry, change the HighCyl value from 21 to 14, and add this line: BootPri = -129

3. Before the LoadWB line in the Startup-Sequence, remove the following lines (lines 18 and 19):

   ```
   Sys:System/SetMap usal ;Activate the ( )/* on keypad
   path ram: c: sys:utilities sys:system s: sys:prefs add
   ```

4. Insert the following lines in your Startup-Sequence, between the FastMemFirst and BindDrivers lines (5 and 6). You may wish to modify the ''copy'' lines of the following script to place some different commands in the ramdrive. The directories and some of the commands (such as NEWSHELL, ENDCLI, RUN, FORMAT and DISK-COPY) are required.

   ```
   failat 30
   assign >NIL: RAD: exists
   if warn
   echo "Mounting RAD:..." NOLINE
   mount RAD:
   if not exists RAD:c
       echo "Setting up RamDrive..." NOLINE
       relabel drive RAD: name RamDrive
       makedir RAD:c
       copy c:assign|copy|delete|dir|echo|ed|else RAD:c quiet
       rad:c/copy c:endcli|endif|if|info|makedir RAD:c quiet
       rad:c/copy c:newshell|path|prompt|rename|run RAD:c quiet
       rad:c/copy sys:Utilities/More rad:c quiet
       rad:c/makedir RAD:s
       rad:c/copy s: RAD:s all quiet
       rad:c/makedir RAD:System
       rad:c/copy sys:system/CLI|format|diskcopy rad:System quiet
       rad:c/copy sys:Shell|Shell.info RAD: quiet
       rad:c/makedir rad:Expansion
   ```

```
    rad:c/copy sys:expansion rad:Expansion quiet
    endif
endif
failat 10

assign s: RAD:s
path rad:c rad:system sys:utilities sys:system sys:prefs sys: s: add
Sys:System/SetMap usa1
assign sys: RAD:
echo "Done"
```

5. Add these two lines before the LoadWb line:

```
path c: ram: add
assign c: rad:c
```

6. UTILITIES ON THE EXTRAS DISK

Tools

The Tools drawer contains several utilities that let you ''work'' with your Amiga. In a sense, they enhance the basic capabilities of the Workbench. With these programs, you can modify your icons and fonts, check on the system's memory, edit text files, and change the colors of an Intuition screen. The Tools drawer includes:

FED — for modifying existing fonts or creating new ones
FreeMap — for displaying your chip memory usage
IconEd — for changing the appearance of icons
IconMerge — for merging and splitting icon images
KeyToy2000 — for displaying the global keymap of an Amiga 2000
MEMACS — for editing text files
Palette — for changing the colors of an Intuition screen
PerfMon — for monitoring the system's performance

Each of these utilities is explained below, except for MEMACS. MEMACS is such a powerful and complex text editor that it requires its own chapter. It is fully explained in Chapter 7.

FED

FED is a font editor for changing the Workbench fonts or for creating your own new fonts. However, if you plan to change a Workbench font, make sure that you are experimenting with a copy of the Workbench disk. Once you make any changes, you will be unable to restore the original. Please note, FED cannot be used to edit color fonts or a font that is larger than 32 pixels in either dimension.

When you double-click on the FED icon, a window appears filled with gadgets, boxes, switches, numbers and symbols. There is also a menu bar across the top of the screen; the menus are Project, Edit and Attributes. You will do all of your creating/editing in this window. All the menu items and gadgets are described below.

If you're going to edit a Workbench font, the first thing you'll want to do is to load the font into FED. To do this, select Open from the Project menu. A requester will appear listing the available fonts. Select the one you want to load and choose Load It. If you change your mind, choose Whoops! (If you want to work on a brand new font, select **New** instead of **Open**.)

Each character of the font appears in one of the character boxes to the right side of the screen. You can only see a limited number of boxes at a time, there are actually more than 200 boxes for a typical font. The character box reflects the current appearance of a character. As changes are made, the display in the character box also changes.

Each character is made up of a series of dots. Each dot represents one pixel of the character. You use the mouse to edit (or create) a character in the large edit box that fills the left side of the screen.

- To change a dot, click on it with the mouse.
- To draw dots, click on any off dot, then drag the mouse.
- To erase dots, click on any on dot, then drag the mouse.

Each of the gadgets is briefly explained so that you understand their functions. Then you should let your artistry and imagination lead you in producing show-stopping fonts.

The Gadgets

LoRes switch — selects either 320 x 200 or 640 x 200 resolution.

Grid switch — turns the edit box grid on or off.

Scroll diamond — if a character is larger than the display area of the edit box, you can move to the outer limits of the box by clicking on the arrows of the scroll diamond. The character will move only if its outer edge is not displayed. For instance, you cannot move a character to the left if you can already see its left edge — even if that edge is not at the edge of the edit box.

Zoom — allows you to select the display size of the edit box.

Left/Right triangles — shifts the current character to the left or right by one pixel.

Up/Down triangles — shifts the current character up or down by one pixel.

R — reverses the character.

Left/Right arrow — flips the character to the left or the right.

Up/Down arrow — flips the character up or down.

The Boxes

The boxes set the specific parameters for a font. For more explanation of these parameters, consult the Text chapter of the Addison-Wesley *ROM Kernel Manual: Libraries and Devices.*

The Lo and Hi boxes — determine the number of characters in a font or the character range (i.e., which characters to save). For instance, you could have a range that only included capital letters or a range that included only numerical symbols. The range for the fonts on the Workbench is from 032 to 255.

Kern — determines the kerning data; this is usually set to zero.

Space — determines the proportional spacing information or the width of each character rectangle — how many pixels of space the character will fill on the screen.

Width — works with the Space variable to determine the proportional spacing information — how many pixels of actual data are present for the character.

Kern, Space and Width can only be changed for proportional fonts.

BLine — specifies the baseline of the characters. The baseline is represented by the green, horizontal line in the edit box. This represents the number of grid blocks from the top of the character to the baseline. Think of the baseline as a ruled line on a piece of paper.

XSize — represents the nominal width of a font.

YSize — represents the nominal height of a font.

The Project Menu

New — clears the character boxes allowing you to start fresh.

Open — lists the available fonts and lets you load one.

Save — saves the font that you've been working on under its original name.

Save-as — lets you save a font under a new name.

About — lists credits.

Quit — exits FED.

The Edit Menu

NOTE: The items in this menu affect the ENTIRE font and some of them are very difficult to reverse.

Make italic — algorithmically italicizes the entire font.

Make bold — algorithmically changes the entire font to boldface.

Make underlined — algorithmically underlines the entire font.

Copy to — copies a character from one character box to another. To use this option, select a character box, for example K, then choose Copy To. When you select another character box, the K will be copied into that box. Until you select Copy To again, the pointer will remain a "Copy" pointer, and each box you select will receive a copy of the originally selected character box. The copy mode will remain in effect until you select Copy To again.

Erase — erases the character from the selected character box.

All right — shifts all characters to the right by one pixel.

All left — shifts all characters to the left by one pixel.

All up — shifts all characters up by one pixel.

All down — shifts all characters down by one pixel.

The Attributes Menu

This menu lets you set the type and style bits of the font.

Font Type — choose either Proportional or Fixed width spacing.

Font Style — choose Normal, Italic, Bold, Underlined, or Extended. For most fonts, select Normal. If you design a special italic font, for instance, set the italic flag. When the system finds a specially designed italic font, it will use it instead of trying to create its own.

FreeMap

FreeMap gives a graphic display of chip memory usage, showing the blocks used and the free holes. Each pixel in the map area represents a 64 byte block. If all the bytes in a block are free, the pixel is dark. If a byte is not free, the pixel is light. At the bottom of the window, the amount of free CHIP RAM and FAST RAM is listed.

To quit FreeMap, choose Please from the Quit menu. Selecting the Info menu, brings up a brief explanation of FreeMap.

IconEd

IconEd lets you change the appearance of the Workbench icons. It has been moved to the Extras disk from the Workbench's System drawer. IconEd is fully documented in your *Introduction to the Commodore-Amiga manual.*

IconMerge

IconMerge lets you merge the images of two single-image Workbench icons. It also lets you split a double-image icon into two single-image icons. A double-image icon is an icon that changes its shape when it is selected.

When you double-click on the icon, the IconMerge window appears and prompts you to enter either an m, to merge two single-image icons, or an s, to split a double-image icon. For example, to merge the ICONED icon with the FED icon, enter an m at the prompt, and you'll get the following response:

```
MERGE: Icon1, Icon2 (merge)-> Icon3

Enter all icon filenames without ".info."
  Icon1:
```

At the prompt after Icon1:, enter ''IconEd'' and hit RETURN. A second prompt will appear for Icon2:, enter ''FED''. For Icon3:, enter ''Test''. IconMerge will merge the two icons and the window will quickly disappear.

To see your new icon, you must close and re-open the Tools window. The new test icon will be on top of the IconEd icon. Simply drag the test icon to an empty space in the window. Notice that the test icon is an exact copy of the IconEd icon (Icon1). When you click on the test icon, the FED image (Icon2) will appear to show that the icon is selected.

The IconMerge icon is a double-image icon. Try splitting that icon. Double-click on the IconMerge icon to open the IconMerge window. Enter an s at the merge/split prompt, and the following display will appear:

```
SPLIT: Icon1 (split)-> Icon2, Icon3

Enter all icon filenames without ".info.":
  Icon1:
```

Enter IconMerge at the Icon1: prompt, Icon for Icon2:, and Merge for Icon3:. After the IconMerge window disappears, close the Tools window, then re-open it. The two new icons, Icon and Merge, will be behind the original IconMerge icon.

Drag them to an empty space in the window (or select Clean-up from the Workbench's Special menu). Icon will reflect the unselected appearance of the IconMerge icon. Merge will reflect the highlighted appearance of IconMerge.

