Uniplex Technical Guide

Manual version: x.xx
Document version: V1.0

THIS PAGE INTENTIONALLY LEFT BLANK

COPYRIGHT NOTICE

Copyright © 1981-2000 Uniplex Software, Inc.
Unpublished. All rights reserved.
Software provided pursuant to license. Use, copy, and disclosure restricted by license agreement.

IXI Deskterm copyright © 1988-1993 The Santa Cruz Operation, Inc.
Word for Word copyright © 1986-1998 Inso Corporation. All rights reserved.
Multilingual spelling verification and correction program and dictionaries copyright © 1984-1997 Soft-Art, Inc. All rights reserved.
Portions derived from the mimelite library written by Gisle Hannmyr (gisle@oslonett.no) and used with permission.
Portion copyright © 1981-1993 Informix Software, Inc.

Uniplex, Uniplex Business Software, UBS, Uniplex II Plus, Uniplex Advanced Office System, AOS, Uniplex Advanced Graphics System, AGS, Uniplex Document Access, Uniplex Datalink, and Uniplex Windows are trademarks of Uniplex Software, Inc. All other names and products are trademarks of their respective owners.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the U.S. Government or other government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the rights in Technical Data and Computer Software clause at DFARS 252.227-7013. Uniplex Software, Inc., 715 Sutter Street, Folsom, California 95630.

Computer software and related documentation shall not be delivered to any branch, office, department, agency, or other component of the U.S. Government unless accompanied by this Restricted Rights Legend or alternatively, unless licensed expressly to the U.S. Government pursuant to FAR 52.227-19, unpublished -- rights reserved under U.S. copyright laws.

NOTICE

The information in this document is subject to change without notice. Uniplex Software, Inc. makes no warranty of any kind in regard to the contents of this document, including, but not limited to, any implied warranties of merchantability or fitness for a particular purpose. Uniplex Software, Inc. shall not be liable for errors in this document or for incidental or consequential damages in connection with the furnishing, performance, or use of it.

Manual version: x.xx
Document version: V1.0

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

About this Guide

THIS PAGE INTENTIONALLY LEFT BLANK

About this Guide

This technical guide is intended for anyone who wishes to configure Uniplex in order to tailor the software to suit the needs of any particular site. The information in the guide may also be useful for the System Administrator responsible for the day-to-day running of a Uniplex system.

Note for existing Uniplex users:

This guide is an amalgamation of a number of Uniplex Version 8.01 technical guides. These guides no longer exist and the information from them has been distributed as follows:

- Uniplex Configuration Guide (Volumes 1 and 2)

  The greater part of the guide can now be found in this guide. The chapters: "Configuring Terminals" and "Configuring Printers" are not part of this guide; they can be found in the Uniplex Device Configuration Guide.

- Uniplex Administration Guide

  The information has been divided between this guide and the Uniplex Form-Building Tools guide.

- Uniplex Installation Guide and Release Notes (Versions 7.02, 8.00)

  Installation information is now in the retitled Uniplex Installation Guide. Specific information from the Release Notes have been incorporated into the appropriate on-line guide; current Release Notes can now be found on-line.

- Uniplex Windows Configuration Guide

  Relevant information from the guide can now be found in the Uniplex Device Configuration Guide.

Assumptions

In order to carry out the configuration tasks described here, it is assumed that you have a good general knowledge of computing and that you are an experienced UNIX user. You should also be familiar with:

- The major Uniplex applications
- The way that Uniplex is used in your organization

Uniplex is a powerful system that can be used to perform a wide range of tasks. This guide suggests ways you might want to organize how Uniplex is used. In addition, the guide provides some of the background information you need to understand the way Uniplex works. For example, it explains how to set up and use backup files for use with the Word Processor.

Note: We expect you to use this guide in conjunction with an installed version of Uniplex. This enables you to browse the configuration files to view real examples of the information contained in this guide.

Non-English Versions of Uniplex

Some Administration tasks in this guide give the menu options in English only. If you are using a non-English version of Uniplex, you will see that these menu options have been translated in the software.

If you are following an Administration task described in this guide and are not sure of the equivalent menu option for your language, refer to your on-line help for further details.

Shell Syntax

While configuring Uniplex, you may need to set environment variables. The way to do this depends on the operating system shell you use.

For the Bourne shell, set variables using the syntax:

VARIABLE=*value*; **export** *VARIABLE*

For example:

TERM=ansi; export TERM

For the C shell, set variables using the syntax:

setenv *VARIABLE value*

For example:

setenv TERM ansi

Note: Throughout this guide, all examples for setting environment variables show the Bourne shell. If you are using a different shell, substitute the given command for the one appropriate to the shell your system uses.

Terminology

If you are not familiar with the basic concepts of a computer operating system, the following table introduces some of those most commonly used.

Term	Explanation
backup	A *backup* is a copy that is made of one or more files. You make backups to ensure you do not lose valuable work. You store backups on tapes or disks.
default	Uniplex uses some information *by default*. For example, when you complete a form, some of the information is entered for you by the system. Uniplex uses these default values unless you offer an alternative.
shell	The *shell* is the command language you use to communicate with the operating system.
superuser or root	If you login as *superuser* or *root*, you override existing permissions on files and directories. This is useful if you want to change the access permissions to files. To login as a superuser, you need to know the superuser password for your system.
files and documents	Uniplex stores the work you do with the word processor in *documents* and the work you do with other applications in *files*. This guide refers to both Uniplex documents and Uniplex files as files.

Term	Explanation
directories and folders	Uniplex stores your files in *folders*. The UNIX term for a folder is a *directory*. This guide refers to folders as directories.
local UAP central UAP	Used to distinguish between a user's home ($HOME/UAP) area, which may contain personal Uniplex configuration, and the main UAP where Uniplex was installed.

Chapter 1
Administrator Tasks and Configuration Concepts

THIS PAGE INTENTIONALLY LEFT BLANK

System Administrator Responsibilities

This section overviews the maintenance tasks you must perform to ensure the smooth running of the system.

In addition to the maintenance tasks you need to perform on a regular basis, you should also oversee the system so that it meets the requirements of individual users.

Installation of Uniplex

When Uniplex is first installed, there are a number of tasks you may need to perform to ensure the system is set up and ready for use.

Task	Further Information
Set the Uniplex environment for users at individual terminals	Look up: The Uniplex Environment
Define the printer interfaces used by different users in the organization	Refer to: The Installation Guide
Set up Mailboxes and define mail routes to other systems	Look up: Electronic Mail
Define the holidays and personal working time of all users	Look up: Time Manager

Regular Tasks

Uniplex is largely self-administering. However, depending on your system use, the following regular tasks are usually required:

Task	Related Information
Delete old files from user's Uniplex Trashcans.	System Administration Menu (next section) and **housekeeping -trash** command (Program Usage section).
Update the File Manager index to reflect files added or removed from the system, using non-Uniplex applications.	**ufmscan** command (Program Usage section).

Task	Related Information
Clear old Uniplex work files left behind after unplanned terminal disconnection (for example, a system crash).	System Administration Menu (next section) and **housekeeping -ucleanup** command (Program Usage section).

As shown in the previous table, each of these tasks can be performed manually (using a menu option and/or a command). In addition, the **ufmscan** command can run all these tasks.

The simplest way to perform all these tasks is with a job scheduling package such as **cron**. For example, the "root" user could create the following **crontab** entry:

```
# Run Uniplex tasks 2 minutes after midnight:
2 0 * * * /usr/bin/uniplex -run ufmscan -r -a -d -t -c
```

Alternatively, each operation could be run on a different cycle. For example:

```
# Update File Manager Index as required 2 minutes after midnight:
2 0 * * * /usr/bin/uniplex -run ufmscan -r -a -d
# Clear out work files at 5am.
0 5 * * * /usr/bin/uniplex -run housekeeping -ucleanup
# Clear old Trash files at 3am. on Saturday's
0 3 * * 6 /usr/bin/uniplex -run housekeeping -trash
```

Other Periodic Tasks

There are tasks that you may have to perform on an occasional, possibly monthly, basis.

Task	Notes and Related Information
System backup.	As with any other software package that creates and changes data, you should regularly backup all areas of your system that contain or are used by Uniplex.
Telling the File Manager about files relocated using symbolically linked directories.	Look up: Handling of UNIX Links and Symbolic Links in the later section Administering and Configuring File Manager.
Update the Uniplex Dictionary.	Refer to the Uniplex Business Software User Guide Supplement.

The System Administration Menu

The System Administration Menu groups together a collection of options that you can use to help in the smooth running of the system. You can access the System Administration Menu directly from the main menu.

The table below briefly describes the tasks you can perform using the options from the System Administration menu:

Option	Function
Printer Administration	Use this option to define the printers used on the print screen. See the Uniplex Installation Guide for details.
Plotter Administration	Use this option to define the plotters for use with the Advanced Graphics System. See the Uniplex Installation Guide for details.
Delete Old Trash Files	Use this option to empty the Trashcan of old files. You can enter the age in days beyond which you want files to be deleted.
Clean the Uniplex Environment	Checks and, if necessary, removes redundant files after a system crash.
Enable/disable External Mailbox	Add or remove definitions of the frequency at which the Advanced Office System will collect mail from external machines. See the sections Enabling External Mailboxes and Disabling External Mailboxes in the Uniplex Installation Guide for details.
On-line Documentation	Use this option to read or print any Uniplex on-line document. See the Uniplex Business Software User Guide Supplement for a list of available on-line documentation.
Software Installation	Provides access to the Installation Options menu, from which you can change the default Uniplex installation. See the Uniplex Installation Guide for details.

Configuration Possibilities

Uniplex has been designed so that it can be configured or translated to suit the needs of any site. This chapter provides an overview of the configuration possibilities available to you and provides some guidelines to follow when configuring Uniplex.

You can modify Uniplex to suit your requirements in the following different ways:

- Use your choice of Peripheral Devices

  Uniplex supports a wide range of terminals and printers. You can change the way these operate with Uniplex and, if required, provide support for other, as yet unsupported, devices. For example, you can change the way Uniplex uses a terminal's video attributes. See the chapters Configuring Terminals and Configuring Printers in the Uniplex Device Configuration Guide for details.

- Reorganize or Redesign the User Interface

  It is easy to modify the Uniplex menus. For example you can remove menu options to restrict access to certain applications for a group of users or you can add menu options to provide additional applications for a group of users. See the chapter Configuring Menus for details.

  You can change the way users operate Uniplex. For example, you can configure command keystrokes to emulate other software packages, so providing an easier learning path into the Uniplex environment. Additionally, you can configure softkey menus. See the chapters Configuring Softkeys, Ring Menus and Popups, and Configuring Command Keystrokes for details.

- Customize Applications

  All the Uniplex applications can be configured. For example, you can configure the Word Processor operating modes to default to insert mode rather than overwrite mode.

- Translate or Change Screen Information

  You can translate all text, messages, help and menus for foreign language versions of Uniplex. Or you can change the text to be specific to your site.

Configuring Uniplex for Individuals

You can configure Uniplex in different ways for individual users. You create a copy of the central UAP directory in the user's home directory. You copy the system files you need to change into this UAP directory.

When you invoke Uniplex, it checks for the existence of the UAP directory in the user's home directory, and if it exists uses the system files that exist there, and not the central UAP files. When you create a UAP directory in a user's home directory, it is known as a *local UAP directory*.

Create a local UAP directory as follows:

1 Create a UAP directory in the user's home directory. For example:

```
mkdir /usr/john/UAP
```

2 Copy the configuration files you require into this directory. For example, if you want to change the main menu for user John, copy uniplex.menu into his local UAP directory:

```
cp /usr/UAP/uniplex.menu /usr/john/UAP/.
```

You can modify the system files in the local UAP directory. The changes you make are only implemented for the individual user, in this case **john.**

Note: Remember that some system files require compilation after you modify them for the changes to take effect. See the appropriate chapter of this guide for details.

Configuring Uniplex for Groups

Using standard UNIX techniques for linking files, you can extend the ability for individuals to have local configuration files to provide group configuration.

For example, you could do the following to let a group of users share a non-standard menu file:

1 Create the new file in the central area, such as **/usr/UAP/finance.menu**.

2 For each user in the Finance department, create a $HOME/UAP directory and then issue the UNIX command:

```
ln -s /usr/UAP/finance.menu /usr/john/UAP/uniplex.menu
```

Maintaining System Integrity During Configuration

Follow this procedure when configuring system files:

1 Before starting configuration, make a backup of the UAP directory on disk or tape.

2 Create a local UAP directory for yourself. See the previous section.

3 Before configuring a particular file, make a copy of the file.

4 Test your configuration changes.

In most application areas, you need to leave Uniplex, return to the operating system prompt and re-enter Uniplex before your configuration changes are actioned.

5 When you are satisfied your changes work as intended, copy them to the central UAP directory, or the UAP directory of the individual or group you are configuring for.

You can edit all system text files. You cannot configure the following non-text files and directories:

File	Explanation
bin/*	Contains all the program binaries.
/.sk	Contains compiled softkey files.
system.comp	Contains the compiled system file.

Chapter 2
The Uniplex Environment

THIS PAGE INTENTIONALLY LEFT BLANK

The UAP Directory Structure

The Uniplex system files and program binaries are located in the UAP directory structure. At installation, you can define where this directory is placed. UAP contains the following files and directories:

Files	Description
Terminal configuration: Gcap Tcap termcap terminfo termset termreset	These files define the characteristics of the different types of terminals you can use with Uniplex. You can change the way a terminal behaves or provide support for additional terminals using these files. See the chapter Configuring Terminals in the on-line Uniplex Device Configuration Guide.
commands (directory) termdef (directory) TERMINFO (directory)	These directories contain data created from the preceding files by the skcomp and TIC programs.
Printer configuration:	
filters (directory) Fcap Gcap Pcap PRINT (directory)	Directory containing printing filters. These files define the characteristics of the different types of printers you can use with Uniplex. You can change the way a printer behaves or provide support for additional printers using these files. See the chapter Configuring Printers in the Uniplex Device Configuration Guide.
System configuration: system.comp uniplex.sys uniplex.cmd uniplex.menu uniplex.eff	These system files define different aspects of the Uniplex user interface. You can configure Uniplex using these files to suit the needs of your site. See the chapters Configuring the System Parameters, Configuring the System Commands, and Configuring Menus.
Binary Directory: bin	The sub-directory *bin* contains the Uniplex binaries. You cannot change these files. A complete list of the binaries and their invocation flags is given in the appendix Program Usage and Invocation.

Files	Description
Script Directory:	The sub-directory *cmds* contains the Uniplex scripts.
cmds	
Application Directories:	Each application has a sub-directory that contains files required by that application only.
dbs	Database Links
diary	Time Manager (usdiary)
pc	External Windows
ped	Presentation Editor
popup	Popup PTOS, etc.
pr	Printing
tapes	Keyrecorder
uc	Spreadsheet
ucard	Card Index
ucdiary	onGO Character Client Time Manager
uchart	Presentation Graphics
ucmail	onGO Character Client Mail
ufilemgr	File Manager
ufill	Formfill and Screen Builder
uform	Database Forms
ugraph	Character Graphics
umail	Electronic Mail (usmail)
ureport	Report Writer
urmenu	Report Writer menu
uspell	Speller
usql	Database Query
wp	Word Processor
Electronic Mail	Electronic Mail system files.
uniplex.alias	
Installation	Installation support scripts and directories.
install	
install.cmds (directory)	
.installation (directory)	
files_*	A list of the files in UAP.
uniplex.key	License key information.
Demonstration files:	The sub-directory (demo) contains the files supplied to help users learn how to use Uniplex.
demo (directory)	

Files	Description
Miscellaneous:	
DATA (directory)	Contains the File Manager index database.
dict (directory)	Contains all the dictionaries used by the spellchecker (uspell).
dict.src (directory)	Files used to (re)build dictionaries.
general (directory)	Common softkey and message files.
informix (directory)	Informix database engine(s).
outlines (directory)	Contains the mailmerge format files.
sort.weights	Uniplex sorting rules.
unsupported (directory)	Directory containing useful, but unsupported scripts and programs.
NVO (directory) NVO.subset (directory)	onGO Character Client and Document Access support.
uxwindows (directory) XW (directory)	Uniplex Windows Uniplex Windows

The Uniplex Startup Script

You invoke Uniplex using the startup script *uniplex*. It itself performs most processing by calling *UAP/cmds/uniplex.start*.

This Bourne shell script is installed, as */usr/bin/uniplex*, though this name and parent directory can be changed during installation.

If Uniplex Windows is installed, then its front-end script, *uxwindows*, is installed in the same directory and, unless the system has been upgraded from an earlier version of Uniplex Windows, this is a link to the main front-end script.

Also, whenever Uniplex Windows is installed, the main front-end script has the fixed name *uxuniplex* linked to it to provide a guaranteed mechanism for opening a new application window.

This script, which is dynamically built during Uniplex installation (by the UAP/install program), imposes a negligible processing overhead, but does perform the following very important functions:

1 Displays the UNIPLEX banner and copyright notice.

2 Unless preset, sets the *Uredirect* environment variable to point to the directory in which Uniplex is installed.

3 Unless preset, sets the *TERMINFO* or *TERMCAP* environment variable (as appropriate to your system) to address the Uniplex supplied terminal description database (within *$Uredirect/UAP*).

4 Extends the user's *PATH* environment variable to include both the *UAP/cmds* and *UAP/bin* directories.

If invoked with the **-run** argument, also adds *UAP/install.cmds* to the PATH to simplify the invocation of **ufmbuild**.

5 If there is no Uniplex-compiled description for the user's terminal, it compiles one and sets up any other appropriate environment variables.

To make it easier to integrate other applications or administration procedures with Uniplex, the startup script can be invoked with the argument **-W** which causes it to report the directory in which Uniplex is installed.

It also provides the argument **-run** to enable any Uniplex program to be run when not within a Uniplex session. This pre-sets the required environment (PATH, TERMINFO, and so on) before using the remaining command line arguments as a command to run.

For example, the following could be used to start up **uclock** from the system reboot file (*/etc/rc*):

```
/usr/bin/uniplex -run ustartclock
```

Pre-empt the PATH in Uniplex Environment

To better enable localized configuration and customization, you can arrange to "pre-empt" any Uniplex programs by setting the environment variable **UprePATH** before running Uniplex.

This is processed by the front-end script and lets you specify directory(s) to be set on the Uniplex PATH before UAP/cmds and UAP/bin.

Environment Variables

You set up some aspects of a user's Uniplex environment using *environment variables*. Some variables must be set for Uniplex to work correctly, most variables are optional.

Uniplex uses some operating system variables and some Uniplex-specific ones.

You can set environment variables in a number of ways:

- In system files, the system administrator can set the common environment variables that each user requires. For example, the setting of the PATH environment variable.
- In the user's *startup* file. The operating system reads and executes this file as soon as the user logs in. The name of the startup file and the syntax you use to set the variables depends on the shell your system uses:

Shell	Startup File
Bourne	.profile
C	.cshrc

  You can edit these files using Uniplex or using any operating system editor (for example, *vi*).

- From an operating system prompt; if you do this, they are only set for this work session, when you log out the setting is lost.

There are four types of variable used by Uniplex:

- Operating system environment variables
- Uniplex-specific environment variables
- Variables set automatically by Uniplex
- Variables that Uniplex recognizes, but that you do not need to set in normal circumstances

The following sections explain all these types of variables.

Operating System Environment Variables

The operating system variables available depend on the version of the operating system you are using. For Uniplex to operate correctly, make sure the following three standard operating system variables are set: HOME, PATH and TERM. Usually these are set by the system administrator in the appropriate system files. For example, for UNIX Version 5.2, running the Bourne shell, the HOME variable is set in */etc/passwd*.

Variable	Syntax	Description
HOME	HOME=*dir* For example: `HOME=/usr/andrea`	Sets the user's home directory.
PATH	PATH=*dir*:*dir*:...*dir* For example: `PATH=/bin:/usr/bin`	Specifies the directories that the system searches for executable programs. Whenever you invoke a program or command by entering its name, the system searches for the directories specified with PATH for the program to execute. Make sure that you specify the location of the startup script *uniplex* in the list you specify for PATH. The Uniplex front-end script always ensures that PATH contains the necessary Uniplex ($Uredirect/UAP/bin and $Uredirect/UAP/cmds) and Uniplex Windows (e.g. /usr/bin/X11) directories.
TERM	TERM=*type* For example: `TERM=ansi`	Specifies the *type* of terminal the user is using. Each supported terminal type is defined in the Uniplex *termcap* or *terminfo* files and in the *UAP/Tcap* system file.

Uniplex Environment Variables

There are a number of variables specific to Uniplex. You do not have to set these variables, but they allow you to modify a user's Uniplex environment.

Variable	Syntax	Description
DBPATH	DBPATH=*dir:dir:...dir* For example: `DBPATH=/usr/UAP/demo:/doc`	An Informix-only variable. Specifies the directories that Uniplex searches for databases. Whenever you request a list of all your databases Uniplex searches all the directories specified by DBPATH.
DBDEFAULT	DBDEFAULT=*link* For example: `DBDEFAULT=INFORMIX5SE`	Specifies the default database link. This will override the setting in the file *UAP/dbs/default*.
DBTEMP	DBTEMP=*dir:dir:...dir* For example: `DBTEMP=/overflow/tmp`	An Informix-only variable. Specifies the directory Informix uses for work files, such as temporary tables when doing a complex SELECT. Defaults to */tmp*. Note: The File Manager interface always sets DBTEMP to $Utemp (in the UAP/dbs/interfaces file).
Ucalcautomacro	Ucalcautomacro= *macro name* For example: `Ucalcautomacro=ONread`	Defines a named spreadsheet macro which can be run whenever a full sheet has been read in (for example, on edit sheet, or File-Retrieve).
UprePATH	UprePATH=*dir* For example: `UprePATH=/test/bin`	See earlier section, "Pre-empt the PATH in the Uniplex Environment".

Variable	Syntax	Description
U_MAXNUMFIELDS	`U_MAXNUMFIELDS=100`	When running in Uniplex Windows mode, defines the maximum number of "sensitive" fields that the terminal emulator can handle. From version 8.10, the maximum, and the default, is 200, which is the number that the UNIX/Motif UTERM can handle. However, earlier versions only supported a value of 100. If you use a terminal emulator that was written for the old 100 limit (for example, onGO Office for Windows), you may need to set this if using small fonts in tall windows (which can result in many sensitive fields with, for instance, the File Manager).

Variables Set by Uniplex

The following table shows the variables set by the startup script *uniplex*:

Variable	Example Setting	Description
Uredirect	Uredirect=/work	Specifies the directory containing the UAP directory.
TERMCAP	TERMCAP=/usr/UAP/termcap	Specifies the location and name of the Uniplex termcap.
TERMINFO	TERMINFO=/usr/UAP/TERMINFO	Specifies the location and name of the Uniplex terminfo directory structure.
Ufrontend	Ufrontend=uniplex	Specifies the startup script being used, to prevent recursive execution of Uniplex.

Variable	Example Setting	Description
NVO	NVO=$Uredirect/UAP/NVO	Specifies the location of the onGO subsystem used by Uniplex.
U_1stimechk	U_1stimechk=Done	Used to optimise the check that this is the first time a user has run this version of Uniplex.

Note: Depending on your operating system, Uniplex uses either the terminfo directory structure or the termcap file, not both. See the chapter Configuring Terminals in the Uniplex Device Configuration Guide for more details.

The following table shows the environment variables set automatically by Uniplex binary programs (those in UAP/bin) during operation.

Note: Uniplex shell scripts (in UAP/cmds), which are designed to be run from Uniplex binaries, expect some of these variables to be pre-set (in particular, Uniplex).

Variable	Example Setting	Description
Ucutfile	Ucutfile=/tmp/WPCP0tty05	Specifies the name and location of the cut and paste buffer (clipboard) 0.
Ufile	Ufile=report.doc	Specifies the name of the document the user is currently editing.
Uidcode	Uidcode=WP	Specifies the product code of the product the user is currently using:

Code	Application
WP	Word Processor
SS	Spreadsheet
DB	Database
TM	Time Manager
EM	Electronic Mail
CI	Card Index
SB	Formfill
GR	Graphics

Variable	Example Setting	Description
Ulocal	Ulocal=/usr/john/UAP	If you are using a local UAP directory, specifies the name of this directory.
Umfile	Umfile=*pathname* For example: `Umfile=/usr/UAP/umail/text/jld/mbox1/I/0d1fb.00003412`	Variable set by usmail, whenever an incoming message has been selected for read, to the full pathname of the message file in the mailstore.
Uniplex	Uniplex=/usr/UAP	Same as Unode, unless using no longer supported Uniplex network configuration, when it specifies the location of the network UAP directory.
Unode	Unode=/usr/UAP	Specifies the location of the central UAP directory.
Uppid	Uppid=4633	Specifies the parent process identification number.
Urealid	Urealid=104	Specifies the user's identification number, as defined in the system password file /etc/passwd for the login name the user first logged in as. Normally, this is the same as Uuserid; this variable is useful when a user has logged in more than once, without logging out (using su).
Utemp	Utemp=/tmp	Specifies the temporary directory as defined in the #SYSTEM of uniplex.sys.
Utmpfile	Utmpfile=/tmp/UC1234	Specifies the name of the file created by the F155 function in the Word Processor, and the Spreadsheet command "print in TMPFILE".

Variable	Example Setting	Description
Utrash	Utrash=/tmp/trash	Specifies the trashcan directory, as defined in uniplex.sys.
Utty	Utty=tty05	If the environment variable UNINAME is not set, specifies the name of the current device. Uniplex uses up to 9 non-/ characters following the second slash (/) to determine the name. For example: **TTY Name** / **$Utty** /s9/tty10 / tty10 /dev/tty10/0 / tty100 /dev/remote/A/tty/0 / moteAtty0 If UNINAME is set, Utty is set to the same value as UNINAME.
Uuserid	Uuserid=104	Specifies the user's identification number as defined in the system password file /etc/passwd, for the login name the user is currently logged in as.
Uusername	Uusername=john	Specifies the user's login name as defined in the system password file /etc/passwd.
Uxwindows	Uxwindows=2	When set to a non-zero value, this variable tells Uniplex applications that they are running in an X environment, as opposed to on a character-based terminal.

Variable	Example Setting	Description
U_plid U_pmid U_windows U_xpid	U_plid=12345 U_pmid=12345 U_windows=1 U_xpid=12341	Variables set and read by Uniplex processes to manage the interactions of Process Switching.
WIN*name*	WIN*name*=*size*	Defines the size and position of the window, when an application is invoked in a window. Where *name* is the name of the application. This variable is not normally set by configurers. It is set automatically in softkey calls, where required. See the chapter Configuring Softkeys, Ring Menus and Popups for details.
XENVIRONMENT	XENVIRONMENT=/usr/tmp/file	This is set by **uxspawn** to point to a temporary file created during initialization to hold all relevant X resources for Uniplex applications it invokes. The contents of the file include the loaded server file, the files *UAP/XW/resources.host* and *UAP/XW/resources.comm*, and other standard X sources.

Restricted Use Environment Variables

The following table lists a number of other environment variables that Uniplex recognizes, but which you do not normally need to set.

Variable	Example Setting	Description
PRTOPT	PRTOPT=off	When set to "off", **uprop** does not optimize reading of the Fcap file by remembering the location of the sections.
Ucalcformat	Ucalcformat=*value* For example: `Ucalcformat=old`	Use an old, machine-dependent file format when saving spreadsheets.
Ucalcmacro	Ucalcmacro=*program name*	Defines the API which causes spreadsheet macro requests to be passed to an external program for execution.
Ucalcmode	Ucalcmode=ucalc ¦ issi For example: `Ucalcmode=issi`	Specifies the user interface mode for running the Spreadsheet. This must be the Industry Standard Spreadsheet Interface (issi), or the original Uniplex Spreadsheet interface (ucalc). The main difference between these is that issi is a command-menu driven interface and ucalc is a text-command driven interface.
Ucharset	Ucharset=iso-8859-2	The name of the character set used on this system. Used when creating MIME-encoded outgoing mail. The default is iso-8859-1.
Uclock	Uclock=/usr/shared	Specifies the location of the uclock locking file (.uclock.lock). Default is $Unode/diary.
UCLOCK_V7MODE	UCLOCK_V7MODE=no ¦ yes	Retains the V7 logic of not delivering alarms unless the user’s tty is writeable.

