ALTOS

Uniplex

System Administrator's Guide

UNIPLEX SYSTEM ADMINISTRATOR'S GUIDE

TABLE OF CONTENTS

INTRODUCTION

This manual has been designed to help you get the best out of Uniplex. The information has been broken down into 2 sections.

These sections cover:

Installation of the Word Processor

Configuration of the Word Processor

INTRODUCTION

This section of the manual lists the files required for the Uniplex module and describes how these files should be installed.

UNIPLEX MODULES

The Uniplex Word Processing module incorporates the following:

The Menu System
Uprint, the print program
Rmerge, the Mail Merge program
Viewprint, print and mail merge screens

INSTALLATION PROCEDURE

Uniplex can be installed through the Altos Business Solution version by following the procedures in the Altos Xenix Applications Software Guide.

To install Uniplex outside of the Altos Business Solution:

1. Login to XENIX as the root user.

2. Type **umask 0**

2. At the # prompt type **cd /usr**

NOTE

If you have limited space in /usr, you may want to put Uniplex in another directory, for example in **/usr2**. To do so type **cd /usr2** at the # prompt.

3. Insert the Uniplex diskette in the floppy drive.

4. At the # prompt type **tar xv**.

Installing the dictionary will write over the following files, if they exist: /usr/dict/words, usr/dict/hlista, /usr/dict/hstop, and /usr/dict/spellhist. If you want to install the dictionary for spelling checking, put the dictionary floppy disk drive and type : **tar xv**

5. Using the table on the next page, verify that the file names have been put in the correct dictionaries and have the correct permissions.

NOTE

If the Uniplex directory is in /usr (i.e. you typed cd /usr in step 2), type : **rm uniplex/Redirect**

6. Change from your current directory to the Uniplex subdirectory by typing **cd uniplex** at the # prompt.

7. Run the install program by typing **install** at the # prompt. The install program will prompt you for information about your printers.

To reconfigure printers (if printers are added or changed after installation, change directories to the uniplex directory and type **reconfig.**

INSTALLING NEW VERSIONS OF XENIX

Before you install new versions of Xenix it is **very important** that you backup the following files: **usr/lib/lpd** and **/bin/lpr.** Follow the procedure below:

1. Insert a blank formatted diskette in the floppy disk drive.

2. Type **tar cv usr/lib/lpd /bin/lpr.**

3. After installing Xenix, restore the above files by inserting the diskette and typing **tar xv.**

THE UNIPLEX FILES

The table below contains all the files that should be on your system, what the functions of these files are, and where they should be located. To verify that the installation process has established the correct filenames and permissions for your system, check them against the table. If there is a pathname in the file /usr/uniplex/Redirect, any directory specified below as /usr/uniplex should be specified "Redirect pathname"/uniplex. (eg. "/usr2/uniplex").

Of the files below, the files **termcap.aom** and **printers** are **UNIX*** files, all other files are specific to the **UNIPLEX** product. The directories these files are in should have general read-execute permission. As a general rule, all binary files are stored in /usr/bin and all other files are located in /usr/uniplex.

NORMAL PERMISSION REQUIRED	FILE	TYPE OF FILE	LOCATION OF FILE	PURPOSE
-rw-r--r--	termcap.aom	text	/etc	Includes terminal variables
WORD PROCESSING FILES				
---x--x--x	uniplex	binary	in /usr/bin or in /usr/uniplex and link with /usr/bin	Operates editor, menu and command file - uniplex.rc
-rwxr-xr-x	vmi	script	in usr/bin	Script to run Uniplex
---x--x--x	uprint	binary	in /usr/bin or in /usr/uniplex and link with /usr/bin	Operates printing
-rwx--x--x	lpr	binary	in /bin	Print spooler
-rwx--x--x	lpd	binary	in /usr/lib	Print daemon
-x--x--x	cutspool	binary	in /usr/bin	Printer for interactivity
---x--x--x	Rmerge	binary	in /usr/bin or in /usr/uniplex and link with /usr/bin	Operates mail merge

*UNIX is a trademark of Bell Labs

WORD PROCESSING FILES continued

-rw-r--r--	key	text	/usr/uniplex	Security file prevents unauthorized use
-rw-r--r--	uniplex.help	text	in /usr/uniplex or in local uniplex directory	Help file for Uniplex editor
-rw-r--r--	uniplex.rc	text	in /usr/uniplex or in local uniplex directory	Command file Configures Uniplex - menus and commands.
---x--x--x	unidir	binary	in /usr/bin	finds uniplex directory
-rw-r--r--	Tcap	text	in /usr/uniplex	Peripheral characteristics file. Sets terminal and printer effects
-rwxr--xr--x	uspell	script	in usr/bin	Special Uniplex shell script to run Unix spell
---x--x--x	vpc	binary	in /usr/bin or in /usr/uniplex and link with /usr/bin	Viewprint compiler vpc -p compiles print screen vpc -m compiles mail merge screen
---x--x--x	vp	binary	in /usr/bin or in /usr/uniplex and link with /usr/bin	Viewprint runtime vp -p = print runtime vp -m = mail merge runtime
-rw-r--r--	vmerge	text	in directory where you execute vpc -m	Source merge screen
-rw-r--r--	vprint	text	in directory where you execute vpc -p	Source print screen
-rw-r--r--	vmerge.vp	control text	in /usr/uniplex or in local uniplex directory	Compiled merge screen
-rw-r--r--	vprint.vp	control	in /usr/uniplex	Compiled print

NORMAL PERMISSION REQUIRED	FILE	TYPE OF FILE	LOCATION OF FILE	PURPOSE
		text	or in local uniplex directory	screen
WORD PROCESSING FILES continued				
-rw-r--r--	vp.help	text	in /usr/uniplex or in local uniplex directory	Help file for merge and print screens
-rw-r--r--	vp.message	text	in /usr/uniplex or in local uniplex directory	Message file for the Viewprint run time (vp) User definable
-rw-r--r--	vpc.message	text	in /usr/uniplex or in local uniplex directory	Message file for the Viewprint compiler (vpc) User definable
-rw-r--r--	mark move print find alter cursor delete emphasis exit insert merge modes often prim1 prim2 prim3 prim4 rulers scroll	text	/usr/uniplex/help	Word processing help files. These files are optional and need not necessary be installed.
rwx--x--x	tail	binary	in /bin	puts up calendar
rwx--x--x	head	binary	in /usr/bin	view end of file
rwx--x--x	wc	binary	/usr/bin	word count

NORMAL PERMISSION REQUIRED	FILE	TYPE OF FILE	LOCATION OF FILE	PURPOSE
WORD PROCESSING FILES continued				
rwx--x--x	spell	binary	/usr/lib	part of spelling checker
rwx--x--x	deroff	binary	/bin	part of spelling checker
rwx--x--x	tee	binary	/bin	part of spelling checker
rwx--x--x	cal	binary	/bin	calendar display
-rw-rw-rw-	spellhist	text	/usr/dict	spell history file
-rw-r--r--	words	text	/usr/dict	words used
-rw-r--r--	hstop	hash table	/usr/dict	pointers to invalid spelling
-rw-r--r--	hlista	hash table	/usr/dict	pointers into words file to valid spellings
-rwxr-xr-x	prconf	binary	/usr/bin	reconfigures printers for Uniplex changes /etc/printers
-rwxr-xr-x	prinit	binary	/usr/bin	original printer configuration on installation changes /etc/printers
-rwxr-xr-x	pfinit	binary	/usr/bin	moves baud rate into form etc/ttys to /etc/printers
-rw-r--r--	printers	text	/etc	printer description created by prinit and precon
-rw-r--r--	printers	text	/usr/uniplex	predefined printers used by prinit and prconf
-rwxr-xr-x	reconfig	script	/usr/uniplex	reconfigure printers

NORMAL PERMISSION REQUIRED	FILE	TYPE OF FILE	LOCATION OF FILE	PURPOSE
drwxrwxrwx	lock	directory	/usr/uniplex	avoids file editing conflicts - should be cleared after any system crash.
WORD PROCESSING DEMONSTRATION FILES				
-rw-r--r--	uni.intro	text	in /usr/uniplex/demo	Introduction demo
	uni.adv	text		Full features WP demo.
	system	text		Text editing
	thanks	text		Paragraphs to merge
	pleased	text		into file.
	finance	text		Tabular text to merge into file.
	h	text		Pre-stored header and footer.
	s	text		File with stored standard text.
	printime	text		File with all print format commands.
MAIL MERGE DEMONSTRATION FILES				
-rw-r--r--	mail.list	text		Records
-rw-r--r--	std.let	text		Standard letter

All directories should have general read execute permission and should be owned by root.

When limited space is available in /usr on your system, you can create a new file called **/usr/uniplex/Redirect.** The contents of this file should be a valid pathname of the directory that will contain the Uniplex files. For instance, if you want the files to be in a directory called **/WP/files,** the files should be placed in **/WP/files/uniplex.** Uniplex expects to find the files in the uniplex sub-directory, so in this case, the entry should appear in Redirect as simply **/WP/files.**

NOTE: Improperly formatted text in the word processor is often caused by Uniplex's inability to identify the terminal being used. This is often the result of supplying an incorrect TERM environment variable on login.

DEMONSTRATION COPIES

Uniplex has been supplied with a serialization keyfile, which has been encoded with the serialization number of your system. This is /usr/uniplex/key. If the key file is not found or is unreadable, Uniplex reverts to demonstration mode. When in demonstration mode the following differences apply:

1. The line at the top of each menu will read 'DEMONSTRATION COPY'

2. When creating or editing a file, the user will be warned that Uniplex is in demonstration mode.

3. A maximum of 24 lines will be written to any output file.

4. Only 1 paste to an named file will be allowed.

INTRODUCTION

This section describes how to configure the word processor using the Uniplex Menus and Viewnix Screens.

The Menu System and Viewnix Screens are used to provide a friendly user interface to the word processor.

Redesigning the system can be done by simply editing a text file, using Uniplex Word Processing or any other standard text editor. To reconfigure the system, help files and command files can be customized. These files can be stored in the users local uniplex directory, as well as their respective system directories. This enables individual users to have commands and help screens personal to themselves. The system first looks for the files in the users local uniplex directory.

The following chapters explain how to reconfigure each of the modules.

MENUS AND UNIPLEX WORD PROCESSING

The table below explains which aspect of the system is configured by each Uniplex Word Processor file. All these files are stored in /usr/uniplex or <home>/uniplex, with the exception of the Tcap file which can be stored only in /usr/uniplex. Tcap configures terminal and printer attributes. It is not necessary for users to have a local Tcap file.

FILENAME	CONFIGURES
uniplex.rc (the Command File)	Menus System messages Editing keystrokes
Tcap	Terminal and Printer attributes
uniplex.help	Uniplex help file
<various help files>	Optional help files accessed from from the menu system stored in /usr/uniplex/help directory.

THE COMMAND FILE

Most user-variable Uniplex Word Processing parameters are contained in the **Command File** module. The Command File is composed of sections that define how Uniplex Word Processing appears to the user -- all menus, the word processing system messages, the keystrokes necessary to initiate editing commands, and certain general aspects of the editing environment.

This chapter provides an overview of the Command File and describes syntax that is common to all sections of the Command File. Subsequent chapters discuss each section of the Command File in detail. Refer to Appendix A for a listing of the Standard Uniplex Command File.

NOTE: The Standard Uniplex Command File distributed with Uniplex is used in this guide as the basis for all explanatory Command File syntax. Modified sections are used to illustrate customized command files.

IMPORTANT
Always make a copy of the Command File before making any editing changes. Then, edit the copy, saving the original as a back up.

Command File Syntax

The command file is organized into units called **sections;** each section is a collection of parameters that covers a particular area of the software.

For instance, the #COMMANDS section contains definitions of keystrokes that initiate Uniplex editing commands. Two others, #USER and #MESSAGES, contain all system messages.

There are three basic types of Command File sections:

1) Those sections that are necessary to the operation of Uniplex, (noted as mandatory sections below).

2) Additional #COMMANDS sections that define command keystrokes for particular terminal types.