When you select either of these new icons, they are simply highlighted. They don't change form like a double-image icon would. Splitting a double-image icon allows you to use IconEd on each of the images. Your modified images may then be merged back together with the merge option of IconMerge.

To run IconMerge from the CLI, the format is:

ICONMERGE [<icon1> <icon2> <icon3> [OPT S]]

The default for IconMerge is to merge the two icons. If no icons are specified, the IconMerge window will appear and you can proceed as outlined above. If you specify OPT S, IconMerge will split the icons named as <icon1> and <icon2>.

A Few Things to Remember:

1) Icon names must be entered without the .info suffix.
2) When merging icons, be sure they are the same size.

KeyToy2000

Double-clicking on the KeyToy2000 icon results in a display of the current global keymap for the keyboard accompanying the Amiga 2000 computer.

The initial display reflects the characters that are output when each key is pressed alone. To see the characters that are output when qualifier keys are pressed simultaneously with a character key, either click on the corresponding gadget in the KeyToy display or hit the key on the key-board. The qualifier keys are Control, both Shifts, and both Alts.

The following list is a guide to interpreting the KeyToy display:

- All KeyToy keys labeled in blue are not used by KeyToy (That's why CTRL, SHIFT and ALT appear in blue in the initial display. After you select one or more of these keys, the label changes color.)
- Any key labeled in red is a dead key, except for the two AMIGA keys and the CAPS LOCK key.
- Any key labeled in bold-italic is deadable.
- Any key labeled with $$ is defined by a character string longer than one character.
- A character preceded by ^ or ~ is a control character.
- If a key is blank, it is undefined for the current choice of qualifier(s).

Palette

With Palette you can modify the colors of a screen. It differs from the color setting capabilities of Preferences in that Preferences is limited to the colors of the Workbench screen. However, color changes made with Palette are only temporary. They cannot be saved to disk.

To open Palette from the Workbench, double-click on its icon. The Palette Tool window will appear in the frontmost screen, usually your Workbench screen. To modify the colors of a different screen, you need to bring that screen to the front of the display before opening Palette. Open the desired screen, then slide it down so that the Workbench screen is also visible. Open the Tools window on the Workbench screen, and double-click on the Palette icon. When the Palette Tool window opens, it will be on the frontmost screen.

To run Palette from the CLI, the format is:

PALETTE [<bitplanes>] [<screentype>]

By specifying values for the <bitplanes> and <screentype> options you can open a custom test screen. The values for <bitplanes> and <screen-type> are as follows:

<bitplanes>   specifies the depth of the test screen; 1 = 2 colors, 2 = 4 colors, 3 = 8 colors, 4 = 16 colors, and 5 = 32 colors

<screentype>   specifies the resolution of the test screen; 0 = 320 x 200 pixels, 1 = 320 x 400 (interlaced screen), 2 = 640 x 200, and 3 = 640 x 400 (interlaced screen)

NOTE: The value for <bitplanes> is restricted to 4 or less if the value for <screentype> is equal to either 2 or 3.

The Palette Tool window contains several gadgets and sliders. Across the top of the window are color rectangles showing the colors that can be modified. Beneath these rectangles are three color sliders: red, blue and green. Select a color rectangle to be modified, then drag the color sliders until you have the color you want. The color changes are reflected in the vertical box that runs alongside the color sliders. Repeat this process with each color rectangle until you have the screen configured as you want it.

Choose the OK box to implement the color change. Choose RESET to return the colors to their original states. Or choose CANCEL to quit Palette without making any changes. WARNING: If you open Palette on a screen other than the Workbench screen, you must first close Palette before exiting or closing the application that opened the new screen.

PerfMon

Double-click on the PerfMon icon, and the Performance Monitor window appears. The top portion of the window shows the amount of CPU time currently in use (calibrated for a 68000 CPU). The black center line indicates 100% CPU utilization. The bottom portion of the window shows the amount of CHIP RAM currently in use. The bottom black line indicates 100% chip memory utilization. The amount of fast memory available is displayed at the bottom of the window.

You can choose the time interval for the performance update from the PM Menu. The options are: .5, 1 and 2 seconds.

PC Utilities

PC Utilities allows you to use a 5¼" floppy disk drive with your Amiga system. With a 5¼" disk drive you can copy files from a PC to your Amiga, and vice versa. In many cases when you add an expansion device to your system, such as a hard disk, you must MOUNT it — tell the Amiga of its existence by adding a command to your startup-sequence. However, you do not need to do this to use PC Utilities. If you do MOUNT the 5¼" drive, PC Utilities will not work correctly.

If you get a message telling you that 5¼" drives are not available, double-check that your drive is connected properly and that it is turned on. Also, be sure to boot the computer after your 5¼" drive has been connected to the system.

PCCopy

PCCopy lets you copy files from a 5¼" PC disk onto a 3½" Amiga disk. When you open PCCopy, a window appears so that you can enter the names of the files you want to copy. The files in the root directory of the 5¼" disk will be displayed. You can either click on the name of the file to have it appear in the FROM gadget, or type it in. You must also enter the name of the Amiga file you want as the destination. Be sure to enter the complete path of both files.

When copying to an Amiga disk, you cannot create a new directory or subdirectory. Any directories specified must already exist on the disk. To create a new directory, either duplicate the Empty drawer and rename it or use the AmigaDOS MAKEDIR command.

After the correct filenames are entered, click on the COPY gadget. A new box will display:

```
Total bytes to copy <xxxxx>
Bytes copied so far <xxxxx>
```

The number of bytes will be updated throughout the copying process. To exit PCCopy either select CANCEL or click in the window's close gadget.

PCFormat

If you are going to copy a file from an Amiga disk to a 5¼" disk using ToPCCopy, the 5¼" disk must be formatted properly by using PCFormat. When you open PCFormat, a window will appear displaying options for Drive, DOS Version, Sides, and Sectors/Track. In general, you do not need to change the default settings that are present. However, you may want to specify a volume name for the disk. To do this, simply type a name, and it will appear at the prompt in the Volume Name gadget.

To begin formatting the disk, make sure a disk is in the drive and click in the Format gadget. DO NOT start formatting if there is no disk in the drive. A box will appear to let you know that the formatting is in progress:

```
Formatting Track <xx>
Tracks Remaining <xx>
```

After each track is formatted, the utility then goes through and verifies each track. If you need to save time, you can select the Abort gadget during the verifying process.

ToPCCopy

ToPCCopy lets you copy files from a 3½" Amiga disk to a 5¼" PC disk. When you open ToPCCopy, a window appears with gadgets that let you enter the complete path for the FROM (Amiga) file and the TO (PC) file. The files that are included on the Amiga disk will be displayed in the box in the bottom-half of the window. If you click on the name of the file, it will automatically appear in the FROM gadget.

If you want to create a subdirectory on the PC disk, include the new directory in the path you are entering in the TO gadget.

After you have entered the correct file names, click in the COPY gadget. Just as with PCCopy, a box will appear displaying:

```
Total bytes to copy <xxxxx>
Bytes copied so far <xxxxx>
```

This information will be updated as the copying process continues. When copying is complete, click on the CANCEL gadget to exit.

Filtering

When copying vanilla ASCII files between the Amiga and the PC, you may want to take advantage of the text filtering gadget. Your options are Text-7, Text-8, and OFF. When you copy an ASCII file using Text-7, the data does not have the high bit in the byte set. Text-8 includes the high bit.

7. MEMACS

MEMACS (which stands for MicroEmacs and is pronounced M-Emacs) is a screen-oriented editor that allows you to edit multiple files at one time. The only restriction is that the entire body of each file must be able to fit into memory at one time, since MEMACS performs all of its operations on memory resident text.

The length of the lines you can edit is limited to 80 characters. Characters beyond the 80th character of the line are not lost; they simply do not show on the screen. The only way to see those characters is to break the line or to delete some of the displayed characters. When entering new characters, you can keep typing past the 80th character of a line, but what you type will not show on the screen.

You can run MEMACS from both the Workbench and the CLI. From the Workbench, double-click on the icon. From the CLI, the format is:

MEMACS [<filename>] [goto <n>] [OPT W]

[<filename>]   specifies the file to read into MEMACS

[goto <n>]   specifies that the cursor should appear on the nth line of the file

[OPT W]   opens MEMACS in a Workbench window

Notational Conventions and Special Terminology

Throughout this chapter certain key combinations/sequences will be referred to in the following manner:

^(char)   A caret (^) followed by a character is a ''Control-key combination.'' This means that you should hold down the control key as you press the designated character key.

'(char)'   Whenever apostrophes enclose a character, it means that upper or lower case does not matter. It is the keycap itself that selects the function. (This notation is widely used in the summary at the end of this chapter.)

<ESC>   Represents the Esc (Escape) key on your Amiga keyboard.

<DEL>   Represents the Del (Delete) key on your Amiga keyboard.

<TAB>   Represents the Tab key on your Amiga keyboard.

<RETURN>   Represents the Return key on your Amiga keyboard.

There are some special terms associated with MEMACS that you should be familiar with:

Buffer: A memory area that MEMACS controls. There is always at least one buffer used by MEMACS, and it will contain zero or more characters of text.

Dot: The current cursor position.

Mark: A cursor position that you can specify. (Each buffer has its own dot and mark.) The menu item Set-mark allows you to ''mark'' the current cursor position. You can then move forward or backward in the file, adding or deleting text. Then, when you wish to return to the place that you ''marked'', you simply select the Swap-dot&mark command.

You can also set a mark to indicate the beginning of a block of text that you want to duplicate, move, or delete. The ''block'' will encompass all the characters starting with the mark and continuing to the current cursor position.