Variable	Example Setting	Description
UCORE	UCORE=REMOVE	Reserved for use under direction from Uniplex.
Udbname	Udbname=salesdb	Set by **uform** whenever the user selects a database name from its menu.
Ufilter	Ufilter=gd_tekconfig	Specifies the graphics filter to use to identify and setup a graphics terminal for running the graphics applications. There are different filters available for different types of terminals. This variable overrides the setting of the FILTER flag in Tcap.
UFM_CLOSE_DB	UFM_CLOSE_DB=yes	Set to a non-empty string to make the File Manager issue a "close database" command when it exits. By default, the File Manager relies on the database engine terminating cleanly when it exits.
UFM_LWM UFM_HWM	UFM_LWM=1000 UFM_HWM=4000	Set the File Manager directory content cache low water (UFM_LWM) and high water (UFM_HWM) marks. The values are limits on the total of files/folders which occur in any cached lists. If neither are set, the default values are 750 for each. If just one of the variables is set, both high- and low-water marks are set to that value.
Ugrep	Ugrep=/tmp/file.list	Name of file containing the default list of files to search using the **grep** UFILL screen.

Variable	Example Setting	Description
Uleavecp	Uleavecp=1	If set to a non-null string, stops Uniplex removing cut and paste buffers on exit. Used by Uniplex DOS to allow data to be preserved across more than one host login session.
Ulstdir	Ulstdir=*ii*:*mmm* For example: `Ulstdir=128:10048`	Controls the memory allocation when reading and sorting the names in a pick and point list of files, or the softkey X(,,,L,...) command. *ii* defines by how much to expand the memory when required (from an initial allocation sufficient for 64 files). *mmm* defines the maximum to allocate. Both numbers are in numbers of items in the list.
Umailaddr	Umailaddr=no ¦ yes	Specifies whether the address fields (TO, CC, BCC) in send forms should allow up to 200 characters instead of the default 110 (which often is not enough). Since earlier versions of Uniplex cannot safely receive messages with long addressee fields, this is not set by default.
Umailsync	Umailsync=no ¦ yes	Specifies whether to synchronize UMAIL/UMAILEXEC so that UMAIL does not exit before UMAILEXEC has completed. Setting this variable causes UMAIL not to exit until all outstanding backend activity has completed.

Variable	Example Setting	Description
UNDO	UNDO=0 ¦ 1 For example: `UNDO=1`	Disables (0) or enables (1) the Word Processor undo facility. Single line undos continue to work when you set this flag to zero. More complex tasks do not. Set this flag to zero to improve performance.
Unetwork	Unetwork=/net/uniplex	No longer supported (specifies the location of the UAP directory for the network layer).
UNINAME	UNINAME=steve	Specifies an alternative name for the cut and paste buffer. Uniplex normally uses the tty name to create the buffer name. Set this variable for systems that support windows, for example workstations. In this case each window has a separate TTY name.
UNIPLEX	UNIPLEX=/usr	Specifies the location of UAP and overrides *all other* mechanisms for specifying the location of Uniplex.
UNIPLEX_ EXTRACT	UNIPLEX_EXTRACT= 'tar xvf /CDdrive/app'	Sets the default command used on the Add-on and Upgrade media extraction forms.
UNIUTMP	UNIUTMP=*device name*	Defines a non-standard subset of the TTYNAME in the /etc/utmp/file.
UNOOPT	UNOOPT=YES	Disables cursor optimization for all products. The product uses the cursor movement command as defined in *termcap* or *terminfo* instead of optimizing. To enable cursor optimization this must be unset.

Variable	Example Setting	Description
Unowpdptrack	Unowpdptrack=no	Set to any value to let **popup ptos** display all text in an arbitrary text file (see **popup** in the appendix Program Usage Control and Invocation).
UPC_LOCK	UPC_LOCK=*integer* For example: `UPC_LOCK=5`	Controls the locking of the MASTER table for the once-per-user-per-terminal update required to register a new user. Values are: 0 No locking. non-0 Lock the table, then sleep for the specified number of seconds before performing the update and locking. -1 Lock, no sleep (default).
UPC_PATCH	UPC_PATCH=no ¦ yes	Enables or disables the UPC_PATCH fix. Extremely rapid Process Switching can cause Uniplex to crash or hang if the user makes two Process Switch requests faster than UNIX can schedule processes. In practice this only affects automated user input using a Remote Terminal Emulator (RTE). For example, using an RTE, run up two WP windows and then fire a constant stream of Switch Process (ESC x s) commands at them. Correct RTE Practice is to allow an amount of think time on each screen display.
Uprinters	Uprinters=*filename* For example: `Uprinters=plotters`	Specifies the file containing details of available printers or plotters on your system. This file is used by the print interface manager **uprtcmd**.

Variable	Example Setting	Description
Uproc	Uproc=*n* For example: `Uproc=2`	Uproc specifies the number of processes (*n*) you can run using External Windows. It overrides the definition of the PROC flag in uniplex.sys. Each operating system has a limit to the number of processes you can specify, although Uniplex has none. However large numbers of concurrent processes can seriously impact system performance. See the chapter Configuring System Parameters (uniplex.sys) for details.
Use_tabs	Use_tabs=y	Unless this environment variable is set UBS will not exploit the tabbing capabilities of the display being used. NOTE that to use the tabbing capabilities the Tcap description must include a **PT** entry, and the UNIX stty tabbing options must be set in an appropriate manner.
USHELL	USHELL=/bin/bsh	Reserved for use under direction from Uniplex.
USORT_LANG	USORT_LANG= *section-name* For example: `USORT_LANG=LATIN2`	Specifies the section in *UAP/sort.weights* that specifies the 8-bit sort sequence.
UTEST_INDEX	UTEST_INDEX=off	Forces the File Manager's "Use Index" preference to "No".
Utest_nofhdr	Utest_nofhdr=no	When set to a non-empty value, overrides the setting of the Word Processor "Generate Style Information" working option, forcing it to "No" (same as using the **-L** flag on the **uniplex** command line).

Variable	Example Setting	Description
UXENVIRONMENT	UXENVIRONMENT=ncd	If set to the name (not full path) of a server file, **uxspawn** uses it to load the relevant resource file, looking in the user's local *UAP/XW/servers* directory for the file before searching in the Uniplex nodal or network layers.

Chapter 3
Configuring Menus

THIS PAGE INTENTIONALLY LEFT BLANK

The Menu System

Uniplex provides a standard menu driven interface for all aspects of the software. You can modify existing menus, add new menus and integrate external applications into the Uniplex environment. You can use the flexibility of the menu system to:

- Change the wording or layout of menus to suit the users at your site. For example, add your department name to the main menu.
- Create menus for particular users or groups of users, containing only the options they require. For example, create a menu for your manager that only provides options for Electronic Mail and Time Manager.
- Restrict access to particular Uniplex applications or tools. For example, remove the System Admin. option from the main menu.
- Modify options to invoke applications differently. For example, invoke the Spreadsheet with a file selected.
- Create or modify options to execute external applications or operating system commands. For example, provide an option on the System Administration menu that lists active processes on the system.

The menu system has a hierarchical structure. The options on each menu do one of the following:

- Provide access to another menu. For example the Database Management option on the Database Forms menu displays a menu.
- Leave the menu, and return to the menu from which you accessed it. For example, if you select Quit from the Spreadsheet menu, Uniplex returns you to the Main menu.
- Invoke an application. For example, the Edit a Document option on the Word Processor menu invokes the Word Processor.
- Execute an operating system command.
- Display a help menu or a help screen.

The Menu File

The menu file, *UAP/uniplex.menu*, contains each menu used by Uniplex. It is an ASCII file that you can edit using the Word Processor. You configure this file to tailor the operation of the menu system. When a user invokes Uniplex, the menu system reads this file.

The Menu File Format

The menu file is divided into sections, each section defines a different menu. Each section includes a pictorial representation of the menu, and specifies the actions Uniplex performs when the menu options are selected. Each menu section definition contains the following:

- The menu name

  This is used by other Uniplex system files to identify the menu. You must specify the name using the correct syntax for Uniplex to be able to find and display the menu.

- The menu title

  This is displayed at the top of the menu.

- The menu ruler

  This controls the layout of the menu. You must follow the ruler when laying out the text for the menu for Uniplex to be able to display and execute the menu correctly.

- The menu text layout

  The words and numbers that make up the menu. These must be formatted according to the menu ruler. See above.

- Menu actions

  The actions that are invoked by each of the options.

- End of section marker))

For example:

```
#UFORM.MENU
D A T A B A S E   M A N A G E M E N T
--------1-------------------------------------2---------------------R
        "TASKS                                "UTILITIES

        1 - Inquire On Records                P - Printing
        2 - Amend/Create Records              T - List Tables
        3 - Select Customized Forms           L - List Forms
        4 - Report Writer

        5 - Database Query (USQL)             S - Select Database

        6 - Database/Table Administration     H - Help
        7 - Build Customized Forms            Q - Quit
))
1 b
2 a
3 V
4 g
5 * 'exec usql -c',,,C
6 > #UFORM.DBMENU
7 > #UFORM.FMMENU
P > #PRINT2
L S
T H
S D
H > #HELPUFORM
Q <
))
```

The following sections explain each of the parts of a menu section.

The Menu Name

You use the *menu_name* to identify the menu section to Uniplex. The menu first displayed when you access an application is known as the *startup* menu. The names for these menus are fixed, and are given in the table that follows. You can create menus with other names as described in this chapter. Follow these syntax rules when naming a menu:

- Precede the name with #.
- Do not use more than 12 characters in the name.

Reserved Menu Names

Module/function	Program	Startup menu name
Main menu/Word Processor	uniplex	#SYSMENU
Word Processor/ integrate (ESC yi)	uniplex	#EDITMENU
Database Forms	uform	#UFORM.MENU
Report Writer menu manager	urmenu	#URMENU.MENU
Time Manager	usdiary	#DIARY.MENU
Advanced Graphics	uchart	#BGU.MAIN #BGU.FORM1 ... #BGU.FORM5
Electronic Mail	usmail	#UMAIL.MENU
Formfill	ufill	#UFILL.MENU
Screen Builder	ufill	#USCR.MENU

The Menu Title

The line following the menu name is the menu title. When you invoke the menu, Uniplex automatically displays this line centered at the top of the screen. It effects the line in the default effect, normally reverse video (or whatever effect is defined by the paint string in the Tcap entry for the terminal). The menu title should not exceed 40 screen columns in length.

The Menu Ruler

The following line, after the menu title, is the ruler. The ruler can contain the following characters:

n	Where *n* is a number, in the range 1 to 3, which indicates the left edge of the *n*th column of the ruler.
R	The right-hand edge of the menu box.
. ¦ - ¦ _	You can pad the ruler to the required length with periods (.), hyphens (-) or underscores (_). Remember that this ruler maps to the screen; therefore, do not create it more than 70 characters wide.

For example:

```
---1-----------------2-------------------3------------------R
```

You must position the letters or numbers for the menu options exactly underneath the column numbers on the ruler. You can have a maximum of three columns on a menu. See the next section The Text Layout.

The Text Layout

You use the text layout to produce a pictorial representation of the menu. This representation must be laid out in columns aligned under the column numbers used in the ruler. The text layout can contain:

Display-only text	Text to display on the menu at runtime, that you do not want interpreted. You enter display-only text with a leading double quote. For example: `"Printing`

Options	The words and numbers that make up the options that can be selected by the user. For example:
	
	`1 - Word Processor`
	`2 - Spreadsheet`
	
	See Selecting Options below.

Blank Lines	To be displayed on the menu at runtime.
))	Indicates the end of the text layout section.

Selecting Options

You can enter options into the text layout of a menu using either of the following syntax methods:

X - option

X- option

Where:

X	is the character the user presses to select the option
option	is the name of the option

If there is a space between *X* and **-**, the user does not have to press RETURN after highlighting the option. If there is no space between *X* and **-**, the user must press RETURN. For example:

```
1 - Word Processor
```

The user selects the option using pick and point, or by pressing 1.

```
1- Word Processor
```

The user selects the option using pick and point, or pressing 1, and pressing RETURN.

Menu Actions

You enter menu actions into a menu section using the syntax:

X action

where *X* is any ASCII character (no case differentiation) which triggers the menu option, and *action* is one of the following types:

- Generic Menu Actions: These actions are available throughout Uniplex.
- Application Specific Menu Actions: These actions are only appropriate to individual applications.

See the following sections for details of these.

Generic Menu Actions

The following table shows the available menu actions:

Action	Description	Example
> #*menu_name*	Display the named menu (#*menu_name*) from the menu file.	> #DOCPREP
<	Quit from the current menu, return to the previous menu in the hierarchy.	<
<<	Quit from the current application. If this is the only active Uniplex application, then the user will also be quit from Uniplex, back to the operating system.	<<

Action	Description	Example
***** *command*	Execute an operating system *command* through the shell. See the section Execute Operating System Commands.	*'ls -l *.x',,P,C
! *command*	Execute an operating system *command* directly. See the section Execute Operating System Commands.	! usql,,,C
?	Display the default help text.	?
? #*help_menu*	Display the named #*help_menu*.	? #HELP
? *file topic*	Display the named *topic* from the named help *file*. If file does not start with a '/', then it is assumed to be a name relative to a directory one level below the central UAP directory.	? ../wp/wphelp CUT

Execute Operating System Commands

You can use two forms of menu action to execute operating system commands from a menu:

o Execute the command directly using !

You use this form when you do not require the operating system to expand the command before execution. This form is more efficient because a new process is not spawned to parse the arguments.

Use the syntax:

! '*command***'[,['***message***'][,[P][,[C]]]]**

You must specify *command*, all other arguments are optional. If *command* is only one word, then it need not be quoted. See the table on the following page for details of this syntax.

o Execute a shell command using *

You use this form of the command when you require the operating system to expand the command before execution. For example, you can include shell expansion meta characters like *.

The syntax is identical to that used with "!".

Operating system command syntax:

Argument	Description	Example
'*command*'	Is the operating system command to execute.	`! 'ls -l'`
'*message*'	Is a message you want displayed at runtime. You can use this to provide additional information about the command, or to prompt the user. *Note: A single quote in either the command or message can be escaped with a backslash. For example:* *! 'adminpgm \'<OTHER>\','Enter administrator\'s password'* *You can specify a message only when you include the command keyword <FILE> or <OTHER> in your command. See the next section, Command Keywords.*	`! 'uniplex <FILE>', 'Enter filename:',,`
P	Causes Uniplex to always prompt the user for RETURN after executing the command and before returning to the menu. For instance, when you want the user to see the results of executing a command (or any error messages it produces) before returning to the menu. *Note: If the executed command returns a non-0 exit code, then Uniplex will always pause for a return, regardless of whether the menu statement had a 'P'.*	`! usql,,P,`
C	Causes Uniplex to clear the screen, from just below the title line, before executing the command.	`! 'umail -m',,,C`

Include a comma after each argument. The position of each argument is fixed. When not specifying all possible arguments, use commas to indicate the position of an argument. For example, if you want to specify C, but not P or a message:

```
! 'umail -m',,,C
```

You can include keywords within *command* to specify additional arguments to the command. The following section explains these keywords.

Command Keywords

You can include the following keywords as part of the command:

Keyword	Summary Description
<FILE>	Displays a prompt requesting a filename. Uniplex substitutes the user's response into the command line at this position and then executes the command. Uniplex ensures that the entered name is an existing file. The user can also press **down arrow** in response to the filename request to use the File Manager to select a file. Using syntax supported in earlier releases, **<FILE>** can also be written as **<FILE=***XXYYY***>** but all characters from and included in the '=' are now ignored.
<OTHER>	Displays a prompt at the top of the screen requesting a string. Uniplex substitutes the user's response into the command line at this position and then executes it.
<PASTE>	Inserts the name of the default cut and paste clipboard (0) in the command.

Multiple Menu Actions

You can configure the menu system to execute multiple menu actions from a single trigger, by linking them with the <BRK> keyword. For example:

```
1 ! 'cp stdfile wkfile' <BRK> ! 'uniplex wkfile'
```

This example copies stdfile to wkfile and then invokes the Word Processor with wkfile.

Note: Any trailing spaces at the end of a menu action are ignored.

Examples of Menu Actions

```
! usql,,,C
```

Invokes the Database Query module, usql. You use commas to indicate null arguments in the command. C indicates that Uniplex will clear the screen from just below the title line before executing the command.

```
* '<OTHER>','Enter shell command to execute :',P,C
```

<OTHER> prompts the user for a string. The message text 'Enter shell command to execute :' informs the user that the string should be an operating system command which Uniplex will execute. P indicates that Uniplex will prompt for RETURN after the command has been executed, before redisplaying the menu. C indicates that Uniplex will clear the screen from just below the title line prior to executing the command.

Application Specific Menu Actions

The menu system is a generic facility built into several Uniplex application programs. In addition to the generic actions, each application supports *application-specific menu actions*.

You enter application-specific menu actions using the syntax:

X option [filename¦directoryname]

where *X* is the character which triggers the menu option, *option* is a single character, *filename* is the name of a specific file (for use with Word Processor only), and *directoryname* is the name of the directory to change to (for use with Word Processor only).

The following tables list the application-specific menu actions.

Word Processor (uniplex)

Character	Option
C	Create new document
E	Edit a document
D	Select new folder
L	Call the File Manager
K	)
R	) These are reserved (had significance in earlier releases, they now generate a warning
B	) if selected).
F	)

Where the action allows an option argument (for example, 'E' in Uniplex to edit a file) then the following characters are treated specially in the argument:

Character	Meaning
~ (tilde)	The use of a leading tilde represents the $HOME variable.
$ (dollar)	$NAME is replaced by the contents, if any, of the environment variable NAME. For example: X E '$Utemp/.workfile'

Time Manager (usdiary)

Character	Option
H	At-a-Glance
I	Monthly
J	2-Weekly
U	Planning
B	Add new event
A	Inquire/Amend
D	Add alarm
E	Delete alarm
S	Print alarms
N	Amend rights
R	List rights
M	Make calendar
G	Select calendar
K	Delete calendar
C	Print calendar
F	Calls the pop-up calendar
T	Amend worktime
L	Search calendars
V	Diary Search
W	Daily/Weekly
X	Monthly
Y	Conference
Z	Block booking

Presentation Graphics

Character	Option
a	Create a new graph
b	Select existing graph
c	Data entry form
d	Text entry form
e	Reset all
f	Save current graph
h	View graph - full screen
i	Switch to split screen mode
k	View graph, split screen if possible, otherwise full screen mode
l	Use split screen softkey line
n	Clear graph (if in split screen mode)
r	Include/exclude data series
o	Graph Type Selection form
s	Include/exclude groups
t	List graphs
v	)
w	) These are reserved (had significance in earlier releases, they now generate a warning
x	) if selected).
y	)
z	)
0	Switch Options
1	Graph Text Attributes
3	Axis scaling form
4	Explode pie segments
5	Re-order groups
6	Re-order data series
9	Shade line marker attributes
A	Calls the Switch Options form
I	Sets split screen to OFF
Z	Calls the Display Format menu

Report Writer (urmenu)

Character	Option
t	Create report
q	Edit report
v	Run customized report
p	Delete a report
r	Copy a report
o	Rename a report

Database Forms (uform)

Character	Option
a	Create/Amend records
b	Inquire on records
c	Run customized form directly
e	Select table directly, in create/amend mode
f	Select table directly, in inquire mode
V	Customized Forms
g	Run reports
D	Select database
H	List tables
S	List forms
A	Create database
B	Delete database
E	Rename database
C	List databases
X	Create standard form
T	Create new form
Q	Edit form
N	Build form
U	View build log
P	Delete form
R	Copy form
O	Rename form
L	View table schema
F	Create table
W	Amend table schema
G	Delete table
J	Rename table
K	Copy table

Electronic Mail (usmail)

Character	Option
m	Send memo
s	Send letter
l	Check mail
i	Read incoming mail
o	Read outgoing mail
a	Read archived mail
B	Reserved for future use
R	Set auto-reply
S	Unset auto-reply
F	Set auto-forward
G	Unset auto-forward
M	Create mailbox
N	Delete mailbox

Formfill (ufill)

Character	Option
V	Enter data
T	Create template
Q	Edit template
N	Build template (Screen Builder mode)
U	View build log
A	Generate template (Formfill mode)
C	Create a document
I	Install a document
B	Edit a document
E	Create a report
D	Edit a report
P	Delete files
R	Copy files
O	Rename files
S	List files

Chapter 4
Configuring Softkeys, Ring Menus and Popups

THIS PAGE INTENTIONALLY LEFT BLANK

Overview

This section explains how to configure softkeys, popups, and ring menus using information held in "softkeys" (.fn) files. You configure them in the same way, using the same syntax and the same variable keywords.

You use softkeys, popups, and ring menus to provide the following:

- An easy-to-use interface to all Uniplex modules

  Softkeys enable you to tailor the operation of Uniplex for novice or infrequent users of the software. Uniplex softkeys display command menus, showing the commands to press and provide help and message text. They let naive users avoid using escape and control command sequences.

- A macro-programming language

  You can build macros of commands to execute from a single key press. This enables you to tailor softkey menus to suit particular applications. Experienced Uniplex configurers can build softkey menu systems to execute any Uniplex command and to call external operating system utilities.

- Control over product integration

  You use a series of popup menus to control the integration between different Uniplex modules. These include the Desk and Utilities popups. These menus offer integration options throughout Uniplex.

Terminology

Softkey Menu

A softkey menu is a single line menu, displayed along the bottom of the screen. For example:

```
F9>Desk F10>Help F11=Softkey Home F12>Utilities Desk  F7< F8>More
```

In addition to using the function keys at the top of the keyboard, you can use the ESC key in conjunction with the number keys to select options from the softkey menus. This is useful if a keyboard does not have function keys, or if the function keys have not been configured.

Note: On keyboards that do not have an ESC key, you can specify an alternative key. See the chapter Configuring Terminals in the Uniplex Device Configuration Guide for details.

Popup Menu

Popup menus appear on a portion of the screen, temporarily overwriting part of the existing display. Popups are similar to softkey menus, except the options are listed vertically instead of horizontally. For more details about the popup menus, see the Uniplex II Plus User Guide.

Ring Menu

The Word Processor and Spreadsheet command menus use a horizontal pick and point list of options. These are generally referred to as Ring menus.

All three types of menu are defined using similar syntax within a softkey file.

Uniplex Conventions

Uniplex uses the following convention to distinguish between options which execute an action and options which display another menu level.

A greater than sign (>) after a function key number indicates that pressing the function key will cause another menu to be displayed.

For example:

```
F5>Effect
```

An equals sign (=) after a function key number indicates that the option will execute an action directly. For example:

```
F1=Save&Exit
```

Compiling the Softkey Files

Uniplex uses compiled versions of the softkey files in order to improve system performance. If you make changes to the .fn files, you must recompile them afterwards in order for the changes to take effect.

When you compile any file with a suffix of .fn, Uniplex converts it to a file with a suffix of .sk. It is this file that Uniplex references when it runs the binary.

To compile a softkey file, follow these steps:

1 Access an operating system prompt. It is recommended that you compile softkey files from within Uniplex in order that the correct environment variables are operating.

2 At the prompt enter:

skcomp *filename*.**fn**

Note: If you are not in the correct directory, you must specify the pathname of the source softkey file.

For example, if you enter:

```
skcomp wpring.fn
```

Uniplex compiles the information into the file wpring.sk.

If you enter:

```
skcomp /usr/UAP/wp/wpring.fn
```

Uniplex compiles the information into the file /usr/UAP/wp/wpring.sk.

For the changes to take effect, you must exit from the binary and re-access it.

File Format and Layout

Most modules have their own softkey file. The softkey file is identified by the suffix .fn. For example, for the spreadsheet (with the issi interface) this file is called issi.fn; for the Word Processor this file is called wpring.fn.

Each .fn file is made up of the following sections:

- Messages

  Contains a list of messages which are displayed as user prompts for that module. This section is mandatory in all .fn files.

- Softkey Menus

  Defines the softkey menus used by that module. A fixed main softkey menu is mandatory in all .fn files.

- Popups

  Defines the popup menus used by that module.

- Ring Menus

  Defines any command menus used by the module.

Softkey files may also reference files for inclusion (include files). These define the files that you want to be included with the .fn file.

The sections that follow include details on all of these.

Note: Softkey compiled files (.sk) may only contain a maximum of 65,535 characters. The compiler will report if you exceed this limit.

Syntax Used

The following subsections describe the syntax to use for the various softkey and popup sections.

Note that each section starts with **#***section_name* and the following names are reserved:

#MESSAGES
#DEFAULTS
#COMMANDS

Message Section Syntax

You define any messages referenced in softkey commands in the message section. You can define messages for softkey or popup actions which call an external process, or which specify a message should be displayed (D or X commands).

The following is a summary of the syntax to use for the message section:

Syntax	Explanation
#MESSAGES	Start of the messages section.
1 *string*	Where *string* is message one.
2 *string*	Where *string* is message two.
3 *string*	Where *string* is message three.
.	
.	
.	
100 *string*	Where *string* is message 100.
))	End of section marker.

Messages 100-150 are reserved for use by Uniplex. For example, messages 100-110 are used for Uniplex Desk.

Note: Unlike other sections, you can specify multiple #MESSAGES sections. This facilitates logical split-up of messages across #include'd files.

Softkey Section Syntax

The following table shows the syntax to use for a softkey section.

Syntax	Summary Description
#*section_name* **F***1*=*option1* **F***2*>*option2* **F***3*=*option3*...	Start of the softkey section. Menu of options as it will be displayed on the screen. You can use any text you require up to a recommended limit of 77 characters. Enclose the menu in single quotes, if the first character of the menu is a number or a space.
1=action1 *2=action2* *3=action3* . . .	Corresponding action for each of the options where *n* is the number of a function key on the softkey menu.
))	End of section marker.

Popup Section Syntax

The following table shows the syntax to use for a popup section.

Syntax	Summary Description
#*section_name* **' '**	The name Uniplex uses to identify the popup. Two single quotes immediately beneath the section name or a blank line indicate a popup rather than a softkey. You can use blank lines elsewhere in the menu to improve readability.
"*menu title*	The first non-blank line beneath the section name is taken to be the title of the popup. Precede the *menu title* with a double quote.
"*text*	You can include *text* in your popup that is not part of a menu option. Include text to provide information on the function and use of the popup. For example, prompts. Precede *text* with a double quote.

Syntax	Summary Description
n - option1 *n - option2* *n - option3* . . .	The list of options on the menu, where *n* is an option number. See Selecting Options below.
))	End of screen section.
n=action1 *n=action2* *n=action3* . . .	Corresponding action for each of the options where *n* is the number of one of the menu options. You define popup actions in the same way as you define softkey actions. See the following sections for more details.
))	End of section marker.

Ring Menu Section Syntax

The following table shows the syntax to use for a ring menu section.

Syntax	Summary Description
##*name*	The menu name.
command1 command2 command3 ...	List of selectable options.
Text shown while command1 is highlighted *Text shown while command2 is highlighted* . . .	The appropriate text is shown when a ring menu command is highlighted. There is one line of text for each command listed.
))	End of section marker.
1=*action for command1* **2**=*action for command2* . . .	Specifies the corresponding actions for each of the options.
))	End of section marker.

Selecting Options

Enter the options a user can select following the standard Uniplex menu option syntax:

n- text description

n - text description

Where *n* is the numeric character(s) the user presses to select the option, and *text description* is the name of the option. If there is a space after *n*, the user does not have to press RETURN after pressing *n*, or highlighting the option. If there is no space after *n*, the user must press RETURN to select the option.

For example:

```
1 - Word Processor
```

The user selects the option using pick and point, or by pressing 1.

```
1- Word Processor
```

The user selects the option using pick and point, or pressing 1, and pressing RETURN.

Include no space where there is conflict between the digit or character you press to select the option. For example:

```
1- Word Processor
10- Spreadsheet
```

The Default Softkey Main Menu

All .fn files must contain a fixed main softkey menu. The names of these menus differ from module function to function, as defined in the following table. These menu names must not be changed as they are called by their corresponding binary.