3) The sections that define menus.

Each component of a Command File section is called an **entry.** Each section entry must conform to the following rules:

Each entry must start at column one.

Each entry is limited to a single line of text; the Uniplex text editor handles lines of up to 256 characters.

Individual definitions on a single line are separated by a comma.

Literal strings are deliminated by single quotes.

The first and last entry of each section mark the beginning and end of each Command File section. The first section entry is the **section header,** which identifies the section.

The last section entry is always two right-hand parentheses. This concludes the section.

Section headers always start at column one and are unique to the current Command File, and begin with the crosshatch symbol (#). Headers can contain any combination of spaces and upper and lowercase alphanumeric characters.

Header names for mandatory sections cannot be changed. Mandatory sections are: #SYSTEM, #COMMANDS, #SYSMENU, #USER, #EDITMENU, #MESSAGES, #RULERS, and #EFFECTS. Other command file section headers can be changed as desired.

NOTES:

Between the end of one section and the beginning of the next, blank lines and remarks can be included to enhance readability.

For the sake of consistency, standard command file sections are identified by upper-case name descriptions that describe the section contents.

The header and double right parentheses conventions make each Command File section easily accessible by Uniplex's **Text Merge** feature. Command File sections can be automatically copied from other files. For details, see the **Uniplex User's Guide.**

Standard Command File Sections

The Standard Uniplex Command File contains 23 sections, which define menus or keystrokes. Of these, seven are mandatory (in some form) for any command file:

#SYSTEM	General parameter definitions
#MESSAGES	Text for system messages
#COMMANDS	Default keystroke definition table
#RULERS	Default ruler settings
#EFFECTS	Printer effects defined for system
#SYSMENU	Main Menu definition
#USER	Messages for menu usage

Six other sections define menus for the integrated standard release:

#EDITMENU	Document escape menu (escape to menu from within the editor)
#DOCPREP	Word Processing menu
#SYSCOMMS	System access menu
#CHECKER	File and checking menu
#FILEMANAGE	File management menu

You also have the following two help related sections:

#HELP	Help menu
#MOREHELP	Command summary menu

Other sections in the Standard Integrated command files include keystroke definitions for various terminals:

#COMLOCATE - Keystroke cross reference table
#COMMANDS - alt2
#COMMANDS - vtl00
#COMMANDS - wyse
#COMMANDS - tvi912
#COMMANDS - tvi910
#COMMANDS - tvi950
#COMMANDS - tvi925
#COMMANDS - viewpoint

All sections of the command file are read in when Uniplex is invoked, with the exception of the menu section. Because the menus are being continually accessed, it is sensible to locate the most often used menus near the beginning of the Command File.

It is obligatory that the first section of the Command File is #SYSTEM. Therefore, menus sections are best located after this section.

In the following chapters, each of these sections is fully described along with selected examples of customized sections.

SUMMARY

* Most user-variable Uniplex parameters are found in the **command file**, which defines all menus, system messages, and keystrokes necessary to initiate editing commands.

* Another Uniplex module is Tcap, which controls the terminal and printer effects.

* The Command File is an ordinary text file and can be edited using Uniplex or any other text editor.

* Multiple command files can be used, each one addressing different user levels.

* The command file is organized into units called **sections**, each one a collection of parameters that covers a particular area of the software.

* Sections can be separated by blank lines and comments.

* Each menu is defined in an individual section.

* Each part of a section is called an **entry**.

* Entries must begin in column one.

* The first entry in any section is the section header, which must be unique to the current command file, and must begin with the crosshatch symbol (#).

* Each Command File section is concluded with two right parentheses in column one on a separate line.

* Command File sections can be copied from other files using Uniplex's **Text Merge** facility.

* Although a section entry is limited to a single line, lines can be of up to 252 characters.

* Seven sections constitute the minimun requirement for any Command File. In the Standard Uniplex Command File, twenty three sections are configured.

MENUS

Sections defining menus, like all Command File sections, are easily edited. The #SYSMENU section defines the Main Menu that is the gateway to the entire menu system. By modifying this menu, and creating others, hierarchies can be developed to tailor Uniplex to individual requirements.

General Rules of Menu Definition

All Command File sections begin with the section header entry. As explained in Chapter 2, section headers must start with the crosshatch character in column one. (Technically, of the menu sections, only the mandatory #SYSMENU section needs begin with the crosshatch; however, it is recommended that this convention be preserved in all sections for consistency). Refer to Chapter 2 for more about section headers.

The second line of a menu section specifies the menu title. This menu name is delimited with single quotation marks. The menu title appears on line four of the screen when the menu is displayed.

The last line of a menu section must contain two righthand parentheses starting in column one.

Menu sections between the second and last line are comprised of a series of entries, each of which defines an option (or remark) of the menu. Each entry is made up of two parts: the first part is the **option** as it appears on the menu, and the second is the **action** that Uniplex is to take when the option is selected.

The two parts are separated by an equal sign (=). The form for menu section entries is:

Menu Option=Action

Each are defined more explicitly below.

SUMMARY OF MENU CONSTRUCTION

Line one is the header label.
Line two is the title line and is printed on line four of the screen.
Line three onward are menu options or comment lines.
Last line must be)) at the start of the line.

MENU OPTIONS

Uniplex recognizes the **first** character of the Menu Option as the trigger that causes it to carry out the option.

Uniplex automatically converts lower-case characters received into upper-case, so users need not concern themselves with shift keys. Lower-case option characters are not permitted within menus, but all upper case alphanumerics, and other printable characters may be used.

Menu options can be of two types, visible or hidden.

A visible menu option is a remark or description of the action to be performed. As a literal string it should be enclosed in single quotes. A visible menu option appears on the screen when the menu is selected. A hidden option is not displayed as a menu option when the menu is selected. The hidden value is defined as its ASCII value equivalent and represents a single character keystoke sequence.

Using Hidden Options, direct access to Uniplex facilities can be provided without disturbing the menu environment important to many users.

ACTION

The second part of the menu section entry defines the **type** of action to be taken. These types of actions can be specified as follows:

Meaning	Action
Jump to a menu	M('menu name')
Comment line	>()
Go back to last menu	^()
Change directory	X() or X('directory name')
List files or a directory	?() or ?('directory name')
Create a file	C() or C('filename')
Edit a file	E() or E('filename')
Duplicate a file (**cp**)	D()
Rename a file (**mv**)	R()
Kill a file (**rm**)	K()
Do this UNIX command line	*('unix line','prompt no','P/R')
Exit from Menu System	*()

Information necessary for a particular action is included in parentheses and is delimited by quotation marks. If more than one piece of information is necessary, they are separated by commas.

As an example, consider the first menu option entry in #SYSMENU:

'1 - Word Processing Menu'=M('#DOCPREP')

The first part of this entry,

'1 - Word Processing Menu'

is the option as it appears on the menu. The option is surrounded by single quotations, as are all literal strings in the Command File.

Uniplex recognizes the first character of the Menu Option as the **trigger** that causes it to carry out the option. Here, the trigger is the number **1**; when the user presses **1** at this menu, Uniplex carries out the action that follows the equal sign.

The second part of the menu option entry is the **action definition**, which in the above example is:

M('#DOCPREP')

The first character following the equal sign, in this example, **M**, defines the **type** of action to be taken.

The **M** action causes Uniplex to display the menu indicated in parentheses. Here, this is the #DOCPREP menu:

M('#DOCPREP')

Some actions do not require any information. For instance, consider this entry from the #SYSMENU section:

'D - Change Directory'=X()

Here, the **X** defines the action, which is change directory. This option, as defined above, causes Uniplex to prompt for a directory name. Since no extra information is required, the parentheses are empty.

An example of a hidden option would be:

3=C()

The numeral **3** is a ASCII code equivalent for (Ctl) C, which is what the user types to select the option. The option in this case is Create a File. A table of characters and their equivalent codes can be found in Appendix B.

In the remaining part of this chapter, the action definitions are described, using examples from standard Command File menus.

M Action: Jump to a menu

The general form is:

M('menu name')

This action provides the means to link menus together, with options on one menu invoking other menus.

The **M** action causes Uniplex to display the menu indicated.

For example:

'2 - File Management Menu'=M('#FILEMANAGE')

As soon as the number '2' is entered, Uniplex searches for the #FILEMANAGE menu in the Command File and displays it on the screen.

In menu action definitions, the section header of the menu being jumped to is included between parentheses.

If Uniplex is unable to find or read the menu specified between parentheses, the following message from the #MESSAGES section is displayed:

Selected menu not available or badly formatted.

This message, like all Uniplex messages, can be rewritten.

> Action: Comment line

The general form is:

>()

The Comment Action is used to enhance the visual appearance of menus by insertion of either blank or text lines.

For example, to cause blank lines between entries in a menu:

' '=>()

Note that a single blank space is required between the quotation marks when specifying blank lines.

The Comment Action can also be used to add text lines to menus.

For example:

'----- Documents & Files -----'=>()

and

'------ Directories ------'=>()

or

'Press (Escape) to go back a menu'=>()

The comment lines are used to either visually improve the menu, or they are used to prompt the user, as in 'Press (Escape) to go back a menu'.

Note that the parentheses are always left blank when using the Comment action.

^ Action: Return to Last Menu

The general form is:

^()

This Action makes Uniplex return to the previous menu. The use of this action definition permits the user to return back up the chain of menus following exactly the route taken on the way down.

Actions That Involve Files

Seven menu actions provide access and information about files. Using these actions you can manipulate files, change directories, and obtain listings of file directories.

X Action: Change Directory

The general form is:

X() or
X('directory name')

This action can be included at any menu to change directories without leaving Uniplex.

In the example taken from #SYSMENU that was introduced earlier in the chapter:

'D - Change Directory' = X()

The user is prompted for the pathname of the requested directory because nothing is put in parenthesis.

In order to prespecify the directory, use:

'D - Change to /usr/uniplex/demo directory' = X('/usr/uniplex/demo')

? Action: List Directory

This action uses the UNIX utility **ls** to examine the contents of a directory. The data is fed back from **ls** automatically into Uniplex, where it is presented neatly in pages, with pauses in between.

As with other actions, information can be included in parentheses to mold the action to specific requirements.

A few examples:

=?('*')	List all files in this directory and all sub directories
=?('.')	List all files in this directory

The qualifier that is included between the parentheses can specify which directory(s) are to be listed. It can also be left blank, in which case the user is prompted for a directory specification.

In #SYSMENU, the action has been used in this way:

'L - List Files' = ?('*')

A few other possibilities for implementing this action:

'1 - List documents on this directory'=?('.')
'2 - List available games'=?('/usr/games')

C Action: Create a File

The general form is:

C() or
C('filename')

This action causes Uniplex to prepare for file creation. If the parentheses are left blank, the user is prompted for the file name.

An example of a visible option would be:

'1 - Create a new file'=C()

An example of a hidden option would be:

3=C()

The numeral **3** is an ASCII code equivalent for (Ctl) C, which is what the user types to select the option.

When the parentheses are left blank, as above, Uniplex verifies that a file of the same name as the user specifies does not already exist. If so, Uniplex displays this message:

Not a good filename!

and the user is prompted for another file name.

Note: Messages shown in this guide are taken from the standard sections, #MESSAGES and #USER. Of course, these messages can be changed as desired.

E Action: Edit a File

The general form is:

E() or
E('filename')

This Action tells Uniplex to edit a file. Syntax and operation are identical to the C Action described above.

An example of a visible option would be:

'2 - Edit an Existing File'=E()

An example of a hidden option would be:

5=E()

The numeral **5** is an ASCII code equivalent for (Ctl) E, which is what the user types to select the option.

When the parentheses are left empty, as above, Uniplex verifies that the file the user specifies exists. If not, Uniplex displays this message:

Not a good file name!

When the file name is specified, Uniplex bypasses the prompting routine and enters the file immediately (assuming it exists). This is a useful way to use this Action, as users often find themselves editing certain files often. A good example is:

'X - Edit the Command File' = E('uniplex/uniplex.rc')

This option provides rapid access to the Command File named in parentheses.