Kill: Kill commands remove text from the screen and save it in a kill buffer. This text can be retrieved and put back into your document by using the Yank command. As you issue successive Kill commands (without selecting Yank in between), each block of text that you kill will be **added** to the existing text in the kill buffer.

Yank: The Yank command copies the contents of the current kill buffer to the line just above the one in which the cursor is positioned. You can copy a block of text from one buffer to another by killing that block, then, **without moving the cursor,** immediately Yanking it back into the same buffer. Move the cursor to a new position, and choose Yank again. The text you want to copy will still be in the kill buffer and will remain there until you mark and kill another block of text.

Window: A window in MEMACS is somewhat different than an Intuition window in Workbench. In MEMACS, the screen can be split into multiple slices so that you can edit and display more than one buffer or two or more portions of the same buffer. Each ''slice'' is a MEMACS window.

Read a File: When you ask MEMACS to read a file (using the menu command Read-file), the contents of the current buffer are replaced with the contents of the file you want to read.

Visit a File: If you want to access a new file without replacing the contents of the current buffer, you can ask MEMACS to visit a file (using the Visit-file command). MEMACS assigns a new buffer to the file you are visiting.

Select Buffer: You can switch back and forth between the buffers you are working with by choosing the Select-buffer command and specifying the name of the buffer you wish to use. MEMACS sometimes assigns a shorter name to a buffer than the file-name to which it corresponds. Be careful to specify the correct buffer name when you want to switch back and forth.

Modified Buffers: When you make any changes to a buffer, even if you only hit <RETURN> then delete it, ME-MACS remembers and will mark that buffer as a modified buffer.

You can see which buffers have been modified by using the List-buffers command. Any modified buffers are signified with an asterisk (*). If you try to exit MEMACS without saving any changes, a prompt will tell you that modified buffers exist and will ask you if you really want to quit. Once you save a buffer, the modified status is removed.

Opening MEMACS

When you open MEMACS, a new screen appears. At the bottom of this screen are the words ''MicroEMACS — main''. This line displays the name of the buffer that is currently in use. In this case, it is the ''main'' buffer. Remember, a buffer contains zero or more characters of text.

Usually you will invoke MEMACS with a filename specified. If the file exists, it will be read into a buffer. Otherwise, the file will be created when you save your work.

You can have several buffers in use at one time, and you can show one or more on the screen at the same time. Menu options let you switch back and forth between them. At all times, what you see on the screen is what is actually in the buffer.

If the contents of a buffer have been either read from or written to a file, that buffer will be associated with that file. In this case, the bottom line of the screen will display the name of the buffer along with the filename with which it is associated.

MEMACS has two modes of operations: normal and command. When MEMACS is in normal mode, you can:

- move the cursor using the cursor keys
- move the cursor to the edge of the window by holding down the SHIFT key and pressing the appropriate cursor key
- move the cursor by clicking the left mouse button in the desired place on the screen
- insert characters at the current cursor position simply by typing them
- delete the character at the current cursor position by pressing <DEL>
- delete the character to the left of the cursor by pressing <BACKSPACE>
- perform other special functions as explained in the menu section and command summaries that follow

When MEMACS is in command mode, the cursor jumps to the bottom line of the display, and the program asks you for certain additional information. The command mode is entered through various menu items which are explained later.

Mouse Commands

You can also use the mouse to interface with MEMACS. If the MEMACS window is inactive, clicking in it with the mouse will activate the window. You can then move the cursor by moving the mouse's pointer to the spot where you want the cursor to be and clicking the left mouse button. You can also use the mouse to switch between buffers. However, to do this both buffers must be visible on the screen.

MEMACS Menus

MEMACS offers the following main menu items:

Project — system and file-oriented items
Edit — file editing commands
Window — controls the characteristics of the MEMACS windows
Move — controls the placement of the cursor
Line — line-oriented operations
Word — word-oriented operations
Search — search and search/replace options
Extras — controls the numerical value of arguments, and lets you execute a series of operations as though it was a single special command

This section will explain each of these menus and their commands. Each of the commands also has a keyboard shortcut. The shortcuts appear in the menus, to the right of the command, and in this text, along the right-hand margin.

The Project Menu

The commands in the Project menu, except for Visit-file, affect the buffer associated with the current cursor position.

Rename   ^XF

Changes the name of the file associated with the current buffer. This command is useful if you are saving versions of a program or text file as you go along. You can perform a Save command for the first version, modify a few things, Rename the file associated with this buffer, and then save the new version.

When you select Rename, MEMACS prompts:

```
New file name:
```

If you simply press <RETURN> without specifying a filename, the buffer becomes disassociated with any filename. You must specify a name here if you want the buffer to be appropriately associated with a file.

Read-file   ^X^R

Replaces the contents of the current buffer with the contents of a file. When you select Read-file, MEMACS moves the cursor to the bottom line of the display and requests:

```
Read File:
```

Enter the complete path of the file, including the volume name, directory, and file, then press <RETURN>. The file is read into the current buffer, overwriting the data that was stored there.

If you do not want to read a file, simply press <RETURN> without specifying a filename. MEMACS will ignore the request and return you to normal mode.

Visit-file   ^X^V

Lets you work with additional files, aside from the first file you open. You must already be editing something before you can visit another file. This command is useful for programmers who are creating a program and want to extract pieces from or refer to other programs.

When you issue this command, MEMACS moves the cursor to the bottom line and asks:

```
Visit File:
```

Type the complete path of the file, and hit <RETURN>. MEMACS will read the file into a buffer, if it is not already there. If the file you want to visit is on a different disk, AmigaDOS will display a requester asking you to insert that particular disk into any drive. If the file is already in a buffer, MEMACS will switch you to that buffer automatically.

Insert-file   **^X^I**

> *Inserts the contents of a file into the current buffer.* When you issue
> this command, MEMACS moves the cursor to the bottom line and
> asks:

    Insert File:

> Enter the complete path of the file to use, and hit <RETURN>.
> MEMACS will read it into the current buffer at a point one line above
> the current cursor position.

Save-file   **^X^S**

> *Writes the contents of the current buffer to the filename associated*
> *with that buffer.* The filename associated with the buffer was deter-
> mined when the contents of an existing file were read to the file
> (**Read-file**) or when the file associated with the current buffer was
> renamed (**Rename**).

> If there is no filename specified on the status line, MEMACS tells you
> ''No File Name'' and refuses to perform the save.

> After a successful Save, MEMACS uses the bottom line of the screen
> to tell you how many lines it has written out to the designated file.

Save-as-file   **^X^W**

> *Allows you to specify the name of a file to associate with a buffer.*
> When you issue this command, MEMACS prompts:

    Write File:

> MEMACS is requesting the name of the file in which it should save
> the current contents of the buffer. If you provide a complete path
> and press <RETURN>, the buffer will be written out to that file. (If
> you don't provide a name and press <RETURN>, you are returned
> to normal mode.) On the status line for the buffer, the following
> notation will appear:

    File: <filename>

> From now on, that file will be used to save the current contents of
> this buffer when you issue a Save command.

Save-mod   **^X^M**

> *Writes the contents of all modified buffers to the disk.* Use this item
> with caution to be sure that you don't accidentally modify a buffer
> associated with a file you have visited but don't intend to change.

Save-exit   **^X^F**

> *Saves all modified buffers then exits MEMACS.* It is simply a combi-
> nation of the Save and Quit items. Again, use this item with caution
> (see Save-mod).

Brings up an entirely new CLI window called "Spawn Window."
You can issue as many CLI commands in the spawn window as you want without interfering with MEMACS. To return to MEMACS, use the ENDCLI command. The spawn window disappears, and ME-MACS is restored to its previous state.

CLI-Command ^X!

Allows you to execute an AmigaDOS command while you are still in MEMACS. It is similar to issuing a RUN command while in the CLI. When you select this menu item, MEMACS moves the cursor to the bottom of the screen and provides you with a prompt (!). You can then type a command for AmigaDOS to process on this line. ME-MACS temporarily suspends operation, and AmigaDOS executes your command. The output of the command appears in a temporary buffer called spawn.output.

Quit ^C

Exits MEMACS. If one or more of the buffers has been modified since you last saved it to a file, MEMACS prompts:

```
Modified buffers exist, do you really want
to exit? [y/n]?
```

MEMACS is giving you a last chance to save your work. If you don't want to exit, simply press <RETURN>. If you do want to quit, you must press the 'y' key then hit <RETURN>.

Before quitting, you can check which buffers MEMACS is referring to by selecting List-buffers in the Edit menu. MEMACS lists the names associated with each buffer and shows an asterisk by each buffer that has been modified since you last saved it to disk.

There are circumstances under which you will not want to save all buffers back to the original files. For example, let's say you were writing a program and copying pieces from other existing programs as you went along. Some of the files you visited may have been accidentally modified or may have been visited on a write-protected disk.

If you are simply using an old program as temporary source material, you will not want to destroy the original program. When you are finished writing the new program, save your new material and exit MEMACS without saving the modified buffers of the source program.

Two alternate keyboard commands for the Quit command are ^X^C and <ESC>^C.

The Edit Menu

The commands in the Edit menu affect the editing of your buffers and their associated files.

Kill-region   ^W

Deletes blocks of text from the current buffer and saves it in a kill buffer, a special buffer for text that has been deleted from buffers by using Kill commands. (Text can be pulled back into the document by using the Yank command, described below.)

If a block of text has been ''marked'' using the Set-mark command (explained below) and the cursor has been positioned away from the mark, the area between those two points is considered a block and can be deleted by selecting Kill-region.

You can also use Kill-region to copy a block from one section of the buffer to another. Simply mark the block, select Kill-region, then **without moving the cursor,** immediately select Yank. The block will be restored to its original position, but there will also be a copy of the block in the kill buffer.