Fixed Softkey and Ring Menu Names (By Module)

Module/function	Program	Fixed Menu Names
Word Processor/main-menus	uniplex	#PROMPT
Word Processor/editor	uniplex	#EDIT ##COMMANDS
Spreadsheet, ISSI mode	ucalc	##ISSI ##GRAPH
Spreadsheet, UCALC mode	ucalc	#PROMPT
Database Forms	uform	#EDIT #PICK #FMMENU #FMEDIT #TBAMEND #TBMAKE
Electronic Mail	usmail	#MINIEDITOR #PASSVER #GETPASS #PICKCOM #PICK
Time Manager	usdiary	#EXTNFILL #FORMFILL #PICK #EDIT
Card Index	ucard	#DPICK #PICK #FINDFORM #SLABELS #FORMFILL #FORMEXAM
Report Writer menu system	urmenu	#EDIT #PICK

Module/function	Program	Fixed Menu Names
Presentation Graphics	uchart	#ATTFORM #ATTFORM2 #PROMPT3 #DATENT #PROMPT4 #SAVCUR #SELEX #ATTMEN #PROMPT
Formfill/Screen builder	ufill	#PICK #FMMENU #FMEDIT #TBAMEND #EDIT
Various popup functions	popup	#CLIP_PROMPT #CLIP_VIEW #PROMPT #MPROMPT #FPROMPT ##PTOSRNG1 ##PTOSRNG2 ##PTOSFND1 #PTOSFND2

Softkey Actions

The syntax for an action is:

n=*action*[:*action* ...]

where *n* is the numeric character(s) which selects the menu option, and *action* is the command that will be executed. Any action can contain sub-actions, separated by colons. The maximum length of any action is 255 characters (lines may be continued with a backslash).

The following table shows the available softkey actions:

Action	Description	Example
F*nn*	Enter the number of a uniplex.cmd command which performs the action you require. Refer to the chapter Configuring Command Keystrokes (uniplex.cmd) for details of the command numbers to use.	2=F62 (the quit command)
G*nn*	Enter the number of a uniplex.cmd generic command which begins to perform the action you require. Refer to the chapter Configuring Command Keystrokes (uniplex.cmd) for details of the command numbers to use.	2=G05 (the generic root for the cut command)
C(F*nn*,**S***nn)*	Use immediately before a menu call to temporarily convert the named command to perform the same action as the named softkey. This is used to trap direct commands and convert them into softkeys. The example shown traps the RETURN key and makes it perform the same action as softkey 1 (Save and Exit).	1=F61:C(F11,S1) :M(#EDIT)
M()	Return to the current main softkey menu.	1='Memo':M()
M(#*name***)**	Transfer to the named softkey menu section.	1='Memo':M(#EDIT)

Action	Description	Example
M(#*name***,M)**	Transfer to the named softkey menu section and make it the current main menu.	1='Memo':M(#EDIT,M)
ˆ()	If this is the only action on a menu selection, it causes all other menu selections to be "held" until this function key is pressed. Non-held menus always return to the main menu after each action.	8=ˆ()
	If this is the last action on a multi-action line, it is the same as **M()**.	3=F23:ˆ()
D*n*	Display message number *n* from the #MESSAGES section of the softkey file.	1=G05:D4
D(*message***)**	Display the message in parentheses as user information towards the top of the screen.	1=G05:D(Mark area)
W(*char***)**	Wait - keep displaying the message until the next character specified by the syntax: **\r** - carriage return **\n** - same as \r **\t** - tab	1=D(Mark area):W(\r)
W(F*nn***¦F***nn***)**	Keep displaying the message until one of the commands in brackets is pressed.	1=D(Busy):W(F13¦F16)
W(*string***)**	Keep displaying the message until the user types the string.	1=D4:W(Memo)
T(F*nn***)**	Acts like wait (**W**) except that the command is trapped and not passed back to the application module.	1=D(Enter formula):T(F11)
I(F*nn***)**	Ignore command F*nn* if it is input by the user.	1=D(Enter formula):I(F11)
I()	Cancel the previous ignore command.	

Action	Description	Example
H(*#section***)** **H(***section***)**	Display the named section from the current applications help file. *Note: The section name "formhelp" is a reserved word in UFORM and UFILL for a form's :HELP section.*	H=H(#GEN_REC)
X()	Call an external process. See the next section for more details.	
'*string*'	Return the characters between the quotes as if they had been entered at the keyboard. Within the quotes the following have special meaning: **\r** - RETURN character **\t** - TAB character	1='Memo'

Call an External Process

You use the external process call to provide integration between the current module and other Uniplex modules or external operating system commands. The syntax for the external call is:

X([*rows***],[***cols***],[***env_var***],***options***,***program***[,***msg_number***])**

The parameters are position dependent. The parameters, *msg_number*, *rows*, *cols* and *env_var* may be omitted, though their separating commas are compulsory. The following table explains each parameter:

Parameter	Description
rows	If the program can run in a window of less than full screen size, you use this parameter to set the number of rows in the window. Defaults to the value 0. Special values: **0** - full screen height. ***** - current window height.

Parameter	Description
cols	If the program can run in a window of less than full screen size, you use this parameter to set the number of columns in the window. Defaults to the value 0. Special values: **0** - full screen width. ***** - current window width.
env_var	Some Uniplex binaries can operate in a smaller window than a full screen. The following variables specify the window size of the called program. These are designed to be used only by the following Uniplex programs:

Variable	Module	Program	Minimum Window Size
WINWP	Word Processor	uniplex	(10 rows 60 columns)
WINUCALC	Spreadsheet	ucalc	(10 rows 40 columns)
WINUFORM	Database Forms	uform	(12 rows 80 columns)
WINUCARD	Card Index	ucard	(16 rows)
WINUDIARY	Time Manager	udsiary	(12 rows 80 columns)
WINUFILL	Screen Builder	ufill	
WINDECIDE	Decision Maker	popup	(10 rows 33 columns)*
WINCALC	Calculator	popup	(13 rows 20 columns)*
WINCAL	Calendar	popup	(10 rows 24 columns)*
WINCLOCK	Clock	popup	(7 rows 38 columns)*
WINSKETCH	Sketchpad	popup	(10 rows 30 columns)
WINUCLIP	Show clipboards	popup	(13 rows)

Note: Sizes marked ***** *are mandatory.*

A twelve digit string defines the window position.

For example, a WINUCALC setting of `000015079023` causes the spreadsheet window to open at column `000`, row `015` and close at column `079`, row `023`.

options — These control the way the program is invoked and how any return values to the calling module are treated. The first option letter is the type of external call, and controls how any return values will be processed.

Parameter	Description
	The following list shows the available option types that can be used in the first letter of *options*: **S** Returns a single value from the standard output of the program. For example, returns a literal string. **L** Returns multiple values from the standard output of the program as a pick and point list. Returns a single value to the calling module when an option is selected from the list. **D** Displays multiple values from the standard output of the program as a pick and point list. No value is returned (display only). **C** Places all returned values in a cut and paste buffer (clipboard 0). **I** Independent program. Standard output is sent to the terminal. No value is returned to the calling program. This option should always be specified when invoking a Uniplex program that calls sub-programs. **N** No value is returned. Standard output from the process is discarded. You can use further qualifying options to control the appearance of the screen during the invocation of the external process. Qualifying options should follow the option types: **B** Box. Draw a box around the window, while the program is loading. **O** Overlay. When using the B option, the window environment variable is set by default to be inside the box. Some external processes such as Uniplex popups display their own border box. You use the overlay option to ensure that the window size includes the box. **R** Redraw. Redraw the whole display after running the external program.

Parameter	Description
program	The name of the program to be run. This is passed to the shell, so it can include a series of shell commands separated by semi-colons. If the command contains commas then the whole string should be enclosed in single quotes. If the string contains the **)** or **(** character, it must be escaped with a ****. For example, \). *Note: There is no way to specify a single quote within the string.* If the last program defined in *program* is going to run for a while (for instance, one of the Uniplex modules), invoke it using the **exec** command. This avoids the calling shell waiting until the program is complete.
msg_number	An optional message number, from the #MESSAGES section of the softkey file.

Softkey and Popup Examples

Example 1

The following lines will put a standard header in your file:

```
4=’.HE4’:F11:’Page #’:F33:F21:F65:’!udate "+%a %d %h %y"\r’:\
F11:F11:F11:F11:M(#EDIT)
```

This automatically enters a 4 line header containing the current date and centered ’Page #’ for page numbering.

The !udate command is followed by the arguments "+%a %d %h %y" which return the day, date, month and year.

\r specifies an automatic carriage return.

Example 2

```
1=F65:’/usr/john/memo\r’:F14:F14:F14:F14:F14:F16:F16:F65:\
’!udate "+%a %d %h %y"\r’:F15:D14:W(\r):F11:F11:F11:F11:\
F11:M(#EDIT)
```

This produces a standard memo by merging in and adding to a file containing a 9-line memo heading as follows:

```
FROM:                   John

TO:                     All

DATE:                   Thu 17 Oct 88

SUBJECT:

_____________________________________________________________
```

F14 and F16 are used to position the cursor at the correct point on the ’DATE’ line.

The **udate** command with its specified flags is used to enter the current date in the form:

Thu 17 Oct 88

F15 is used to position the cursor at the correct point on the ’SUBJECT’ line.

```
D14:W(\r):
```

Displays a customized message:

Please Enter Subject, and press RETURN to end :

Return will end, the remaining F11s position the cursor below the separator line ready for the user to type the memo.

Example 3:

The following example shows a pop-up menu:

```
#DESK2
''
"Uniplex UTIL: Page 1 of 2
"
1 = List Documents
2 = View Clipboards
3 = Clock
4 = Decision Maker
5 = Calculator
6 = Phone & Address List
7 = Card Index
8 = Personal Organizer
"
9 > Next Page
0 > Previous Page
"
"ESC Q to Quit
))
1=X(,,,L,ls)
2=X(13,*,WINUCLIP,NB,exec popup clipboard)
3=X(7,38,WINCLOCK,NOB,exec popup clock)
4=X(10,33,WINDECIDE,NOB,exec popup decide)
5=X(13,20,WINCALC,SOB,exec popup calc)
6=X(,,WINUCARD,IOR,exec ucard -t1 $Uniplex/ucard/ucards/addresses)
7=X(,,WINUCARD,IOR,CDIR=$HOME/UAP;((cd $CDIR\)\
>/dev/null 2>&1\) ¦¦ mkdir $CDIR;exec ucard -c $CDIR/card-index)
8=M(#ORGANISER)
9=M(#DESK2.2)
0=M(#DESK1.2)
))
```

```
1=X(,,,L,ls)
```

Option 1 executes an external command, the operating system command ls. The window size is not defined because an operating system command is to be executed and the popup will therefore calculate the window size required for the output. For the same reason, no environment variable is required. A L(ist) of values is returned into the popup.

```
2=X(13,*,WINUCLIP,NB,exec popup clipboard)
```

Option 2 executes the external command **popup clipboard** in a window 13 rows by the current number of columns. The external command uses the environment variable WINUCLIP. (N)o value is returned from the command. It will be (B)oxed.

```
3=X(7,38,WINCLOCK,NOB,exec popup clock)
```

Option 3 executes **popup clock** in a window 7 rows by 38 columns. The external command will use an environment variable WINCLOCK. (N)o value is returned from the command which (O)verlays an existing process and is (B)oxed.

```
4=X(10,33,WINDECIDE,NOB,exec popup decide)
```

Option 4 executes **popup** with the argument decide (decision maker). (N)o value is returned from the command which (O)verlays an existing process and is (B)oxed.

```
5=X(13,20,WINCALC,SOB,exec popup calc)
```

Option 5 executes **popup calc** (calculator) in a window 13 rows by 20 columns. A (S)ingle value can be returned from the command, which (O)verlays a new process and is (B)oxed.

```
6=X(,,WINUCARD,IOR,exec ucard -t1 $Uniplex/ucard/ucards/addresses)
```

Option 6 executes ucard with the -t flag.

```
7=X(,,WINUCARD,IOR,CDIR=$HOME/UAP;((cd $CDIR\)\
>/dev/null 2>&1\) ¦¦ mkdir $CDIR;exec ucard -c $CDIR/card-index)
```

Option 7 executes a multi-command shell script in a full screen window. The shell script ends by loading ucard with the -c flag.

```
8=M(#ORGANISER)
```

Option 8 displays the menu named in parentheses, in this case #ORGANISER.

```
9=M(#DESK2.2)
```

Option 9 displays the menu named in parentheses, in this case #DESK2.2.

```
0=M(#DESK1.2)
```

Option 0 displays the menu named in parentheses, in this case #DESK1.2.

```
))
```

Indicates the end of the section.

Example 4

The following example shows how you can set up a series of softkey menus to allow you to automatically sort a file.

```
#EDIT
1: F1=Save&Exit F2=Menu F3=Last Menu F4=Quit F5>Draw F6=Sort  F7>
1=F061
2=F159
3=F160
4=F062
5=M(#SK.DRAW)
6=M(#SORT)
7=M(#EDIT2,M)
))
```

The above example changes the standard edit menu so that when a user presses F6, Uniplex displays the Sort softkey menu.

Define the Sort softkey menu as follows:

```
#SORT
Move to the first line to be sorted and press F1   F8>More
1=F21:F52:M(#SORTIT)
8=M(#EDIT,M)
))
```

```
1=F21:F52:M(#SORTIT)
```

Pressing F1 moves to the beginning of the current text line, and marks the character as the beginning of a block. Uniplex then displays the SORTIT menu.

```
8=M(#EDIT,M)
```

Returns to the Edit Menu.

Define the SORTIT menu as follows:

```
#SORTIT
Move to the last line to be sorted and press F1   F7<  F8>More
1=F21:X(,,,N,rm -f .SORTFILE):F72:'.SORTFILE':F11:F55:X(,,,C,sort .SORTFILE)
7=M(#SORT)
8=M(#EDIT,M)
))
```

```
1=F21:X(,,,N,rm -f .SORTFILE):F72:'.SORTFILE':F11:F55:X(,,,C,sort .SORTFILE)
```

Moves the cursor to the beginning of the next text line and calls an external program which deletes any files called .SORTFILE. Then writes the area the user has selected to a file called .SORTFILE. Then calls another external process, (the sort utility) and places the results of the sort in a file called .SORTFILE.

```
7=M(#SORT)
```

Returns the user to the SORT menu.

```
8=M(#EDIT,M)
```

Returns the user to the EDIT menu.

Other Sections and Syntax in a Softkey File

#DEFAULTS Section

The #DEFAULTS section can be included in a softkey file to define softkey actions that are common to more than one menu.

Syntax	Summary Description
#DEFAULTS	Start of section
' '	Dummy or real blank line (required)
n=action *n=action* . .	Default action for softkey *n*
))	End of section marker

For example, the following #DEFAULTS section could be used to ensure that, unless specifically included in a softkey menu, the F10 key should call the #HELP menu.

```
#DEFAULTS
''
10=M(#HELP)
))
```

Comments

Comments can be interspersed anywhere in a softkey file by preceding them with an asterisk. Blank lines outside of sections are ignored, and can be used to improve readability.

For example:

```
* This is a line that is just a comment

1=F123 * This is a comment at the end of an action line
```

#include Command

This command allows nesting of softkey files. Its syntax is:

#include *file*

where *file* is the name of another softkey file. If *file* is not found directly, the softkey compiler, skcomp, attempts to use a file of the same name, as follows:

1. Use **$HOME/UAP/general/***file*, if it exists, otherwise
2. use **$Uredirect/UAP/general/***file*, if it exists, otherwise
3. abort - *file* not found.

All the standard softkey files contain the line:

```
#include ../general/general.fn
```

to pick up the common softkey functions, such as DESK.

Use of the #include command can result in more than one of the same section being part of a softkey file. Such duplicates are treated as follows:

#MESSAGES	Multiple sections allowed. If multiple messages exist with the same number, then the *last* one is used.
#COMMANDS #DEFAULTS #section_name	Multiple sections are not allowed. If multiples exist, the *first* one is used and the rest are ignored.

#COMMANDS Section (Pseudo-commands)

The #COMMANDS section can be included in a softkey file to define softkey actions driven from "spare" command numbers in *uniplex.cmd* -thereby allowing the creation of "Pseudo" commands.

Syntax	Summary Description
#COMMANDS	Start of section
F*nnn=action*	Define action for the named pseudo-command
F*nnn=action*	
.	
.	
))	End of section marker

For instance, the following #COMMANDS section, which is used in the standard Uniplex configuration, defines the pseudo-command **F241**, which is associated in *uniplex.cmd* with the keystroke sequence ESC ESC $ (**F241=&-&-'$'**), as being an escape to shell operation.

```
#COMMANDS
F240=M(#DESK2)
F241=X(,,,IOR,echo;PS1="Uniplex ${PS1-\$ }";export PS1;exec sh)
F242=X(,,,IOR,eval ufill print "’’" ‘uprtcmd -g $Uidcode‘)
))
```

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5
Configuring System Parameters
(uniplex.sys)

THIS PAGE INTENTIONALLY LEFT BLANK

File Format and Layout

You use the system command file, *UAP/uniplex.sys*, to define the general system keywords for the entire product.

- You can define general system keywords such as the standard Uniplex date format.
- You can define specific word processing keywords, such as the default operating modes and preset rulers.

This file is organized using the Uniplex file format and layout:

#*section_name*
keyword
keyword
.
.
.
keyword
))

It contains the following two sections:

Section Name	Description
#SYSTEM	Sets general system keywords such as the Uniplex date format. It also contains specific word processor entries, such as the default operating mode.
#RULERS	Sets the 10 preset rulers used in the word processor.

The later sections give details of each keyword in the #SYSTEM and #RULERS section. You can also refer to the supplied uniplex.sys file for information.

Syntax for the #SYSTEM Section

You specify entries in #SYSTEM using the following syntax:

- *KEYWORD=string* for most *KEYWORDs*

  The *string* is made up of:

  - The decimal value of a character. For example: 166
  - A literal string in single quotes. For example: 'abc'

  One or more literal string may be entered, separated by hyphens (-).

  Note: Single quote (') or backslash (\) characters within a literal string must be prefixed with a backslash. For example: 'a\\b\'c' represents the 5 characters a\b'c.

- *KEYWORD=integer* for *KEYWORD*s setting an actual integer value.

  Note: Very few keywords use this form. Some, such as PROC, require an integer specified in a literal string (PROC='2'). Be very careful to follow the exact syntax specified in this section.

You can include comments by prefixing them with an asterisk (*). All lines outside a section are assumed to be comments.

Examples:

Decimal value character	`ALLOW=167-'{}'-201-233`
Literal string	`PAGE='60'`
Integer	`HLEN=6`
comment	`* Sets default page length`

Compiling uniplex.sys

After you have made changes to the uniplex.sys file, you need to compile it. To do this, use the ESC ESC $ escape sequence to display the operating system prompt, and then type in **syscomp -s**.

Running syscomp creates a binary version of your local or central uniplex.sys in the corresponding directory's system.comp file. If you already have a system.comp file, it will be overwritten. Before the changes take effect, you must quit and re-invoke Uniplex.

When you run syscomp, the preset Word processor rulers and most of the Word Processor working options are taken from the old system.comp rather than from the new uniplex.sys file. The options taken from system.comp are:

Hard returns (yes/no)
Column count (line/column/off)
Hyphenation (automatic/manual/off)
Automatic reformatting (yes/no)
Tab mode (yes/no)
Autosave (in number of characters)
Autosave (in minutes)
Command hints (yes/no)
Character to enter menus (character)
Cursor wrap (on/off)
Blank line to align columns (yes/no)
Generate style information (yes/no)

All other option information will be taken from the new uniplex.sys. (This includes mode options ACDFIOUW and $.) Note that insert mode is available to reset as a working option, but is automatically reset from uniplex.sys.

For example:

1 Assume you have used the Word Processor, with the working options popup to set Cursor Wrap mode to OFF. This will be set to OFF in your local system.comp file. If you then reconfigure uniplex.sys and set Cursor Wrap to ON (by including > in the MODE string), after compilation by syscomp Cursor Wrap will be OFF.

2 Assume you have set Clock on Menu Line (mode C) in your existing system.comp. If you then reconfigure uniplex.sys with it unset (no clock), after compilation, there will be no clock on the menu line.

System Keywords

Define 2SPACE and 3SPACE

You use 2SPACE and 3SPACE to define those characters, currently followed by at least one space which will be followed by two, or three trailing spaces when you reformat text in a Word Processor document.

For example:

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

This example defines the punctuation characters ! ? : ; . with two trailing spaces after reformatting.

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

This example defines the punctuation characters ! ? : ; to have 2 trailing spaces and the period to have 3 trailing spaces after reformatting.

Define the Word Processor Backup File - BACKUP

Each time a user edits a document using the Word Processor, and subsequently saves it, Uniplex makes a copy of the original. The BACKUP entry defines the name of this backup file. If you do not define the file, no backup is made.

In the BACKUP entry, the following items are expanded if they are the first characters in the string:

~ (tilde)	is expanded to the user's home directory (the contents of the $HOME variable) if it is the first character in the string..
^ (caret)	is expanded to the current filename.
$*name* or $(*name*)	(where: *name* is an environment variable), are expanded to the contents of the variable. The variable *name* can consist of any characters except space or slash (/).

For example:

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

Sets the backup to a file called wp.back.up in the user's home directory.

or:

```
BACKUP='$Utrash/$Uusername/^'
```

Sets the backup to *filename* (where *filename* is the name of the file currently being edited) in the users trashcan. Recovery can then take place using the trash management menu.

or:

```
BACKUP='~/wp.bak.^'
```

Sets the backup to wp.bak.*filename* in the user's home directory, where *filename* is the name of the file currently being edited.

Note: If the specified BACKUP file is already in use (for instance, if you have more than one Word Processing session active at a time, using multiple windows), Uniplex allocates and uses a unique backup file in the TEMP directory.

Define Checking for Messages - CHECKTIME/CHECKCHARS

CHECKTIME and CHECKCHARS define when and if Uniplex checks whether to display a boxed message, such as `You have new mail`.

When you specify the interval between checks, you must define both CHECKTIME and CHECKCHARS.

Note: If you define CHECKTIME and CHECKCHARS, see the section Enable UNIX tty Driver Timeouts - TTYTIMING later in this chapter, for details of enabling UNIX tty driver timeouts.

o CHECKTIME

  This specifies that message arrival is to be checked for if:

  - The specified number of seconds has elapsed, and:
  - The user has not pressed a key for the specified number of seconds.

The number of seconds specified must be in the range 0 to 12. If CHECKTIME is set to 0, no time-based checks are made (neither are time-based autosaves; see the later section, Enable the Word Processor Autosave - SAVETIME/SAVECHARS, for further details). If not set, CHECKTIME defaults to 15 seconds if TTYTIMING is 'N', or 12 seconds if it is 'Y'.

For example:

```
CHECKTIME=5
```

This checks for messages after 5 seconds if no key has been pressed. If the user does not leave a gap of 5 seconds between key presses, no time-based check will be made (nor will a time-based autosave).

- CHECKCHARS

  This specifies that message arrival is to be checked for if the user has made the specified number of keystrokes.

  The number of keystrokes defaults to 1000. If CHECKCHARS is set to 0, no keystroke-based checks are made.

  For example:

```
CHECKCHARS=500
```

  This checks for messages after 500 keystrokes.

Define the Date Flag - DATEFLAG

Uniplex users can enter a standard string into word processor documents to represent the date. Uniplex converts this string to the current date at print-time. For example:

```
DATEFLAG='$TODAY'
```

At print-time, Uniplex converts all occurrences of $TODAY to the current date.

Define the Date Format - DATEFMT

Uniplex recognizes dates entered by the user in the standard date format. This format is defined using the DATEFMT flag in #SYSTEM.

The date string must be enclosed in single quotes and is composed of the following day, month and year identifiers, which must be in the order month-day-year, day-month-year or year-month-day.

MM	Month
DD	Day
YY	Two digit year
YYYY	Four digit year

Characters such as / - . are used as separators. The two separators used in a date string need not be the same.

For example:

```
DATEFMT='MM/DD/YY'
```

Displays August 10th 1990 as 08/10/90.

```
DATEFMT='DD-MM-YY'
```

Displays August 10th 1990 as 10-08-90.

```
DATEFMT='YYYY/MM-DD'
```

Displays August 10th 1990 as 1990/08-10.

Four digit years are available for the menu header line, and for the following applications:

- Screen Builder
- Formfill
- Database
- Time Manager
- Card Index

Before a Uniplex application will display a four digit year, in addition to specifying YYYY in the DATEFMT string, you must set the relevant DATEMODE flag. For more details see the next section Defining the Date Mode. Uniplex uses a two digit year as the default.

Note: The default setting of the DATEFMT flag differs according to the language version of Uniplex you are using.

Define the Date Mode - DATEMODE

By setting DATEMODE you can specify which:

- o two digit years are interpreted as the century 1900-1999, and which as the range 2000-2098.
- o Uniplex applications can have a four digit year displayed if also set in DATEFMT.
- o day is the start of the week in Time Manager.

The following table explains each flag:

Flag	Explanation
C*nn*	Interprets all two digit years less than *nn* as 20*xx*. For example if C90 is entered, 4/1/90 is interpreted as April 1st 1990 but 4/1/89 is interpreted as April 1st 2089. *Note: When set, this flag affects every Uniplex application. It cannot be changed for individual applications. It has no effect on Database Query input which is passed to the database engine, see Chapter 8 for details. The default is set to C41. This means that two digit years are assumed to be in the range 1941-2040.*
D	Allow four digit years in Database Forms.
F	Allow four digit years in Screens and Formfill.
N	If TIMEFMT is a 12 hour format, display Noon as 12:00pm rather than 12:00am.
Q	Allow four digit years in Database Query. This controls the default output when the "format column" statement is not used.
T	Allow four digit years in Time Manager.
W*n*	Use day *n* as the start of the week when displaying calenders in Time Manager, where *n* must be an integer between 0 and 6, with 0 representing Sunday. The default is W0 (Sunday).
X	Allow four digit years in Card Index.

For example:

```
DATEMODE='C90FD'
```

This example sets the century break to be 90, and allows four character years in screens, formfills and database forms.

Define the Decimal Tab Alignment Character - DECTAB

The DECTAB flag defines which character is used as the decimal separator.

You use the hash character to create a decimal tab ruler. You enter numbers beneath this ruler and press tab to align the numbers automatically.

For example:

```
DECTAB='.'

L.......T.......#.......T.......T.......T.......T.......T.......T....R
                20.0
            10,000.00
               100.00
```

This example sets the separator character to a decimal point. Uniplex aligns the first number before the decimal point underneath the # on the current ruler.

If you set the separator character to a comma, Uniplex aligns the first number before the comma underneath the #.

Note: You will not normally change DECTAB without changing the FIELDSEPS string (see below).

Define Translated Dot Commands - DISPDOTS

You use DISPDOTS to translate the dot commands as stored in files into another representation to be displayed on screen. DISPDOTS commands are what you enter, and what you see when documents are displayed. DOTS commands are filed in documents. You are advised not to change DOTS commands unless you are prepared to compromise the portability of the document. Such commands are position dependent within the DISPDOTS flag.

For example:

```
DOTS='HE:HM:FO:FM:PL:PA:PN:SN:PM:SP:JY:JN:RE:ME:ST:SB:FN:GR:FT:IX:IC:FS:FE:NS:PT:UP'

DISPDOTS='KO:KR:FU:FR:SL:NS:SN:DR:SU:ZA:AF:EF:BE:MI:DA:DE:FN:GR:FT:IX:IV:FA:FE:NA:ZD:RP'
```

This example maps German DISPDOT commands onto the US/UK DOTS string.

If the DISPDOTS command is not defined, the DOTS string is used to define the dot commands.

Define Translated Ruler Characters - DISPRULER

You use DISPRULER to translate ruler characters as stored in files into another representation to be displayed on screen. DISPRULER commands are what you enter, and what you see when documents are displayed. RULER commands are stored in documents. You are advised not to changed RULER commands unless you are prepared to compromise the portability of the document. Such DISPRULER commands are position dependant within the flag.