D Action: Duplicate a File

The general form is:

D()

This action provides an easy means of making copies of files. During the specification of the file name, file verification and error handling are automatic as with previous options described above.

This example is from the #FILEMANAGE section:

'1 - Copy a File' = D()

The user is prompted for the names of both the original and the new file.

Note that this Action only copies files for which the user has **read** permission. Also, the copy is made only if the user has **write** permission in the specified directory.

R Action: Rename a File

The general form is:

R()

This option allows the user to rename a file without leaving Uniplex.

The user is asked for the original name of the file. Then, after this has been accepted by Uniplex, the new name is requested.

Permissions and both the name for the original file and the new name entered are checked for existence by Uniplex to provide file protection.

This example is from the #FILEMANAGE section:

'3 - Rename a File' = R()

K Action: Kill a File

The general form is:

K()

This Action allows the user to remove a file without leaving Uniplex.

A file name is requested and its existence verified. As with other options, the user is notified of an unusable filename. After verification the user is asked to confirm the erase action:

Enter "*" key to confirm or <Return> to abandon:

Note that this Action only erases files if the user has **write** permission for the directory.

* Action: External Function

The general form is:

=*('UNIX Command Line','#USER Message Number','Pause/Return to Menu')

This Action provides access to UNIX, and as such, is one of the key ingredients to customizing Uniplex.

By including information in parentheses, any UNIX call can be executed.

Three qualifiers must appear in the above order only, separated by commas and included between single quotation marks. Spaces between quotes, commas, and parentheses are not allowed.

A summary of the qualifiers is detailed below:

UNIX Command Line
Type in the UNIX line as you would enter it at a UNIX prompt line, but put ^ where user input is required and ^^ if the input should be an existing filename.

#USER Message Number
The second qualifier is used when some user input is required to carry out the specified system call. It specifies the number of a message in the #USER section. This message is then displayed. (The standard #USER is in the Command File in Appendix A).

The messages in #USER can be changed, but are position dependent. The messages are numbered from **zero.** Therefore, message number 3 is the fourth message in the #USER section. Any #USER message number which does not correspond to a valid message number is converted to zero and therefore it is wise to use the zero entry in #USER as an error message.

Pause/Return to Menu
The third qualifier tells Uniplex to either pause before returning the user to the menu (P), or return directly without pause (R). Pause preserves on screen any information provided by the call.

Consider this example:

'1 - Where am I'=*('pwd','','P')

In this example, the second qualifier is not being used, so it is specified as a null string. The third qualifier tells Uniplex to pause. This means the following message is to be displayed before returning the user to the menu:

Press any key to continue

This preserves on screen any information provided by the call. In this case, the working directory is preserved on the screen until the user presses any key.

NOTE: The system defaults to the P qualifier when none is specified, so the above example could be written as,

'1 - Where am I'=*('pwd','','')

Sometimes no useful information is provided by a system call, so there is no need to tell the system to pause before returning the user to the menu. For example,

'2 - Remove back-up file'=*('rm backup','','R')

This tells Uniplex to do the call, in this case remove a file, and return (R) the user to the menu directly.

The second qualifier is used when some user input is required to carry out the specified system call. It specifies the number of a message in the #USER section. This message is then displayed.

For example,

'3 - Rename back-up file'=*('mv backup ^','10','R')

The caret (^) following the call tells Uniplex to prompt the user for input. The '10' tells Uniplex to print the message number 10 in the #USER section, which describes to the user the type of input that is necessary.

Whenever the caret symbol is used, it is good practice to enclose it between **double** quotation marks to avoid possible misinterpretation by the XENIX or UNIX shell. Thus, the last example would be more properly written as,

'3 - Rename back-up file'=*('mv backup "^"','10','R')

A **double** caret (^^) can be used when Uniplex's **Point and Pick** facility is defined. (Point and Pick allows the user to select a file name visually from a table of names, and is fully described in the **Uniplex User's Guide**).

For example,

```
'5 - View the End of a File'=*('tail "^"','5',)
```

would cause Uniplex to prompt for a file name before doing the system call, **tail**. The caret tells Uniplex to expect user input and to prompt for it by displaying the message number 5 in #USER (for example, ' Enter which document to look at').

The use of a double caret instead of a single one causes Uniplex to allow file name selection using Point and Pick in addition to the standard file selection prompt:

```
'5 - View the end of a file'=*('tail ^^','','')
```

*() Action: Exit from Uniplex

The * Action, when there is nothing in the parenthesis, is used to provide the option of leaving Uniplex, as in this example:

```
'* - Leave Uniplex' = *()
```

If the user's shell has been arranged to run Uniplex immediately on login, this Action could be used to log the user out upon leaving Uniplex, as in this example:

```
'* - Log out of the computer'=*()
```

Document Escapes

The #EDITMENU section defines the menu seen when the user gives the **Escape to Menu** command. The standard section looks like this:

```
#EDITMENU  '***  Document  Menu  ***'
'1 - Comprehensive Help Menu'=M('#HELP') '
'2 - UNIX Command Line '=*('^','5','P')
'3 - Find the Local Command File'=*('ls $HOME/uniplex/uniplex.rc')
'Press (Escape) to get back to the document'=>()
' '=>()
27=*()
))
```

NOTE: Although this menu section can be edited to provide almost any Uniplex and UNIX facility, it should be used with caution. Certain activities should NOT be attempted from the #EDITMENU. For instance, a second file should NEVER be opened while escaped from the original file.

As such, the **C** and **E** Actions are not accepted as entries in the #EDITMENU section.

SUMMARY

* The #SYSMENU defines the Main Menu that is the gateway to the entire menu system.

* The first line of a menu section is the header that identifies the section; the second line is the menu title as it appears to the user.

* Menu section entries define an option (or a comment) of the menu. Each menu is made up of two parts, the **option**, and the **action** that Uniplex is to take when the option is selected.

* The first character of the option is the trigger that causes Uniplex to carry out the action.

* Uniplex automatically converts lower case characters to upper case (when used as menu selectors), so that users need not concern themselves with the shift key.

* There are twelve distinct actions that Uniplex can carry out.

* The M Action causes Uniplex to jump to the menu specified.

* The > Action causes Uniplex to display the specified remark (which could be a blank line).

* The X Action causes Uniplex to change directories.

* The ? Action causes Uniplex to list the current or specified directory.

* The C Action causes Uniplex to create a file.

* The E Action causes Uniplex to edit a file.

* The D, R, and K options cause Uniplex to duplicate, rename, and remove a file, respectively.

* The ^ Action causes Uniplex to return to the last menu.

* The * Action causes Uniplex to carry out an operating system call.

* The * Action causes Uniplex to terminate when the parentheses are left blank.

Summary (cont)

* The form for the * Action is: =*('UNIX Command Line','#USER message number','Pause/Return to menu')

* The second qualifier tells Uniplex to print the indicated message from the #USER section. When not used, this qualifier must be included as two adjacent quotes.

* When a message is specified as above, a caret (^) must be included in the first qualifier, to let Uniplex know that user input is expected.

* A double caret is used to tell Uniplex to allow the user to select a file using Point and Pick, rather than the routine prompt.

* When either a single or double caret is used, it should be surrounded by double quotes to prevent misinterpretation by the shell.

* The third qualifier tells Uniplex to pause before returning the user to the menu (P), or to return directly (R).

* Hidden options can be included in any menu to provide quick access for more advanced users.

* The #EDITMENU section defines the menu seen when the user gives the **Escape to menu** command. Other files must not be edited while escaped from the original file.

The following chapters describe reconfiguration of Uniplex Word Processor system messages, editing keystrokes and other aspects of the editing environment.

System Details

The #SYSTEM section of the Command File includes general information necessary to the functioning of the Uniplex Word Processor. The standard #SYSTEM section looks like this:

```
#SYSTEM
BACKUP='^.backup'
2SPACE='.!?:;',                    3SPACE=''
RULER='.#CIJLTRH'
WDEL=';:,.!? ',                  HLEN=5
DOTS='HE:HM:FO:FM:PL:PA:PN:SN:PM:SP:JY:JN:RE:ME:ST:SB'
KEYIN='N'
MODE='T',                           PAGE='66'
PNUM='#'
SPELL='/bin/sh /usr/bin/uspell ^'
DECTAB='.'
STOP=' *!&$'
STATUS='S'
))
```

Each entry defines variables that correspond to a particular area of the software. Entries can be included in free format, in any order you choose. If you include more than one entry per line, be sure to put a comma between them, as in this line:

```
2SPACE='.!?:;',                    3SPACE=''
```

Entries not being used do not have to be included at all. For example, the 3SPACE entry is defined as a null string above only to note its existence. Alternately, it could have been eliminated all together.

Each entry is individually described.

BACKUP

This entry defines the name of the file in which Uniplex stores a backup of the file being edited. Backup of a file is automatic and is taken prior to editing. In the standard release, this file is named **^.backup:**

```
BACKUP='^.backup'
```

Uniplex replaces the circumflex (^) with the name of the file currently being edited, and concatenates it with the character string. Thus, when the BACKUP entry is included as above, and the file being edited is called **advent**, the backup copy is saved as **advent.backup.**

The backup filename can either be a single file or a pathname. If a complete filename is not specified, the backup file is stored in the current directory. Examples of both a single file in the current directory and a full pathname are:

```
BACKUP = '^.backup'
BACKUP = 'usr/uup/^.backup'
```

The backup entry can also contain the tilde (~), which is replaced with the user's environment variable, $HOME. For example:

```
BACKUP='~/wp.backup.up'
```

2SPACE and 3SPACE

These entries define characters that require special spacing during paragraph formatting by Uniplex. The characters listed in 2SPACE will have two spaces added after them when Uniplex is formatting paragraphs. Likewise, the characters listed in 3SPACE will have three characters added after them during format. This adjustment is applied only if the characters are followed by at least one space in the document; Uniplex automatically adjusts the number of spaces to two (2SPACE) or three (3SPACE).

These entries are used in the standard release:

2SPACE='.!?:;', 3SPACE= ''

DECTAB

Defines the decimal point character that Uniplex is to recognize when doing decimal tabbing. In the standard release, this is defined as a period:

DECTAB='.'

RULER

Ruler contains a list of valid ruler characters that must appear in the correct order as noted in this table:

Position:	Function:	Standard Setting:
1	Blank position	.
2	Decimal tab	#
3	Center offset	C
4	Indent	I
5	Justify right edge	J
6	Left edge	L
7	Tab stop	T
8	Right edge, no justify	R
9	Paragraph hang	H

The characters can be changed as long as they maintain the order as above. The standard entry is included as:

RULER='.#CIJLTRH'

WDEL

WDEL contains the characters designated to be word delimiters during the preparation and formatting of text within Uniplex.

This information is used during the execution of certain functions and processes in the editor such as: word-wrap, reformat, tab and so on.

Normally, WDEL is set to include the space character and all punctuation characters in normal use. However, if Uniplex is used with non-English character sets, WDEL will need to be expanded.

The standard entry is:

```
WDEL=';:,.!? '
```

HLEN

This entry is used to control the Uniplex hyphenation system. The number entered here indicates to Uniplex the number of unfilled spaces permitted at the end of a line, before attempts to hyphenate will occur.

Trial and error indicates that a suitable value for English is 5 spaces. Non-English languages will probably require different settings of this variable.

Thus, the standard entry is:

```
HLEN=5
```

DOTS

This entry defines the two letter codes that name Print Time Commands. The standard entry is:

DOTS='HE:HM:FO:FM:PL:PA:PN:SN:PM:SP:JY:JN:RE:ME:ST:SB'

These commands define certain parameters at print time:

Print-Time Command	Tells Uniplex To ...
.HEn	The next n lines are printed as a header at the top of each page.
.HMn	Start printing text n lines after the end of the header
.FOn	The next n lines are printed as a footer on each page
.FMn	Stop printing text n lines before the beginning of the footer
.PLn	Set the page length to n lines
.PA	Start a new page here
.PNn	Set the page number to n
.SNcode	Sends code to printer.
.PMn	Start a new page here if fewer than n lines remain
.SPn	Insert (n-1) spaces between each printed line
.JY	tells Uniplex to reformat all text, according to current rulers, until it reaches a ".JN" command or the end of the document.
.JN	tells Uniplex not to reformat subsequent text.
.RE	denotes comment line.
.ME	merges indicated file at print time.
.STcode	sends indicated code to printer for top of page controls.
.SBcode	sends indicated code to printer for bottom of page controls.