If you repeatedly select Kill-region on different areas of text, without performing a Yank, each successive kill segment is appended to the kill buffer. When you perform the first Yank, it marks the end of the kill buffer.

Yank   ^Y

Copies the contents of the kill buffer to the line immediately above the current cursor location in the current buffer. Yank reverses the action of Kill-region, but it does not change the contents of the kill buffer. Therefore, you can repeatedly move the cursor to another buffer, select Yank, and copy the contents of the kill buffer. The next time you kill a block of text, however, the contents of the kill buffer will be replaced with the new material, and the old contents will be lost.

Kill-region and Yank are often used together to move text from one buffer to another.

Set-mark   ^@

Marks the cursor position in a buffer. When you select Set-mark, the position of the cursor is **marked** in the current buffer. From then on, any other position of the cursor is referred to as the dot. You can move back and forth between the mark and the dot by selecting the Swap-dot&mark command in the Move menu.

You can use Set-mark to mark the beginning of a block of text that you want to duplicate or move somewhere else in the buffer. Set the mark on the first character you want to include in the block. As you move the cursor through the file, you are essentially blocking out a portion of text.

An alternate keyboard shortcut for Set Mark is <**ESC**>-.

Copy-region   \<ESC\>W

Copies the contents of the marked region to the kill buffer. This new
text replaces any previous contents of the kill buffer.

Upper-region   ^X^U

*Changes the text of the entire marked region, the area between the
mark and the current cursor position (dot), to upper case.*

Lower-region   ^X^L

Changes the text of the entire marked region to lower case.

List-buffers   ^X^B

*Splits the current buffer's window and provides you with a list of the
buffers that MEMACS is currently maintaining.* The list has 4 col-
umns. For example:

```
C   Size    Buffer     File
*   17260   emacs.doc  dfl:docfiles/emacs.doc
```

C is an abbreviation of ''Changed'' and will display an asterisk if the
buffer has been modified since it was last saved to a file.

Size shows how many characters there are in a particular buffer.

Buffer shows the name given to this particular buffer. If you have
read in a file, this will usually be the name of the file itself, minus the
full path. For example, if the file you are editing is dfl:docfiles/e-
macs.doc, then its buffer name will be emacs.doc.

File shows the name of the file, including the full path. This shows
you where MEMACS will write this file if you choose Save-file or
Save-exit while your cursor is in that buffer.

When you choose List-buffers, the status line at the bottom of the
screen displays ''MEMACS — [List]''. Even though List-buffers
brings up a window display, it is not listed as an available buffer. If
you edit the List-buffers window, it can be made to act just like any
other buffer. If, for example, you open a file in the List-buffers
window, the name of the buffer will continue to be [List], and the
name of the file you have opened will become associated with the
List-buffers window.

If you should leave the List-buffers window on the screen but use a
different window to modify the listed buffers, the List-buffers display
will not be continuously changed to reflect the current changes. To
get current information, you must select List-buffers again.

Select-buffer   ^Xb

*Lets you select which buffer you wish to edit in the **currently
selected window,** the window where your cursor is positioned.*
When you choose Select-buffer, MEMACS moves the cursor to the
bottom line and asks:

```
Use buffer:
```

You must provide a name that is the same as one of those shown in the List-buffers listing. If you specify one of the available names, that buffer replaces the contents of the currently selected window.

If you specify a name that is not in the List-buffers listing, you are telling MEMACS to create a new buffer with that name. In this case, there is no filename associated with the new buffer and you will have to rename the file or select Save-as-file when you are prepared to save the buffer's contents to a file.

If you simply press <RETURN>, the command is ignored.

Insert-buffer   <ESC>^Y

Inserts the contents of a named buffer into the current buffer at the line above the current cursor position. When you select Insert-buffer, MEMACS asks:

```
Insert buffer:
```

You must type the name of the buffer to insert, then press <RETURN>.

Kill-buffer   ^Xk

Deletes the contents of a chosen buffer. MEMACS can only edit a file if the entire file will fit in the available memory. To make room in the system's memory, you can use Kill-buffer to delete the contents of one or more buffers. This command returns the buffer's memory to the memory manager for reuse.

When you choose Kill-buffer, MEMACS asks:

```
Kill buffer:
```

You must then enter the name of the buffer you wish to delete. You cannot kill a buffer if its contents are currently displayed.

Justify-buffer   ^XJ

Removes all blank spaces and tabs from the left-hand edge of all the lines in the current buffer. The text is rearranged so that it aligns with the current margins.

Redisplay   ^L

Causes a complete redrawing of the entire screen.

Quote-char   ^Q

Lets you ''Quote'' a character and make it part of the text file. Some keyboard selections have been assigned as MEMACS control characters (for instance, the menu command shortcuts). If you try to insert such a selection into your text, MEMACS will react as if you chose a menu item.

For example, Control-L (^L) tells MEMACS to redraw the display, but ^L is also useful as a printing control to insert a formfeed character. By selecting Quote-char, the next character you type will

be taken "literally" by MEMACS and will be inserted into the text file, instead of being treated as a menu command.

To designate a Control-key combination, use the caret (^) followed by the character, as used in this manual. As MEMACS manipulates the buffer, the combination of the caret and the character is treated as a single character, both by the cursor keys and the character counter.

You can also use Quote-char to insert a <RETURN> key into the text or to insert any other control character that may be needed during a macro command. Even ^Q can be inserted by typing it twice.

An alternate keyboard shortcut for Quote char is ^Xq.

Indent ^J

Moves the cursor to the next line, automatically indenting the same amount of spaces as the previous line.

Transpose ^T

Swaps the positions of two adjacent characters. Place the cursor on the rightmost of the two characters.

Cancel ^G

Ends an ongoing menu command, such as a query search and replace.

The Window Menu

A window in MEMACS is not the same as a window in Intuition (the Workbench's display system). MEMACS splits the screen into multiple slices, allowing you to edit a separate file (buffer) in each MEMACS window. The Window menu lets you control how you view your buffers on the screen.

One-window   **^X1**

Makes the current buffer a single, full-sized window on the MEMACS screen. All other buffers remain invisible, allowing you maximum space to work on the current buffer.

Split-window   **^X2**

Splits the current window in half, positioning the current buffer identically in both windows. This lets you edit two segments of the buffer at the same time. Any changes made in either window affect the entire buffer. This is convenient when you want to see what you wrote in an earlier part of your document while working on a later section.

Next-window   **^Xn**

Moves the cursor ''down'' to the next window and makes that window available for editing.

Prev-window   **^Xp**

Moves the cursor ''up'' to the next window and makes that window available for editing.

The Next-window and Prev-window commands wrap-around. If you move the cursor down as far as it will go, the cursor will automatically move up to the top window. Selecting Prev-window when the cursor is in the top window will move the cursor to the bottom window.

Expand-window   **^Xz**

Adds a line to the current window and simultaneously deletes a line from the adjacent window.

Shrink-window   **^X^Z**

Deletes a line from the current window and simultaneously adds a line to the adjacent window.

Next-w-page   **<ESC>^V**

Displays the next page of the next window. For instance, if you have split a window and are working in the top one, selecting Next-w-page will move the contents of the bottom window (the one you aren't working in) to the next page. This doesn't make the window available for editing; it just lets you view the contents.

Prev-w-page   **^Xv**

Displays the next page of the previous window. If only one window is displayed, it displays the next page of that window.

The Move Menu

The commands in the Move menu let you move the cursor rapidly through the current buffer.

Top-of-buffer   **<ESC><**

Moves the cursor to the top line of the current buffer.

End-of-buffer   **<ESC>>**

Moves the cursor to the bottom line of the current buffer.

Top-of-window   **<ESC>,**

Moves the cursor to the top of the current window.

End-of-window   **<ESC>.**

Moves the cursor to the bottom of the current window.

Goto-line   **^X^G**

Moves the cursor to a specific line number. When you select Goto-line, MEMACS moves the cursor to the bottom of the screen and asks:

```
goto-line:
```

Enter a line number, press <RETURN>, and MEMACS moves the cursor directly to that line. If you specify a line number larger than the total number of lines in the buffer, MEMACS moves the cursor to the last line of the buffer.

Swap-dot&mark   **^X^X**

Places a mark at the current cursor position and moves the cursor to where the mark had been set. If you have not yet set a mark in the window, MEMACS replies, "No mark in this window." This command lets you move quickly to and from a preset location in your buffer. Selecting this item again, restores the cursor to where it was before you selected Swap-dot&mark the first time.

Next-page   **^V**

Moves the text within the window toward the end of the buffer by one full window, less one line. The cursor is repositioned so as to stay on the screen.

Prev-page   **<ESC>v**

Moves the text within the window toward the beginning of the buffer by one full window, less one line. The cursor is repositioned so as to stay on the screen.

Next-word   **<ESC>f**

Moves the cursor forward to the next non-alphanumeric character after the current word.

Previous-word   **\<ESC\>b**

Moves the cursor back to the first letter of the previous word.

Scroll-up   **^Z**

Moves the text within the window towards the end of the buffer by a single line.

Scroll-down   **\<ESC\>z**

Moves the text within the window towards the beginning of the buffer by a single line.

The Line Menu

The commands in the Line menu let you move the cursor within or between lines and let you perform operations involving entire lines.

Open-line   **^O**

Splits the line the cursor is in, forcing the character on which the cursor rests to become the first character of the following line. This command leaves the cursor in the original line so that you can type new characters beginning at the current cursor position.

If you select Open-line by mistake, immediately pressing the \<DEL\> key will close up the line.

Kill-line   **^X^D**

Deletes the line in which the cursor is located and places the text in the kill buffer. If you have not selected Yank since the last Kill command, the text will be **appended** to any existing text in the kill buffer.