For example:

```
RULER='#CIJLTRHAM¦acv"

DISPRULER='#ZEBLTRHAM¦acv"
```

This example maps German DISPRULER commands onto the US/UK RULER string.

If the DISPRULER command is not defined, the RULER string is used to define the ruler characters.

Define the Print-time Commands - DOTS

You use two letter dot commands to set print-time commands from within the Word Processor. To define these codes you use the DOTS flag in #SYSTEM.

For example:

```
DOTS='HE:HM:FO:FM:PL:PA:PN:SN:PM:SP:JY:JN:RE:ME:ST:SB:FN:GR:FT:IX:IC:FS:FE:NS:PT:UP'
```

Note: Any changes required to the dot commands should be made by using DISPDOTS. This allows documents to be printed using the local configuration while retaining a standard interchangeable configuration on disk.

The following table explains each code:

Code	Function
.HE*n*	Print the next *n* lines as a header at the top of each page.
.HM*n*	Start printing text *n* lines after the top of the page or the end of the header.
.FO*n*	Print the next *n* lines as a footer at the end of each page.
.FM*n*	Stop printing text *n* lines before the bottom of the page or the beginning of the footer.
.PL*n*	Set the page length to *n* lines.
.PA	Start a new page here.
.PN*n*	Force the page number to *n*.
.SN*code*	Send indicated *code* to printer now.
.PM*n*	Start a new page here if fewer than *n* lines remain.
.SP*n*	Define the line spacing. If *n* is omitted, Uniplex adjusts the spacing to a single line spacing value appropriate to the current font. *Note: For Pre-Styled documents, you can set line spacing using fixed units by adding the suffix '**a**' (absolute) to the command. Uniplex prints using fixed units rather than that determined by the current font.*
.JY	Reformat all text according to current rulers until a .JN command or the end of document is encountered.
.JN	Do not format subsequent text.
.RE	Comment line.
.ME*filename*	Merge indicated *filename* at print-time.
.ME**!***command*	Merge result of indicated command at print-time.

Code	Function
.ST*code*	Send indicated *code* to printer for top of page controls.
.SB*code*	Send indicated *code* to printer for bottom of page controls.
.FN*name*	Change to the named font.
.GR*filename*	Draw a graph using the named graph *filename*.
.FT*x*	Controls footnote text: .FTS save all subsequent footnote text. .FTD print all saved footnote text.
.IX*label*	Compile index using references on this line.
.IC*label*	Compile table of contents using references on this line.
.FS	Start footnote text.
.FE	End footnote text.
.NS*code*	Set numbering style.
.PT	Set ruler pitch.
.UP0 *program_name*	Replace *pfilter* with the specified program at print-time.
.UP1 *language_code*	Change language dictionary.
.UP2	Stop spelling of text.
.UP10	Used in the Pre-Styled document header to define printer style and class.
.UP11	Allow text on previous line to be justified regardless of the content of the next line.

.SN, .ST AND .SB Commands

You can include keywords in a printer Pcap section, to send a single command sequence to the printer. Uniplex uses the keywords in conjunction with the commands .SN, .ST and .SB so that, instead of including the sequence in the document, the name can be used. Suppose the sequence to print a change bar one line high in the right-hand margin is **ESC 99C**, and the sequence to print a change bar two lines high in the right-hand margin is **ESC 99C2**, then you might enter the following flags in Pcap:

```
CN=$-'99C'
C2=$-'99C2'
```

In the document, you could use the following .SN commands to print change bars:

```
.SNCN
```

This text would print a change bar one line high in the right-hand margin.

```
.SNC2
```

This text would print a change bar two lines high in the right-hand margin.

For further information, see the chapter Configuring Printers in the Uniplex Device Configuration Guide.

Define Field Separator - FIELDSEPS

You use this flag to set the field, record and thousand separator characters. These are used by the Spreadsheet, Database Query, Report Writer and Word Processor (calculate command).

If the DECTAB command has been set to '.' then the default characters for these separators are ',;,'.

For example:

```
DECTAB='.'
FIELDSEPS=',;,'

name1, name2
records1; records2
10,500
```

If the DECTAB command has been set to ',', then the recommended setting for FIELDSEPS is ';:.' for field, record and thousand separators.

```
DECTAB=','
FIELDSEPS=';:.'

name1; name2
records1:  records2
10.500
```

Discontinued Flag - FOLIO

This flag, supported in earlier versions of Uniplex, is no longer used.

Define the Hard Characters - HARDCHARS

You use the HARDCHARS flag to define various special *hard* characters as 15 positional-dependent values. Hard characters are characters entered in a file which Uniplex does not remove or alter when the file is reformatted. For example, Uniplex may add or remove spaces entered with the SPACEBAR when it reformats text. If you enter a *hard space*, this indicates that the space will not be modified by Uniplex.

Character position	Meaning
1	Hard space*
2	Soft hyphen*
3	Hard tab*
4	Hard line break
5	Preferred hard return*
6	Remark marker
7	Contents marker
8	Index separator character
9	Footnote marker*
10	Abbreviation period*
11	Leader dot
12	Decimal dot
13	EM space
14	EM dash
15	Thin space

* Uniplex usually uses only those characters marked with an asterisk.

For example:

```
HARDCHARS=160-173-137-138-'~'-146-148-152-134-129
```

Define the Minimum Hyphenation Length - HLEN

In the Word Processor, you use the hyphenation mode to hyphenate text at reformat time. You can use the HLEN flag to set the minimum word length below which Uniplex will not attempt hyphenation. HLEN takes an integer value. For example:

```
HLEN=5
```

Note: Uniplex uses the hyphenation points from words in the dictionaries to produce automatic hyphenation.

Disable Word Processor Type-ahead Processing - KEYIN

By default, the Word Processor checks whether the user has entered a command, after displaying each line of the screen. In this way it avoids unnecessary screen redrawing. For example, if the user enters the command to scroll down twice or three times, it is unnecessary to redraw the screen for the intermediate screens, it is only necessary to redraw the screen when the appropriate point is reached.

You can modify this type-ahead processing using the KEYIN flag, by setting it as follows:

Setting	Description
KEYIN='N'	The default setting. Uniplex checks for user input after displaying each line of the screen display.
KEYIN='X'	Disables type-ahead processing. Uniplex only checks for user input after displaying the complete screen.
KEYIN='*n*'	Where *n* is the number of lines after which you want Uniplex to check for user input. Uniplex checks for user input after displaying the number of lines you specify here.

Define the Lower Conversion Characters - LOWER

You use the LOWER flag to set lowercase conversion. For example, suppose [is mapped to lowercase a grave (à), the entry:

```
LOWER='['
```

would ensure that you could use the Convert to Lower command in the Word Processor to correctly convert this foreign language character from uppercase to lowercase.

Each lowercase character in LOWER must have a corresponding uppercase equivalent in UPPER.

For example:

```
LOWER=224-225-233                  * X/OPEN a grave, a acute, e acute
UPPER=192-193-201                  * X/OPEN A grave, A acute, E acute
```

Define the MERGE! Flag

In Uniplex, you can merge the results of an operating system command into the current application by entering the merge command and using the following syntax:

!*command*

You can use the MERGE! flag to define the valid commands. For example:

```
MERGE!='date udate ls pwd who ps sort fgrep grep'
```

You can edit this list to add or remove valid operating system commands.

Define the Default Operating Mode - MODE

You can use the MODE flag to define the default operating modes, mainly for the Word Processor. You define modes by assigning one or more of the following option letters to the MODE flag:

Note: Entries marked by an asterisk (*) can be overridden with the Operating Modes form in the Word Processor. See the note about working options in the earlier section on "Compiling uniplex.sys".

Letter		Details
A		Enable beep whenever cursor moves onto ruler.
B	*	Align multi-columns with a blank line.
C		Enable the clock on the menu line.
D		Prevent reformatting of text under a ruler containing decimal tabstops.
H	*	Set hyphenation mode as default.

Letter		Details
I		Set Insert mode as default. If **I** is not set, the default is overtype mode.
J	*	Set auto-justify mode as default.
L	*	Disable the creation of document headers when creating new Pre-Styled documents.
O		Operate in "Version 6 compatibility mode". This makes the Word Processor operate with softkeys only and not ring menus. The Uniplex program will then look for softkeys from the file *uniplex.fn* instead of *wpring.fn*.
R	*	Set Hard return mode as default.
T	*	Set Tab mode as default.
U		Do not underline spaces when using obsolete command F034.
W		Enable wait-for-spell mode. Causes the speller to write correction information to a file rather than the screen. When the speller is finished, Uniplex opens and reads the file, letting the user interactively control the corrections in the normal way.
/*n*	*	Use *n*, by default, to enter the last level of the command menu. Where *n* can be any character.
?	*	Set hint mode as the default.
>	*	Set cursor wrap mode as the default.
$		Set $*name* to be expanded to the value of the corresponding variable when entered as part of a filename. *Note: If you include $ as part of the MODE string, you must remove it from the STOP string (described later) and recompile uniplex.sys before the expansion will take place.*
-	*	Set auto-hyphenation mode as the default.

Letter	Details
F1	Set file listings to be in reverse time order, newest files first, and short format.
F2	Set file listings to be in alphabetical order and long format.
F3	Set file listings to be in reverse time order, newest files first, and long format.
F4	Set file listings to be in alphabetical order and short format (this is the default file listing format).

For example:

```
MODE='ITC'
```

Sets Insert and Tab mode and displays the clock on the menu line.

Define the Network Type - NETWORK

This flag was available in earlier releases; it is no longer supported.

Define the Location of the System Name - NODE

Applications such as Electronic Mail and Time Manager require each computer system to have a name. You use the NODE flag to define the file in which the system name is stored. For example:

```
NODE='/etc/sysid'
```

Note: If this flag is not set, or the file specified does not contain a node name, Uniplex will use the node name from the kernel if available.

Define the Default Page Length - PAGE

The PAGE flag defines the default page length for documents edited using the Word Processor. If PAGE is not defined, Uniplex uses the default of 66. You can set PAGE to any number of lines you require.

For example:

```
PAGE='55'
```

Define the Delimiter Between Paragraph Numbering - PARA

You use the PARA flag to define the delimiter between levels of automatic paragraph numbering. For example:

```
PARA='.'
```

This example defines the delimiter as a period, for example, 1.1.

```
PARA='/'
```

This example defines the delimiter as a slash, for example, 1/1.

Define the Page Numbering Character - PNUM

By default, Uniplex recognizes the hash character in a header or footer as the page numbering character. This character is replaced by the page number at print-time. You use the PNUM flag to define an alternative character.

For example:

```
PNUM='#'
```

This example sets the page numbering character to a hash (#).

```
PNUM='^'
```

This example sets the page numbering character to a caret (^).

Define the Name for Output to the Printer - PRINT

You use the PRINT flag to define the name you can enter to send output directly to the printer, in certain modules, when prompted for a filename. For example, when prompted for a filename in Database Forms, you can specify that a selected group of records is written to a file, or you can enter the definition of PRINT to specify that you want the selected records sent directly to your default printer.

For example:

```
PRINT='PRINTER'
```

Define the Maximum External Window Processes - PROC

You use the PROC flag to define the number of processes that can be switched between using External Windows. The number plus one defines the maximum number of switchable processes. For example:

```
PROC='2'          * Users can switch between 3 processes
```

Define the Valid Ruler Characters - RULER

You use rulers to format text in the Word Processor. A ruler consists of ruler characters, which are defined in the RULER flag.

```
RULER='.#CIJLTRHAM¦acv'
```

Note: Any changes required to the ruler characters should be made by using DISPRULER. This allows documents to be printed showing local ruler characters while retaining a standard interchangeable configuration on disk.

Characters in the RULER flag are position dependent. The following table explains each ruler character:

Position	Default	Explanation
character 1	. (period)	Pads the ruler to the desired length.
character 2	#	Marks the decimal alignment positions on the ruler.
character 3	C	Marks the point around which text is centered.
character 4	I	Indents the first line of a paragraph, or can be used for multi-word hangs.
character 5	J	Justifies text at the right margin.
character 6	L	Marks the left margin.
character 7	T	Marks a tab or alignment position.
character 8	R	Leaves text ragged at the right margin.
character 9	H	Hangs text to the left of the left margin.
character 10	A	Aligns text at the right margin.

Position	Default	Explanation
character 11	M	Centers text between the margins.
character 12	¦	Delimits multiple columns of text.
character 13	a	Right-aligning tab stop (reserved).
character 14	c	Center-aligning tab stop (reserved).
character 15	v	Box alignment mark (reserved).

Enable the Word Processor Autosave - SAVETIME/SAVECHARS

SAVETIME and SAVECHARS define when and if the Word Processor autosaves the current document.

Autosaving causes the document you are editing to be saved automatically at regular intervals. It is saved to a file in your trashcan with the name, **WP.AUTO___***nnnnn*. You can use this file, which is automatically removed at the end of an edit session, to recover work in the event of a system crash.

You can enable autosave by defining either or both of SAVETIME and SAVECHARS.

Note: You can override SAVETIME and SAVECHARS by using the Operating Modes form in the Word Processor. See the note about working options in the earlier section on "Compiling uniplex.sys".

Note: If you define SAVETIME and/or SAVECHARS, see the section Enable UNIX tty Driver Timeouts - TTYTIMING later in this chapter, for details of enabling UNIX tty driver timeouts.

o SAVETIME

This specifies that autosaving is to be performed if:

- CHECKTIME has been set (to check for message arrival; refer to the earlier section, Define Checking for Messages -CHECKTIME/CHECKCHARS, for further details), and:
- The specified number of minutes (the SAVETIME interval) has elapsed, and:
- The user has pressed one or more keys during the specified number of minutes, and:

- The number of seconds defined by CHECKTIME (the CHECKTIME interval) has elapsed since the SAVETIME interval and the user has not pressed a key during that time.

If SAVETIME is set to 0, no time-based autosaving is performed.

For example:

```
SAVETIME=10
```

This autosaves the current document after ten minutes plus the number of seconds specified by CHECKTIME.

If the user continues to press keys without leaving a gap longer than CHECKTIME, Uniplex will not autosave.

o SAVECHARS

This specifies that autosaving is to be performed if the user has made the specified number of keystrokes.

The number of keystrokes defaults to zero (no keystroke-based autosaving).

For example:

```
SAVECHARS=1000
```

The current document will be autosaved after 1000 keystrokes.

Define the End of Sentence Markers - SENTEND

You use the SENTEND flag to define the characters Uniplex recognizes as marking the end of a sentence. These are used by the Go to Sentence command and can be configured for use with foreign language translations.

For example:

```
SENTEND='.!?'
```

Define the Softkey/Popup Flag - SOFTKEY

You use the SOFTKEY flag to define whether Uniplex displays only softkey menus, or softkey and popup menus. The options are:

```
SOFTKEY='Y'
SOFTKEY='P'
```

The default is **Y**.

The **P** option supports an extension to softkey/popup menu definitions, where a menu can be defined to have both a softkey and a popup menu, using the syntax:

#*menuname*	Start of menu
'F1=.... F2=.....'	Softkey label line
" *title*	Popup menu definition
.	
1 -	
.	
))	End of popup
1=*action*	Actions, common to both menus.
2=*action*	
.	
))	End of menu.

If the **P** option is set, then menus in this format will be displayed as popups, whereas if **Y** is set they will be displayed as softkeys.

Define the Spell Checking Keywords - SPELL

The SPELL flag in uniplex.sys defines which language dictionary and thesaurus to use as the default.

If you have more than one language dictionary or thesaurus on your system, you can change the setting of SPELL.

The syntax is:

SPELL='exec uspell -e -d *language_code* -s ^'

Where:

exec prevents the existence of an additional shell program between uniplex and uspell.

uspell is the name of the spell checker program.

-e is the extra information protocol flag.

-d *language_code* indicates that the language dictionary specified by the *language_code* is the default.

The *language_codes* are:

Language Code	Language Dictionary
001	American English
044	British English
033	French
049	German
039	Italian
002	Canadian French
031	Dutch
045	Danish
034	Spanish
047	Norwegian
046	Swedish
351	Portuguese
358	Finnish
055	Brazilian
061	Australian

If this option is not present, or if the number used does not appear in the above table, then uspell assumes American English by default.

-s indicates that supplementary dictionaries are to be used.

^ is dynamically replaced by the name of a temporary file containing the text to spell check.

For example:

```
SPELL='exec uspell -e -d 001 -s ^'
```

will set the default language dictionary to American English.

Note: See the appendix Program Usage and Invocation for more details about the uspell program.

Define the Spreadsheet Backup File - SSBACKUP

Each time a user edits a spreadsheet, and subsequently saves it, Uniplex makes a copy of the original spreadsheet. The SSBACKUP entry defines the name of this backup file. If you do not define the file no backup is made.

In the SSBACKUP entry, the following characters are expanded:

~ (tilde) is expanded to the user's home directory

^ (caret) is expanded to the current filename.

For example:

```
SSBACKUP='~/ss.back.up'
```

This example sets the backup to ss.back.up in the user's home directory.

```
SSBACKUP='~/ss.bak.^'
```

This example sets the backup to ss.bak.*filename* in the user's home directory, where *filename* is the name of the file currently being edited.

Note: If the specified SSBACKUP file is already in use (for instance, if you have more than one Spreadsheet session active at a time, using multiple windows), Uniplex allocates and uses a unique backup file in the TEMP directory.

Define the Default Line/Column Counter Status - STATUS

You can define the default line/column count mode in the Word Processor using the STATUS flag. The options for this flag are:

STATUS='N' (do not display line or column counter)

STATUS='Y' (display line and column counter)

STATUS='X' (display line counter only)

Note: You can override the STATUS keyword by using the Operating Modes form in the Word Processor. See the note about working options in the earlier section on "Compiling uniplex.sys".

Define Invalid Characters in Filenames - STOP

You use the STOP flag to set the characters not allowed in filenames. This is useful for trapping filenames containing characters that could be interpreted by the operating system, such as &. For example:

```
STOP='\\!&$'
```

Define the Directory to Contain Temporary Workfiles - TEMP

You use the TEMP flag to define the name of the directory where Uniplex stores temporary workfiles. Modules such as the Word Processor create workfiles during the course of an editing session.

For example:

```
TEMP='/tmp'
```

Places the workfiles in /tmp.

Define the Time Format - TIMEFMT

You use the TIMEFMT flag to define the time display in Advanced Office System applications. To do this, set TIMEFMT by using one of the following, enclosed in single quotes:

Setting	Explanation
HH:MM	24 hour time, with zero padding. For example, `TIMEFMT='HH:MM'` displays nine-thirty pm as 21:30
hh:mm	12 hour time with am/pm and zero padding. For example, `TIMEFMT='hh:mm'` displays nine-thirty pm as 09:30pm
hh:MM	12 hour time with AM/PM and zero padding. For example, `TIMEFMT='hh:MM'` displays nine-thirty pm as 09:30PM

Note: The separators in the TIMEFMT string can be one of: `: . ; , -`

Define the Directory to Contain the Trashcan - TRASH

You can use the Uniplex filing system to deposit documents in a trashcan. Subsequently, you can delete the contents of the trashcan, or retrieve documents from it. You use the TRASH flag to define the name of the trashcan directory. For example:

```
TRASH='/tmp/trash'
```

Sets the trashcan directory as /tmp/trash.

Enable UNIX tty Driver Timeouts - TTYTIMING

You can define the TTYTIMING flag to enable UNIX tty driver timeouts. These are used to control CHECKTIME and SAVETIME timing. When TTYTIMING is set, the system is more CPU-efficient than the default UNIX alarm mechanism.

To enable the UNIX tty driver timeouts enter:

```
TTYTIMING='Y'
```

Define the Upper Conversion Characters - UPPER

You use the UPPER flag to define Uppercase conversion. For example, suppose { is mapped to uppercase A grave (À), the entry:

```
UPPER='{'
```

would ensure that you could use the Convert to Upper commands in the Word Processor to correctly convert the case of this foreign language character from lower case to uppercase.

Each uppercase character in UPPER must have a corresponding lowercase equivalent in LOWER. For example:

```
LOWER=224-225-233              * X/OPEN a grave, a acute, e acute
UPPER=192-193-201              * X/OPEN A grave, A acute, E acute
```

Define the Valid Word Delimiters - WDEL

You use the WDEL flag in to define the valid word delimiters used during reformatting in the Word Processor. Normally, these are the standard punctuation characters:

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

Define the Case Insensitivity Marker for Searching - WPCASE

You use the WPCASE flag to define the marker to be used to indicate a case insensitive search or replace in the Word Processor. The syntax for this entry is:

```
WPCASE='xy'
```

where:

x is the character used as the first character in a search or replace string to indicate that the following string is to be searched for case insensitively.

y is the character used to prefix *x* if the user wants to search for a string starting with the character *x*.

For example:

```
WPCASE='\\='
```

Note: Two backslashes are needed to represent one, to satisfy uniplex.sys string syntax.

Using this example, the user starts the search or replace string with **** to indicate case insensitivity. The user starts the search or replace string with **=** to search for a literal **** within the document.

Define the Preset Word Processor Rulers

You use the #RULERS section of uniplex.sys to define the 10 preset rulers (numbered 0-9) used to format text in the Word Processor.

You can configure this section to meet special formatting requirements such as extra wide documents. The rulers in this section are position dependent, the first ruler being 0. By default, Uniplex always displays ruler 0 when a user first edits a document. You can create rulers containing a maximum of 254 character positions.

For example:

```
#RULERS
L.......T.......T.......T.......T.......T.......T.......T.......T....R
H....L..........T.......T.......T.......T.......T.......T.......T....R
L.......T.......T.......T.......T.......T.......T.......T.......T....J
L.......I.......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....M
................L.......T.......T.......T.......T.......T.......T....A
L..............#...............#...............#...............#....R
.L.....................J..L.....................J..L..............J
L.......T.......T.......T.......T.......T.......T.......T.......T..R.
))
```

You use the ruler characters defined in the RULER and DISPRULER sections of #SYSTEM to configure a ruler.

Chapter 6
Configuring Command Keystrokes
(uniplex.cmd)

THIS PAGE INTENTIONALLY LEFT BLANK

File Format and Layout

You use the system command file, *UAP/uniplex.cmd*, to define the command keystrokes for the entire product.

The file is organized using the Uniplex file format and layout:

#*section_name*
keyword
keyword
.
.
.
keyword
))

It contains the following three sections:

Section Name	Description
#COMMANDS	Sets the keystrokes pressed by the user to execute commands.
#COMMANDS-*term_name*	Sets an alternate set of command keystroke sequences for the named terminal, instead of those defined in #COMMANDS.
#COMMANDS2-*term_name*	Sets extra command keystroke sequences for the named terminal, in addition to those defined in #COMMANDS.

These sections are covered in later sections of this chapter. You can also refer to the supplied uniplex.cmd file for information.

Syntax for uniplex.cmd

You use the #COMMANDS section of uniplex.cmd to define the keystroke sequences used to execute commands throughout Uniplex. The syntax of the section is:

#COMMANDS
entry [***** *comment*]
entry [***** *comment*]
.
.
))

All comments are optional. A comment must begin with an asterisk (*).

The syntax for each entry is one of the following:

- *program command*=*sequence*
- *program command*=**K***nnn*
- **K***nnn*=*sequence*
- **M***nnn*=*sequence*

where:

program command	is a function number used by Uniplex to identify and execute a program function, for example, delete a line, or redraw the screen. You cannot change program numbers defined by Uniplex.
sequence	is the keystroke sequence the user enters to execute the program command.
K*nnn*	is a logical number assigned to a keyboard key. K1 is used by Uniplex as the keyboard key to use for F1, K2 for F2, and so on. **K** numbers are the *low level* definition to Uniplex as to what sequences are generated by the keyboard function keys. They are always mapped to the Uniplex softkey functions (**S** numbers) in standard configuration. As supplied, the **K** numbers for supported terminals are defined in *uniplex.cmd*.

M*nnn* — **M** numbers are used to define an input character *sequence* to map to the character stored as *nnn*. This can be used to define a set of default input-only mappings to avoid extensive use of the MAP directive in Tcap sections.

For example: to map the input sequence **control-a** followed by **a** to be stored in a file as character 160, you can use **M160=1-'a'** which performs the same function as the first two sections of the Tcap statement: **MAP=1-'a',160,$-'[am'**

The following table shows the different types of *program_command*:

Type	Summary Description
F*nnn*	Defines a program function.
G*nnn*	Defines a generic program function such as delete.
P*nnn*	Defines a graphics program function.
S*nnn*	Represents a softkey number.

*Note: In #COMMANDS sections, use **&** to represent the "direct command leadin character" which defaults to the Escape key and is shown in the User Guides as **ESC** or **Esc**.*

For all these types of commands, as well as the **M***nnn* statements, the case of the first letter (F, G, S, P or M) is significant, as follows:

UPPER CASE — Any alpha characters in the associated *sequence* (even if assigned via a **K** number) can be typed in either upper or lower case. For instance:

F123=&-'P'

means that function 123 can be invoked by typing either **ESC P** or **ESC p**.

LOWER CASE — Typed alpha characters must be in exactly the same case as defined in the associated *sequence*. For instance:

f123=&-'Pq'

means that function 123 can only be invoked by typing **ESC P q** and not by **ESC P Q**.

The following table shows the different elements you can use in a keystroke sequence:

Element	Summary Description	
- (dash)	You link elements together into a sequence using a dash.	
n	An ASCII decimal number. You use ASCII decimal numbers in the range 1 - 32 to indicate CTRL sequences.	
'string'	A literal string enclosed in single quotes. If the literal string is to contain a single quote (') or a backslash (\) character, these must be "escaped" by prefixing with a backslash. For example: **'a\\b\'c'** represents the 5 characters **a\b'c**.	
&	The command key defined with the CC flag in termcap (cmdch flag in terminfo). If this flag is not defined, **&** is taken to represent escape.	
$	Escape (ASCII decimal 27).	
T	Tab	(as defined in TERMINFO/TERMCAP)
U	Cursor up	(as defined in TERMINFO/TERMCAP)
D	Cursor down	(as defined in TERMINFO/TERMCAP)
L	Cursor left	(as defined in TERMINFO/TERMCAP)
R	Cursor right	(as defined in TERMINFO/TERMCAP)

Example Commands

You can use any unique combination of keystrokes to execute a program function.

For example:

```
F27=4
```

In this example, you can trigger program command 27 (Scroll Down) by pressing CTRL d (ASCII decimal 4).

```
F27=&-&-D
```

In this example, you can trigger program command 27 (Scroll Down) by pressing ESC ESC cursor-down.

```
F27=4
f27=&-&-'d'
```

In this example, you can trigger program command 27 (Scroll Down) by pressing either CTRL d or ESC ESC d.

```
F011=13
```

In this example, you can trigger program command 011 (Carriage Return) by pressing RETURN, (ASCII decimal 13).

Example Generic Commands

You use generic commands to define a keystroke sequence as a root for a series of related commands. For example:

```
G001=&-'D'
F003='W'
F004='R'
F005='L'
F007='B'
F080='S'
```

In this example, you trigger the generic root - program command G001 (Generic Delete) - by pressing ESC d. Uniplex displays the Word Processor delete options, prompting the user for the key to press to complete the command.

All following **F** commands which consist of a single keystroke are interpreted as "branches" to the generic command - options which will complete the generic command. Available "branches" end at the next non-**F** command definition or **F** command that is assigned a *sequence* that does not consist of a single character.

For example:

```
F003='W'
F004='R'
F005='L'
F007='B'
F080='S'
```

are all options which will complete the generic command because they consist of a single character. So, for instance, **ESC d w** is the sequence to delete word (F003).

Compiling uniplex.cmd

After making changes to the uniplex.cmd file, you need to compile it. Compiling the file converts it from ASCII to binary format. You do this by entering **syscomp** at an operating system prompt, using the ESC ESC $ escape sequence to reach UNIX.

This works in conjunction with the terminfo/termcap and Tcap files, to create a compiled version of the *uniplex.cmd* information in both the UAP/termdef and UAP/commands directories.