Any code can be renamed with a unique two letter combination. The only condition is that the codes are position dependent; to replace **HE** with **HD**, for instance, type **HD** in the first position.

The code indicated by the **.SN, .ST and .SB** should be expressed as an ASCII decimal equivalent. Dash (-) should be used to separate one value from another. Short forms like $ for Escape should not be used. Refer to Appendix B for the ASCII decimal equivalent codes.

KEYIN

This entry is used to switch off the external keyboard reader in Uniplex, should this be required. The entry in such situations would be:

KEYIN='N'

Note: In the XENIX operating system, the KEYIN entry should be set to N as shown above.

The standard entry is shown above.

KEYIN='Y'

can be used with other systems.

Mode

This entry specifies modes that are automatically activated when the user enters the document editor. The Mode is displayed on the first status line of the editor.

The characters specified are capital letters denoting **Insert mode** (I), **Hyphenation mode** (H), **Stop space underline mode** (U), or **Auto tab mode** (T).
For example,

MODE='T'

causes the user to automatically be in **Tab mode** whenever the document editor is entered. The user can, of course, turn off any preset mode after entering the file.

PAGE

This entry specifies the default page length. The page length is displayed on the first status line of the editor. If not specified in the Command File, 66 lines is used as the default page length. The example below can be used to redefine the default page length.

PAGE='66'

PNUM

This entry defines the character used to represent the automatic page numbering in headers and footers. At print time, this character is replaced with the current page number.

For example, to use the crosshatch character for the page number:

PNUM='#'

SPELL

This variable specifies the location of the spell program. In the standard release this is,

SPELL = '/bin/sh /usr/bin/uspell ^'

The caret tells Uniplex to run spell on the current file being edited. Uspell is a modified version of the standard UNIX spelling checker. If your implementation of UNIX does not have a spelling checker then uspell will not function and you will need to provide a routine that outputs the badly spelled words one per line in the order in which they are found, to standard output.

If the -b option is specified then the British spell checking dictionary is used. The entry should be included as follows:

SPELL='/bin/sh /usr/bin/uspell -b ^'

STATUS

The **STATUS** entry enables the status line during file edit. The standard entry is:

STATUS='S'

This enables the complete status line. The column counter on the status line (which indicates the cursor location) is disabled when the entry is included as:

STATUS='X'

This may be preferable in certain instances, as the system requires a little time to update the counter each time the cursor is moved.

STOP

This entry defines characters that are not allowed in file names specified by the user. For example, when the entry is included as below, the asterisk (*) is not allowed in file names given by the user:

STOP='*'

When the character named in this entry is subsequently used in file name specified by the user, the **Not a good file name!** message appears and the user is reprompted for a valid file name.

SUMMARY

* The #SYSTEM section contains various details relating to the operation of Uniplex.

* Each entry defines variables that correspond to a particular area of the software.

* Entries can be arranged in free format, and those not being used can be omitted entirely.

* The BACKUP entry specifies the name of the backup file in which Uniplex saves a copy of the file being edited.

* The 2SPACE and 3SPACE entries define characters that are to be followed by two or three spaces when Uniplex formats text.

* The RULER entry defines valid ruler characters. If changed, the entries must preserve the order as included in the standard release.

* The WDEL entry defines characters used as word delimiters.

* The DOTS entry defines two letter codes used for print time commands.

* The PAGE entry defines the default page length in number of lines.

* The PNUM entry establishes the character used to specify the current page number in headers and footers.

* The DECTAB entry defines the character used for decimal tabbing in text.

* The STOP entry specifies characters that are not allowed in file names.

* The MODE entry specifies modes that are automatically activated when the user begins editing a document.

* The HLEN entry tells Uniplex when to attempt to hyphenate words when hyphenation is in effect.

Commands

Uniplex editing command keystrokes are completely assignable, making it compatible with a wide variety of terminals, including those with function keys. Using the function keys found on many terminals, a single keystroke can invoke any command.

NOTE: Always make a copy of the Command File before making any editing changes. Then edit the copy, and save the original as a back up.

Characteristics of specific terminal types can be included in any Command File. A single Command File can serve a group of users by specifying keystroke definitions for several terminals. Or, sites with many diverse terminals can be easily accommodated with personal command files.

The keystroke sequences that invoke editing commands are assigned in the #COMMANDS section. Keystrokes necessary to invoke commands can be changed by simply altering the entries in this section.

Commands are represented by codes (F01-F75); the correspondence of codes to editing commands is contained in the #COMLOCATE section:

```
#COMLOCATE
F01=Delete character     F02=Destructive Backspace   F03=Delete word     F04=Delete right
F05=Delete left          F06=Delete line             F07=Delete blanks   F08=Insert space
F09=Insert line          F10=Insert blank lines      F11=Return          F12=Left
F13=Right                F14=Up                      F15=Down            F16=Tab
F17=Previous word        F18=Next word               F19=Bring line up   F20=Bring line down
F21=Go left              F22=Go right                F23=Go up           F24=Go down
F25=Top of screen        F26=Bottom of screen        F27=Scroll down     F28=Scroll up
F29=Top of file          F30=Bottom of file          F31=Convert to lower case   F32=Convert to upper case
F33=Center line          F34=Emphasize text          F35=Show effect     F36=Line split
F37=Do again             F38=Format paragraph        F39=Help            F40=Redraw screem
F41=Find next occurrence F42=Find pattern            F43=Search and Replace      F44=Format Document
F45=Terminate command    F46=Go to page              F47=Recall ruler    F48=Store ruler
F49=Use ruler            F50=Enter mode              F51=Leave mode      F52=CP mark (block)
F53=CP blank             F54=CP leave                F55=CP remove       F56=Not In Use
F57=CP Overlay           F58=CP Insert               F59=CP elbow        F60=Escape to menu
F61=Exit & save          F62=Quit no save            F63=Write no exit   F64=Not In Use
F65=Merge insert         F66=Restore text            F67=Enter insert mode       F68=Leave insert mode
F69=Start print effect   F70=Stop print effect       F71=CP mark (ser)   F72=Save CP text
F73=Merge overlay        F74=Save to file            F75=Spell
```

Note: #COMLOCATE is only used for administrator reference, and as such, is not really a command file section. Note the absence of the double right parentheses; this is done so that when #COMLOCATE is included just before #COMMANDS, they can be copied to other locations as a set. Technically, #COMLOCATE is a series of remarks between sections.

The #COMMANDS Section

In the #COMMANDS section each editing command, represented by a function number, is assigned the keystoke sequence that command will use when in the editor.

The keystroke sequence is defined by an ASCII decimal equivalent, or is in quotes if a literal string or one of the short forms described below. The dash (-) is used to join these definitions if required.

The ASCII codes are found in the table in Appendix B. The codes are useful for defining keystrokes that cannot be included literally.

Short forms can be used in combination with literal strings to make up complete entries.

Short Forms	Keys
$	Escape
T	Tab
U	Cursor Up
D	Cursor Down
L	Cursor Left
R	Cursor Right
_	Delete

NOTE: You can instruct Uniplex to read single quotes literally by including a backslash (\) before the quote.

Most default Uniplex keystroke sequences in the standard release are **case independent**, meaning that the user can type **either** an upper or lower case letter to invoke the same command. This is done by using F in defining keystroke sequences, as above. When it is desirable to have a lower case letter invoke one command, and the same letter in upper case invoke another. For example, **<ESC> a** might invoke a particular command, and **<ESC> A** another one. In these situations, the code is specified with a lower case f. The general syntax is:

f nn = <keystroke sequence> where keystrokes typed are read literally as upper or lower case

F nn = <keystroke sequence> where keystrokes are case independent; either case can be typed by the user

To see how keystroke sequences are assigned to the command codes, here is the first line of the standard #COMMANDS section:

F01=3, F02=127, F03=23, F04=$-'DR'

By consulting #COMLOCATE, we find that **F01** corresponds to the **Delete character** command. The numeral **3** is the decimal numeric form of the ASCII character value for control-C (**^C**).

Note that the above line preserves Command File syntax by including a comma after each entry. The next entry, **F02=127**, translates to the **Erase previous character** command, the Rub or Delete key.

F04=$-'DR' is the definition for the **Delete right** command. This entry uses the $ short form to specify Escape key.

The next part of this entry can be quoted literally, and so is included in quotes as command file syntax demands. The entire translation of this entry, then, is:

Delete right = <ESC> DR (or **dr**, as the F convention was used)

Configuring Function keys

ASCII codes for special keypads and function keys can also be used. Furthermore, the same command can be assigned to different keystrokes in the same #COMMANDS section. For example, codes F12-F15 specify cursor movements using codes from the above table (L,R,U,D). Additionally, F12-F15 are also defined on the last entry line of #COMMANDS as 8,12,11,10. These codes are defined in the ASCII table in Appendix B as ^H,^L,^K,and ^J. This preserves cursor motion commands as used in other UNIX software.

To assign a command to a function key, consult the manual for the terminal being used to obtain the ASCII code sequence generated by the function key or keypad key; this is then used in the definition of the Uniplex command. Find out what string is produced when each function key is pressed. For example, function key #1 on the Altos II produces this string when pressed:

```
1-'@'-13
```

To assign the Exit and Save command to this key, for example, include this entry:

```
F61=1-'@'-13
```

Function keys can also be combined with other keys to produce commands. For example, if a function key generates:

```
1-'E'-13
```

then the **Mark Bottom and remove** command can be defined as:

```
F53=1-'E'-13-'r'
```

and the **Mark Bottom and leave** command as:

```
F71=1-'E'-13-'l'
```

This assigns the function key to mean "mark bottom", and "l" and "r" must be typed in to indicate leave or remove.

Entries in the #COMMANDS section can be included in **free format**, that is, without regard to any particular order; however, the standard #COMMANDS section presents definitions in order according to function number, and is structured in tabular format.

#COMMANDS sections for specific terminals can be built using the syntax described above. The header must include the TERMCAP name of the terminal. For example, the header of #COMMANDS section for the Altos II terminal reads:

#COMMANDS-alt2

"alt2" is the TERMCAP code for the Altos II, and the dash tells Uniplex that this is a special terminal definition. Uniplex obtains the terminal type being used and matches it with this header to find the proper section. It uses the standard section (#COMMANDS) by default when no match is found.

Following simple rules of syntax, commands can be assigned to almost any set of keystrokes by simply editing the #COMMANDS section, and by developing other #COMMANDS sections for other terminal types as needed.

Note that the command for 'stop command in progress' (F45) is an exception, in that only a single character can be used in its code sequence. This is due to technical reasons associated with the UNIX operating system.

SUMMARY

* By assigning keystrokes to commands in the #COMMANDS section, single keystroke commands are possible.

* Separate Command Files can be developed to suit different needs. A central Command File can serve a group of users, while local ones can serve individuals.

* Command names are represented by codes defined in #COMLOCATE.

* Keystrokes are defined by codes defined in Appendix B, or by literal strings.

* Literal entries are surrounded by single quotation marks.

* Entries in the #COMMANDS section can be entered in free format, separated by commas if on the same line.

* COMMANDS sections can be developed for specific terminals to allow full use of terminal capability.