Kill-to-eol   **^K**

Deletes the text between the current cursor position and the end of the line. This text enters the kill buffer and will be **appended** to the existing text if a Yank has not been recently performed. The text can also be restored to your file by immediately selecting Yank.

Start-of-line   **^A**

Moves the cursor to the leftmost position on a line.

End-of-line   **^E**

Moves the cursor to the rightmost position on a line. If you have typed more than 80 characters on a line, a dollar sign ($) appears at the right edge of the line. Moving to the end of the line places the cursor logically on the rightmost character even though you cannot see it. Physically the cursor is positioned over the dollar sign. If you use the left-arrow cursor key to move the cursor to the left, it will take as many key presses as there are unseen characters before the cursor actually begins to move.

Next-line   **^N**

Moves the cursor down one line.

Previous-line   **^P**

Moves the cursor up one line.

Line-to-top   **<ESC>!**

Moves the line containing the cursor to the top of the window.

Delete-blanks   **^X^O**

Deletes blank lines, proceeding forward from the current cursor position, until MEMACS gets to the next line on which text exists.

Show-Line#   **^X=**

Displays information on the present cursor position. For example:

```
Line 17 Column 1 (2%)
```

In this example, the cursor is on the 17th line of text, in the first column. The percentage shows that the cursor is in a position 2% of the way from the top of the buffer. In other words, if the cursor was on the last character of text, the percentage would be equal to 100.

The Word Menu

The Word menu contains word-associated operations.

delete-forw   **<ESC>d**

Deletes the character on which the cursor is positioned and all remaining characters to the right until the next non-alphanumeric character is found, (i.e. a blank space, tab, or punctuation mark).

For instance, if the cursor is positioned on the ''s'' in the word **wordsuffix,** choosing delete-forw will delete ''suffix'' from the word. If the cursor is positioned on a blank space, it must be moved forward to the start of a word to delete that word.

delete-back   **<ESC>h**

Deletes the character on which the cursor is sitting and all remaining characters to the left of the cursor until it finds the first character of a word.

An alternate form of this command is **<ESC><DEL>**.

Upper-word   **<ESC>u**

Changes a word to upper case, starting at the character where the cursor is positioned and proceeding to the last character of the word.

Lower-word   **<ESC>l**

Changes a word to lower case, starting at the character where the cursor is positioned and proceeding to the last character of the word.

Cap-word   **<ESC>c**

Changes the character where the cursor is positioned to upper case.
It also changes the characters to the right of the cursor, up to the end
of the word, to lower case.

Switch-case   **<ESC>^**

*Changes the case of a word, starting at the current cursor position
and proceeding to the right until it reaches the end of the word.* If a
word is upper case it changes it to lower case, and vice versa.

The Search Menu

The Search menu allows you to search through the current buffer for
specific text strings. The case (upper or lower) of the string is not signifi-
cant in the search itself. However, if you are using text substitution
(search and replace), the text will be replaced in the same case as that of
the replacement string.

Search-forward   **^S**

*Searches through the text starting at the current cursor position and
moving forward to the end of the buffer.* When you issue this
command, MEMACS moves the cursor to the bottom line of the
screen and asks:

```
Search:
```

Enter the string of characters that you want MEMACS to search for,
and press <RETURN>. If the string is found, MEMACS positions
the cursor immediately following the last character of the string.

If MEMACS cannot find the string, it replies ''Not found.''

An alternate form of this command is **^Xs.**

Search-backward   **^R**

*Searches through the text from the current cursor position back-
wards to the beginning of the buffer.* This command operates in the
same manner as Search-forward. An alternate form of this command
is **^Xr.**

Search-replace   **<ESC>r**

*Operates the same way as Search-forward, except that it allows you
to replace the string with different text.* When MEMACS finds the
first occurrence of a specified string, it asks:

```
Replace:
```

You must enter the string of characters that should replace the found
string. Remember, the characters will appear in the same case as you
type them. When you press <RETURN>, MEMACS will automati-
cally forward-search the rest of the file and replace the search-string
with the replacement-string. After MEMACS completes this com-
mand, it reports:

```
Replaced (xx) occurrences
```

(xx) stands for the number of times the string was replaced.

Query-s-r   **<ESC>q**

Operates the same way as Search-replace, except that it allows you to choose whether or not to replace each occurrence of the string. When you select Query-s-r, MEMACS prompts:

```
Query replace:
```

As it finds a matching string, it always asks:

```
Change string? [y/n/c/^G]?
```

The options are: y (yes); n (no); c (changes all occurrences of the string); and ^G (abort). This gives you a chance to control the replacement process. After MEMACS completes this command, it reports:

Replaced (xx) occurrences

Fence-match   **<ESC>^F**

Finds the next occurrence of a character that is the same as the one at the current cursor position. For instance, if the cursor is resting on an asterisk (*), choosing Fence-match will move the cursor to the next occurrence of an asterisk in the text.

The Extras Menu

The Extras menu contains commands to let you tell MEMACS how to operate. Many of these operational commands require that you specify a numeric argument before selecting the command itself. This menu also includes several macro commands. A macro command is actually a sequence of commands or other keystrokes that are executed by selecting the menu item **Execute-macro.**

Set-arg   **^U**

Lets you specify a numeric argument for the operational commands. When you issue this command, MEMACS responds by moving to the bottom line and prompting:

```
Arg: 4_
```

If you select Set-arg again, MEMACS multiplies the argument value by 4.

If you press a numeric key (0-9), MEMACS accepts an integer argument. If you press a minus sign first, MEMACS accepts a negative integer argument, starting at -1.

Examples: (Each started by a single press of ^U)

Arg: -1   (pressed ''-'' as the first key)

Arg: -23   (pressed ''- 2 3'' as a 3-key sequence)

Arg: 12   (pressed ''3 ^U'' as a 2-key sequence)

MEMACS accepts the argument value as a key for whatever you do next. To add 12 blank lines at the cursor position, specify an argument of 12, then press <RETURN>. To add 20 minus signs, select

an argument number of 20, **do not press** <**RETURN**>, and press the minus sign on the keyboard. (Note: Don't use the keypad's minus sign; it is mapped to a different value.)

To set one of the MEMACS operational parameters (described below), select the value of the argument, **do not press** <**RE­TURN**>, then select the appropriate menu item. MEMACS will use the argument to set the value.

Set   <**ESC**>**s**

Allows you to choose various MEMACS parameters. When you choose Set, MEMACS prompts:

    Set:

You can then enter one of the following:

Screen   places the MEMACS display in a Workbench window or back onto a custom screen

Interlace turns the interlace mode on or off

Mode   results in a second prompt "Mode:"; you can enter **cmode** (for editing c programs) or **wrap*** (to enable automatic wordwrap when the text reaches a set cursor position). Cmode provides automatic fence matching. Use **+ mode** or **− mode** to add or subtract a mode.

Left*   determines the left margin

Right*   determines the right margin

Tab*   sets the increment for tab spacing

Indent*   used in cmode to determine how far to indent each level of nesting

Case   turns case sensitive searches on or off; default is off

Backup   turns on or off MEMACS' backup function. Your options are ON (renames the current file <filename>.bak and saves that backup file to the T: directory), SAFE (this option checks to see if a file already exists for the buffer; if so, it will not overwrite the existing file), and OFF (this is the default option; MEMACS does not perform any backup)

*Each of these entries results in a prompt for a numerical argument, unless the numeric argument is given along with the entry.

Start-macro   **^X(**

Tells MEMACS to start recording any subsequent keystrokes or menu selections. This is a macro command and is used in conjunction with the Stop-macro and Execute-macro commands.

Stop-macro   **^X)**

Tells MEMACS to stop recording keystrokes.

Execute-macro **^Xe**

Repeats keystrokes and menu selections that were entered between Start-macro and Stop-macro. They are repeated as if you had freshly entered the entire sequence.

Set-key **^X^K**

Allows you to redefine all of the function keys, the Shifted function keys, the Help key, or any key on the numeric keypad as keyboard macros. This means that if you select one of these redefined keys while recording macro commands, the new key definition will be recorded in the command. One definition, having as many as 80 keystrokes, can be recorded for each of these keys.

NOTE: If you want to insert the Set-mark command into any of the keyboard macro definitions, you can't use the menu shortcut of ^@. This does not function correctly when used in a macro command. Instead, you must use the alternate form of Set-mark, <ESC>-. This alternate form is acceptable in macro commands.

When you choose Set-key, MEMACS asks:

```
key to define:
```

Press one of the 10 function keys, the Help key, or a numeric keypad key. MEMACS responds:

```
def: [commands]:
```

[commands] is a display of the current commands bound to that key. Enter the new string of characters (up to 80) that you want to have MEMACS respond to when this key is pressed. Pressing <RE-TURN> terminates the entry.

Remember that when entering commands that involve function keys, for example <ESC>< (go to top of buffer), you must use Quote character [^Q] to properly insert the function key keystroke into the definition.

The chart below contains the default values of the function keys when used in macro commands:

KEY	DEFAULT VALUE	KEY SEQUENCE
fkey 1	clone line	^A^K^Y^M^Y
fkey 2	delete line	^X^D
fkey 3	execute keyboard macro	^Xe
fkey 4	next screen	^V
fkey 5	previous screen	<ESC>v
fkey 6	split window	^X2
fkey 7	one window	^X1
fkey 8	scroll window up	^Z
fkey 9	scroll window down	<ESC>Z
fkey 10	save file and exit	^X^F
help	insert line and indent	^J
keypad enter	insert line and indent	^J

The numeric, period, and minus keys on the numeric keypad default to their normal values (i.e. keypad 1 defaults to 1, keypad 2 defaults to 2, etc.).