It creates a file for each terminal type that has been configured, in both directories. For example if your system has been configured for use on an altos II terminal, syscomp will create the following files:

UAP/termdef/altosII

and

UAP/commands/altosII

Note: The use of syscomp is closely linked to terminal descriptions, so please see details of its use in the chapter Configuring Terminals in the Uniplex Device Configuration Guide.

#COMMANDS Section of uniplex.cmd

This section lists the keystroke commands found in #COMMANDS of uniplex.cmd. The commands have been sorted into numerical order by function number. Not all functions are used in the current product.

Note: For a list grouped by functionality, see the uniplex.cmd file on your system.

Command	Description
F001	Delete character
F002	Erase left character (DEL/RUB)
F003	Delete word
F004	Delete right
F005	Delete left
F006	Delete whole line
F007	Delete blank lines
F008	CTRL-E - Add space
F009	CTRL-O - Open line
F010	Add blank lines
F011	Carriage return
F012	Cursor left
F013	Cursor right
F014	Cursor up
F015	Cursor down
F016	Tab
F017	Previous word
F018	Next word
F019	Scroll line up
F020	Scroll line down
F021	Beginning of text line
F022	End of text line
F023	Top of screen
F024	Bottom of screen
F025	Top left of screen
F026	Last character on screen
F027	Next screen
F028	Previous screen
F029	Goto top of file
F030	Goto bottom of file
F031	Convert to lower case
F032	Convert to upper case
F033	Center line
F034	Underline with character
F035	Mode : Show print effects
F036	Split line

Command	Description
F037	Repeat last command
F038	Format paragraph
F039	Help!
F040	Redraw screen
F041	Find next
F042	Find forwards
F043	Locate and replace
F044	Format document
F045	Abandon global commands
F046	Page number
F047	Read ruler
F048	Save ruler
F049	Use ruler
F050	Mode : Preferences dialogue box
F051	Join lines
F052	Mark top left block
F053	Mark bottom blank
F054	Mark bottom leave
F055	Mark bottom remove
F056	Mode : Column counts
F057	Paste: Overlay
F058	Paste: Insert
F059	Paste: Elbow
F060	Window: Integrate (direct call to #EDITMENU in WP)
F061	Save and exit (enter in other applications)
F062	Quit, abandon - no update
F063	Write to file
F063	Save file
F064	Backtab
F065	Read and insert
F066	Undo last delete
F067	Mode : Toggle insert/overtype
F068	Force insert mode to off
F069	Print effect on
F070	Print effect off
F071	Mark top left serial
F072	Cut to named file
F073	Read and overlay
F074	Save to named file
F075	Locate spelling errors
F076	Erase block
F077	Clipboard number
F078	Right align
F079	Calculate: Will do calculations on numbers
F080	Default all form fields
F081	Default form page

Command	Description
F082	Default form field
F083	Reserved
F084	Save screen (to clipboard)
F085	Mark bottom right, no action
F086	Locate string backwards
F087	Replace text backwards
F088	Thesaurus lookups
F089	Spell serial block
F090	Page left
F091	Page right
F092	Remove effects from block
F093	Backspace
F094	Outline edit/outline view
F096	Append to file
F097	Alternate expand
F099	Expand
F100	Reserved
F101	Start block/paragraph
F102	End block/paragraph
F103	Next paragraph
F104	Start sentence
F105	End sentence
F106	Next sentence
F107	Hide marked area
F108	Spell marked area
F109	Draw box
F110	Align columns
F111	Hard tab
F112	Indent left margin
F113	Un-indent left margin
F114	Undo last line change
F115	Redo last undo
F116	Indent right margin
F117	Un-Indent right margin
F118	Convert block to uppercase
F119	Copy marked area
F120	Move marked area
F121	Hard line break
F122	Explicit hard return
F123	Hard space
F124	Footnote mark
F125	Fill block with graphic effect
F126	Index
F127	Table of contents
F128	Reference or footnote
F129	Notes or remarks

Command	Description
F130	Division
F131	Block or paragraph
F132	Convert block to lowercase
F133	Start page
F134	End page
F135	Next page
F136	Cut and blank
F137	Cut and leave
F138	Cut and remove
F139	Cut and write
F140	Cut and append
F141	Forwards
F142	Effect block
F143	Switch
F144	Backwards
F145	Reserved
F146	Move
F147	Grow
F148	Process (process switch menu)
F149	Open full new line, even in multi-columns
F150	View graph, used for graphics user interface
F151	Read from file
F152	Mouse event
F153	Redraw graph
F154	New file
F155	Save to unique file ($Utmpfile)
F156	Go to end of word
F157	Go to marked position (top left of cut)
F158	Re-edit file
F159	Go to top level of ring menus
F160	Go to last level accessed of ring menus
F161	Search and replace dialogue box
F162	Page format dialogue box
F163	Create ruler dialogue box
F164	List rulers popup
F165	Reserved
F166	Export document as an ASCII file
F167	Numbering styles dialogue box
F168	Paragraph numbering dialogue box
F169	Reserved
F170	Window: Open
F171	Window: Forwards
F172	Window: Close
F173	Window: Switch
F174	Window: Backwards
F175	Reserved

Command	Description
F176	Window: Link
F177	Window: Unlink
F178	Reserved
F179	Window: Desk (Direct call to #DESK softkey menu)
F180	Tape: Insert
F181	Tape: Begin
F182	Tape: End
F183	Tape: Continue
F184	Tape: Save
F185	Tape: Recall
F186	Tape: Attach
F187	Tape: Play
F188	Tape: Menu
F189	Tape: Library select
F190	Internal Test: Command for Utest
F191	Internal Test: Keystroke logging on/off
F192	List Files: Next mode
F193 - F197	Reserved
F198	Mode: Wysiwyg (PWP)
F199	Mode: PS/draft mode (PWP)
F200	Reserved
F201	Expand outline
F202	Compress outline
F203	Outline view
F204	Outline full expand
F205	Outline body text
F206	Edit a print command
F207	Create header/footer dialogue box
F208	Effects popup box
F209	Font selection dialogue box
F210	Draw mode ON
F211	Erase mode on
F212	Draw and erase mode off
F213	Create a bookmark
F214	Find a bookmark
F215	Create ’Print to Screen’ display
F216	Switch to ’Print to Screen’ display
F217	Edit Pre-Styled document header
F218	Reserved (File Manager change context)
F219	Use File Manager in SELECT FILE mode to return the name of a file
F220	Activate the File Manager in BROWSE/MANAGE mode
F221 - F238	Reserved
F239	Never used for any Uniplex function. Can be used in terminal drivers to map "illegal" characters or key sequences so that Uniplex beeps when entered

Pseudo Commands	Description
F240	Utility desk
F241	Escape to sub-shell
F242	Surrogate print key (ESC CTRL-P)

Generics	Description
G000	Generic find
G001	Root for delete
G002	Root for insert (add)
G003	Root for format
G004	Not used (Mark ??)
G005	Root for cut
G006	Root for paste
G007	Generic read
G008	Go to paragraph start/end/next
G009	Go to sentence start/end/next
G010	Generic format controls (PWP)
G011	Generic window command
G012	Root for goto
G013	Root for serial cut
G014	Generic saves
G015	Root for tapes
G016	Not used
G017	Not used
G018	Generic option start
G019	Go to page start/end/next
G020	Change screen menu (lines)
G021	Change screen menu (solids)
G022	Generic external windows
G023	Change screen menu (markers)
G024	Graphics add
G025	Change screen menu (text)
G026	Bitmap graphics pan and zoom functions
G027	Miscellaneous graphics cmds

Input maps	Description
m129	Abbreviation period
m160	Hard space
m163	Pound sign
m173	Soft hyphen
m175	Tilde

Graphics Commands	Description
P001	Add a line
P002	Add a box
P003	Add a marker
P004	Add a segment
P005	Add an arc
P006	Add a poly
P007	Add Text
P008	Add Circle
P010	-- revert to default/normal full image zoom
P011	-- zoom in
P012	-- zoom box
P013	-- zoom out
P014	-- pan left
P015	-- pan right
P016	-- pan up
P017	-- pan down
P020	-- text horizontal justification
	-- fill pattern
P021	-- fill color/hue
	-- text size
P022	-- marker size
	-- text type
	-- line width
	-- outline width
P023	-- text face
	-- line style
	-- outline style
	-- marker style/type
P024	-- line color
	-- marker color
	-- outline color
	-- text color
P025	-- text rotation
P030	Grid on/off toggle
P031	Snap on/off toggle
P032	Set grid pitch
P033	Switch number of graphs displayed
P034	Redraw current graph
P100	Internal Test : Bitmap graphics

Define a Set of Command Keystrokes for a Specific Terminal

You create #COMMANDS2-*term_name* sections in *uniplex.cmd* to define a set of command keystrokes for a particular terminal, to be used in addition to those in #COMMANDS.

All fully supported terminals supplied by Uniplex use the #COMMANDS2-*term_name* to define:

1 The sequences generated by the terminal's function keys (**K** numbers).

This is normally done by using the statement **include=#***keyboard_type*, since many different terminals' function keys generate identical sequences, which are then defined in a suitably named sub-section. For example:

include=#vt100STYLE-KEYPAD

Note that the supplied keyboard definition sub-sections assign function keys not used by Uniplex (K17-K20) so that, if the user presses them, the message:

```
You have not configured this function key. Press F1 to continue
```

is displayed. If you want to configure these keys for other Uniplex operations, you will have to re-work this section of the terminal's #COMMANDS2-*term_name* section.

2 The standard Uniplex mapping of **K** numbers to **S** numbers. This should always be done by using the statement:

include=#UNIPLEX-CONFIGURATION

3 Any other specially labelled and accessible keyboard keys that can be usefully assigned to a Uniplex function - such as assigning the Uniplex Scroll Up function (F28) to a key engraved with *Previous Page*.

For example, if the TERM environment variable for an Altos II is altosII, the header for the section would read:

```
#COMMANDS2-altosII
#COMMANDS2-altosII
```

When you invoke Uniplex using this type of terminal, Uniplex would use the set of command definitions in #COMMANDS2-altosII in addition to those in #COMMANDS.

The supplied *uniplex.cmd* file describes this terminal as follows:

```
#COMMANDS2-II,COMMANDS2-alt2,COMMANDS2-altosII,COMMANDS2-altos3,...
include=#WyseSTYLE-PF1-PF16
include=#UNIPLEX-CONFIGURATION
                        * KEYBOARD KEY                      UNIPLEX OPERATION
                        * ------------                      -----------------
f39=1-'P'-13            * Help
f08=$-'[@'              * Insert char
f27=$-'[S'              * Next Screen
f28=$-'[T'              * Previous Screen
f01=$-'[P'              * Delete  character
f09=$-'[L'              * Insert line
f06=$-'[M'              * Delete line
f25=$-'[f'              * Home Key
f04=$-'OP'              * CLR key                           Delete to end of line
* Uncomment following line on Altos II terminal to enable XOPEN-MAPS
*include=#XOPEN-MAPS
))
```

*Note: As a general rule, in #COMMANDS2 section, use **$** and not **&** to represent escape.*

Define an Alternate Set of Command Keystrokes for a Terminal

You can create #COMMANDS-*term_name* sections in *uniplex.cmd* to define an alternate set of command keystrokes for particular terminals.

For example, if the TERM environment variable for your terminal is 3161, the header for the commands section could read:

```
#COMMANDS-3161
```

When you invoke Uniplex using this type of terminal Uniplex would use the set of command definitions in #COMMANDS-3161 in preference to those in #COMMANDS.

You will normally only have to do this for terminals whose keys generate key sequences that conflict with common Uniplex direct command sequences. For instance, if the terminal's function keys generate the sequences **ESC a**, **ESC b**, **ESC c**, and so on.

Other Statements

include= command

The statement:

include=#*section_name*

can be used to include named sub-sections into other sections. For instance, the supplied configuration makes extensive use of the sub-section #**UNIPLEX-CONFIGURATION**, which is included by all #COMMANDS2-*term_name* sections by the statement:

include=#UNIPLEX-CONFIGURATION

See the earlier section, Define a Set of Command Keystrokes for a Specific Terminal, for more details of how the include statement is used.

Pseudo Commands

Command numbers that are not defined internally by Uniplex programs (those greater than 240) can be defined in *uniplex.cmd* and assigned an associated *sequence*.

If, when a given program is running, it has a #COMMANDS section defined in its softkey file which contains the same command number, then the defined *sequence* will invoke the defined softkey *action*.

Using this technique, you can create "new" Uniplex commands - referred to as "pseudo commands". See the chapter Configuring Softkeys, Ring Menus and Popups for more details.

Additional Commands Appended by the APP

The APP appends the section *#COMMANDS2-$TERM* to the uniplex.cmd file for each of the newly installed terminals. These sections consist of include statements referring to other sections which are added as part of the installation procedure. The *#COMMANDS2-$TERM* section defines the character sequences sent by the keyboard. The Uniplex functions they invoke are basically independent from the product translation but dependent upon the terminal language and the type of keyboard used with the terminal.

For for details about terminals, see the Configuring Terminals chapter in the Uniplex Device Configuration Guide.

Chapter 7
Help and Message Configuration

THIS PAGE INTENTIONALLY LEFT BLANK

Configuring Help

You use the Uniplex help facilities to provide on-line help and documentation for Uniplex modules. You can modify the help text provided with Uniplex. Additionally, you can modify the messages that Uniplex displays.

Uniplex provides an extensive on-line help system. The help system has been designed to show users how to make the most effective use of its applications.

You can display help at any point while using Uniplex:

- While displaying a menu
- While using an application
- While using a form

This chapter describes how to change Uniplex help in any of the following ways:

- Change the help text by changing the text in the help files. Make sure you take a backup of the original file before attempting any changes.
- Add more help text to the existing help text.
- Change the way help text is accessed.
- Change the help menus.

Additionally, when creating your own forms, menus or applications, you can use the existing Uniplex facilities to provide help for these.

This section is divided into the following sub-sections:

Section	Sub-section
File Format and Layout	Describes the structure of help files and the syntax to use.
Create Help for Menus	Describes how to add a Help option to a menu which provides help on how to use that menu.
Create Help for Applications	Describes how to provide Help for any Uniplex application. Help for applications is context sensitive.
Create Help for Forms	Describes how to provide Help for any Uniplex screen-based form and explains how to complete the form.

File Format and Layout

Each Uniplex module has a help file, which is named with the suffix, **.help**. For example, the word processor help file is uniplex.help; the spreadsheet (ISSI interface) help file is issi.help.

Each help file contains help sections, using the syntax:

#topic1, topic2 ... topicn
section_title
help text
.
.
.
.PA
.
.
.
))

Where:

#topicn	is the name of the help section.
section_title	the first text line of the section is taken as the help title. At runtime, this text is displayed on the Uniplex title line in reverse video.

.PA	can be used to break the help text into screens.
))	is the end of section marker.

You can include help text in free format, using print effects and graphics boxing to enhance readability.

Help on Menus

While you are accessing a menu, the help that is displayed depends on which method you use to access it; you can either press ESC h or pick and point the Help option.

If you press ESC h, Uniplex displays the help defined in the currently active softkey menu for the application. See the section Help While Using Applications later in this chapter for more details of this kind of help.

If you pick and point the Help option, Uniplex displays a help menu. This help menu gives you access to the following different types of help:

1 Help on the menu the user is accessing. For example, if the user is accessing the Word Processing menu, the help explains the different options available on that menu.

2 General Help about the way Uniplex works. For example, explains how to use softkeys.

3 A subset of the help available for the application. For example, explains how to print a document.

Follow these steps to provide access to help via the help option on any menu following these steps:

1 Include the help option and definition on the menu.

The system file uniplex.menu contains the layout and definition of each menu. The majority of existing menus contain the specification for the H option. If you have defined a new menu, you need to include the help option and definition on that menu. See the section, Define Help on Menus, for more details.

2 If required, create help menus.

You define help menus in the same way as you define other Uniplex menus. See the section Create a Help Menu for more details.

3 Create or modify the help text.

You must make sure the help files and the sections you have specified exist and are relevant to the menus you have defined. See the earlier section File Format and Layout for a description of how to format help files.

Full details on how to carry out all these tasks are in the sections that follow.

Define Help on Menus

When modifying or editing a menu, you can add a Help option to the menu in the same way that you add any other option.

In the specification of the menu (immediately below the menu layout), specify the Help option in one of the following ways:

H > *menu_name*

This indicates that this option accesses another menu. For example:

```
H > #MAIN_HELP
```

H ? *help_file section_name*

This indicates that this option accesses a help file section. For example:

```
H ? ../wp/uniplex.help main
```

This indicates that the Help option displays the help section called *main* in the standard help file *uniplex.help* which is in the directory *UAP/wp*.

Create a Help Menu

If you have added help to a menu, you may need to create a help menu to allow access to the specified help section.

When you specify that an option from a Help menu accesses a section from a help text file, use the following syntax:

n **?** *filename section_name*

Where:

n is the number of the option

filename is the name of the help file

If *filename* does not start with a '/', then it is assumed to be a name relative to a directory one level below the central UAP directory. For example, use:

```
../uc/issi.help
```

to reference the file **UAP/uc/issi.help**

section name is the name of the help section in the help file

For example, you can create a help menu as follows:

```
#HELP_GENERAL
G E N E R A L  H E L P   M E N U
------1------------------------------2-----------------------R
      "INTRODUCTION                  "OPTIONS

       1 - Overview                   A - Option One
       2 - General                    B - Option Two
                                      C - Option Three
       3 - Finishing
       4 - Going
                                      Q - Quit
))
1 ? ../wp/uniplex.help START
2 > #HELPGEN
3 ? ../wp/uniplex.help FINISH
4 ? ../wp/uniplex.help GOING
A ? ../wp/uniplex.help ONE
B ? ../wp/uniplex.help TWO
C ? ../wp/uniplex.help THREE
Q <
))
```

Note: See the chapter Configuring Menus for more details about configuring help menus.

Help While Using Applications

Context-sensitive help is provided for each of the applications. In most cases, Uniplex uses the current softkey menu to identify the current context of the application. For example, if the user has the Draw softkeys displayed while using the Word Processor, Uniplex displays the Drawing Lines help text.

Each softkey menu contains a help entry that points to a popup help menu. These are contained in the .fn file for that application. For example, the popup help menu for the word processor is contained in uniplex.fn. The popup help menu for the Spreadsheet (issi interface) is contained in issi.fn.

Each option on the help menu either points to another help popup menu or a section in the help file.

In this way, as the user changes softkey menus, so the associated help menu changes.

Every softkey menu file (.fn file) specifies the popup menu which should be displayed if a user requests help. You specify this using the following syntax:

H=M(#*name*)

Where *name* is the name of the menu.

Each help menu is defined as a popup menu. See the next section, Define Popup Help Menus for details.

Define Popup Help Menus

When defining popup help menus, you use the following syntax:

n=*x*(*name*)

Where:

n	is the number of the option
x	H specifies a help section
M	specifies another menu
name	is the name of either the section in the help file, or the name of the menu in this file.

For example:

```
1=H(#WP_MOVE5)
```

Displays the help section #WP_MOVE5 in uniplex.help.

```
2=M(#GENERAL_H)
```

Displays the popup menu #GENERAL_H.

The help menu for each softkey menu is organized as follows:

menu_name

1- Full Help
2- General Help
3- Quick Reference

4- *softkey_section_name*
5- *softkey_section_name*
6- *softkey_section_name*
7- *softkey_section_name*
8- *softkey_section_name*
9- *softkey_section_name*

Options 4 through 9 provide help on specific softkey menu options.

For example, if the current softkey menu is the Draw softkey menu:

```
F1=Line Draw F2=Erase Lines F3=Normal Mode F4=Quit
```

The definition for help for this softkey menu in wpgeneral.fn is:

```
H=M(#H_DRAW)
```

If the user presses ESC h, the following help popup is displayed:

```
Drawing Lines

1- Full Help
2- General Help
3- WP Quick Lookup

4- Draw Lines
5- Move While Drawing
6- Erase Lines
```

This menu is defined in wpgeneral.fn as follows:

```
#H_DRAW
''
"Drawing Lines
"
1- Full Help
2- General Help
3- WP Quick Lookup
"
4- Draw Lines
5- Move While Drawing
6- Erase Lines
"
"ESQ Q to Quit
))
1=M(#H_FULL)
2=M(#GENERAL_H)
3=H(#QUICK_WP)
4=H(#WP_DRAW1)
5=H(#WP_DRAW2)
6=H(#WP_DRAW3)
))
```

Context Sensitive Help While Using Ring Menus

While using an application which contains a ring menu, such as the Spreadsheet or Word Processor, you can access context sensitive help by pressing ESC h when the highlight is on the ring menu option you want to display help about.

For example, if the highlight is on the Spell option and you access help, Uniplex displays help concerning that option.

You define these sections in the same way as other help sections, except that you use the following syntax for the name of the help section:

#menuname.option1,menuname.option2,...menuname.optionn

Where:

menuname is the name of the menu, in the relevant *.fn* file, that contains the option.

option is the option on that menu.

For example, the name of the help section for the Spell option in the Word Processor is:

```
#TOOLS.Spell
```

Context Sensitive Help While Using the Spreadsheet

While using the Spreadsheet, in issi mode, context sensitive help is accessible directly, after entering a function, as well as through a popup help menu.

For example, if you enter **@sum** and then access help, uniplex displays the help section concerning the **sum** function immediately, without having to select an option from a menu.

In order to display the relevant help, Uniplex accesses the #KEYWORDS section of issidefs and uses the entry for the current function as a link to the appropriate entry in issi.help.

For example, if you have accessed help after entering **@sum** in a spreadsheet, Uniplex will find the following entry in the #KEYWORDS section of issidefs:

```
SUM=sum
```

It will then use this entry to find the relevant entry in issi.help. For example:

```
#SSSTAT1,sum
```

Uniplex will then display the relevant help section on the screen.

If you change the function name in the #KEYWORDS section of issidefs, you must therefore also change the relevant entry in issi.help, otherwise Uniplex will be unable to display help on the screen.

Help While Using Forms

You can display help while using most Uniplex forms, to explain the purpose and scope of the form.

To provide help on a form, you create a :HELP section in your screen template. You also need a softkey of the form **H=H (formhelp)** in effect.

For example, the forms provided to create an Easiletter were created using the Screen Builder.

```
:HELP

            Complete this form as follows:

            Entry 1   Make your first entry in this field.

            Entry 2   Make your second entry in this field.

            Press ESC e when you have completed this form.
))
```

See the Screen Builder chapter in the Uniplex Form-Building Tools guide for details of creating forms and how to provide help for them.

Indexing Help Files

After making any changes to a help file you should index it. This enhances performance when Uniplex is accessing the file.

To index a help file, at a system prompt, using the **!** (shell command) option from the main menu, enter:

uindex *file*

Where *file* is the name of the help file you want to index.

For example, to index the electronic mail help file you enter: **uindex umail.help**

Configuring Messages

You use the Uniplex message facilities to provide the following:

- Additional help and information on runtime operation.

  For example:

  Messages prompt the user for input, such as a document name, or the keystrokes to complete a command.

- Information on error conditions or invalid user input.

  For example:

  Messages indicating that a user does not have permission to read/write to a document, or has input an invalid command.

- Module specific operating parameters.

  For example:

  A list of the valid data types in the Database, or the valid day names in Electronic Mail.

All messages are stored in ASCII text files so that they can be configured or translated if required.

File Layout and Format

Each Uniplex module has a message file, which is named with the suffix .msg. For example, the Word Processor message file is uniplex.msg; the Spreadsheet (issi interface) message file is issi.msg. Entries in the message file use the syntax:

n (,)message_text

where:

n	is a program message number. You cannot change message numbers as they are hard coded in the binary programs.
,	(comma) is the character used to right-align translated print-time commands. This is only effective on messages 300 to 349 in the file *uniplex.msg*.

message text is the text that is displayed at runtime.

As a general rule, do not enter messages longer than 65 characters. This is the width of the window in which Uniplex displays some messages.

As far as possible, maintain the same maximum character lengths for each of the messages.

If the *message_text* begins with a non-alphabetic character, for example a number or a space, it must be preceded by a single quote (’). For example:

```
123 ’33 is the answer
```

To make the message:

```
Forced Page Break
```

appear on the right-hand side of your screen, you should change the message in *uniplex.msg* to:

```
307 ,Forced Page Break
```

Chapter 8
Configuring Uniplex II Plus Applications

THIS PAGE INTENTIONALLY LEFT BLANK

Configuring the Word Processor

You use the system command file, *UAP/uniplex.sys*, to carry out the majority of the configuration for the Word Processor. Additionally, you can configure how the Word Processor is invoked.

Configure the Word Processor Parameters

You define the Word Processor parameters in the #SYSTEM section of uniplex.sys. For example, you can define the default page length and the dictionaries used by the spell checker. See the chapter Configuring System Parameters (uniplex.sys) for details.

Define the Preset Word Processor Rulers

You use the #RULERS section of *UAP/uniplex.sys* to define the 10 *preset* rulers (numbered 0-9) used to format text in the Word Processor.

You can configure this section to meet special formatting requirements such as extra wide documents. By default, Uniplex always displays ruler 0 when a user first edits a document. You can create rulers containing a maximum of 254 character positions. See the chapter Configuring System Parameters (uniplex.sys) for details.

Configure the Word Processor on Invocation

You can configure the way the Word Processor is invoked using command line arguments. See the section **uniplex** in the appendix Program Usage and Invocation.

Backup Files

You use word processor *backup files* to help prevent users from losing documents in the event of a system failure.

How Uniplex uses Backup Files

Each time you edit an existing document, Uniplex makes a copy in a file called **wp.back.up**. By default, this is located in your home directory. It always contains a copy of the document, as it was when you last saved it.

During the edit session, changes can be written away using the Save and Continue command (**ESC w**). These changes are always written away to the original document, not to the backup.

When you save and exit the document, all changes are written away to the original document. The next time you edit a document, a copy of it is written away, overwriting the old copy in **wp.back.up**.

You should use the Save and Continue command (**ESC w**) frequently to ensure that in the event of a system failure, the document will always contain all the changes you have made, up to the last **ESC w** you entered.

Word Processor Limitations

The maximum number of character columns that can be used on a line in a document depends on the number of "effect changes" (areas on the line that are effected with a differing effect from the preceding area), and is no greater than: 253 - (2* number of effect changes).

For instance, if a line containing the 3 characters "aba" with both "a"s in effect A, and "b" in effect "B", then the number of characters used in the buffer is 3 + 2*3 = 9. Also, line-drawing characters count as effects. Thus a single horizontal line across the full width of a 160-character-column uses 160+2 characters. But if the box contains 30 line-drawing "descenders" separated from each other by at least one character- column, then WP would need 160+60 bytes in its line buffer.

If you try to use more effect changes than can be supported (for example, by trying to create a table with many vertical lines), the Word Processor beeps.

Administering and Configuring File Manager

This section describes the following File Manager administration and configuration facilities:

- Keeping the Index in step with the UNIX file system.
- Handling of UNIX links and symbolic links.
- Describing multiple-pathed directories (for example, symbolically linked or on a network-mounted file system).
- Defining the work areas and folders accessible on the Open Folder dialog.
- Restricting File Manager access to parts of the UNIX file system.
- Controlling the size of filenames and some Index attributes.

Keeping the Index in Step with the UNIX File System

Files added or removed from the system without using Uniplex can result in missing or incorrect entries in the File Manager's Index database.

The File System Scanner utility, **ufmscan**, is provided to correct the Index entries for such files, and should normally be run regularly. See the earlier section, System Administrator Responsibilities, for suggested invocations.

ufmscan should normally be run when the system is inactive, but when all file systems containing indexed files are mounted. The mount directory of any unmounted file system should have its "Type" property set to "MOUNTABLE UNIX DIRECTORY" to stop **ufmscan** from removing index entries for its children.

Handling of UNIX Links and Symbolic Links

File Manager handles UNIX "linked" files and directories in the same way as the rest of Uniplex.

It has no special understanding of "hard" links (more than one name for the same file). Such files appear in folder lists in the usual way, but each link results in a separate (not linked) entry in the Index database (for example, each link has a separate Title).