Messages

All system messages (with the exception of those used in menus, see Chapter 5), are loaded from the #MESSAGES section:

```
#MESSAGES
'[?]  Select an option'
'File already in use'
'Selected menu not available or badly formatted.'
'Press <RETURN> to continue:'
'Unable to run system command/program.'
'Not a good file name.'
' selected.'
'No write permission. '
'Please enter the file name or "*" to return to menu '
'Copying file to '
'Unable to open file for edit.'
'Cannot create new backup file.'
'Cannot write to file '
'*** Document copy ***'
'*** Document re-name ***'
'*** Document erase ***'
'Requested task is now complete.'
'Operation FAILED.'
'Original file'
'New file'
'Enter "*" key to confirm or <RETURN> to abandon:'
'Merge file section not found.'
'Enter name of file :'
'Enter string to search for:'
'Option ? '
'Sorry, search failed.'
'Enter directory name or "."=current, "*"=complete, "-l"=statistics'
'Unable to list selected directory'
'Enter "*" to exit or <RETURN> for more '
'*** List documents ***'
'*** External function ***'
'Busy saving the document......'
'*** Edit a file ***'
'*** Create a file ***'
'"-" to hyphenate or <RETURN> to skip:  '
'Enter "*" to replace or <RETURN> to skip '
'Enter new characters or <RETURN> to delete '
'Enter "*" for global or <RETURN> for interactive '
'Invalid command entered'
'Enter "*" to confirm quit or <RETURN> to continue '
'Busy executing command'
'Enter <RETURN> for next or "*" to quit '
'Enter ruler number (0-9)'
```

```
'Enter (B)lank, (L)eave, (R)emove or (S)ave to file'
'Enter (O)verlay, (E)lbow or (I)nsert'
'Enter 1 (tab), 2 (hyphen), 3 (underline text), or 4 (insert) :'
'Enter character to underline with'
'Enter <RETURN> for next page,"*" <RETURN> to quit'
'INSERT'
'HYPHEN'
'TAB'
'Current directory : '
'Getting your directory listing'
'Directory, enter "*" to change directory'
'No write permission, enter "*" for read only '
'Sorry -- no read permission on this file'
'Move cursor to table of names or enter a file/directory name'
'*** Change directory ***'
'Enter a directory name then press <RETURN>'
'Sorry -- cannot change to requested directory'
'Enter page number, <RETURN> :'
'Enter "*" for more '
'File exists - enter "*" to overwrite or <RETURN> to abandon '
'Enter "*" to find word or <RETURN> to skip '
))
```

You can change these messages as desired, keeping in mind that they are position dependent, should be delimited by single quotes and should not exceed the screen width.

Rulers

The #RULERS section specifies ten standard rulers, numbered Ø through 9, that are available to the user for control of formatting. The standard rulers are accessible by using the **Recall ruler** and **Use ruler** commands.

The default rulers in the standard Command File are:

```
#RULERS
L.......T.......T.......T.......T.......T.......T.......T.......T....R.
L.......T.......T.......T.......T.......T.......T.......T.......T....J.
........L.......T.......T.......T.......T.......T.......T.......T....R.
........L.......T.......T.......T.......T.......T.......T.......T....J.
..............L.......T.......T.......T.......T.......T.......T....R.
..............L.......T.......T.......T.......T.......T.......T....J.
......................L.......T.......T.......T.......T.......T....R.
......................L.......T.......T.......T.......T.......T....J.
........L.......T.......T.......T.......T.......T.......T.......J......
T.......T.......T.......T.......T.......T.......T.......T.......T......
))
```

NOTE: The rulers are too long to be fully displayed here.

The #RULERS section can be changed so that rulers for special formatting requirements can be accessible to the user. The rulers as shown above are position dependent; the first ruler is Ruler Ø, the second position is reserved for Ruler 1, and so on.

Note that if you change any ruler characters when modifying #RULERS, you must also specify them in the RULER entry in the #SYSTEM section.

Note also that Ruler Ø is the default ruler when a file is entered, and is displayed on the ruler status line.

Printer Effects

The Uniplex module **Tcap** contains codes that control how printer effects are displayed on the particular terminals being used. This file also includes codes that enable different printers to correctly carry out printer effects. Each Tcap entry specifies codes for a particular terminal or printer. Tcap is fully described in Chapter 9.

The #EFFECTS section defines the printer effect prompt that the user sees when the **Start effect** is given. The standard #EFFECTS menu looks like this:

```
#EFFECTS
A=Bold
B=Double Strike
C=Underscore
D=Underscore text
E=Superscript
F=Subscript
G=Elite
H=Big
I=Small
```

Each line of EFFECTS can contain one entry only; note that quotes are not needed as delimiters in this section.

Up to 26 effects can be included, using the capital letters **A** through **Z**, although letters **A** through **D** must be reserved for the effects as noted above. Entries can be arranged in free format, will appear on the user's screen in the order in which they are included here.

Since only one print effect can be on at one time, if the user wants two or more print effects at once (for example bold and underlining), a separate effect can be set up to do this (for example, J=Bold and Underscore).

SUMMARY

* The Uniplex module **Tcap** contains codes that control how printer effects are displayed both at the terminal, and on the printer. Each Tcap entry specifies codes for a particular terminal. Tcap is fully described in Chapter 9.

* The **#EFFECTS** section defines the printer effect prompt that the user sees when the **Start effect** command is given.

* Each line of EFFECTS can contain one entry only.

* Quotes are not needed as delimiters in this section.

* Up to 26 effects can be included, using the capital letters **A** through **Z**.

* Entries can be arranged in free format, appearing on the user's screen in the order in which they are included in **#EFFECTS**.

The Tcap file

The **Tcap** file, resident in /usr/uniplex, includes printer effect codes used by each different kind of printer and terminal in the system.

Uniplex expects to find the file **/usr/uniplex/Tcap.** If it doesn't, or if there are errors in this file, the message **cannot read Tcap file** appears, and Uniplex terminates.

A separate section is reserved for each different printer and terminal. Each Tcap section defines the codes necessary to produce the effect on the terminal or printer. The actual effects being used in the system are named in the #EFFECTS section of the command file (see Chapter 8).

If a code is represented in the #EFFECTS section, but not defined in Tcap for the terminal in use, the standard terminal standout mode (as defined by the termcap entries **so** and **se**) is used to represent the effect on the video terminal.

If no sequences are given for the effects A-D for the printers then Uprint will produce the effects in a standard way internally in the software. If no other controlling entries are present and text needs to be bolded, the text will instead be triple struck. Any other sequence needed by the printer to produce an effect that is not present in Tcap (ie. effects E-Z) is ignored, leaving text on the line remaining the same.

Defining Printer Effect Codes

Each section of the Tcap file is indicated by the name of the printer or terminal for which effect codes are being described. The name is preceded by the hash sign (#), as is the convention for sections of the command file. Thus an effect definition section header of Tcap might appear as:

#diablo

or

#vt100

Each defined effect takes the form:

<code>=<Sequence to start effect>,<sequence to end effect>

where **code** is a capital letter A-Z.

A sample definition might look like:

```
A=27-93-65,27-93-91
```

In this example, the sequence 27-93-65 starts print effect A and 27-93-91 stops it. Syntax rules for code definitions in Tcap correspond to those used in the Command File.

Using the ASCII table in Appendix B, we find that **27-93-65** corresonds to ^[^] **A**, and **27-93-91** translates to ^[^] [.

NOTE: Standout mode is used for a terminal when a particular printer effect is not defined here, or if the terminal is not defined in Tcap at all.

Using Optional Switches

Two sets of switches, one for printer definition, one for terminal definition can also be included within a Tcap section. These switches provide extra control over peripheral devices.

Six terminal options are available; when used, each should appear on a separate line.

ns — Don't use terminal standout mode. This is a must if the standout mode places a phantom blank space (called an **attribute byte**) before the string to be highlighted.

nt — Don't use terminal insert line/delete line capabilities. This can be useful where scrolling is faster if performed by Uniplex than by insert/delete line at the terminal.

nu — Cursor motion sequences may send null characters to screen. This option invokes special routines in Uniplex version-3 that bypass this problem.

rw — Place terminal in RAW mode rather than CBREAK mode. This is allied with the **nu** option, though some terminals which need **nu** do not need to be in raw mode. If this option is present, the **Terminate command** feature of the editor is lost.

co — Use full screen width for printing. The default in Uniplex version-3 is terminal width minus one since many terminals will scroll if a character is printed in the rightmost column, particularly on the last line. This option should only be used if the terminal does not scroll when a character is in the rightmost column.

TERMINAL
MAP Three sequences can be specified for each map location and there may be up to 24 different map's for each terminal entry in Tcap.

Field 1 = Character sequence typed in from keyboard to be mapped.
Field 2 = Single character that will store in the text file.
Field 3 = Character string to display on screen. (MUST end up as one character).
To print the displayed character, the character in field 2 is mapped in the PRINTER MAP sequence.
Field 2 can be any character with an ASCII decimal value of less than 128. The syntax is:
MAP=<character sequence typed>,<character stored>,
<character sequence displayed>
See also PRINTER MAP below.

Five printer options can be used; again, when used, each should appear on a separate line.

PITCH This location is used if the printer itself does not have the capability to bold text by means of a code sequence. This is equivalent to the PITCH1 and PITCH2 locations in implementations of Uniplex version-2. The syntax is: PITCH=<sequence to set character pitch small, reset sequence>

FF This option is used to cause the blank lines at page ends to generate a form-feed. This is sometimes required when using cut-sheet feeders. The syntax is: FF=<sequence to effect a form-feed> eg. FF=12

INIT The code entered here is sent to the printer just prior to printing the document and can be used for a number of purposes such as printer reset. The syntax is: INIT=<sequence to initialize the printer>

DEINIT The code here is sent to the printer after the document printing has been completed. The syntax is: DEINIT=<sequence to reset the printer>

PRINTER
MAP Two sequences can be specified for each map location and there may be up to 24 different map's for each printer entry in Tcap.

Whenever the from-character is located in the document the to-sequence is substituted and sent to the printer. This can be used to drive a special character sequence across to the printer from the document such as might be required for graphics or other controls. This switch used in conjunction with the TERMINAL MAP is convenient for languages other than English, where a letter is typed in as two characters, but is displayed and printed as one, as is the case of accented letters. The syntax is: MAP=<mapped-from character>, <mapped-to sequence>

Example Tcap entry

For a terminal defined by the TERM variable: "cs"

```
#cs
B=$-'[5m',$-'[m'
C=$-'[4m',$-'[m'
D=$-'[7m'-$-'[4m',$-'[m'
E=$-'[4m'-$-'[5m',$-'[m'
F=$-'[7m'-$-'[5m',$-'[m'
nt
co
))
```

Specifying printer types

The last part of the Tcap file is labeled PRINTERS. This section is used by the printer configuration program to describe which printers each printer type applies to. If a printer type is added to the Tcap file, this section should be updated.

SUMMARY

This chapter described how the Tcap file works in conjunction with the command file to define printer effects for the particular terminals and printers being used.

* Tcap is resident in /usr/uniplex; if Uniplex cannot find it here (the user may also not have permission to read), or if there are errors in the file, Uniplex terminates with the message, **cannot read Tcap file.**

* A separate section is used for each different printer and terminal type.

* Each Tcap section describes the codes necessary to produce each effect on the particular terminal or printer.

* The actual effects being used in the system are named in the #EFFECTS section of the command file.

* Standard terminal standout mode (termcap entries **so** and **se**) is used to represent effects on a terminal not defined in Tcap, or for any effects not defined for a terminal.

* For the printer, if no sequences are given for effects A-D, Uprint produces the effects in a standard way internally in the software.

* Any other sequence needed by the printer to produce an effect not present in Tcap is ignored, leaving text untouched.

* Syntax rules for code definitions in Tcap correspond to those used in the command file.

* Two sets of optional switches (one for terminals, one for printers) can be included in Tcap sections.

* The last part of the Tcap file (PRINTERS) describes which printer each type is used for.

UNIPLEX WORD PROCESSING HELP

Users have the option of on-line help that is accessible either from within a document or from the menu system. All help files can be rewritten and the help system can be expanded to be as comprehensive or as specific to individual users as required.

There are two elements to the Help System. First is the Text Editor help file **uniplex.help** which is accessed when either editing or creating a file. Secondly is the Menu Help System. These help files are accessed from any menu when outside the editor or from the EDITMENU from within the editor.

TEXT EDITOR HELP FILE

The standard help file **uniplex.help** is stored in the directory **/usr/uniplex.** Individual users can however have, their own local help files in the users local uniplex directory.