Reset-keys   **<ESC>k**

Returns any keys defined by Set-keys to their original default state.

Execute-file   **<ESC>e**

Allows you to execute a program file within MEMACS. When you select this command, MEMACS prompts:

```
File:
```

Enter the name of the file you wish to access. This file is executed as a file of MEMACS commands.

Execute-line   **^[^[**

Sets MEMACS to the command mode. When you choose Execute-line, MEMACS asks:

```
execute-line:
```

You can then enter any menu command and its parameters by simply typing it at the prompt. You must use the exact format used in the menus, including hyphens, or you will receive an alert and "command error" message. For instance, you can't type:

```
execute-line: insert file <filename>
```

You must type:

```
execute-line: insert-file <filename>
```

An alternate shortcut for execute-line is **<ESC><ESC>**.

Commands Not Installed in Menus

The following commands have not been installed in menus and are only accessible through the keyboard.

Describe Key   **<ESC>^D**

Tells you if any functions are bound to a key or key-sequence. When you select <ESC>^D, MEMACS prompts for the key to describe. If you enter a key sequence, such as ^L or <ESC>k, MEMACS will respond with the corresponding function. In this case, Redisplay and Reset-keys, respectively.

Bind Key   **<ESC>^B**

Allows you to bind a key to a function. When MEMACS prompts for the key to bind, enter the function (following the format used in the menu items) then the key or key sequence. To check if the key was bound properly, use Describe key (<ESC>^D).

Unbind Key   **<ESC>^U**

Allows you to return a bound key to an unbound state. When MEMACS prompts for the key to unbind, enter the key or key sequence. MEMACS will then reply ''Key is not bound.''

Echo   **<ESC>^E**

Displays the string typed in the command line. This command is usually used when creating or editing executable MEMACS script files.

Move to Edge of Window   **<SHIFT> + Cursor Key**

By holding down the Shift key and a cursor arrow key, MEMACS will move to the top, bottom, left, or right edge of the screen. This is subject to the amount of text available.

Delete the Next Character   **^D**

Deletes the character at the current cursor position. This is the same as hitting the <DELETE> key.

Delete the Previous Character   **^H**

Deletes the character to the left of the current cursor position. This is the same as hitting the <BACKSPACE> key.

Move to Next Line   **^M**

Inserts a newline character after the current cursor position and moves the cursor to the start of the new line.

Move Cursor by x number of Characters   **(forward)**   **^F**
                                                (backward)   **^B**

Allows you to move the cursor forward or backward a specified number of spaces. The default value of this command is one character. However, you can establish a higher value by using ^U to set the argument value. Then select ^F or ^B to move that number of characters.

Adding MEMACS Startup Commands

When MEMACS is opened, it reads the contents of a file called emacs__ pro to see if there are any commands that it should automatically execute. Emacs__pro does not exist; you have to create it. MEMACS first looks in the current directory for emacs__pro. If it is not there, it looks in the disk's s:emacs directory.

You can create more than one emacs__pro file if you wish. You can create a global file that would execute a series of commands each time MEMACS is opened. And, you can create more specified local files with startup commands particular to a certain file that you may use frequently. In the case of both local and global emacs__pro files, the local startup command files override the global file.

Functional Summary of Commands

Operations:

Replace buffer with new file (Read-file) .... ^X^R
Open an additional file (Visit-file) .... ^X^V
Insert a file into current buffer (Insert-file) .... ^X^I
Rename the buffer .... ^XF
Display a list of buffers .... ^X^B
Select a buffer to edit .... ^Xb
Insert contents of buffer into current buffer .... <ESC>^Y
Write buffer to new file (Save-as-file) .... ^X^W
Save buffer to existing buffer .... ^X^S
Save all modified buffers .... ^X^M
Save all modified buffers and exit .... ^X^F
Set numeric argument value .... ^U
Set MEMACS parameters .... <ESC>s
Show the current cursor position (Show-line#) .... ^X=
Run one CLI command .... ^X!
Open a CLI window .... ^-
Cancel a menu command .... ^G
Quit MEMACS .... ^C
                                                ^X ^C
                                            <ESC>^C

Moving Cursor:

Swap dot and mark .... ^X^X
Move forward x number of characters .... ^F*
Move backward x number of characters .... ^B*
Move to start of line .... ^A
Move to end of line .... ^E
Go to a specified line .... ^X^G
Move to next line .... ^N
Move to previous line .... ^P
Move to next page .... ^V
Move to previous page .... <ESC>v
Move to start of buffer .... <ESC><
Move to end of buffer .... <ESC>>
Move to next word .... <ESC>f
Move to previous word .... <ESC>b

*Used in conjunction with Set-arg (^U)

Windows:

Make current buffer a full-sized window (One-Window) .... ^X1
Split current window .... ^X2
Expand window .... ^Xz
Shrink window .... ^X^Z
Move cursor to top of window .... \<ESC\>,
Move cursor to end of window .... \<ESC\>.
Move current line to top of window .... \<ESC\>!
Scroll window up one line .... ^Z
Scroll window down one line .... \<ESC\>z
Move cursor to next window .... ^Xn
Move cursor to previous window .... ^Xp
Redraw the screen .... ^L

Text:

Set a mark in the text .... ^@
                                                          \<ESC\>-
Quote a character .... ^Q
                                                          ^Xq
Justify text in buffer .... ^XJ
Transpose characters .... ^T
Make a word upper case .... \<ESC\>u
Make a word lower case .... \<ESC\>l
Make a region upper case .... ^X^U
Make a region lower case .... ^X^L
Change the case of a word .... \<ESC\>^
Capitalize a word's first letter .... \<ESC\>c
Insert a tab .... ^I
                                                          \<TAB\>
Insert line, cursor is moved to next line .... ^M
                                                          \<RETURN\>
Insert line, cursor is moved to next line and indented same number of
  spaces as previous line .... ^J
Split the line (Open-line) .... ^O

Delete and Copy:

Delete next character . . . . ^D

<DEL>

Delete previous character . . . . ^H

<BACKSPACE>

Delete next word . . . . <ESC>d

Delete previous word . . . . <ESC>h

<ESC><DEL>

Delete text from cursor to end of line . . . . ^K

Delete entire line . . . . ^X^D

Delete region between dot and mark . . . . ^W

Delete blank lines . . . . ^X^O

Delete entire buffer . . . . ^Xk

Copy contents of marked region into kill buffer . . . . <ESC>w

Copy contents of kill buffer into current buffer (YANK) . . . . ^Y

Search:

Search forward . . . . ^S

^Xs

Search backward . . . . ^R

^Xr

Search forward and replace . . . . <ESC>r

Search forward, query and replace . . . . <ESC>q

Fence match . . . . <ESC>^F

Macro Commands:

Start macro commands . . . . ^X(

Stop macro commands . . . . ^X)

Execute macro commands . . . . ^Xe

<ESC>~

Execute line . . . . ^[^[

<ESC><ESC>

Execute file . . . . <ESC>e

Define function keys (Set-key) . . . . ^X^K

Reset keys . . . . <ESC>k

Describe key . . . . <ESC>^D

Bind key . . . . <ESC>^B

Unbind key . . . . <ESC>^U

Echo string . . . . <ESC>^E

Alphabetical Summary of Commands

Remember: When a key is shown enclosed in apostrophes it means that the case of the key does not matter; it is the keycap itself which controls the function.

Control-Key Combinations

Set-mark .... `^'@'`
New-CLI .... `^'_'`
Execute-line .... `^'['`
Start-of-line .... `^'A'`
Move backward x number of characters .... `^'B'*`
Quit MEMACS .... `^'C'`
Delete next character .... `^'D'`
Move to end of line .... `^'E'`
Move forward x number of characters .... `^'F'*`
Cancel a menu command .... `^'G'`
Delete previous character .... `^'H'`
Insert a tab .... `^'I'`
Insert line, cursor is moved to next line and indented same number of
   spaces as previous line .... `^'J'`
Kill-to-eol .... `^'K'`
Redisplay .... `^'L'`
Insert line, cursor is moved to next line .... `^'M'`
Next-line .... `^'N'`
Open-line .... `^'O'`
Previous-line .... `^'P'`
Quote a character .... `^'Q'`
Search-backward .... `^'R'`
Search-forward .... `^'S'`
Transpose .... `^'T'`
Set argument value .... `^'U'`
Next-page .... `^'V'`
Kill-region .... `^'W'`
Yank .... `^'Y'`
Scroll window up .... `^'Z'`

*Used in conjunction with Set-arg (^U)

Control-X/Control-Key Combinations

List-buffers .... ^X^B
Quit MEMACS .... ^X^C
Kill-line .... ^X^D
Save-exit .... ^X^F
Goto-line .... ^X^G
Insert-file .... ^X^I
Define function keys (Set-key) .... ^X^K
Make a region lower case .... ^X^L
Save all modified buffers .... ^X^M
Delete-blanks .... ^X^O
Read-file .... ^X^R
Save-file .... ^X^S
Make a region upper case .... ^X^U
Visit-file .... ^X^V
Save-as-file .... ^X^W
Swap-dot&mark .... ^X^X
Shrink-window .... ^X^Z

Control-X/Key Combinations

CLI-Command .... ^X!
Start macro commands .... ^X(
Stop macro commands .... ^X)
Show-Line# .... ^X=
One-window .... ^X1
Split-window .... ^X2
Select-buffer .... ^X'b'
Execute-macro .... ^X'e'
Rename buffer .... ^X'F'
Justify text .... ^X'J'
Kill-buffer .... ^X'k'
Next-window .... ^X'n'
Previous-window .... ^X'p'
Quote character .... ^X'q'
Search-backward .... ^X'r'
Search-forward .... ^X's'
Next page of previous window .... ^X'v'
Expand-window .... ^X'z'