Files that are UNIX "symbolic links" (functionally similar to "hard" ones, but allow directories to be linked, with machine-dependent results) will effectively be handled the same as "hard" linked ones.

Directories that are symbolic links should normally be declared using the "path prefix synonyms" feature described below.

Describing Multiple-pathed Directories

To avoid (often system specific) ambiguities of symbolically-linked directories, File Manager lets you define such linkages using the "path prefix synonyms" utility (**ufilepps**), described in the Program Usage section.

If you have symbolically-linked directories, you should use this utility to describe each link. Failure to do this will result in files underneath any such directory having separate Index entries (that is, one entry for each pathname to the file).

For example, if you decide to relocate directory **/usr/memos** to a different file system as **/disk2/memos** but use a symbolic link to retain the same **/usr/memos** access path, you should use the following command to describe this:

ufilepps -c /usr/memos /disk2/memos

Note: If you upgrade the Index database to a networked one, such as Informix-NET, you can use machine names to qualify pathnames for shared mounted file systems.

Define Work Areas and Folder Access Controls

You can use the file *UAP/filemgr/folder.ctl* to define the list of work areas available from the Open Folder dialog, and any restrictions on use of Change Folder (via the dialog and by using F5=Open on a folder name). This file is valid at both the central UAP area and the user's local one (which totally overrides a central one), but would not normally exist in the latter.

It consists of lines with three vertical-bar separated fields:

Flag ¦ *Directory* [¦ *Text*]

The fields are:

Flag One or more characters indicating what form of directory this line relates to. The first character can be one of:

H Disable access to this directory. The content of this directory is not normally displayed in File Manager. Such directories can can only be changed to using the "Named" option if available, and are always "blocked" (that is, the **H** flag implies the **B** one too).

If File Manager needs to display the contents of such a directory, it just shows the string **** No access to this folder ****, exactly as if the user had no UNIX read access to it. The Text field is not used on this line.

N The "Named" folder option is to be shown in the Open Folder list. The directory field is not used on this line.

F General purpose folder. The folder will only appear on the Open Folder dialog if the corresponding directory is readable and searchable.

T Template directory. The directory, and any subdirectories, are a Template area.

Note: even if you don't change the supplied definitions, you may want to create templates to suit your organization in one of the defined System Template areas (for example, $Unode/outlines/wp). For details, see the Uniplex Business Software User Guide Supplement.

P The "Parent" option (open parent of current file) is to be shown in the Open Folder list. The directory field is not used on this line.

X The "Home Folder" option is to be shown on the Open Folder list. The directory field is not used on this line.

Y The "Trashcan" option is to be shown on the Open Folder list. The directory field is not used on this line.

The following qualifying characters can be used with some of the main flags:

C Create-on-Open: (valid with a main flag of **T**) if the user opens the named directory, it will be auto-created if required.

B "Block": (valid with a main flag of **F**, **T**, **X** or **Y**) the user cannot change to the parent of this directory. In its folder list, the normal "../" string is replaced with the string **** No access to parent folder ****.

Directory The name of a directory. Can be blank if the *flag* does not require it.

Text Optional field containing the text shown on the Open Folder dialog.

Comments and blank lines can be included in the file. Comments consist of any characters from and including a '#' on a line. Blank characters (spaces and tabs) immediately preceding a '#' are also ignored.

The *Directory* field can include environment variables using the following shell syntax:

$*name*	The value of variable *name*
${*name***}**	The value of variable *name*
${*name*:=*word***}**	The value of variable *name* if it is set to a non-null value, otherwise the string *word*

The name on an **H** line must expand to the name of an existing directory, otherwise the File Manager will fail to start, with the error "Cannot read folder control file".

Note: User-dependent variables should be used with care.

If the file contains more than one line with a *Flag* of **N**, **P**, **X** or **Y**, the last one in the file is used. Where restrictions implicit in the rules conflict, the flags have precedence as follows (highest precedence first):

B
T
H
F, **P**, **N**, **X** and **Y**

Accessing the Index Database

The Index database is an Informix 5SE database named *$Unode/DATA/ufilemgr.dbs*. As with its predecessor "folios" database, it is writable by all users.

Provided your Datalink selection is set for Informix 5 ("Uniplex-supplied Informix Database" on the Alter Database Link form), you can examine the database schema and access its data using the Uniplex Database Forms and Query applications. These tools also enable you to update the database, but, other than changing field sizes with Database Forms, as described below, this should only be done under instruction from Uniplex.

To avoid casual access to the database, the *UAP/DATA* directory is installed with read and write access only to "root". This means that non-"root" users can only access it using Database Forms and Query by setting the DBPATH variable and invoking it by name.

The database contains three tables (whose fields are summarized in the file $Unode/ufilemgr/ufilemgr.sql):

paths Contains the full pathname of every indexed folder, along with the name of the machine from which this path is valid.

Includes the field, **native**, which is set to indicate whether this directory is permanently mounted, or not, by setting its "Type" property to "UNIX DIRECTORY" or "MOUNTABLE UNIX DIRECTORY".

documents Contains the filename of every indexed file and folder, its properties, and an index to the name of its parent folder in the **paths** table.

synonyms Contains the Path Prefix Synonyms managed by the **ufilepps** utility.

To gain a better understanding of the database, you can use the **-q** flag on the **ufilemgr** or **ufmscan** programs to request a log of all SQL statements generated in order to perform Index updates or accesses.

There are also two unsupported report scripts, in the UAP/unsupported directory, that give examples of producing statistical reports from the database:

```
sh $Unode/unsupported/ufmrep1
```

and:

```
ureport $Unode/unsupported/ufmrep2
```

Changing the Size of Fields in the Index

If required, you can add other fields to the database without affecting File Manager. You can also alter the size of the following character fields to save disk space or suit other site requirements:

Table	Field	Min	Default	Max	Used for
documents	filename	14	40	62	UNIX file (not path) name.
documents	title	1	100	255	Title text.
documents	keywords	10	255	255	Keyword/phrases (always in upper-case, separated by lower-case "s").
paths	machine	10	40	255	Machine name that this path is valid on. This field needs to be large enough to hold Uniplex host name (stored, by default in */etc/sysid*) for any system using this index.

File Manager automatically adjusts its operation to match the values set (the **filename** setting affects the naming of UNIX files only for users who have their **Use Index** preferences set to **Yes**).

Reducing the size of a field will just truncate the corresponding data in the index. For **filename** fields, this could destroy the link between the UNIX file and its Index entry.

*Note: Non-indexed files with names longer than the size of the **filename** field, but still no longer than its maximum value, can be seen and selected with File Manager. However, you may need to use the Properties form (scrolling the name field horizontally, if necessary). to see all of the name.*

To alter the size of one of these fields:

1 Make sure that no-one is using File Manager.

2 As "root", access the Index database from Database Forms.

3 Use the Amend Table Schema option on the Database/Table Administration form to set the new value(s).

The ufilemgr.rc File

This file contains additional File Manager configuration. It should not normally need modification and contains the following:

#DBCONFIG	This section contains one entry, **DBNAME**, which sets the name of the File Manager database (the string used on the SQL invoke command). The rest of the File Manager database interface description is contained in the **FILE_MANAGER** section of the *UAP/dbs/interfaces* file.
MAIL_COMMAND=*command*	Specifies the command executed by the File->Actions->Mail menu to mail a file.
MAX_COMMAND_SIZE=*nnn*	Specifies the maximum number of characters that the system's shell can have on a command line. If this is less than that required to invoke the PRINT or MAIL commands, they will report an error. If omitted, the File Manager assumes that the shell has no limit.
PRINT_COMMAND=*command*	Specifies the command executed by the File->Actions->Print menu to perform file printing.

License Controls

Unless the appropriate software license is installed, File Manager operates with restricted functionality. Please see the Release Notes for a list of these restrictions.

Configuring the Spreadsheet

You configure the spreadsheet using the file *UAP/uc/ucdefs* for ucalc mode or the file *UAP/uc/issidefs* for issi emulation mode. You must compile these files after changing them. Both files contain the same standard sections, differing only in default settings for the operating modes and date formats.

To compile these files, enter:

ucbld issidefs

or:

ucbld ucdefs

If the compilation is successful, Uniplex produces a compiled version of the command file, called *issi.rc* or *ucdefs.rc*.

Uniplex reports errors if the compilation is not successful.

File Format and Layout

The command files, issidefs and ucdefs, contain the following sections:

Section	Summary Description
#COMMANDS	Defines the words used to start a spreadsheet command. This is used primarily for compatibility with ucalc.
#KEYWORDS	Defines the words used to build the remainder of a spreadsheet command. This is used primarily for compatibility with ucalc.
#DEFAULTS	Defines the default state for a number of spreadsheet options, particularly modes.
#MAPS	Maps word processor type keystroke commands to spreadsheet type commands.
#FORMATS	Defines the available date formats and user definable numeric formats.
#HELP	Defines the special on-line help sections for the spreadsheet.

Define Spreadsheet Commands

The #COMMANDS section of issidefs and ucdefs contains one entry per line defining the words used to start spreadsheet commands. The syntax of each entry is:

program command=user command

where:

program command — is the word used internally in the spreadsheet program defining the action Uniplex will execute. You cannot change this.

user_command — is the word the user enters to execute the program command. You can change this.

For example:

```
CALC=c
```

where CALC is the internal program command to calculate the spreadsheet and c is the command the user enters to execute the command.

When the user enters a command, the spreadsheet program scans the #COMMANDS section of issidefs to find the longest leading string that matches. For example:

```
CALC=c
COPY=co
```

Entering	**c** **ca** **cal** **calc** **calculation**	All execute the CALC program command.
Entering	**co** **cop** **copy**	All execute the COPY program command.

Define Spreadsheet Keywords

The #KEYWORDS section contains one entry per line defining the words used to complete spreadsheet commands. The syntax of each entry is:

program_keyword=user_keyword

where:

program_keyword is the word used internally in the spreadsheet program. You cannot change these words.

user_keyword is the word the user will type in to complete a command. You can change these words.

For example:

```
ALL=all
```

where *ALL* is the internal program keyword and *all* is the user keyword.

Note: Unlike commands, the user must enter the keyword in full; it cannot be abbreviated.

Define the Spreadsheet Defaults

You use the #DEFAULTS section of the spreadsheet command file to define general operating defaults, as follows:

Keyword	Description
CALC=on ¦ off	Defines whether the spreadsheet defaults to automatic calculation mode on invocation or whether this must be explicitly selected by the user.
COLNAME=left-to-right ¦right-to-left	Defines the direction of the column names **left-to-right** is the normal English direction of reading. The **right-to-left** option is intended for use in languages that are read from right to left, such as Arabic.
COLS=*n*	Defines the maximum number of columns possible. Where *n* is in the range 100 to 4999. The default is 256.
COMPRESSED=on ¦ off	Specifies whether you want spreadsheets printed in compressed mode or in normal sized print. By default, Uniplex prints spreadsheets in normal sized print. Set to **on** to print in compressed mode.

Keyword	Description

CURSOR=*character(s)*	Defines how the cell pointer is displayed. A single *character* is interpreted as an effect letter; the effect is used to highlight the cell pointer. Two *characters* are interpreted literally as the left and right markers of the cell pointer.
	
	For example:
	
	`CURSOR=Z`
	
	This sets the cell pointer display to effect Z.
	
	`CURSOR=()`
	
	This sets the cell pointer display to ( ).

EFFECT=*character(s)*	Specifies the effect characters, where each *character* maps to the normal Uniplex print effects. The entry should include six characters. By default, the entry is: `EFFECT=AHCDEI`
LENGTH=*n*	Defines the length of a printed page. Where *n* is the length in lines of a printed page. By default, Uniplex uses a page length of 42 lines. (Uniplex prints spreadsheets in landscape mode by default.)
NEGZERO=**on**¦off	Defines whether small negative numbers, that are rounded down to zero, should be shown with a minus sign (**on**) or not (**off**).
NUMOVERLAP=**on**¦off	Defines whether numbers that require more characters than the current column width to display are replaced by a string of asterisks (see also the later section Define Spreadsheet Formats). The default setting allows wide numbers to extend as far as necessary into any available blank space to the right. If there is not enough blank space, the number is truncated.

Keyword	Description
PERCENT=*n*	Specifies the multiply factor, *n*, when values are formatted as a percentage. By default the value is set at 100.
ROWS=*n*	Defines the maximum number of rows possible. Where *n* is in the range 100 to 9999. The default is 1024 rows.
SCROLL=on ¦ off	Defines whether the spreadsheet scrolls in single columns/rows (on) or by screen (off).
STATUS=on ¦ off	Defines whether or not the status line is displayed. Switching off the status line speeds up performance.
STDROUNDING=**on**¦off	Defines whether to use Uniplex's predictable rounding algorithm (**on**) or the Operating System's own (as provided by the *sprintf* function). There should be no noticeable performance difference between the two. STDROUNDING=**on** may cause slightly different calculation results from the 7.01 version. It applies the conventional *round up on .5* logic (that is 1.5 to 2, 1.55 to 1.6, -1.5 to -2, and so on).
STRTOVAL=on¦off¦**warn**	Defines whether numeric values embedded in text are to be included in calculations. For example, if summing a range including a string of the form `Adjustment=10`, then setting STRTOVAL=**on** would use a value of 10 in the calculation, rather than 0. The default setting (**warn**) also generates warnings or errors whenever such a calculation is attempted. The warning text gives the address of the cell whose calculation is affected, not the one containing the offending text string. Any macro that runs into the same problem is aborted and the warning text gives the cell containing the line that the macro was aborted at. These warnings can be suppressed by setting the value to **off**.

Keyword	Description
TEXTOVERLAP=**on**¦off	Set this to **off** to suppress the default, whereby long text or date strings surround data in following cells. *Note: When setting to **off** users are best advised to increase the default column width from 6 to 8 to prevent automatic truncation of common data, such as date strings.*
UNDO=**on**¦off	Controls whether the UNDO operation is available. If not available (**off**), then any attempt to UNDO causes a beep. *Note: if UNDO=**on** and the spreadsheet runs short of memory, it automatically switches UNDO to **off** for this session only.*
UNDOSAVE=**on**¦off	Controls whether Worksheet Erase (**;new**) and File Retrieve (**;get**) are allowed. In order to be able to UNDO either of these commands, the Spreadsheet saves data to a temporary file (with name of the form $Utemp/UCSVxxxxxx). If the sheet is large, this can be slow and use a lot of disk space. Setting UNDOSAVE=**off** with UNDO=**on** disables the ability to UNDO these two commands. DECALIGN=right¦left Defines whether to align numbers to the **right** or left of a cell. The default is to the right. The **left** option is intended for use in languages that are read from left-to-right, such as Arabic.
VIEW=on ¦ off	Defines whether or not the spreadsheet displays the contents of the current cell on the status line.
WIDTH=*n*	Defines the width of a printed page. Where *n* is the width in characters. By default, Uniplex uses a page width of 112 characters. (Uniplex prints spreadsheets in landscape mode by default.)

Keyword	Description

WINDOW=*character(s)*	Defines how the window numbers are displayed. A single *character* is interpreted as an effect letter, the effect is used to highlight the window number. Two *characters* are interpreted literally as the left and right markers to enclose the window number. For example:
	
	`WINDOW=[]`
	
	The window number will be enclosed in 2 brackets.
	
	`WINDOW=Z`
	
	The window number will be highlighted with effect Z.

Define Spreadsheet Maps

You use the #MAPS section of the spreadsheet command file to map standard Uniplex keystroke sequences to spreadsheet commands. The syntax for a map is:

program_command=spreadsheet_command

where:

program_command is a command from the Uniplex command file uniplex.cmd.

spreadsheet_command is a spreadsheet command, combining commands, keywords and spreadsheet macros. For example:

```
F03={NC}kill @
```

F03 is the product-wide Uniplex command to delete a word. {NC}kill @ is a spreadsheet macro equivalent:

{NC}	no confirmation required.
kill	delete command.
@	the current cell.

The user will be able to enter the product-wide command F03 (CTRL w by default) to delete a cell from the spreadsheet.

Define Spreadsheet Formats

You use the #FORMATS section of the spreadsheet command file to define the available date formats and to enter user definable formats. Note that \s is equivalent to space (so that a space can be added visibly to the end of a line).

o Set a Date Format

Line 1 is a list of the valid day names for display in the spreadsheet.

Line 2 is a list of the valid month names for display in the spreadsheet.

Line 3 should be the default date format, as defined in the #SYSTEM section of the Uniplex system file, uniplex.sys.

Lines 3-18 can be used for definable date formats; these correspond to the date formats 1-16 that can be selected from within the spreadsheet at runtime.

The key characters in lines 3-18 are:

DAY	Day name (in full)
DY	Day name (abbreviated)
D or DD	Day number 1-31 (not zero-padded)
0D	Day number 1-31 (zero-padded)
MONTH	Month name (in full)
MON	Month name (abbreviated)
M or MM	Month number 1-12 (not zero-padded)
0M	Month number (zero-padded)
Y or YY	Last 2 digits of year number
YYYY	Year Number

For example:

0M/0D/YY	formats the date as 01/06/88
MM/DD/YYYY	formats the date as 1/6/1988

o User Defined Numeric and Currency Formats

Lines 20 to 36 are for user definable numeric formats; these correspond to formats 1-16 that can be selected from within the spreadsheet at runtime. The syntax for each line is:

[*xxx*:]**{***comparison value*[**A-F**][**c ¦ r ¦ l**]**}***text* #.##

where:

xxx:	optional prefix for currency formats (see below).
comparison value	is a spreadsheet macro test
A-F	is an optional effect letter (A to F)
crl	specifies the formatting: center/right/left justify
text	is any text
#.##	is the number

For example:

```
{>=0}# {<0}(#)
```

This example will put parentheses around negative values. If the number is greater than or equal to 0, do nothing; if the number is less than 0 it is displayed in parentheses.

```
{>=0}#{<0Ar} OD
```

This example will display negative numbers in bold (A) and right justified (r).

o Currency Formats

The Currency Format Prefix (up to 3 characters followed by a colon) on user formats defines the exact set of symbols recognized as Currency Symbols when pasting or importing data.

The leading or trailing status of the Currency Symbol (CS) is set by the first non-space character following the first close-brace (}) in the format specification itself. If the character is a hash (#), then the CS must trail the digits for the string to be recognized, otherwise it must lead them.

When scanning for the CS, if it is of leading type, then it must come:

- First in the string. Example: DM10, or
- Immediately following a minus sign. Example: -DM10 or
- Immediately following an open parenthesis. Example: (DM10)

If it is of a trailing-type, then it can only be either:

- At the end of the string. Example: 10 F
- Immediately followed by a close parenthesis. Example: (10F)

For example:

Format	CS position if Import/Paste is to be recognized as formatted currency
DM:{>=0}DM#{<0}(DM#)	Leading
ÖS:{>=0}#\sÖS{<0}-#\sÖS\s	Trailing

o Over-width Symbol (user-defined format 37)

This format can be used to change the default "*" character used when NUMOVERLAP is set to **off** and a number does not fit the cell width.

For example, to show over-wide numbers with >>>>>>, include the following:

```
37  >
```

Change the Default Interface Mode

When you invoke the spreadsheet, Uniplex sets the ISSI interface as the default interface for each user. You can change the default interface for particular users as follows:

1 Log in as the user who wants their default interface changed.

2 At the shell prompt, enter:

```
Ucalcmode=ucalc; export Ucalcmode
```

3 If the user wants to return to having the ISSI interface as default, reset the Ucalcmode variable as follows:

```
Ucalcmode=issi; export Ucalcmode
```

List File Processing

Before reading a list file, the current calculation mode is saved. While processing the file, no calculations are done unless a "calc" command is read, which is executed when found. Any "calc on" or "calc off" command in the file is ignored (they are also not written to list files). When the list file has been fully processed, the calculation mode is restored to what it was before reading the list file. If the mode is "auto", then the sheet is calculated.

UNDO Facility

If you enter a spreadsheet command by mistake, Uniplex now provides the facility to undo the last operation or command. For example, if a row is deleted the row can be retrieved by undoing the delete command.

The UNDO facility operates on all input and data change commands including import and consolidations, specifically on:

Data cell creation
Formula cell creation
Cell deletion using space-bar
Cell deletion using Range Erase
Change cell value
Change cell formula
Move cells
Copy cells
Insert rows/columns
Delete rows/columns
Erase Worksheet (see notes 1 and 2 below)
File Retrieve (see note 2) - entire file (see note 1) or part file
Data Fill
Data Zero
Data Initialize
Data Sort
Paste
Range Format changes
Column width changes
Creation or deletion of range names (labels)
Graph links
Undo UNDO (see note 1 below)

1 In these cases, UNDO or UNDO-UNDO do not work if the result is an empty worksheet. This is because, in these cases, UNDO operates by creating a temporary spreadsheet save file. If the worksheet is empty this is not done and so there is nothing to UNDO. Instead, you can just use /Worksheet Erase to get the empty worksheet.

2 To cut down system load, these two operations can be disabled independently of other UNDO capabilities (see the UNDOSAVE flag in the earlier section Define the Spreadsheet Defaults).

If a macro or list file was the last command run, then only the last command within the macro or file is undone, not the entire operation.

If UNDO runs out of memory to save information, it turns itself off and any further attempts to UNDO result in a beep.

When entering data into a series of cells, only the last changed data cell is restored by an UNDO.

Configuring Database Query

Configuring the USQL Commands

USQL commands are defined in the #KEYWORDS section of the file *UAP/dbs/dbs.rc*. Each line of this file defines one word of the language. They are in the format:

KEYWORD=configurable keyword

Where:

KEYWORD	is the USQL command.
configurable keyword	is the word the user enters to action the USQL command.

The KEYWORDs in the file are in alphabetical order and must be kept this way. Do not change the KEYWORD. You can change the configurable keywords.

For example, you want the configurable keyword *retrieve* to perform the same function as the select KEYWORD. Make the following entry, in the correct alphabetic position in the file:

SELECT=retrieve

Now you can use the retrieve to perform selects on the database. You do not need to delete existing configurable keywords from the file to add new ones.

You must not use any of the words in this file for database, table, column and index names.

To restrict access to specific KEYWORDs, for example *drop database*, you can delete them from *UAP/dbs/dbs.rc* local to the user.

The spreadsheet and database use the invoke KEYWORD. Configurable keywords can be added which will perform the same function as the invoke KEYWORD, enabling this KEYWORD to be translated and configured, but this line should never be removed.

The #DB_CONFIG section of the file *UAP/dbs/dbs.rc* contains the line:

BANG=!

This line defines the character to precede a shell command when using database query. You can change the exclamation mark (**!**) to any other character.

For example:

```
BANG=@
```

would change the shell command character to **@**. To prevent shell escape, enter:

```
BANG=
```

Configuring the USQL Error Messages

There are two files containing USQL error messages: *UAP/dbs/dbs.msg* and *UAP/usql/usql.msg*. You can configure these files in the same way as any other message file (see the chapter Help and Message Configuration for details).

The file *UAP/dbs/dbs.msg* contains error messages specific to INFORMIX and the file *UAP/usql/usql.msg* contains local error messages. You cannot configure ORACLE error messages.

If the error message to be displayed is an ORACLE or INFORMIX message, then Uniplex will precede the message with one of the following statements:

`Oracle Error:`*message*

`Informix Error:`*message*

Datalink

Uniplex Datalink is a suite of interfaces to enable you to exchange information between third party database systems such as Oracle and Uniplex UBS.

Uniplex Datalink interfaces to other databases so that information can be read directly into or updated by Uniplex applications, including accessing data that is spread across a number of remote systems. Distributed databases such as Informix-OnLine are supported allowing you to use your corporate or personal database while still using the familiar interface of Uniplex applications.

Uniplex comes supplied with two databases; Informix-SE 2.10, referred to as the 'Uniplex Database' and Informix-SE 5.10, referred to as the 'Uniplex-Supplied Informix Database.

They are installed in a standard Informix hierarchy under UAP/informix. For instance, the Informix 2.1 version of "bcheck" is installed in:

```
UAP/informix/2.1/bin/bcheck
```

and the Informix 5 Standard Engine in:

```
UAP/informix/5SE/lib/sqlexec
```

To use the installed versions of Informix outside the Uniplex environment you will need to manually set the environment variables INFORMIXDIR, SQLEXEC and PATH. For example:

```
INFORMIXDIR=/UNIPLEX_NEW/UAP/informix/5SE
PATH=$INFORMIXDIR/bin:$PATH
SQLEXEC=$INFORMIXDIR/lib/sqlexec
```

When using Uniplex, these settings are set from the Datalinks UAP/dbs/interfaces file. In most circumstances this file will not need to be changed.

To switch from using Informix-SE 2.10 to use Informix-SE 5.10:
From the main Menu, select 'A - Administration' then 'I - Software Installation' and select the option 'Change Database Link'.

CAUTION: Using the 5SE engine to access a database created by 2.1, and vice-versa, can corrupt the database. Therefore, unless you are sure which database you will use and which engine was used to create it, DO NOT CHANGE YOUR DATABASE LINK BETWEEN THE TWO.

Interface Files

The file *dbs/interface.idx* lists the Datalinks available on this particular system.

The file *UAP/dbs/interfaces* contains details of all available database links; there should be no need to alter it unless you want to use an external database (one not supplied by Uniplex) supported by Datalink. Refer to the file for entry examples. Each section in the file contains a number of keywords: some are database specific and some are generic.

The following table describes the labels contained in each section.

Entry	Explanation
DATE	(Interface-specific) Specifies an extended date format as used by USQL. This tells uform how to format the date before it passes it to the underlying interface. If not specified, then the DATEFMT string from the #SYSTEM section is used in preference. DATE should be specified on a per-interface basis and should be set up like a format column n date "string" command in USQL. An example would be: **DATE='DD MONTH YYYY'** See the information below for specific details of DATE usage in each database interface. *Note: Dates output by a database are converted to a Uniplex date format string by both database forms and database query before display.*

Entry	Explanation
DBLIST	indicates that this link can provide a list of database names. That is, it can contain multiple databases.
DBADMIN	indicates that this link allows the statements **create database**, **rename database** and **delete database**.
DBSERIAL	indicates that this link allows use of the **serial** data type.
DBINTERVAL	indicates that this link recognizes the **interval** data type, although it is not actively supported. (Informix-specific)
DBDTIME	indicates that this link recognizes the **datetime** data type, although it is not actively supported. (Informix-specific)
DBYTE	indicates that this link recognizes the **byte** data type, although it is not actively supported. (Informix-specific)
DBTEXT	indicates that this link recognizes the **text** data type, although it is not actively supported. (Informix-specific)
DBVARCHAR	indicates that this link recognizes the **varchar** data type, although it is not actively supported. (Informix-specific)
ENV	Sets an environment variable and supports $NAME syntax to represent existing environment variable values (Note: no other shell-like syntax, such as ${NAME-default}, is supported).

Entry	Explanation
FORCENV	Same as ENV, but, whereas ENV only sets missing variables, FORCENV ignores any existing value of the variable.
INFNET	indicates you are running Informix-Net. (Informix-specific)
INFTURBO	indicates you are running Informix-Turbo. (Informix-specific)
INFONLINE	indicates that you are running Informix 4 On-Line. (Informix-specific)
II_SYSTEM	Sets the Ingres database directory path (Ingres-specific)
II_AUTHORIZATION	Set if using an evaluation copy (Ingres-specific)
II_DATE_FORMAT	Format dates in Ingres-expected formats (Ingres-specific)
INGRES	(Ingres-specific)
INGRES_HOME	(Ingres-specific)
LABEL*n*=*text*	specifies the field label. Where *n* is the field number and *text* is the title which appears on the screen. The maximum number of labels is three.
LENGTH*n*=*a*	specifies the field length. Where *n* is the field number and *a* is the length in characters. OPTIONAL*n* specifies that the field is optional. Where *n* is the field number.
NOPREV	indicates that this link does not provide support for the **previous** command in Database Forms.
ORADATE	Format dates in Oracle DD-MON-YY format. (Oracle-specific)
ORACLE_BASE	(Oracle-specific)
ORACLE_HOME	Sets the Oracle database directory path.(Oracle-specific)
ORACLE_SID	Sets the name of the Oracle server (Oracle-specific).
ORACLE_TERM	(Oracle-specific)
ORACLE8	(Oracle-specific)
ORACLE8_SID	(Oracle-specific)
ORAENV_ASK	(Oracle-specific)
OPTIONAL *n*	specifies that the field is optional. Where *n* is the field number.