Uniplex looks first for the file **uniplex.help** in the local uniplex directory. If there is no such file then the **uniplex.help** file in /usr/uniplex is used.

To access this help file, the user types the command <Esc> h (standard command) from within the text editor. Refer to the Uniplex User Manual for further information.

If required, the user can translate this file for use in a foreign language or rewrite the contents of the file. The standard **uniplex.help** file lists all the text editing functions and the keystrokes required to execute these functions. There is no limitation on the length of the **uniplex.help** file. If longer than the screen size then, the prompt **'Enter "*" to exit or <Return> for more'** instructs the user there is more text to follow.

MENU SYSTEM HELP FILES

The following files stored in the directory /usr/uniplex/help have been included in the standard implementation of Uniplex Word Processing. These files and the related menus #HELP and #CONSUM are optional and do not have to be implemented if not required.

FILES:

alter	move
cursor	often
delete	print
emphasis	prim1
exit	prim2
find	prim3
insert	prim4
mark	rulers
merge	scroll
mode	

The location and name of the standard help files is specified within the menus #HELP and #COMSUM in the command file, **uniplex.rc.** The ability for users to have their own menus enables individuals to have their own personalized help files or for the user to develop their own Menus and Help files.

To access the Help Menu from outside the text editor type **?** at any of the menus (as configured on standard menus). From within the editor type **<Esc> !**, to access the EDITMENU. Refer to Uniplex User Manual for further information.

VIEWPRINT

Viewprint provides a screen driven, controlled and definable environment in which to perform mail merging and printing for the Uniplex system. It is used as a front end to the uprint and Rmerge modules. The uprint module is covered in the next chapter, and the Rmerge module is covered in the Mail Merge section of the User's Guide.

Viewprint is given one argument that indicates the task required, and an optional file name. A screen-based form is presented to the user with fully protected fields for the operator to enter information about the task.

For example: to print a file the user may be presented with a Viewprint screen which has default values for options like interactive print, start page, device, etc. and the user need only fill in the file name. However the user may overwrite any of the default values to suit the present print requirements.

Viewprint is composed of 2 sections. The first is the compilation section ("vpc"), where the system administrator can define the fields to appear on the screen in a text file ("vprint" or "vmerge"), and then compile this screen using the program "vpc."

The second section is the run-time section ("vp") which uses the compiled screens to collect the information needed to run the particular job. Inputs can be very tightly controlled and entries verified against predetermined parameters. Operator prompts can be printed for individual fields. Default values can be specified, and entered values can be mapped to other values. Once the user has correctly entered the required input and wants to process, Viewprint will assemble the entered data into a predetermined command line which will be executed under Unix.

Viewprint Purpose: Provides form-type front end for merging and printing.

```
Usage:          compiler stage:  vpc -p
                           or:  vpc -m

                run-time stage:  vp -p [filename]
                           or:  vp -m [filename]
```

Where -p selects the print screen, and -m selects the merge screen.

Operation

Build the instruction file ("vprint" or "vmerge") and compile it in the directory where the file resides, using "vpc -p" or "vpc -m".

If the compilation is not successful, a report file is generated indicating the problem. This report file will be either vprint.er or vmerge.er. After a successful compilation a control file for the run-time module is produced, either vprint.vp or vmerge.vp. This output file must be placed in either the /usr/uniplex directory or the $HOME/uniplex directory.

Layout of Instruction file

The instruction file acts as the source file for the Viewprint compiler "vpc". All the information that Viewprint will require at run-time is taken from the compiled instruction file. Information is grouped into three sections.

1. #DEFINE Runtime instruction sequences
2. #SCREEN SCREEN layout and field identification
3. #FIELDS FIELD definition data

Each section begins with one of the three labels above and ends with the terminating sequence "))" at the start of a line.

#DEFINE section

The DEFINE section of the Viewprint instruction file contains information required by Viewprint to interpret and run the screen successfully.

Entries are single entry per line, no delimiters.

Example section:

```
#DEFINE
-c@h @j @f -f@i -S@c -E@b -m@d -p @g @e @l -s@a @k
FIELDS = [_]
HELP = $-'h'
QUIT = $-'q'
GO = $-'e'
RUB = 127
RESET = '#'
GRESET = $-'#'
))
```

The first line contains a template for the arguments of the binary to be executed (either "uprint" or "Rmerge"). The characters after each @ character refer to specific fields, as defined in the #SCREEN section.

The next seven lines define various command sequences, one per line, and the delimiters for the fields on the screen. All of these sequences have default values so it is possible to provide just the command line for the screen and the compiler will fill in the desired values. These seven lines have a similar syntax, i.e. the variable, then an '=', then the string.

The "FIELDS" variable specifies the delimiter for the start of a screen field, the paint character, then the delimiter for the end of a screen field. The paint character is used to fill the actual space between the field delimiters.

The format for the remaining six sequences is the variable, followed by an '=', followed by a string consisting of either decimal character codes and/or literal characters in single quotes, each seperated by a hyphen (-). Note the escape key can be specified by the dollar ($) character.

The default values for the screen delimiters and paint character are:

```
Left field delimiter [
Paint character _
Right field delimiter ]
```

The section terminates by '))' marker.

#SCREEN Section

The screen section is used by Viewprint for screen layout. Fields are defined here for position, size and identifying flag character.

Form layout may take up to 20 screen lines. The top four lines of the screen are reserved for messages and instruction prompts.

Note that the run-time module will amend a screen that will not fit onto a terminal, using termcap to find the maximum number of columns and rows.

Example section:

```
#SCREEN
                    ------ UNIPLEX PRINT SET-UP ------

  Enter name of file to print              [eeeeeeeeeeeeeeeeeeeeee]

  Number of the first page to print        [cccccc]
  Number of the last page to print         [bbbbbb]
  Number of copies                         [hhhh]

  Name of Printer                          [gggggggggggggggg]

  Print page-by-page (y/n)                 [j]
  Paper length (lines)                     [iii]
  Print alternate pages (y/n)              [f]
  Left hand margin offset                  [ddd]

                        TO PRINT PRESS  <ESC> e
                        TO QUIT PRESS   <ESC> q
                        HELP PRESS      <ESC> h
))
```

Any visible characters may be used to identify fields with the exception of the three defined as the field delimiters and the paint character. Note that a field is considered to be a left field delimiter, followed by one or more identical characters, followed by a right field delimiter.

This means that the field delimiter characters can be used on the screen as part of the screen layout, rather than as field markers, provided they do not enclose a string of identical characters, or a single character. For instance, if an option required a yes/no reply then a line such as this could be used:

```
Do you want a copy [y/n] [a]
```

Notice that the first set of field delimiters will be ignored because they do not enclose characters that are all the same. However the second set will tell the compiler that this is the location of a single character field whose identification character (for the #FIELDS section) is "a". There is a limit of 132 characters on the size of a field , and the number of fields is limited to 80 per screen.

#FIELDS Section

The #FIELDS section of the compiler input file enables you to specify precisely what a valid response in a field is - not only what type of data, but for certain cases you can specify that only a certain choice or range of replies is valid.

This section controls all the data validation of the users replies.

Field definition types:

Summary-	Type:	Verification on input data:
	i	can be opened as file for input
	o	can be opened as file for output
	a( )	Alpha (choice, range or type)
	n( )	Numeric (choice, range or type)
	m( )	Map string-1 to string-2
	p()	Prompt string
	+	Discretionary input
	&( )	Default value
	@	Ghost field
	g	General field type

Definition qualifiers:

Individual entered values in field definition qualifiers are delimited by spaces unless the whole string is delimited by quotes.

Example:	Values seen:
a(one)	one string one
a(one two)	one string one two
a(one two three)	three strings one two three
a('one two three')	one string one two three

Definition Types:

Field Type 'i' Can be opened as file for input
- Example- a=i
- Checks- File exists
  Read permission held

Field Type 'o' Can be opened as file for output
- Example- a=o
- Checks- Valid Unix filename
  Write permission here
  File does not exist

Field Type 'a' Alpha (choice, range or type)
- Example (choice)- a=(a b c d)
- Checks- Input data must be one of 'abcd'
- Example (range)- a=a(a-z)
- Checks- Input data must be one of 'abcdefghijklmnopqrstuvwxyz'
- Example (type)- a=()
- Checks- Input data must not be a digit

```
     Field Type 'n' Numeric (choice, range or type)
         Example (choice)-        a=n(1 2 3 4)
         Checks-                  Input data must be
                                  one of '1234'
         Example (range)-         a=n(1-9)
         Checks-                  Input data must be one
                                  of '123456789'
         Example (range)-         a=n(>0 <10 >=b)
         Checks-                  Input data must be
                                  greater than zero,
                                  less than 10
                                  Greater than or equal
                                     to field b
         Example (type)-          a=n(1-9)
         Checks-                  Input data must be
                                  either a digit or
                                  one of ',.-'

        Field Type 'm' Map string-1 to string-2
            Example-         a=m(cmd 'uniplex/uniplex.rc')
            Checks-          Substitutes 'uniplex/uniplex.rc' for
all                                input 'cmd'
```

(Note that a string can be mapped to null by a=m(abc '') and that a mapping is only performed on a field's reply after any ghost field copying has taken place)

```
        Field Type 'p' Prompt string
            Example-         a=p(Enter a number greater then 0 )
            Checks-          None. Any string between the
                             brackets is accepted

        Field Type '+' Discretionary input
            Example-         a=+
            Checks-          Viewprint assumes that all fields
                             must have an entry unless this
                             type is specified.

        Field Type '&' Default value
            Example-         a=&(wp.back.up)
                             Loads entered value onto field when
                             screen is first displayed.
```

(Note - If a field requiring a filename as input is given the default value of a paint character (eg. a=&(_)), the optional filename argument of the "vp" command will be substituted.)

```
        Field Type '@' Ghost field
            Example-     a=@b
                         The value returned for the screen
                         field b is also placed in ghost
                         field a. This field will not ap-
                         pear on the screen, but will be
                         in the command line.
```

Field Type 'g' General field type
Example- a=g
The input for field a will not be checked.

Example FIELDS section

```
#FIELDS b=n( >Ø < 32767 >=c)
b=p(Enter a number in the range 1 to
32766 and greater than the one above)
b=&(32766)
c=n( > Ø < 32767)
c=p(Enter a number in the range 1 to 32766)
c=&(1)
d=n(Ø-132)
d=p(Enter a number for the left margin offset) d=&(Ø)
e=i
e=&(_)
e=p(If file selected from the directory listing is
    incorrect retype filename)
f=a(Y N y n)
f=p(If only even pages are to print change first page to print
   '2')
f=&(N)
f=m(n  '')
f=m(y  '-a')
f=m(N '')
f=m(Y '-a')
j=a(Y n Y N)
j=&(N)
j=p(Enter 'y' for page by page printing, or 'n' for continuous
   printing
j=m('y'  '-i')
j=m('Y'  'i')
j=m('N'   '')
l=@j
l=m('y'  '  | cutspool')
l=m('Y' ' | cutspool')
l=m('n' ' | lpr')
l=m('N' '  |lpr')
k=@j
k=m('y'  '')
k=m('Y' '')
k=m('n' '&')
k=m('N'  '&')
h=n()
h=&(1)
h=p(Number typed will be the number of copies
    printed of this file) i=n()
i=&(66)
i=p(Default entry is 66 lines, 78 lines = A4 paper sheet
    feeder)
g=g  ()
a=@g
g=p(Printers:  printer1 printer2 printer3)
g=&(printer1)
```

NOTE

The g field must always be the printer name field. The printer configuration program will change the g entry when printer names or the default printer name are changed.

There are eleven fields in the above example, a through k . In each field the type 'p' indicates an additional prompt line.

Examination of Field b:

```
b=n( >Ø < 32767 >=c)
b=p(Enter a number in the range 1 to 32766 and greater than
    or equal to the one above)
```

This field is for a number greater than Ø, less than Ø, less than 32767 and greater than the reply to field c (line 1) with a default value of 32766 (line 3).

Examination of Field c:

```
c=n( > Ø < 32767)
c=p(Enter a number in the range 1 to 32766)
c=&(1)
```

This field is for a number in the range 1 to 32766 inclusive (line 1) and has a default value of 1 (line 3).