Escape-Key Combinations

Line-to-top of window .... <ESC>!
Switch-case .... <ESC>^
Set mark .... <ESC>-
Move to top of window .... <ESC>,
Move to end of window .... <ESC>.
Top-of-buffer .... <ESC><
End-of-buffer .... <ESC>>
Bind key .... <ESC>^B
Previous-word .... <ESC>b
Quit MEMACS .... <ESC>^C
Capitalize a word .... <ESC>c
Describe key .... <ESC>^D
Delete next word .... <ESC>d
Echo .... <ESC>^E
Execute-file .... <ESC>e
Fence-match .... <ESC>^F
Next-word .... <ESC>f
Delete previous word .... <ESC>h
Reset-keys .... <ESC>k
Make a word lower case .... <ESC>l
Query, search and replace .... <ESC>q
Search-replace .... <ESC>r
Set .... <ESC>s
Unbind key .... <ESC>^U
Make a word upper case .... <ESC>u
Next page of next window .... <ESC>^V
Previous-page .... <ESC>v
Copy-region .... <ESC>w
Insert-buffer .... <ESC>^Y
Scroll window down one line .... <ESC>z
Execute-line .... <ESC><ESC>
Delete previous word .... <ESC><DEL>

Function Keys

Delete next character .... <DEL>
Delete previous character .... <BACKSPACE>
Insert line, cursor is moved to next line .... <RETURN>
Insert a tab .... <TAB>

A. FOR AMIGA 2000 OWNERS WITH HARD DISKS . . .

If you own an Amiga 2000 and have, or are planning to have, a hard disk installed, there are some aspects of Version 1.3 that pertain especially to you. First, if you have an A2090A Hard Disk/SCSI Controller, you can autoboot directly from your hard disk. Secondly, there is a new FastFile-System, FFS, which increases the speed at which you can access and use your hard disk files.

The A2090A

The A2090A is the new Hard Disk/SCSI Controller card for the Amiga 2000. Once the Version 1.3 Kickstart ROM is in place in your A2000, you can automatically boot from your hard disk. There is no need to insert a Workbench, or other bootable application, disk into your floppy disk drive! The A2090A is autobooting because of the interaction between the 1.3 Kickstart ROM and the two Autoboot ROMs on the A2090A. (Your A2090A may not have these ROMs installed. However, complete installation instructions are included with the A2090A board.)

The FastFileSystem

The FastFileSystem (FFS) is a new file system provided for use with any hard disk that you can connect to your Amiga. The only restriction is that the hard disk driver must provide the standard set of disk device commands and present the media in track and sector format. Since Commodore's Hard Disk/SCSI Controller auto-mounts the default file system on the first partition of a hard disk, it is not possible to use the FastFileSystem there. However, using the new 1.3 Version of the MOUNT command, it is possible to make additional partitions that will use the FastFileSystem.

Since most hard drives use DMA (direct memory access) to transfer data from the disk to memory, the old file system is inefficient since it does not use the hardware to its full potential. FFS addresses this problem by storing nothing but data in the data blocks. Wherever possible, data blocks are allocated consecutively; this means that large reads and writes can be performed in one operation. A pleasant side effect of the new data format is a 4.9% increase in the amount of data that can be stored on a given disk. This amounts to about an extra 50K per Megabyte.

The speed increases are a little difficult to quantify. FFS can read unfragmented files as fast as the disk can transfer the data, and it can write files at about 75% of the maximum disk transfer rate. Validation of a regular sized hard disk (about 20 MB) can be completed about 40 times faster with FFS. As the disk transfer speed increases, FFS keeps up and fully utilizes the data bandwidth of a given controller. Compatibility with most existing software has been maintained, with the exception of some Disk-Doctor type programs that only have knowledge of the old data block format. The DiskDoctor program supplied on the Version 1.3 Workbench disk understands the normal file system and the FastFileSystem.

Mounting FFS

The A2090 driver software auto-mounts the first hard disk partition as a normal file system device. The FastFileSystem may be used on any additional partitions. Since the partitions are mounted using the MOUNT command, a different file system may be specified in the MountList entries. However, these partitions must be (re)formatted in FFS format for use with FFS.

1. First, back up or save off the contents of your hard drive.
2. Boot with Version 1.3 Workbench or copy the new l:FastFileSystem, c:Mount, and c:Format files to your boot disk.

3a. IF YOUR HARD DRIVE IS ALREADY SUITABLY PARTITIONED . . .

If your drive is currently partitioned and the first partition size is acceptable to you as your normal file system partition, you will probably be able to avoid re-prepping and re-formatting your entire drive. (The first partition should usually be relatively small.) If this is the case, you simply need to add three lines to your DEVS/Mount-List entries for each additional Amiga partition(s):

```
GlobVec          =1
FileSystem       =L:FastFileSystem
DosType          =0x444F5301
```

(**WARNING:** Do not change the LowCyl or HighCyl for any partition! If you change them, you must re-prep and reformat the whole drive — see 3b. If you change the names of your partitions, remember to also change the names being mounted in your startup-sequence to match.)

Now, reboot your computer. When your startup script mounts these partitions you will get ''Not A DOS Disk'' requesters. Cancel them, then format the FFS partitions with the new FORMAT command. For example, if your partition is named fs1:

```
FORMAT DRIVE fs1: NAME "FAST1" FFS
```

WARNING: ALL DATA ON THAT PARTITION WILL BE LOST! BE SURE TO HAVE A BACK UP OF YOUR FILES.

3b. IF YOUR DRIVE IS NOT SUITABLY PARTITIONED . . .

If your hard drive is not suitably partitioned, you will have to re-prep your drive and reformat all partitions. Edit your MountList and set up entries for one or more FFS partitions. Your LowCyl and HighCyl entries will be dependent on how many cylinders you plan to specify for the initial auto-mount, normal file system partition during PREP.

Some people prefer a small initial partition (cylinders 2 through 3); others like a first partition of at least 880K to hold the Workbench. The following equations will help you determine how many cylinders will hold Workbench:

BytesPerCyl	= BlocksPerTrack x Surfaces x 512
BytesPerFloppy	= 880 x 1024 = 901120
CylsToHoldFloppy	= BytesPerFloppy / BytesPerCyl (round up the result)

For example, with many 20MB ST506 drives, this works out to 26 cylinders.

Use PREP as described in your A2090 manual and set up your partitions.

Our example drive would be prepped for cylinders 0-1 reserved (RES0:), cylinders 2-28 as first partition (auto-mounted normal file system DH0:), and the remaining cylinders 29-611 for your FastFile-System partition.

Once PREP is complete, reboot the machine, then proceed through the following steps. The examples here assume that you are using an ST506 20 Megabyte drive with the first partition occupying cylinders 2 to 28. **Note: if you are using a different kind of drive you will have to adjust the drive parameters.**

1. Edit your DEVS/MountList file, and add an entry to describe your FFS partition. The GlobVec, FileSystem, and DosType lines are required for any FFS entry. For example:

```
FS1: Device = hddisk.device
     Unit = 1
     Flags = 0
     Surfaces = 4
     BlocksPerTrack = 17
     Reserved = 2
     Interleave = 0
     LowCyl = 29
     HighCyl = 611
     Buffers = 20
     Stacksize = 4000
     GlobVec = 1
     FileSystem = L:FastFileSystem
     DosType = 0x444F5301
```

2. Format your first partition (dh0:) for the normal file system:

```
FORMAT DRIVE DH0: NAME AmigaHD
```

3. Mount your FFS partition:

```
MOUNT FS1:
```

4. Format your FFS partition for the FastFileSystem:

```
FORMAT DRIVE FS1: NAME "FAST1" FFS
```

The FFS option marks the disk as being used with the FastFileSystem. If no keyword is used, FORMAT will check the DosType entry in the MountList to see if an alternate file system is specified. If the FastFileSystem is specified, the FFS option is automatically used. You can use the FFS and NOFFS options to override that process.

5. Edit your startup-sequence and add a line, somewhere after BINDDRIVERS, to MOUNT FS1:.

Using the Recoverable Ramdrive with FFS

You can use the Recoverable ramdrive with the FastFileSystem, but you cannot reboot from it. If you are going to use the Recoverable ramdrive with the FFS, you must change the BootPri entry in the ramdrive.device MountList (RAD:) to -129. This indicates that the ramdrive.device is not bootable.

B. REFERENCE LIST OF DIRECTORIES/FILES ON WORKBENCH

C Directory

ADDBUFFERS	Commands the file system to add cache buffers.
ASK	Asks for user input when executing a script file.
ASSIGN	Assigns a logical device name to a directory.
AVAIL	Reports the amount of CHIP and FAST RAM available.
BINDDRIVERS	Binds device drivers to hardware.
BREAK	Sets attention flags in the specified process.
CD	Sets, changes or lists the current directory.
CHANGETASKPRI	Changes the priority of processes started from the CLI.
COPY	Copies a file or directory.
DATE	Displays or sets the system date and/or time.
DELETE	Deletes up to 10 files or directories.
DIR	Displays a sorted list of the contents of a directory.
DISKCHANGE	Informs the Amiga of the changing of a disk in a 5¼" drive.
DISKDOCTOR	Attempts to repair a corrupt floppy disk.
ECHO	Prints a string.
ED	Edits text files (a screen editor).
EDIT	Edits text files by processing the source file sequentially.
ELSE	Allows an alternative in a conditional in a script file.
ENDCLI	Ends an interactive CLI or SHELL process.
ENDIF	Terminates an IF block in a script file.
ENDSKIP	Terminates a SKIP block in a script file.
EVAL	Provides a means of evaluating simple expressions.
EXECUTE	Executes a command file with argument substitution.
FAILAT	Sets the failure condition of a script file.
FAULT	Prints the specified error message(s).
FF	Speeds up the display of text.
FILENOTE	Attaches a comment to a file.
GETENV	Gets the value of an environment variable.
ICONX	Executes an AmigaDOS script file from the Workbench.
IF	Handles conditional operations in script files.
INFO	Gives information about the file system.