PROMPT*n*=*text*	specifies the link program to run.
TITLE=*screen_title*	specifies the user prompt when the user moves onto this field. Where *n* is the field number and *text* is the prompt message.
SECRET*n*	specifies the title line for the screen.
SELECTAS	is not to be displayed. Where *n* is the field number.
	Defines the name that the user sees when selecting a link.

Configuring Database Table Information Commands

As part of processing a **describe database** (from USQL) or **select table** or **list tables** (from UFORM) command, an SQL query is sent to the back-end engine.

This query returns a number of columns of data depending on the interface (usually 3 - table name, table type and table index).

The default query is built-in to each Datalink interface, but can be configured externally by creating the file UAP/dbs/<*program*>.ds, where <*program*> is the name of the Datalink interface binary (for example: oraclev6_if, or inf210_if).

This file can contain one or more lines of (native engine) SQL in plain ASCII text. If the back-end engine does not like new-line characters, these can be "hidden" by putting a hyphen (-) as the last character of a line (as in USQL's line continuation syntax).

Parameter Substitution

In addition, the following strings are recognized as requests for dynamic parameter substitution:

[LOGNAME]	User's UNIX login name (same as $Uusername)
[DATABASE]	Currently selected database, if relevant
[DBLOGNAME]	Database login name (Oracle only).

If not available or relevant, these strings are just deleted, as described in the following table:

	Informix	Oracle	Ingres
[LOGNAME]	Substituted	Substituted	Substituted
[DATABASE]	Substituted	Deleted	Deleted
[DBLOGNAME]	Deleted	Substituted	Deleted

Comments

Any line with a first character of hash (#) is treated as a comment. All other lines, including blank ones, are simply passed to the back-end, after any parameter substitution and "-"/newline stripping.

Statement continuation can span comments. For example:

```
SELECT this -
# Comment here does not stop the statement
# "SELECT this from that" being passed to the back-end.
from that
```

Example

To illustrate, and to show the default used in the Informix Datalink interface, the following is an example of what you could put in this file without altering current functionality:

```
Informix interfaces
# UAP/dbs/inf210_if.ds
# (identical file can be used on all Informix interfaces)
# Following is exactly what is performed if this file does not
# exist.
# Select 3 columns:  table name, table type and index name
# Engine requires one newline-terminated string.
SELECT tabname,tabtype,idxname FROM systables, -
OUTER sysindexes WHERE systables.tabid >= 100 AND -
systables.tabid = sysindexes.tabid
```

Informix Administration

This section describes the differences between Informix and Uniplex's standard database management system which affect the administration of the database.

Set DBPATH and DBTEMP

If you use Informix-based databases, you may need to set the environment variable DBPATH to save changing directory to locate different databases.

If you make complex selects requiring more disk space than is available in */tmp*, you will need to set DBTEMP to the name of a directory on a larger file system.

For more details of this variable, see the chapter The Uniplex Environment.

Recover Data

You can only use recover data commands on your current machine. You cannot issue them for a remote machine. To recover data on a remote machine you must be logged on to that machine.

Date Formats

The DATE= keyword is required for Informix interfaces and controls how Uniplex will set the Informix DBDATE variable before invoking the Informix engine.

DATE (via DBDATE) controls the valid date syntax formats accepted by USQL; it has no effect on the Database Forms user interface. DATE should be set to a string in Uniplex format that matches the national one. For example:

DATE=DD/MM/YY	passes DBDATE=DMY2/ to Informix
DATE=MM/DD/YYYY	passes DBDATE=MDY4/ to Informix

You should use four Y's if you want to enter 4-digit years. Versions before Informix 5.10 always assume 2-digit years are in the range 1900-1999. Informix 5.10 will interpret the date according to the DBCENTURY environment variable. If DBCENTURY is not set, it copies the first two digits from the current date, and so it now assumes the range 2000-2099. (The DATEMODE century change flag has no effect on USQL dates.)

Oracle Administration

This section describes the differences between Oracle and Uniplex's standard database management system which affect the administration of the database. It covers the following:

Oracle V8.0
Oracle V6.0
Oracle SQL*Net

Note: The version(s) of Oracle supported by Uniplex may vary from platform to platform.

Oracle administration differences are:

Audit Trail

Oracle uses audit trails to monitor such activities as users connecting to the database or statements requiring Database Administrator privilege. Oracle generates audit trails differently. For more details see the chapter on database security in your Oracle Administrator's Guide.

Environment Variables

Oracle uses the following environment variables instead of **DBPATH**:

- **ORACLE_HOME**, which sets the database directory path. To set ORACLE_HOME enter the following in your **.cshrc** file:

  setenv ORACLE_HOME '*pathnamelpathname...pathname*'

- **ORACLE_SID**, which sets the name of the Oracle server. To set ORACLE_SID enter the following in your **.cshrc** file:

  setenv ORACLE_SID '*server_name*'

If you use more than one database it is useful to also set these variables in the **interfaces** file. This causes the database directory path and server name to be reset automatically whenever you change the database you are using to Oracle.

For more details about setting environment variables, see your Oracle Administrator's Guide.

Recover Data

Oracle uses a different method for recovering data. For details see your Oracle Administrator's Guide.

Recover Table

Oracle uses a different method for recovering tables. For details see the Recovery chapter in your Oracle Administrator's Guide.

Setting Database Privileges

The grant statements are used to provide users with access privileges once the database has been created. To do this, use the following syntax:

grant *privilege* **to** *name* **identified by** *password*

where *privilege* can be one, or a combination of the following:

- **connect**
- **resource**
- **dba**

Setting Table Privileges

You can only use the **grant select** statement on entire tables. Oracle does not support this statement on individual columns.

Transaction Log Files

Oracle refers to a transaction log file as a re-do log file. This file is generated automatically and is used when you are recovering your database. For details, see the chapter on database recovery in your Oracle Administrator's Guide.

Oracle Configuration

This section describes the differences between Oracle and Uniplex's standard database management system which affect the configuration of the database.

Error Messages

You cannot configure Oracle error messages. When an error is displayed, Uniplex precedes the message with the statement:

```
Oracle Error:
```

Date Formats

Oracle can only understand dates in the input format DD-MON-YY where the month name must be a short name in uppercase and in English. Because of this, the DATE keyword is not useful as part of the Oracle interface configuration. Instead, the keyword ORADATE is used which tells uform to always send dates to the interface in the format DD-MON-YY, uppercase English.

When using database forms with Oracle, users must enter dates according to the Uniplex date format. This is validated by Uniplex before being passed on to Oracle in a format it understands (DD-MON-YY).

When using database query, users must enter dates according to DD-MON-YY. Failure to do so results in an invalid date error from Oracle.

Ingres Administration

This section describes the differences between Ingres and Uniplex's standard database management system which affect the administration of the database.

Audit Trail

Ingres uses a table journal to monitor activities on individual tables in the database. To record journal entries you need to set journal recording for the entire database as well as setting it for the particular table. For details see your Ingres/SQL Reference Manual.

Copy a Table

Ingres uses a different method to copy tables. This means you cannot copy a table.

Creating and Deleting Databases

In Ingres you use the:

- **createdb** command from the shell to create a database.
- **destroydb** command from the shell to delete a database.

For more details, see your Ingres/SQL Reference Manual.

Environment Variables

Ingres uses the following environment variables instead of **DBPATH**:

- **II_SYSTEM**, which sets the database directory path. To set II_SYSTEM enter the following in your **.cshrc** file:

  setenv II_SYSTEM '*pathnamel pathname ... pathname*'

- **II_AUTHORIZATION**, which you set if you are using an evaluation copy. To set II_AUTHORIZATION obtain the environment key from your System Administrator and enter the following in your **.cshrc** file:

  setenv II_AUTHORIZATION '*environment key*'

  If you use more than one database it is useful to also set these variables in the **interfaces** file. This causes the database directory path and environment key to be reset automatically whenever you change the database you are using to Ingres.

  For more details about setting environment variables, see your Ingres/SQL Reference Manual.

Recover Data

Ingres uses a different method for recovering data. For details see your Ingres/SQL Reference Manual.

Recover Table

Ingres uses a different method for recovering tables. For details see your Ingres/SQL Reference Manual.

Renaming a Table or Database

When you use Ingres you cannot rename:

- a column.
- a table.
- a database.

Transaction Log Files

Ingres uses a journal for the entire database as a transaction log file. For details see your Ingres/SQL reference Manual.

Ingres Configuration

This section describes the differences between Ingres and Uniplex's standard database management system which affect the configuration of the database.

Error Messages

You cannot configure Ingres error messages. When an Ingres error message is displayed, Uniplex precedes the message with the statement:

```
Ingres Error:
```

Date Formats

Dates output by the Ingres database engine are output as strings in a format controlled by the environment variable II_DATE_FORMAT. Since the dates output by Ingres need to be understood by Uniplex's Ingres interface program, it is important that Uniplex forces the Ingres database engine to output dates in a known format.

By default, Ingres can understand any of three different date input formats - dd-mmm-yyyy, mm/dd/yy or mmddyy. However, if the II_DATE_FORMAT variable is set, for forcing a particular date format on output, then this may change the allowable input formats.

Configuring dates for Ingres is handled as follows:

- The DATE keyword in the interfaces file is set to one of two values: DD/MM/YY or YY0M0D. Nothing else is allowed and the Ingres database interface complains if it finds a different setting.

- If DATE is set to DD/MM/YY, then database query users can enter dates as dd/mm/yy, or dd-mmm-yyyy, or mmddyy. The Ingres database interface, seeing DATE as DD/MM/YY, sets the environment variable II_DATE_FORMAT to be MULTINATIONAL, overriding any previous setting if necessary, which forces Ingres to output dates as DD/MM/YY strings which are converted and then displayed correctly in the appropriate Uniplex front-end program, database forms or query.

  The above setting of DATE is recommended for English configurations since it allows dates entered in database query to be DD/MM/YY format.

  The alternative setting for DATE is YY0M0D. If set to this way then the Ingres database interface explicitly sets the environment variable II_DATE_FORMAT to be ISO. This setting means that users of database query need to enter dates as mm/dd/yy, or dd-mmm-yyyy, or yymmdd.

  Regardless of the setting of DATE, database forms users must always enter dates according to the Uniplex date format specified in #SYSTEM.

INGRES does not support date functions such as **date** (*value expression*).

Chapter 9
Configuring and Administering
Uniplex Business Software Electronic Mail

THIS PAGE INTENTIONALLY LEFT BLANK

Configuring and Administering Electronic Mail

Note: This chapter refers only to the Uniplex Business Software Electronic Mail system. For details of the onGO Character Client Mail system, refer to the onGO Character Client user and administration guides.

You can set up Electronic Mail to send to:

- Users on the same computer system who use Uniplex mail.
- Users on the same computer system who use other mail systems.
- Users on other computer systems who use Uniplex mail.
- Users on other computer systems who use other mail systems.

In addition, you can set up different ways of sending mail to the same system. For example, you can set up a method of sending mail immediately, or saving it to send in a batch when machine time is cheaper.

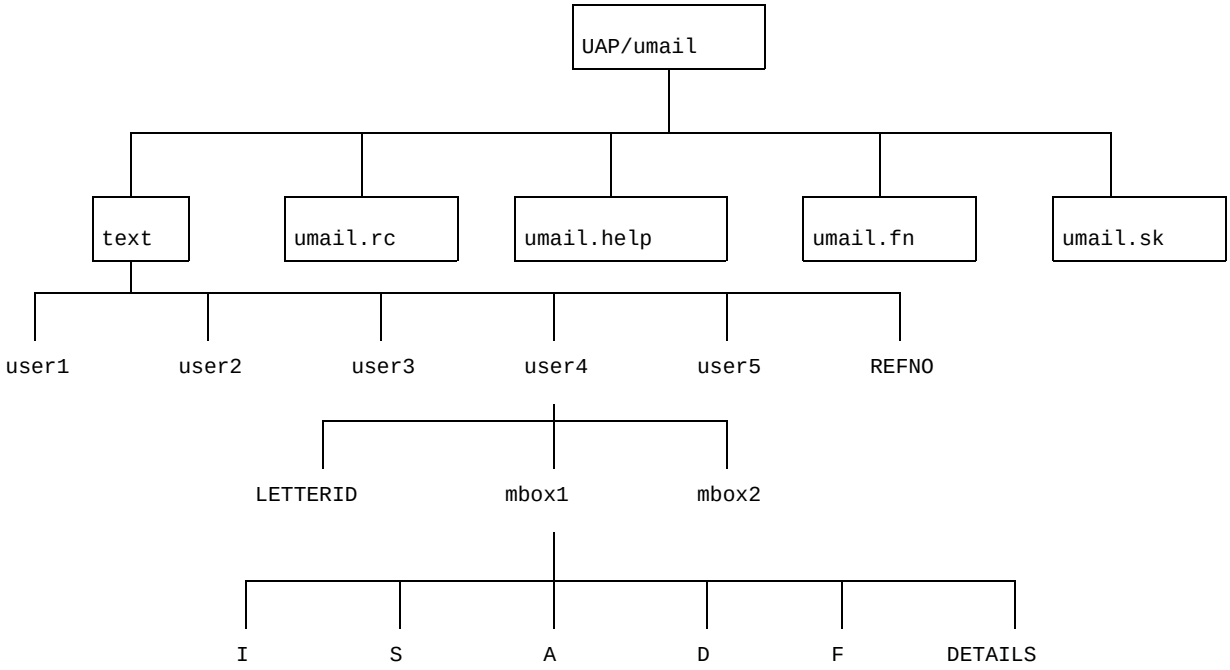
The System Administrator can change the way that mail is set up at an individual site.

This chapter explains how to set up systems to send and receive mail and how to configure the way mail works, using the file UAP/umail/umail.rc.

The IMAP server (**uimap**) allows IMAP clients to access the Uniplex mail system. This is enabled by entries in the system files /etc/services and /etc/inetd.conf. The only part of the umail.rc file specifically for the IMAP server are the keywords UBSABOX, UBSDBOX and UBSFBOX.

Electronic Mail File Structure

The following diagram represents the Electronic Mail file structure:



These files and directories contain the following information:

File or Directory	Explanation
UAP/umail	The umail directory.
text	Directory which contains the mailboxes for each user.
umail.rc	Contains umail system information. See the section Using umail.rc.
umail.help	Contains the help text used by Electronic Mail.
umail.fn	Contains the softkey information used by Electronic Mail.
umail.sk	Contains the compiled softkey information used by Electronic Mail.

File or Directory	Explanation
REFNO	Contains a reference number for the next letter generated on this system (used for retrieve and auto-forward).
user1...	This is a directory where mail for each individual user is stored (using their UNIX login name).
LETTERID	Contains the serial number to be used in creating the next letter in any of the user's mailboxes.
mbox1	Sub-directory containing a user-assigned and named Uniplex mailbox. Each user can have more than one mailbox. mbox1 always exists.
I	Sub-directory for mbox1 or other Uniplex mailboxes. Contains incoming mail.
S	Sub-directory for mbox1 or other Uniplex mailboxes. Contains a saved copy of all sent mail.
A	Sub-directory for mbox1 or other Uniplex mailboxes. Contains a copy of all archived mail.
D	Sub-directory for mbox1 or other Uniplex mailboxes. Contains sent letters. Copies are not saved.
F	Sub-directory for mbox1 or other Uniplex mailboxes. Contains future mail (letters to be sent at some point in the future).
DETAILS	File for mbox1 or other Uniplex mailboxes. Contains a summary of the information contained in the mailbox. See the section, The DETAILS File for more information.

Note: Mail also uses the following unconfigurable program files: uimap, umailexec, umd_clean, umd_runix, usmail. These files are held in UAP/bin. Furthermore, the file UAP/uniplex.alias defines real names for users and sets up system mailing lists.

The DETAILS File

Each named mailbox contains a DETAILS file which summarizes information in that particular mailbox. The file contains 11 lines of ASCII text, arranged as follows:

Line	Content	Detail
1	Mailbox name	User assigned name, or blank for default mailbox.
2	Auto-forward list	Blank if no auto-forward set. Otherwise contains a list of user names to auto-forward to.
3	Auto-reply file	Blank if no auto-reply set. Otherwise contains the name of the file containing the auto-reply message.
4	Number of letters in incoming message mailbox.	Number of files in the I sub-directory.
5	Number of the above that are urgent.	Number of files in the I sub-directory that are marked urgent.
6	Number of letters that are unread.	Number of files in the I directory that are still unread.
7,8	Date and time last message received.	Contains the date and the time the last message was received in UMAIL internal format.
9	The name of the system	Takes the name of the system from the uniplex.sys NODE parameter.
10	User name	Displays the name of the higher level directory. This is normally the user name. For example, if the DETAILS file is in the directory: UAP/umail/text/fcm/mbox1 then this line contains fcm.
11	Mailbox number	Displays the number of the mbox directory.

*Note: You can set auto-forward without save for a mailbox either by using the **umd_migrate** command or by adding the string "NOSAVE:" to the front of the system name in a mailbox's DETAILS file.*

Using umail.rc

The system file UAP/umail/umail.rc contains seven types of entries:

Entry	Description
Keywords	These define how mail is set up for a particular site. Some of the keywords define the defaults for the various option fields on the lower half of the send mail form. *Note: If you are using a supported translation of Uniplex, the defaults for the MAILBOX keywords (DFLTMBOX, MAILBOXES, TRASHMBOX, UBSABOX, UBSDBOX, UBSFBOX) may have been localized.*
#SYSTEM	You need to set up a #SYSTEM entry for each remote computer system to which you want to send mail, and for each different method of sending mail.
#EASISEND	You need to set up a #EASISEND entry if you want to define the defaults to use for sending messages with the EasiSend option.
#PICKLETT	You need to set up a #PICKLETT entry if you want to configure the default widths used in the display fields for incoming mail and sent mail.
#PRINT	You need to set up a #PRINT entry if you want to configure the way a message header will be printed.
#FORMAT	You need to set up a #FORMAT entry if you want to use a mail system other than UNIX or XENIX.
#UMD_RUNIX	You need to set up a #UMD_RUNIX entry to tell umd_runix what the UNIX mailbox will look like.

The Reply To Sender Form is not configured using umail.rc. It can be configured by changing messages 1240-1261 in UAP/umail/umail.msg.

Mail Keywords

You can define the following keywords in umail.rc. The first list contains the keywords that define the defaults for the various option fields at the bottom of the Send Mail form.

Use the following syntax to define keywords:

Either *keyword* = *setting*
or: *keyword* = *list*;
or: *keyword* = **yes ¦ no**

where *list* is one or more comma-separated words.

Send Mail Form Keyword	Description
SAVECOPY=yes¦no	Defines whether Uniplex saves a copy of the message you are sending. Uniplex saves the copy in UAP/umail/text/*user*/mbox*n*/S.
REGISTERED=yes¦no	Defines whether Uniplex sends verification when a remote machine receives sent mail.
VERIFY=yes¦no	Defines whether Uniplex sends verification when the recipient has read the mail on a remote machine.
ENCODE=yes¦no	Defines whether Uniplex encodes remote messages in transit to a remote machine.
CONFIRM=yes¦no	Defines whether Uniplex requests that the user confirm that he or she has read the received mail.
PRIORITY=*setting*	Defines the priority with which to send the mail. See the description of PRIORITY in the later section Defining Mail Routes to Other Systems (#SYSTEM) for details.
NOTIFY=*setting*	Defines the number of days after which Uniplex sends non-read notification.

The remaining keywords are:

Keyword	Description
ADMIN=*name*	Where *name* is the name of the Uniplex mail administrator, if different from the name given by ADMIN in uniplex.sys. If no ADMIN here or in uniplex.sys the use of "root" is assured.
ALARM_DELIMITERS=*string*	Defines a string of characters which are used if "You have new Mail" notification information has to be truncated. Each character represents a "special" character which, if contained within the Senders address, constitutes a point at which the address may be truncated. Example setting: `ALARM_DELIMITERS = .%@!/[]^:`
BRNAME=*command*	Defines the browser invoked when a foreign (non-Uniplex) file is encountered. *Note: Uniplex Mail invokes the configured browser with all files regarded as foreign. The browser will, therefore, have to cope with all non-Uniplex, spreadsheet or RGIP files.*
BROWSER=*yes* ¦ *no*	Defines the default browser. Set to YES or **NO**. If set to NO, the BRNAME-defined browser is invoked when Uniplex Mail encounters a foreign file. If set to YES, the BRNAME-defined browser is invoked in preference to the default mail text browser, regardless of the file type.
COMMBUF=*size*	Defines the default command buffer size. The buffer holds the expanded COMMAND line as defined in the relevant #SYSTEM section. This is passed to the UNIX shell for execution when sending remote mail. The default size is 255 bytes. This can be increased or decreased as required.

Keyword	Description
DAYS=*n*	Defines the period of time for which checked statuses are kept.
DEFSYSTEM=*system_name*	When set, this is the default system to send Mail messages to. If it is set to *mail* (where *mail* is the name of the default mail system) then all mail is passed to this mail system, taking advantage of underlying configuration information such as aliases and E-mail routes. This is particularly useful for Sendmail users. For example: **DEFSYSTEM=***mail* **#SYSTEM;** **NAME=***mail***;** **PRIORITY=123;** **ADDRESS=[USER];** **COMMAND=/usr/lib/sendmail [TO]<[LETTER];** **FORMAT=***unixmail***;** **))** If this facility is set up as described above, the *uniplex.alias* and the *umail.rc* files need not be changed when aliases, mail routes, machine names, etc. are changed. This is because, using this facility, no alias expansions are done by Uniplex and address resolution is assumed to be handled by the underlying mail system.
DELIMITER=*char char*	Defines the delimiter characters around keywords in the umail.rc file. These are shown as square barckets in this guide.
DFLTMBOX=*name*	Defines the name of the default mailbox where incoming messages are received. For English versions of Uniplex, by default, this is **general**.
EDITOR=*yes* ¦ *no*	If set to 'yes', the full word-processor is used for creating messages.

Keyword	Description
EDNAME=*name*	Where *name* is the name of another editor the user can specify for creating memos instead of Uniplex.
FIFO=*yes* ¦ *no*	Incoming messages are displayed with the latest at the top, by default. Setting this keyword to 'yes' displays oldest messages at the top (First In First Out).
FPRINT=*command*	Command to print WordPerfect files. A sample shell script, **wp.print**, is supplied by Uniplex to interface to the WordPerfect editor and print mechanism. The script is located in $Uniplex/cmds.
GROPTS=*options*	When printing RGIP files, Uniplex Mail constructs a Uniplex file containing a **.GR** command. The default .GR command is **.GR filename r 70 70**. To change the options **r 70 70**, use the GROPTS keyword, for example, **GROPTS=r 50 50**.
LOCALSYS=*list*;	Reserved.
LOCKMODE=0	Enables an earlier style of locking when updating Uniplex mailboxes. Use this keyword under direction from Uniplex.
MAILBOXES=*list*;	Defines the mailbox names that Uniplex recommends you use in order to organize the sending of mail within an organization. For English versions of Uniplex, by default, these are: **calendar**, **personal**, and **business**.
MBEXCLUDE=*list*;	Defines the mailboxes that umd_runix does **not** include when it runs. (umd_runix is the background program used for reading UNIX mailboxes, see later in this section for details.) The exclusions are provided for users, for example root, who would normally use UNIX mail in preference to any other.
NOLOCAL=*yes* ¦ *no*	If set to 'yes', the local user names from /etc/passwd are not included on the popup list of users. The default is 'yes'.

Keyword	Description
OPTCOPY=*yes* ¦ *no*	Disable copying. Can be set to **YES** or NO. If set to NO, the user does not have the option to copy the letter using the command menu.
OPTPRINT=*yes* ¦ *no*	Disable printing. Can be set to **YES** or NO. If set to NO, the user does not have the option to print the letter from the command menu.
POSTIE=*username*	Defines the name of the local dispatcher. It defaults to u-mailer.
PRINT=*program*	Defines the command which is executed when the print option is selected from the ring menu.
REVALIAS=*yes* ¦ *no*	If set to 'yes', Uniplex attempts to convert usernames into more meaningful text. The default is 'yes'. *Note: To successfully interpret usernames from the alias file, the alias must include a machine name.*
RMFILTER=*program*	Runs the script or program that processes the mailbox before umd_runix reads it.
SMODE=*string*	The characters **P**, **J**, **N** and **R** may be used in the SMODE string. **P** indicates that Uniplex Mail produces new format forwarded messages. Note that if WordPerfect is to be used as the standard editor/browser then option **P** must be set in the SMODE string. **J** indicates that Uniplex Mail attempts to combine comments with the letter details of a forwarded message (if in ASCII format). This is to avoid having to include the comments option on the command menu when reading a forwarded message. **N** indicates that Uniplex Mail strips effects and hard returns from text prepared in the Notepad when it is saved.

Keyword	Description
SMODE (*continued*)	**R** indicates that the Notepad uses the right margin as set on your default ruler as its right margin on screen.
SPLITINC=*yes* ¦ *no*	If set to 'yes' causes incoming mail to be displayed in three sections; Urgent, Unread and Read mail.
SPLITOUT=*yes* ¦ *no*	If set to 'yes' outgoing mail is displayed in two sections; Saved and Unsaved copies.
SRCHBYTIME=*integer*	This keyword controls the Time functionality of the Uniplex Mail search mechanism. The default negative value disables the functionality. A value of zero enables the time field in the search form. A positive value is a rounding factor in minutes. For example a value of ten, rounds times entered to the nearest ten minutes. This permits a greater granularity to the search logic.
SUBJECT=*yes* ¦ *no*	If set to 'yes', the Subject field is mandatory when sending mail.
TRASHDAYS=*n*	Defines the length of time deleted messages are retained in the trashcan mailbox.
TRASHMBOX=*name*	Defines the name to be used for the trashcan mailbox. For English versions of Uniplex, by default, this is **trashcan**.
TRDATE=*yes* ¦ *no*	If set to 'yes', the format of the date displayed on Incoming and Outgoing mail pick and point screens is truncated to mm/dd or dd/mm (the year is left off). Default is 'no'.
UBSABOX=*name*	Defines the name that the IMAP server gives to its archived folder. Each mailbox will have an entry within this folder.
UBSDBOX=*name*	Defines the name that the IMAP server gives to its folder for mail sent via the UBS system. Each mailbox will have an entry within this folder.

Keyword	Description
UBSDBOX (*continued*)	*Note: This does not hold mail sent from the IMAP client which is usually stored in a top-level folder with a name such as "Sent Mail". Only mail from the usmail front-end will appear in here.*
UBSFBOX=*name*	Defines the name that the IMAP server gives to its future (deferred) mail folder. Each mailbox will have an entry within this folder.
UMD_NO_AUTO_ACTION=*list*	This keyword is used to stop Uniplex Mail from generating auto-replies to, or auto-forwarding of, UNIX Mail non-delivery notifications. Auto-replies will not be generated for UNIX Mail messages received from users in the UMD_NO_AUTO_ACTION list. Similarly, UNIX Mail messages from the users listed will not be auto-forwarded. If auto-forward has been requested with the NO SAVE option set, the latter will also be ignored and a copy of the letter will be saved in the recipient's local mailstore. The default list is set to MAILER-DAEMON only. The list uses the same syntactic rules as for MBEXCLUDE (comma-separated, semi-colon terminated list of names). You may wish to add root to this to ensure that, by default, no UNIX Mail messages from root will be auto-actioned. If you enable the MODE=F flag to force "From:" to override "From" then you should not need to add root to the list. Comparisons against names in the list are always done in a case-insensitive manner, i.e. MAILER-DAEMON will match Mailer-Daemon and mailer-daemon as well as MAILER-DAEMON.

Keyword	Description
UMD_NO_AUTO_ACTION (*continued*)	In sendmail, all non-deliveries come from the postmaster aliased as MAILER-DAEMON which is also the name part of the From: address field. AOS install checks any sendmail configuration on the machine and reports if it's postmaster name is not in the UMD_NO_AUTO_ACTION list.
USERS	This is a reserved keyword.