Examination of Field d:

```
d=n (Ø-132)
d=p (Enter a number for the left margin offset)
d=&(Ø)
```

This field is for a number in the range Ø to 132 (line 1) and has a default value of Ø (line 3).

Examination of Field e:

```
e=i
e=&(_)
e=p(If file selected from the directory listing is incorrect
   retype file name)
```

This field is for an input file (line 1). The default is the paint character '_' (line 2). If vp is called with a filename as an argument, this filename will be substituted in this field. (eg. "vp -p filename" or "vp -p ^^" from a menu action).

Examination of Field f:

```
f=a(Y N y n)
f=p(If only even pages are to print change first page to print '2')
f=&(N)
f=m(n '')
f=m(y '-a')
f=m(N '')
f=m(Y '-a')
```

This is a field requirint a response of 'y' or 'n' or 'Y' or 'N' (line 1). The 'y' or 'Y' response will be mapped to '-a' (line 5 and line 7). The 'n' or 'N' response will be mapped to null.

Examination of Field g:

```
g=g()
g=p(Printer: printer1 printer2 printer3)
g=&(unigate)
```

This is a general field (line 1) which means the user can enter any combination of characters and digits in the field. No checking is done on this field. In order for the printer configuration program to work, field g must always be the Printer Name field.

Examination of Field a:

```
a=@g
```

This field is a ghost field taking its input from the reply to field g (line 1).

Examination of Field h:

```
h=n()
h=&(1)
h=p(Number typed will be the number of copies printed of this file)
```

This field requires its input to be any number (line 1) and has has a default value of 1 (line 2).

Examination of Field i:

```
i=n()
i=&(66)
i=p(Default entry is 66 lines, 78 lines = A4 paper sheet feeder)
```

This field requires its input to be any number (line 1) and has has a default value of 66 (line 2).

File locations:

Binary: 'vpc' within user's path

Instruction files:

Viewprint requires two sets of information to compile:

a)Messages
b)Details of the instruction to be compiled.

The compiler will look for the messages as a file called vpc.message in either $HOME/uniplex or /usr/uniplex. The compiler takes its input file from the current directory.

A profile of compilation operations is as follows:

1. Invoke the compiler :

vpc -p or vpc -m

The compiler will then check out your screen definition file (either vprint or vmerge) and if it is correct a message to this effect will be displayed.

If, however, the screen definition file is not correct a message indicating this fact will be displayed and the name of a report file displayed (usually the filename with the suffix '.er').

This file will contain all the lines correctly processed, the offending line, and under that an error message and if appropriate a pointer to the offending character(s).

The final output from the compiler for a successful screen definition is a control file for the run-time program. This will be the filename with a suffix of ".vp", e.g. vprint.up.

UPRINT

Uprint is the binary executable file that controls printing. The options are listed below and the general format of a Uprint command is as follows:

uprint [options] filename [filenames]

If more than one filename is given, they are printed in the order that they are given. Page numbers and headers/footers are continuous from one file to the next by default (options -r and -h alter this). If no options are given, the file will be printed, stripped of effects, to standard output. If several options or filenames are given, they should be separated by spaces.

The options are as follows:

-t Preview a file with print effect at the terminal.

-p Set the printing device, as specified in the Tcap file, to something other than the standard output (e.g., "uprint -p diablo filename")

-h Print headers and footers separately when printing multiple files.

-r Reset page numbers so that each file will start at page number one when printing multiple files.

-c Set the number of copies of the file to be printed to something other than one (e.g. "uprint -c4 filename").

-f Specify a form length different from the default of 66 lines (e.g. "uprint -f63 filename").

-S Set first page to print at page number other than one (e.g., "uprint - sl0 filename").

-E Set the last page number to be printed (e.g, "uprint -s2 -el0 filename" will print only pages 2 through 10).

-a Print alternate pages for easy collation. This is often used with the start first page to get even or odd pages (e.g., to get even pages use "uprint -a -s2 filename").

-i Interactively drive a cut-sheet printer. This causes uprint to prompt the user to continue printing or stop printing at the start of each page.

-m Set the left hand margin to an offset. This is likely to be used for cut-sheet printers.

```
                    *                                       *
                    *   STAND ALONE UNIPLEX COMMAND-FILE    *
                    *              Version 3                *
                    *                                       *
                    *  (c)  Redwood Bureau Services Limited *
                    *   St. Albans, Hertfordshire, England  *
                    *                                       *
                    *   Customized for Altos Computers      *
                    *                                       *
                    *****************************************
-----------------------------< System Attributes >----------------------
#SYSTEM
BACKUP='~/wp.back.up'
2SPACE='.!?:;',              3SPACE=''
RULER='.#CIJLTRH'
WDEL=';:,.!? ',              HLEN=5
DOTS='HE:HM:FO:FM:PL:PA:PN:SN:PM:SP:JY:JN:RE:ME:ST:SB'
KEYIN='Y'
MODE='T'
PNUM='#'
SPELL='/bin/sh /usr/bin/uspell ^'
DECTAB='.'
STOP=' *!&$'
STATUS='S'
))


---------------------------< Program Message Table >--------------------
#MESSAGES
'[?]  Select an option'
'File already in use '
'Selected menu not available or badly formatted.'
'Press <RETURN> to continue:'
'Unable to run system command/program.'
'Not a good file name.'
' selected.'
'No write permission '
'Please enter the file name or "*" to return to menu '
'Copying file to '
'Unable to open file for edit.'
'Cannot create new backup file.'
'Cannot write to file '
'*** Document copy ***'
'*** Document re-name ***'
'*** Document erase ***'
'Requested task is now complete.'
'Operation FAILED.'
'Original file'
'New file'
'Enter "*" key to confirm or <RETURN> to abandon:'
'Merge file section not found.'
'Enter name of file :'
'Enter string to search for:'
'Option ? '
'Sorry, search failed.'
'Enter directory name or "."=current, "*"=complete, "-l"=statistics'
'Unable to list selected directory'
```

```
'Enter "*" to exit or <RETURN> for more '
'*** List documents ***'
'*** External function ***'
'Busy saving the document......'
'*** Edit a file ***'
'*** Create a file ***'
'"-" to hyphenate or <RETURN> to skip:  '
'Enter "*" to replace or <RETURN> to skip '
'Enter new characters or <RETURN> to delete '
'Enter "*" for global or <RETURN> for interactive '
'Invalid command entered'
'Enter "*" to confirm quit or <RETURN> to continue '
'Busy executing command'
'Enter <RETURN> for next or "*" to quit '
'Enter ruler number (0-9)'
'Enter (B)lank, (L)eave, (R)emove or (W)rite'
'Enter (O)verlay, (E)lbow or (I)nsert'
'Enter 1 (tab), 2 (hyphen), 3 (underline text), or 4 (insert) :'
'Enter character to underline with'
'Enter <RETURN> for next page,"*" <RETURN> to quit'
'INSERT'
'HYPHEN'
'TAB'
'Current directory : '
'Getting your directory listing'
'Directory, enter "*" to change directory'
'No write permission, enter "*" for read only '
'Sorry -- no read permission on this file'
'Move cursor to table of names or enter a file/directory name'
'*** Change directory ***'
'Enter a directory name then press <RETURN>'
'Sorry -- cannot change to requested directory'
'Enter page number, <RETURN> :'
'Enter "*" for more '
'File exists - enter "*" to overwrite or <RETURN> to abandon '
'Enter "*" to find word or <RETURN> to skip '
))
```

----------< Function Table Reference List >----------

```
#COMLOCATE
F01=Delete char.   F02=Destructive Bsp.   F03=Delete word   F04=Delete right
F05=Delete left    F06=Delete line        F07=Delete blanks F08=Insert space
F09=Insert line    F10=Insert Blank lines F11=Return        F12=Left
F13=Right          F14=Up                 F15=Down          F16=Tab
F17=Previous word  F18=Next word          F19=Line Up       F20=Line Down
F21=Go left        F22=Go right           F23=Go Up         F24=Go Down
F25=Top of Screen  F26=Bottom of Screen   F27=Scroll down   F28=Scroll Up
F29=Top of File    F30=Bottom of File     F31=Lower Case    F32=Upper Case
F33=Center Line    F34=Emphasize text     F35=Show effect   F36=Line split
F37=Do again       F38=Format paragraph   F39=Help          F40=Redraw
                                                                Screen
F45=Stop command   F46=Go To Page         F47=Recall Ruler  F48=Store ruler
F49=Use Ruler      F50=Use mode           F51=Stop mode     F52=CP Mark
                                                                (block)
F53=CP blank       F54=CP leave           F55=CP Remove     F56=Not In Use
F57=CP Overlay     F58=CP Insert          F59=CP Elbow      F60=Menu escape
F61=Exit & Save    F62=Quit (no save)     F63=Write no exit F64=Not In Use
F65=Merge Insert   F66=Restore text       F67=Insert on     F68=Insert off
F69=Effect ON      F70=Effect OFF         F71=CP Mark (ser) F72=Save CP text
F73=Merge Overlay  F74=Save to File       F75=Spell
```