INSTALL	Handles the boot block of a formatted disk.
JOIN	Links together up to 15 files to make a new file.
LAB	Specifies a label in a script file.
LIST	Lists specified information about directories and files.
LOADWB	Starts Workbench.
LOCK	Sets the write-protect status of a hard disk using FFS.
MAKEDIR	Creates a new directory.
MOUNT	Creates an AmigaDOS device node.
NEWCLI	Starts a new interactive CLI with a new window.
NEWSHELL	Starts a new interactive SHELL with a NEWCON: window.
PATH	Controls the path that the CLI uses to search for commands.
PROMPT	Changes the prompt string of the current SHELL.
PROTECT	Changes the protection bits of a file.
QUIT	Exits from a script file with the specified error code.
RELABEL	Changes the volume name of a disk.
REMRAD	Removes the recoverable ramdrive.device.
RENAME	Changes the name of a file or directory.
RESIDENT	Loads and adds commands to the resident list.
RUN	Executes commands as background processes.
SEARCH	Looks for the specified text string in the specified files.
SETCLOCK	Sets or reads the real time clock.
SETDATE	Changes the date and time of a file or directory.
SETENV	Sets the value of an environment variable.
SETPATCH	Makes ROM patches in Version 1.2 and Version 1.3 Kickstarts.
SKIP	Skips ahead when executing command files.
SORT	Alphabetically sorts each line of a small file.
STACK	Displays or sets the stack size of the current CLI.
STATUS	Lists information about the CLI/SHELL processes.
TYPE	Types a text file.
VERSION	Finds the version and revision numbers of Workbench, libraries, or devices.
WAIT	Causes AmigaDOS to wait a specified amount of time.
WHICH	Searches the command path for a specified command.
WHY	Prints an error message explaining why a command failed.

Devs Directory

clipboard.device	Controls writing and reading clips to CLIPS: or DEVS:clipboards.
MountList	Contains the descriptions of devices to be mounted.
narrator.device	Controls access to the speech synthesizer.
parallel.device	Controls access to the parallel port.
printer.device	Controls access to the printer device.
ramdrive.device	Controls access to the recoverable ramdrive.
serial.device	Controls access to the serial port.
system-configuration	Contains the current Preferences settings.
Keymaps subdirectory	Contains the usa1 keymap file.
Printers subdirectory	Contains the generic printer driver.
Clipboard subdirectory	Clips are stored here if CLIPS: is not assigned.

Fonts Directory

Fonts	*Sizes*
Ruby	8, 12, 15
Opal	9, 12
Sapphire	14, 19
Diamond	12, 20
Garnet	9, 16
Emerald	17, 20
Topaz	11

L Directory

Aux-Handler	Provides unbuffered serial input and output.
Disk-Validator	Validates disks for writing when they are inserted into a drive.
FastFileSystem	A file system for use with hard drives.
Newcon-Handler	Provides an interactive SHELL window.
Pipe-Handler	Provides an interprocess communication channel between programs.
Port-Handler	Handles the DOS interface for PAR:, SER:, and PRT:.
RAM-Handler	Provides file storage in memory.
SHELL-SEG	Controls the SHELL.
Speak-Handler	Provides speech output for the Amiga.

Libs Directory

diskfont.library	Contains functions for finding and loading resident fonts.
icon.library	Contains functions for creating and reading icon files and string manipulation routines.
info.library	Contains code for the Info function of Workbench.
mathieeedoubbas.library	Contains double precision IEEE basic mathematics functions (addition, subtractions, etc.).
mathieeedoubtrans.library	Contains double precision IEEE transcendental math functions.
mathtrans.library	Contains mathffp transcendental math functions (sine, cosine, etc.).
translator.library	Contains a routine for translating english text into phonemes suitable for the narrator device.
version.library	Contains the version and revision numbers of the Workbench release.

Prefs Directory

CopyPrefs	Copies the system-configuration from a hard disk to the Workbench disk.
Preferences	Allows you to change various settings of the Amiga, including the screen colors, what printer driver is used, the shape and color of the pointer, and serial device settings.

S Directory

CLI-Startup	The file which is executed whenever a new CLI is started.
DPAT	A script file which can add pattern matching to double argument commands.
PCD	A script file, similar to the CD command, which remembers the last directory.
SHELL-Startup	The file which is executed whenever a new SHELL is started.
SPAT	A script file which can add pattern matching to single argument commands.
Startup-Sequence	The file which is executed whenever the Workbench disk is loaded.
Startup-Sequence.hd	A startup-sequence that transfers control to a hard disk.
StartupII	The file which is run by the initial startup-sequence and which makes many AmigaDOS commands resident.

System Directory

CLI	Allows direct communication with the Amiga through AmigaDOS commands.
DiskCopy	Copies the contents of one disk to another.
FastMemFirst	Changes the order of allocation of memory putting FAST RAM first.
FixFonts	Updates the .font files in FONTS:.
FORMAT	Initializes a disk so that it can be used with the Amiga.
InitPrinter	Initializes a printer so that it can be used with the Amiga.
MergeMem	Merges the memory lists of sequentially configured RAM boards.
NoFastMem	Forces the Amiga to use only resident CHIP RAM.
SetMap	Sets the keymap to be used with a keyboard.

Utilities Directory

Calculator	A standard four-function calculator.
Clock	Displays a digital or analog clock on the Workbench screen.
ClockPtr	Changes the Workbench pointer into a digital clock.
CMD	Redirects serial or parallel output to a file.
GraphicDump	Performs a graphic dump of the frontmost Intuition screen.
InstallPrinter	Copies a printer driver from Extras to the Workbench.
MORE	Displays ASCII text files.
Notepad	Allows the creation of short text files.
PrintFiles	Copies multiple files to the printer.
Say	A speech synthesizing program.

Please note that the .info files are not referenced in this section. Files with .info suffixes contain the information needed to display icons and windows and to start applications. Most users do not need to access these files.

C. REFERENCE LIST OF DIRECTORIES/FILES ON EXTRAS

Files included in the FD1.3 and Basic Demos directories pertain to the programming language Amiga Basic. For more information on these directories, please refer to the Amiga Basic manual which was packaged with your Amiga computer. All other directories on Extras are referenced in this appendix.

Devs Directory—Keymaps subdirectory

Keymap	Compatible Keyboard
cdn	French Canadian
ch1	Swiss French
ch2	Swiss German
d	German
dk	Danish
e	Spanish
f	French
gb	Great Britain
i	Italian
is	Icelandic
n	Norwegian
s	Swedish
usa2	Dvorak

Devs Directory—Printers subdirectory

Printer Driver	Compatible Printers
Alphacom—Alphapro—101	Alphacom Alphapro 101
Brother—HR-15XL	Brother HR-15XL
Calcomp—ColorMaster	ColorView-5912
Calcomp—ColorMaster2	ColorView-5912
Canon—PJ-1080A	Canon PJ-1080A
CBM—MPS1000	CBM MPS1000, Canon BJ-130, IBM 5152 Plus printers
Diablo—630	Diablo 630
Diablo—Advantage—D25	Diablo Advantage D25
Diablo—C-150	Diablo C-150
EpsonQ	Epson Q series (LS1500, LQ2500, etc.)
EpsonX	Virtually any Epson compatible printer, Commodore MPS1250
EpsonXOld	Epson X series, Star Micronics Gemini 10-X
Howtek—Pixelmaster	Howtek Pixelmaster
HP—DeskJet	Hewlett-Packard DeskJet
HP—LaserJet	Hewlett-Packard LaserJet, LaserJet Plus and LaserJetII
HP—PaintJet	Hewlett-Packard PaintJet
HP—ThinkJet	Hewlett-Packard ThinkJet
ImagewriterII	Imagewriter, Imagewriter II
Nec—Pinwriter	NEC Pinwriter P5, P6, P7, P9, P2200
Okidata—92	Okidata 92
Okidata—2931	Most Okidata printers with the IBM Plug-n-Print card.
Okimate—20	Okimate 20 (Both Serial and Parallel Plug'n Print Kits)
Quadram—QuadJet	Quadram QuadJet
Qume—LetterPro—20	Qume LetterPro 20
Toshiba—P351C	Toshiba P351C, other Toshiba 24-pin printers
Toshiba—P351SX	Toshiba P351SX, other Toshiba 24-pin printers

Fonts Directory

Fonts	Sizes
Courier	11, 13, 15, 18, 24
Helvetica	9, 11, 13, 15, 18, 24
Times	11, 13, 15, 18, 24

PCUtil Directory

PCCopy	Copies files from a 5¼" PC disk to a 3½" Amiga disk.
PCFormat	Initializes a 5¼" disk for use with ToPCCopy.
Read Me	Contains text describing PCCopy, PCFormat and ToPCCopy.
ToPCCopy	Copies files from a 3½" Amiga disk to a 5¼" PC disk.

Tools Directory

FED	Modifies and creates fonts.
FreeMap	Displays chip memory usage.
IconEd	Changes the appearance of icons.
IconMerge	Merges or splits icon images.
KeyToy2000	Displays the global keymap of an Amiga 2000.
MEMACS	Edits text files.
Palette	Changes the colors of an Intuition screen.
PerfMon	Monitors the system's performance.

Please note that the .info files are not referenced in this section. Files with .info suffixes contain the information needed to display icons and windows and to start applications. Most users do not need to access these files.

INDEX