----------< Keyboard Definition Table >----------

```
#COMMANDS
F01=3,             F02=127,           F03=23,            F04=$-'DR'
F05=$-'DL',        F06=24,            F07=$-'DB',        F08=5
F09=15,            F10=$-'AL',        F11=13,            F12=L
F13=R,             F14=U,             F15=D,             F16=T
F17=16,            F18=14,            F19=$-'^',         F20=$-'V'
F21=$-L,           F22=$-R,           F23=$-U,           F24=$-D
F25=20,            F26=2,             F27=4,             F28=21
F29=$-'T',         F30=$-'B',         F31=$-'KL',        F32=$-'KU'
F33=$-'C',         F34=$-'_',         F35=$-'@',         F36=$-'L'
F37=$-'.',         F38=6-'P',         F39=$-'H',         F40=$-'#'
F41=$-'N',         F42=$-'F',         F43=$-'G',         F44=6-'D'
F45=28,            F46=$-'P',         F47=$-'R',         F48=$-'S'
F49=$-'U',         F50=$-'{',         F51=$-'}',         F52=$-'(B'
F53=$-')B',        F54=$-')L',        F55=$-')R',        F56=0
F57=$-'*O',        F58=$-'*I',        F59=$-'*E',        F60=$-'!'
F61=$-'E',         F62=$-'Q',         F63=$-'W',         F64=0
F65=$-'MI',        F66=18,            F67=$-'I',         F68=$-'O'
F69=$-'<',         F70=$-'>',         F71=$-'(S',        F72=$-')W'
F73=$-'MO',        F74=$-'X',         F75=$-'$'
F12=8,             F13=12,            F14=11,            F15=10
))

#COMMANDS-alt2
F01=3,             F02=127,           F03=23,            F04=$-'DR'
F05=$-'DL',        F06=24,            F07=$-'DB',        F08=5
F09=15,            F10=$-'AL',        F11=13,            F12=L
F13=R,             F14=U,             F15=D,             F16=T
F17=16,            F18=14,            F19=$-'^',         F20=$-'V'
F21=$-L,           F22=$-R,           F23=$-U,           F24=$-D
F25=20,            F26=2,             F27=4,             F28=21
F29=$-'T',         F30=$-'B',         F31=$-'KL',        F32=$-'KU'
F33=$-'C',         F34=$-'_',         F35=$-'@',         F36=$-'L'
```

```
F37=$-'.',          F38=6-'P',          F39=$-'H',          F40=$-'#'
F41=$-'N',          F42=$-'F',          F43=$-'G',          F44=6-'D'
F45=28,             F46=$-'P',          F47=$-'R',          F48=$-'S'
F49=$-'U',          F50=$-'{',          F51=$-'}',          F52=$-'(B'
F53=$-')B',         F54=$-')L',         F55=$-')R',         F56=0
F57=$-'*O',         F58=$-'*I',         F59=$-'*E',         F60=$-'!'
F61=$-'E',          F62=$-'Q',          F63=$-'W',          F64=0
F65=$-'MI',         F66=18,             F67=$-'I',          f68=$-'o'
F69=$-'<',          F70=$-'>',          F71=$-'(S',         F72=$-')W'
F73=$-'MO',         F74=$-'X',          F75=$-'$'
F12=8,              F13=12,             F14=11,             F15=10

f37=$-'OP',         f48=$-'OQ',         f73=$-'OR',         f50=$-'OS'
f28=$-'Ow',         f23=$-'Ox',         f27=$-'Oy',         f51=$-'Om'
f21=$-'Ot',         f26=$-'Ou',         f22=$-'Ov',         f31=$-'Ol'
f17=$-'Oq',         f24=$-'Or',         f18=$-'Os',         f32=$-'OM'
f10=$-'Op',         f07=$-'On'

f09=$-'[L',         f06=$-'[M',         f01=$-'[P',         f08=$-'[@'
f28=$-'[T',         f27=$-'[S',         f25=$-'[f'

f39=1-'P'-13

f61=1-'@'-13,       f62=1-'`'-13
f63=1-'A'-13,       f74=1-'a'-13
f75=1-'B'-13,       f60=1-'b'-13
f65=1-'C'-13,       f43=1-'c'-13

f71=1-'D'-13,       f52=1-'d'-13
f53=1-'E'-13-'b',   f54=1-'E'-13-'l',   f55=1-'E'-13-'r',   f74=1-'E'-13-'s'
f58=1-'e'-13-'i',   f57=1-'e'-13-'o',   f59=1-'e'-13-'e'
f33=1-'F'-13,       f35=1-'f'-13
f69=1-'G'-13,       f70=1-'g'-13

f42=1-'H'-13,       f41=1-'h'-13
f38=1-'I'-13,       f44=1-'i'-13
f47=1-'J'-13,       f49=1-'j'-13
f67=1-'K'-13,       f68=1-'k'-13

f29=1-'L'-13,       f30=1-'l'-13
f46=1-'M'-13,       f40=1-'m'-13
f04=1-'N'-13,       f05=1-'n'-13
f03=1-'O'-13,       f66=1-'o'-13
))


--------------------< Messages for User Defined Prompts >-------------
#USER
'**** AUXILLIARY MESSAGES FOR USE WITHIN UNIPLEX SYSTEM CALLS ****'
'Enter document to look at:'
'Enter subject to look for:'
'Enter name of sub-directory:'
'Enter name of the file to find:'
'Enter Unix command-line:'
'Enter name of file to print:'
'Enter name of standard text file:'
'Enter [year]  or  [month year]  like 1984 or 5 1984:'
))
```

```
----------------------< Default Margin Ruler Definitions >---------------
#RULERS
))

#EFFECTS
A=Bold
B=Double Strike
C=Underscore
D=Underscore text
E=Superscript
F=Subscript
G=Elite
H=Big
I=Small
))

-------------------------------< Primary Menu >-------------------------
#SYSMENU
'*** Main Office System Menu ***'
'1 - Word Processing Menu'=M('#DOCPREP')
'2 - File Management Menu'=M('#FILEMANAGE')
'3 - Additional System Usage Menu'=M('#SYSCOMMS')
' '=>()
'D - Change Directory'=X()
'P - Printing'=*('vp -p ^^','6','R')
'L - List Files'=?('*')
'? - HELP'=M('#HELP')
' '=>()
'* - Leave Uniplex'=*()
3=C()
5=E()
21=*('^','5','P')
24=*('csh','5','P')
2=*()
))


#EDITMENU
'*** Document Menu ***'
'1 - Comprehensive Help Menu'=M('#HELP')
' '=>()
'2 - UNIX Command Line '=*('^','5','P')
' '=>()
'3 - Find the local Command File'=*('ls $HOME/uniplex/uniplex.rc')
' '=>()
'Press (Escape) to get back to the document'=>()
' '=>()
27=*()
))

#DOCPREP
'*** Word Processing Menu ***'
'----- Document Handling -----'=>()
'1 - Create a new file'=C()
'2 - Edit an existing file'=E()
'3 - File Checking Menu'=M('#CHECKER')
'4 - Look at file format before printing'=*('uprint -t 1 more','1','P')
```

```
'5 - Mail Merge'=*('vp -m ^^','7','R')
' '=>()
' '=>()
'D - Change Directory'=X()
'P - Printing'=*('vp -p ^^','6','R')
'L - List Files'=?('*')
'? - HELP'=M('#HELP')
' '=>()
'Press (Escape) To go back a menu'=>()
' '=>()
27=^()
3=C()
5=E()
21=*('^','5','P')
2=*()
))


#FILEMANAGE
'*** File Management Menu ***'
'----- Documents & Files -----'=>()
'1 - Copy a file  '=D()
'2 - Erase a file '=K()
'3 - Rename a file'=R()
'4 - File Checking Menu'=M('#CHECKER')
' '=>()
'------ Directories ------'=>()
'5 - Open a new directory'=*('mkdir ^','3','P')
'6 - Remove an empty directory'=*('rmdir ^','3','P')
'7 - Change directory'=X()
'8 - Directory of files'=?()
' '=>()
'P - Printing'=*('vp -p ^^','6','R')
'? - HELP'=M('#HELP')
' '=>()
'Press (Escape) to go back a menu'=>()
27=^()
3=C()
5=E()
21=*('^','5','P')
2=*()
))

#CHECKER
'*** File Checking and Measuring Menu ***'
'------- File statistics -------'=>()
'1 - Count lines, words, characters in a file'=*('wc ^^','1','P')
'2 - Full status report of a file'=*('ls -l ^^','1','P')
' '=>()
'------- Document Details ------'=>()
'3 - Search for files on a particular subject'=*('grep -l "^" ./*/* *','2','P'
'4 - Find the location of a file'=*('find . -name "^" -print','4','P')
'5 - View the first part of a file'=*('head ^^','1','P')
'6 - View the end of a file'=*('tail ^^','1','P')
' '=>()
' '=>()
'D - Change Directory'=X()
'P - Printing'=*('vp -p ^^','6','R')
```

```
'L - List Files'=?('*')
'? - HELP'=M('#HELP')
' '=>()
'Press (Escape) to go back a menu'=>()
27=^()
3=C()
5=E()
21=*('^','5','P')
2=*()
))

#SYSCOMMS
'*** Additional System Usage Menu ***'
'------------ Status ------------'=>()
'1 - Who is also on the computer'=*('who','','')
'2 - Date and time'=*('date','','')
'3 - Calender'=*('cal "^" | more','8','')
'4 - Find the local Command File'=*('ls $HOME/uniplex/uniplex.rc')
' '=>()
' '=>()
'D - Change Directory'=X()
'P - Printing'=*('vp -p ^^','6','R')
'L - List Files'=?('*')
'? - HELP'=M('#HELP')
' '=>()
'Press (Escape) To go back a menu'=>()
27=^()
3=C()
5=E()
21=*('^','5','P')
2=*()
))

#HELP
'*** Help System Menu ***'
'1 - Command Summary - Quick Reference'=*('u=`unidir`;more $u/uniplex.help',''
'2 - Brief Command Definitions'=M('#MOREHELP')
'3 - Using the Word Processor'=*('u=`unidir`;more $u/help/priml','','P')
'4 - Using Menus'=*('u=`unidir`;more $u/help/prim2','','P')
'5 - Using Files'=*('u=`unidir`;more $u/help/prim3','','P')
'6 - Using Commands'=*('u=`unidir`;more $u/help/prim4','','P')
' '=>()
'Press (Escape) To go back a menu'=>()
27=^()
))

#MOREHELP
'*** Brief Command Definitions Menu ***'
'A - Often Used Commands'=*('u=`unidir`;cat $u/help/often','','P')
'B - Cursor Movement'=*('u=`unidir`;more $u/help/cursor','','P')
'C - Scrolling Text'=*('u=`unidir`;cat $u/help/scroll','','P')
'D - Exiting and Saving Files'=*('u=`unidir`;cat $u/help/exit','','P')
'E - Deleting Text'=*('u=`unidir`;cat $u/help/delete','','P')
'F - Inserting Text'=*('u=`unidir`;cat $u/help/insert','','P')
'G - Altering Text'=*('u=`unidir`;cat $u/help/alter','','P')
'H - Emphasizing Text'=*('u=`unidir`;cat $u/help/emphasis','','P')
'I - Using Rulers'=*('u=`unidir`;cat $u/help/rulers','','P')
'J - Marking Text'=*('u=`unidir`;more $u/help/mark','','P')
```

```
'K - Moving Text'=*('u=`unidir`;cat $u/help/move','','P')
'L - Locating and Replacing Text'=*('u=`unidir`;cat $u/help/find','','P')
'M - Using Print Time Commands'=*('u=`unidir`;more $u/help/print','','P')
'N - Using Modes'=*('u=`unidir`;more $u/help/modes','','P')
'O - Merging Text'=*('u=`unidir`;cat $u/help/merge','','P')
'Press (Escape) To go back a menu'=>()
27=^()
))
```

Decimal	Octal	Hex	ASCII	Decimal	Octal	Hex	ASCII
0	000	00	NUL	32	040	20	SP
1	001	01	SOH	33	041	21	!
2	002	02	STX	34	042	22	"
3	003	03	ETX	35	043	23	#
4	004	04	EOT	36	044	24	$
5	005	05	ENQ	37	045	25	%
6	006	06	ACK	38	046	26	&
7	007	07	BEL	39	047	27	'
8	010	08	BS	40	050	28	(
9	011	09	HT	41	051	29	)
10	012	0A	LF	42	052	2A	*
11	013	0B	VT	43	053	2B	+
12	014	0C	FF	44	054	2C	,
13	015	0D	CR	45	055	2D	-
14	016	0E	SO	46	056	2E	.
15	017	0F	SI	47	057	2F	/
16	020	10	DLE	48	060	30	0
17	021	11	DC1	49	061	31	1
18	022	12	DC2	50	062	32	2
19	023	13	DC3	51	063	33	3
20	024	14	DC4	52	064	34	4
21	025	15	NAK	53	065	35	5
22	026	16	SYN	54	066	36	6
23	027	17	ETB	55	067	37	7
24	030	18	CAN	56	070	38	8
25	031	19	EM	57	071	39	9
26	032	1A	SUB	58	072	3A	:
27	033	1B	ESC	59	073	3B	;
28	034	1C	FS	60	074	3C	<
29	035	1D	GS	61	075	3D	=
30	036	1E	RS	62	076	3E	>
31	037	1F	US	63	077	3F	?

Decimal	Octal	Hex	ASCII	Decimal	Octal	Hex	ASCII
64	100	40	@	96	140	60	`
65	101	41	A	97	141	61	a
66	102	42	B	98	142	62	b
67	103	43	C	99	143	63	c
68	104	44	D	100	144	64	d
69	105	45	E	101	145	65	e
70	106	46	F	102	146	66	f
71	107	47	G	103	147	67	g
72	110	48	H	104	150	68	h
73	111	49	I	105	151	69	i
74	112	4A	J	106	152	6A	j
75	113	4B	K	107	153	6B	k
76	114	4C	L	108	154	6C	l
77	115	4D	M	109	155	6D	m
78	116	4E	N	110	156	6E	n
79	117	4F	O	111	157	6F	o
80	120	50	P	112	160	70	p
81	121	51	Q	113	161	71	q
82	122	52	R	114	162	72	r
83	123	53	S	115	163	73	s
84	124	54	T	116	164	74	t
85	125	55	U	117	165	75	u
86	126	56	V	118	166	76	v
87	127	57	W	119	167	77	w
88	130	58	X	120	170	78	x
89	131	59	Y	121	171	79	y
90	132	5A	Z	122	172	7A	z
91	133	5B	[	123	173	7B	{
92	134	5C	\	124	174	7C	|
93	135	5D	]	125	175	7D	}
94	136	5E	^	126	176	7E	~
95	137	5F	_	127	177	7F	DEL

ALTOS

2641 Orchard Park Way, San Jose, California 95134
(408) 946-6700, Telex 470642 ALTO UI

Printed in U.S.A.
P/N 690-15517-002

August 1984