The following is an example of a umail.rc file foreign editor keyword section:

```
FPRINT = wp.print
EDITOR = wp
EDITOR = yes
BROWSER = no
BRNAME = wp
SMODE = PJ
OPTCOPY = no
OPTPRINT = no
GROPTS=b r 60 60 10
```

Specify the #EASISEND Section

Any of the seven keywords that can be configured for the Send Mail Form can also be configured for use with the EasiSend option. To do this, simply put a #EASISEND entry in umail.rc.

Any of the seven keywords described in the Send Mail Form which do not appear in the #EASISEND entry will use the defaults from the main send form.

If there is no #EASISEND section in the umail.rc file, the default values for this option will be identical to those specified for the main send form.

An example of a #EASISEND entry is given below:

```
#EASISEND
SAVECOPY = no
VERIFY = yes
REGISTERED = no
))
```

Specify the #PICKLETT Entry

You can specify how the message fields will appear in the 'Read Incoming Mail' and 'Check Mail Sent' screens.

This allows you to change the default widths of the From and To fields (16 characters) and also of the Subject field (33 characters).

To do this, you need to put a #PICKLETT section entry in the umail.rc file, including some or all of the following keywords:

Keyword	Description
#PICKLETT	Indicates the start of the section.
FROM_TO = *n*	Specifies the number of characters in the **From** or **To** field, depending on the screen. The default is 16 and the minimum allowed is 4 characters.
SUBJECT = *n*	Specifies the length of the **Subject** field; by default this is 33 characters, and the minimum allowed is 4.
DATE = *yes¦no*	Specifies whether the date is to appear in the display or not.
TIME = *yes¦no*	Specifies whether the time is to appear in the display or not.

Note: The width of the fields in the 'Read Incoming Mail' and 'Check Mail Sent' screens are also affected by the setting of the keyword TRDATE in the umail.rc file. These and the other remaining keywords are described below.

Specify the #PRINT Entry

When a message is printed, the information contained in the header can be configured by putting a #PRINT entry in umail.rc.

The section defining the print header characteristics can include some or all of the keywords described in the table below, in any order:

Keyword	Description
#PRINT	Start of the #PRINT entry in umail.rc
FROM	Name of the sender of the message
SUBJECT	Subject of the message
TO	Name of recipient(s)
CC	Name of users to receive carbon-copies
BCC	Name of blind carbon-copy recipients
DSENT	Date message was sent
DRECEIVED	Date message was received
SENT	Date and time message was sent
RECEIVED	Date and time message was received
))	End of section marker

You may also insert blank lines and lines without keywords in the header information. An example of a #PRINT section is given below:

```
#PRINT
From:                [FROM]
Subject:             [SUBJECT]
To:                  [TO]
Date received:       [RECEIVED]
))
```

The titles given on the left of the keywords can include whatever text you choose, but the actual keywords must be enclosed in square brackets as shown.

Note: If you are using a supported translation of Uniplex, the keyword titles may have been localized.

Defining Mail Routes to Other Systems (#SYSTEM)

You use #SYSTEM sections to tell Uniplex how to send mail to other systems.

You can define different ways of sending mail to the same system if required, by creating a #SYSTEM entry, for each method, in the umail.rc file.

For example, you may have a modem and an X.25 connection to a remote site, each accessed by a different method.

Uniplex provides some example #SYSTEM entries in the umail.rc file. You can use these as a template for any #SYSTEM entries you need to make.

You must make a separate entry for each remote computer system you want to send mail to, and for each different method, even if it is to the same target machine.

Create a #SYSTEM entry as follows:

1 Edit the system file umail.rc, which is normally located in UAP/umail.

2 Move to the bottom of the file.

3 Add a system specification as follows (see below for a detailed description of each field):

```
#SYSTEM
NAME = system_name;
PRIORITY = n...n;
UMAIL = yes¦no¦V6¦ongo;
ADDRESS = address;
COMMAND = command;
FORMAT = format;
USERS  = list;
DISTRIB = yes¦no[,dispatcher];
SYSMODE = string;
))
```

Keyword	Description
#SYSTEM	Indicates the beginning of a system specification. You must include #SYSTEM at the beginning of each new system specification.
NAME = *system_name*;	Where *system_name* is the name by which the system is known to Uniplex mail. You can create more than one #SYSTEM entry using the same *system_name*. For example, if you want to use different addressing methods for different mail priorities. The length of *system_name* must not be more than 15 characters. If the remote system runs Uniplex mail, this name must match the name specified for the system in its systemid file. For example: `NAME = blue;` *Note: The name **mail** is used as the #SYSTEM to use to send mail if the user uses an address containing @ or !.*
PRIORITY = *n...n*;	Where *n* is a number of a priority you can use to send mail to this system. This is an optional field. If you do not specify it, users can send mail with any priority to this system. If you do specify it, users can only send mail to this system with one of the priorities you specify. All messages with priority 1 are marked as urgent for the recipient(s) attention. A message with priority 2 is treated as non-urgent and priority 3 message is treated as normal. It is recommended that you assign at least these priorities to another system. For example: `PRIORITY = 123;`

Keyword	Description
PRIORITY (*continued*)	Specify multiple #SYSTEM sections with the same NAME and different PRIORITY values to arrange to deliver mail by different routes or using different delivery COMMANDs.
UMAIL =yes ¦ no ¦ V6 ¦ongo;	In this field, you specify whether or not the system uses Uniplex mail and if so, what type of header is attached to the message. Uniplex creates its own "mail envelope" which contains all the header information it requires. Specify **yes** if the system does use Uniplex mail (a minimum header will be sent with the mail), otherwise specify **no** (no header will be sent). See the #FORMAT section for information about sending mail using another system. For example: `UMAIL = no;` Specifying **V6** will cause Uniplex mail to act as for the **yes** option except that a "full" rather than a condensed Uniplex message header will be sent. The **V6** option should be used when sending to a system using Version 6 of Uniplex. Specifying **ongo** is the same as putting 'O' in the SYSMODE string.
ADDRESS = *address*;	Where *address* is the address or routing to the other system as referenced by the [TO] replacement in the command, if any. The format of the address depends on what the mail transport mechanism is between your system and the remote system. For example, by uucp. You must set the address when setting up mail for use by remote systems. The address may be, for example, an ARPANET, INTERNET, or UUCP address. The position of the username is defined with the keyword [USER]. See the section Keywords for details.

Keyword	Description

ADDRESS (*continued*)	For example, if this system and the remote system (blue) are connected by uucp:
	
	`ADDRESS = blue![USER];`
	
	Or, if this system and the remote system (blue) are connected by uucp via the system red:
	
	`ADDRESS = red!blue![USER];`
	
	Or, if this system (red) and the remote system (blue) use the ARPANET addressing, the network address could be:
	
	`ADDRESS = [USER]@blue;`
	
	For details of how to set up a uucp connection, see your operating system manuals. For details of making connections using other communications or networking packages, refer to the manuals for the communications or networking package you are using.

COMMAND = *command*;	Where *command* is the name of the command that is needed to send mail to this system. The *command* can include: o The operating system command to invoke the mailing system you want to use. o Any of the keywords required by the mail system. See the section Keywords. o Any special characters required by the command. o Any special operating system characters you want included with the command. However, it cannot include a semicolon character (semicolon is used to indicate the end of the command).

Keyword	Description

COMMAND (*continued*)	For example, this system and the remote system (blue) both have UNIX mail:
	
	`COMMAND = mail [TO] < [LETTER];`
	
	In this example, the address specified is used in the TO field, and must be a valid remote address that the communications system can recognize. See the section Keywords for details of using TO and LETTER.
	
	Where it is available and appropriately configured, **sendmail** is the best command to use. For example:
	
	`COMMAND = /usr/lib/sendmail [TO] < [LETTER];`

FORMAT = *format*;	Where *format* specifies the format required for the mail envelope in conjunction with the command specified in COMMAND. For example: `FORMAT = unixmail;` You must define the format you specify here in a #FORMAT entry. See the earlier section Define #FORMAT Sections.
USERS=*name*,*name...name*;	This optional field identifies the only users on the remote system that can be mailed to. *name* is a username on the remote system. The *names* are also used to extend the list shown in a popup pick and point list of users when using the EXPAND function, or **LIST USERS** softkey, in the send mail form. For example, if #SYSTEM of **NAME = blue** contains: `USERS=jim,helen,brian;`

Keyword	Description

USERS (*continued*)	then the pick and point list would contain:
	
	`blue>jim`
	`blue>helen`
	`blue>brian`

DISTRIB = *yes*, [*dispatcher*]¦*no*;	If this keyword is set to yes, and *dispatcher* is not specified as a DISTRIB parameter, then multi-recipient messages will be sent to the default dispatcher (u-mailer).
	
	However, some systems do not allow '-' as part of a username; in this case, an alternative name can be assigned via the *dispatcher* parameter to the DISTRIB keyword. This name can be up to 14 characters in length.
	
	Note: Check that the recipient system has a login account set up, either for the dispatcher specified, or for u-mailer if no dispatcher is specified, and that umd_runix is configured to process that mailbox on the remote system.
	
	For example:
	
	Sending end
	
	`#SYSTEM`
	`NAME=blue;`
	`DISTRIB=yes,postreader;`
	`.`
	`.`
	`.`

Keyword	Description
SYSMODE = *string*;	The SYSMODE can contain any of the following letters:

A This can be used when the receiving system is Uniplex Version 9 or later. It tells Uniplex not to limit the size of attachment lists to 149 characters. If this flag is not set when doing a remote send, lists longer than 149 characters are converted to "A0 A1 A2 ...".

B This can be used when the receiving system is Uniplex Version 7 or later. Any attached binary files are converted to ASCII by the local machine before being sent. This ensures that binary files, which could otherwise become corrupted (as they contain data outside the standard ASCII range, 0-127), arrive safely.

Note: Expansion of up to 25% can occur when the file conversion (binary to ASCII) takes place. The example below shows what an attached binary file format is like.

```
%UATTACH
%BINARY
...
(up to 78 ASCII characters/line)
...
xBINARY End <checksums>
%UEND
```

D This can be used when the receiving system is Uniplex Version 8 or later. It tells Uniplex not to "flatten" forwarded messages into a single message with *****START OF FORWARDED MESSAGE***** delimiters. Instead, it passes each forwarded message as a separate item (using strings %UFORWARD and %UBODY in the Uniplex envelope.

Keyword	Description
SYSMODE (*continued*)	E This can be used when the receiving system is Uniplex Version 7 or later. Any extended (ISO) ASCII characters in files are converted to four ASCII bytes (*nnn*) by the local machine before being sent (where *nnn* is the octal character equivalent). In this case, the receiving system should have "U" set in its umail.rc MODE keyword. M Messages are sent with MIME encoding. The main message is always of type "text/plain" (unless SYSMODE=W is also specified, see below). Messages with attachments are encoded with type "multipart/mixed", and the attachments become parts with type "application/octet-stream". *Note: The COMMAND for this system must allow RFC 822 header fields in the file input (sendmail allows this), and the FORMAT for this system must not include a blank line in its :HEADER.* O Capital "o". This can be used when the receiving system is onGO UOS Version 2.00 or later. Send a "de-aliased" version of the sender's name in the mail header so the onGO recipient sees the user's full, rather than login, name. For example, the mail appears to come from John Doe/V7 (systemA>jld) rather than jld/V7 (systemA>jld). Alternatively, setting the UMAIL keyword in the #SYSTEM section to "ongo" has the same effect as adding "O" to the SYSMODE string.

Keyword	Description
SYSMODE (*continued*)	X Reverse the order of the names of To and CC recipients in the mail header so that if the onGO user replies (or forwards) to all, the pre-completed names have the forename last instead of first. If this flag is not set, onGO users see these other recipients with the names reversed. For example, Doe John instead of John Doe. *Note: This setting should only be used in the #SYSTEM sections that address onGO users only. If the same system section is used for Uniplex Business Software users, they will then suffer the name reversal problem. Thus, if you have machines that host both onGO and UBS users, you should use different #SYSTEM section to address them. For example:* *onGO userA = o_blue>ona;* *UBS userB = u_blue>uub;* W Used in combination with SYSMODE=M. If the main message includes Uniplex Word Processor formatting, it is sent as type "multipart/alternative" with "text/plain" (without formatting) and "text/x-uniplex" parts. Notes: 1 If setting "B" or "E", it is assumed that the machine receiving the files is running Uniplex Mail Version 7 or later, and is thus capable of automatically converting the files back to their original format. 2 As a rule, when setting up a wide area Uniplex mail network, between two machines running Uniplex connected by a non-8-bit clean mail server, SYSMODE should be set to BE at both ends.

Keyword	Description
FCONVERT=*command*	This keyword is used to configure the name of a converter for non-Uniplex II Plus body text (for example, WordPerfect to ASCII). If this option is set for any #SYSTEM entry, then Uniplex Mail makes changes to the body text of any message which has been constructed with the foreign editor. Having performed the conversion of the body parts, Uniplex Mail applies any SYSMODE settings if necessary.

Define #FORMAT Sections

As shipped, Uniplex lets you exchange mail with other systems that use UNIX mail as a transport method for Uniplex Mail messages.

If you want to exchange mail with a system that does not use either of these, you must provide a #FORMAT entry for the mail system used.

Create a format entry as follows:

1 Edit the system file, umail.rc, which is normally located in UAP/umail.

2 Move to the bottom of the file and enter the following for each format entry:

Keyword	Notes
#FORMAT	Start of section.
NAME = *name*	Specifies the unique name for this FORMAT section.
[:HEADER] *entry_line* *entry_line* . . .	Indicates that the following *entry_line*(s) constitute a sub-section describing the format of the header to be added to the front of every message.

Keyword	Notes
[:FOOTER] *entry_line* *entry_line* . . .))	Indicates that the following *entry_line(s)* constitute a sub-section describing the format of the footer to be added to the end of every message. End of section.

The *entry_lines* in :HEADER or :FOOTER sections can include keywords to be replaced in the actual text of the header or footer. See the next section, Keywords, for a list of the keywords you can use in the #FORMAT entry.

Keywords

You can use the following keywords in the ADDRESS and COMMAND lines of #SYSTEM entries, and in the :HEADER and :FOOTER sub-sections of #FORMAT sections.

Keyword	Function
[LETTER]	The name of a temporary file containing the text of the letter.
[USER]	The usernames of the recipients of the mail.

[TO]	The address list.
	
	The [TO] string is replaced by the address of each user, formatted according to the ADDRESS= specification. For example, if your #SYSTEM entry is as follows:
	
	`NAME=blue;`
	`ADDRESS=red!blue![user]`
	`COMMAND=mail [TO] < [LETTER];`
	
	and the user sends mail to:
	
	`blue>peter, blue>john, blue>ann`
	
	then the following command is executed:
	
	`mail red!blue!peter red!blue!john red!blue!ann < temporary.file.name`

Keyword	Function
[TLIST]	This keyword must only be used in the #FORMAT section, and only if the underlying mail system can be directed to read its username arguments from the message file (for example, as Sendmail can with the **-t** command-line switch). It enables recipient addresses to be embedded in the message header. This means that addresses do not have to be passed in a buffer, and that the underlying mail system only needs to be invoked once, even for multiple recipients. For example, the #SYSTEM section defining the Sendmail system might be defined as follows: **#SYSTEM** **NAME=sendmail;** **PRIORITY=123;** **ADDRESS=[USER];** **COMMAND=/usr/lib/sendmail -t < [LETTER];** **FORMAT=sendmail;** **))** The corresponding #FORMAT section defining the FORMAT specified in the above #SYSTEM section (that is, sendmail) might be defined as follows: **#FORMAT** **NAME=sendmail;** **:HEADER** **To: [TLIST]** **Subject: [SUBJECT]** **))**

Keyword	Function
[TLIST] (*continued*)	The resultant file, passed by Uniplex mail as an argument to Sendmail, will look like this:

```
To:          address1,
             address2,
             .
             .
             addressn
Subject:     subject_of_message

%UNIPLEX
other_header_information

message body
%UEND
```

Note that each address will be on a separate line, terminated by a comma.

[FROM] The name of the user that is sending the mail. (Some mailing systems require this.)

If the other mailing system is not Uniplex mail, but can pick out the name of the sender from the body of a letter, this can be placed in the #FORMAT section.

Sites using several Uniplex systems locally networked but with a common "public" mail address for all users (username@company.com), can add an SMTP Reply-To directive to the "Unix" mail header so that non-Uniplex recipients will see this public address as the reply address, instead of the one for the internal system they work on (username@local-machine). For example:

```
#FORMAT
NAME=unixmail;
:HEADER
Reply-To:  [FROM]@uniplex.co.uk
Subject:  [SUBJECT]
))
```

Note: Although the use of FROM is optional, it may be specified to ensure that message authorship is correctly recorded throughout the mail system, when a UNIX mail message is autoforwarded by Uniplex.

Keyword	Function
[SUBJECT]	The subject of the letter. If the #SYSTEM entry specifies SYSMODE=M and the subject contains non-ASCII characters, then RFC 1522 coding is used.
[SYSTEM]	The name of the system that is sending the mail. (Some mailing systems require this.) If the other mailing system is not Uniplex mail, but can pick out a system line from the body of a letter, this can be placed in the #FORMAT section.
[DATE]	If the other mailing system is not Uniplex mail, but can pick out the date from the body of a letter, this can be placed in the #FORMAT section.

Picking Up Mail From Other Machines (umd_runix)

The program *umd_runix* is the remote mail controller. It extracts messages from users' UNIX mailboxes and places them in their corresponding Uniplex mailbox. You can modify the operation of *umd_runix*; for example, you can exclude some users from being processed. The more common reason to modify *umd_runix* is to specify a filter program to use in conjunction with the program. This allows you to configure Uniplex to handle non-standard UNIX mail.

When you invoke *umd_runix*, it reads each UNIX file you specify and assumes it to be a UNIX mailbox for that user. If the file contains data, it is opened and *umd_runix* extracts all valid messages from it, moving their data to the corresponding user's Uniplex mailbox (in *UAP/umail/text/user_name*).

If an RMFILTER has been specified, it will process the copy of the mailbox, because of this no locks, other than *umd_runix*, are in existence.

If the filter wishes to hide data from *umd_runix* and then alter a user's UNIX mailbox, the filter should be written to obey the locking rules used by UNIX mail. These vary between version of UNIX and it is beyond the scope of this document to detail them. For more details about invoking *umd_runix*, see the appendix Program Usage and Invocation.

You can use the special section, **#UMD_RUNIX** in the central *UAP/umail/umail.rc* (a local *umail.rc* file is ignored by *umd_runix*) to modify the operation of the program *umd_runix*. You can specify the statements in this section in any order, in one of the following two forms:

either: *NAME* **=** *value* [**,** *value* ....] **;**
or: *NAME* **=** *value*

Where *value* specifies text to match. If the last character of a *value* is a hyphen (-) any word starting with the specified characters (including the hyphen) is matched. For example, **X-** matches lines starting **X-FROM**, **X-REF** and so on.

You can use spaces and new lines in any position between the NAME and the end of the statement. This improves the readability of the statement. White space within a multi-word *value* is treated literally.

For example:

```
HEADER_IGNORE = Posted Date;
```

matches `Posted Date`, but not `Posted  Date`.

You can include comments in statements. Prefix each comment with an asterisk.

The recognized statements are:

Statement Name	Explanation
MBEXCLUDE=*list*;	List of user names whose mailboxes should not be processed. For example: `MBEXCLUDE = root, sys, daemon, uucp, lp ;` *Note: This statement may be included elsewhere in umail.rc, as well as inside the #UMD_RUNIX section.*
RMFILTER=*program*	Specifies the name of a program to be used to read the mailbox. By default *umd_runix* reads each mailbox directly. The use of an RMFILTER program allows sites with non-standard UNIX mailboxes to implement suitable interface programs. For example: `RMFILTER = umfilter` *Note:* *a) The RMFILTER name must NOT be followed by a semicolon (;).* *b) This statement may be included elsewhere in umail.rc, as well as inside the #UMD_RUNIX section.* *c) RMFILTER cannot be specified in a local umail.rc file e.g ($HOME/UAP/umail/umail.rc).*

Statement Name	Explanation

RMFILTER (*continued*)	The program specified should read standard input which *umd_runix* connects to a copy of the mailbox to be processed. The program specified is invoked with its standard output connected to a pipe back to *umd_runix*. It should not access the mailbox directly.
	
	Though the logic it implements is not required with this version of *umd_runix*, the sample RMFILTER program shown in the Version 6 release notes can therefore be written as:
	
	`sed '/^From[^:]/a\`
	`' # Add blank line after each "From " line`

FROM_TRIGGER=*list*;	Defines possible UNIX header trigger words (see the section UNIX Header, below).
HEADER_FROM=*list*; HEADER_INFO=*list*; HEADER_SUBJECT=*list*; HEADER_IGNORE=*list*;	Define all other recognized UNIX header keywords (see the section UNIX Header Keywords, below).
DISPATCHER=*list*; DISPATCH_SYSTEM=*list*;	Define special senders (see the section From Formats, below).
MODE=*string*	Define the mode used, where the options string can include any number of the following characters: *Notes:* *1 MODE needs no semi-colon (;)* *2 Although most MODE flags control umd_runix, some control other mail programs.*

Statement Name	Explanation	
MODE (*continued*)	A	use 'Reply-To' field as return address.
	B	keep treating text as header information until the first blank line is encountered, ignore any unknown keywords. *Note: By default, Uniplex Mail identifies the end of a UNIX Mail header when either a blank line, or an unknown keyword, is encountered. By setting this mode, all unrecognized keywords in a UNIX Mail header are ignored; the body of the UNIX Mail message follows the blank line.*
	C	when replying to mail, automatically makes all other recipients of the original message CC recipients on reply. The user can override this by changing the option on the 'Reply To Sender' form. *Note: This flag controls the USMAIL program, not UMD_RUNIX.*
	F	a 'From:' line will override a preceding 'From ' line in a message header
	H	causes all the headers of a non-Uniplex message to be put into an attachment called "%headers". The trigger line is not included, but all other header lines are included (even those ignored because of HEADER_IGNORE or MODE=B).
	L*size*	specifies the size (in bytes) at which the debugging procedure switches from one circular log file to the other

Statement Name	Explanation	
MODE (*continued*)	M	causes MIME-encoded messages to be recognised and decoded. The return address (used for auto-replies and if the user selects "Reply") for a MIME-encoded message is prefixed with "mime>". Also allows folded RFC 822 header lines to be understood, and makes the checking of HEADER_... keywords case-insensitive (except "FROM").
	S	causes the full Sendmail address of the originator (as specified in the Sendmail header) to be used. *Note: This flag also turns off reverse aliasing in all Uniplex Mail programs.* *Note: The 'S' flag must not be used unless you have an RFC 822 Sendmail system.* This flag automatically uses the 'F' flag described above. Also, the address on the From: line will be used in preference to %FROM in messages.
	U	used together with the SYSMODE = E; option, this causes this machine to convert encoded characters from octal \nnn format to the original 8-bit format
	X	normally used in combination with MODE=S, this requests that all (non-Uniplex) UNIX mail "remote from" messages are shown as coming from the mail system defined in DEFSYSTEM in umail.rc, or, if that is not set, from the system named "mail". If a MIME-encoded message is decoded with MODE=M, then "mime" is used as the name of the system (not the DEFSYSTEM value or "mail").

Statement Name	Explanation

MODE (*continued*)

Note: The following flags control the UMAILEXEC program, not UMD_RUNIX, and so affect all Uniplex Mail and Diary operations.

R — use the #SYSTEM entry for the local (that is, originating) system when delivering local mail.

Note: The effect of this flag is similar to that of DEFSYSTEM=localsys except that it will also affect USDIARY notifications.

G or g — If either of these flags is set, UMD_RUNIX and UMAILEXEC binaries can have any "setuid root" removed but still must have "setgid mail" set. For BSD systems (that require an external chown) use 'G', for SysV systems, use 'g'. These flags control the permissions forced onto mailstore files and directories as follows:

Mode G:(BSD)

	Mode	Owner	Group
Letters:	0660	bin	mail
Other files:	0660	bin	mail
Directories:	0755	bin	mail

Mode g:(SysV)

	Mode	Owner	Group
Letters:	0620	user	mail
Other files:	0660	bin	mail
Directories:	0755	bin	mail

Otherwise:

	Mode	Owner	Group
Letters:	0600	user	mail
Other files:	0666	anyone	mail
Directories:	0777	anyone	mail

Debugging Information

It is sometimes useful to obtain debugging information to provide assistance in configuring umd_runix and Uniplex Mail.

The data obtained is stored in either the file /usr/spool/umail/*username*.log or /tmp/*username*.log if umd_runix cannot create the directory in the first pathname.

To avoid consuming system resources, two log files are used cyclically. When the first file has reached a predefined size (which is 25000 bytes by default but can be configured using the L option in the MODE keyword), the routine automatically cycles the file to *username*.log-1. There are two methods of obtaining debugging data:

- by specifying the run-time flag **-D** with umd_runix. Debugging is carried out for all users' UNIX mailboxes that are parsed by umd_runix.

- by putting the keyword **USRLOG** in the umail.rc file, and specifying either selected users' names, separated by commas, or using the option *ALL* to produce debugging data for all users' mailboxes.

  The examples below illustrate the use of this keyword.

```
USRLOG = george, bernard, joyce;
USRLOG = ALL;
```

For more information about the #UMD_RUNIX section, refer also to the supplied umail.rc file.

Definitions

In the descriptions of mailbox processing below, the following terms are used with specific meaning (hexadecimal character values are given in brackets):

Definition	Explanation
blank character	SPACE (0x20) or TAB (0x09).
blank line	LINEFEED (0x0A) followed by zero or more *blank characters*, followed by another LINEFEED.
keyword delimiter character	Any *blank character* or colon (:).
printable character	Any character in the ASCII range SHRIEK (**!** - 0x21) through TILDE (~ - 0x7E).

UNIX Mailbox Format

The program **umd_runix** understands the format for UNIX mailboxes, whereby a mailbox is a contiguous file containing the following:

UNIX PREFIX
UNIX HEADER (message 1)
UNIX DATA (message 1)
UNIX HEADER (message 2)
UNIX DATA (message 2)
UNIX HEADER (message *n*)
UNIX DATA (message *n*)

Details of each of these are given in the sections that follow.

UNIX Prefix

There is usually no prefix data, but some UNIX mail systems allow the mailbox to start with a line such as **Redirect to** *username* to cause auto-redirection of mail coming into the mailbox.

The program *umd_runix* will recognize that a mailbox is being forwarded and will cease processing that mailbox.

UNIX Header

The header consists of one or more lines, the first of which is a *trigger* line (see below). Each line can be up to 255 characters of ASCII text followed by a linefeed character, and must start with a recognized *Header keyword* (see the section UNIX Header Keywords, below). A keyword is terminated by a *keyword delimiter character*. From the sample *umail.rc* shown earlier, the following are all valid header lines:

```
Apparently-To joe
Reply-To:  <jones@host>
Reply-To dorothy
DATE:  Tue, 24 May 88 14:15:51-0000
Date 24 May 88
```

If *umd_runix* uses information following a keyword, this information is assumed to start with the first non-blank character following the keyword delimiter character.

A header line starting with one of the recognized keywords may be continued on a second line. Any such continuation line starts with at least one blank character. A completely blank line is also a valid continuation line - and is inserted by some mail systems.

For example, the three lines:

```
Date:  May 12 1988

Subject:  This is text for the header detail line SUBJECT
```

could also be phrased as the three lines:

```
Date:  May 12 1988
Subject:  This is text for the
    header detail line SUBJECT
```

Each header starts with a *trigger* line, which is recognized if it starts with a word specified as a FROM_TRIGGER in *umail.rc*, immediately followed by a blank character, not a colon.

Note that, until the end of header has been detected, FROM_TRIGGER words are not special. It is therefore common to find FROM_TRIGGER words (usually the word **From**) also specified as HEADER_FROM or HEADER_IGNORE keywords, since they are often repeated when mail is sent across systems.

The header extends until the next line that is not a header keyword line, or a continuation line.

UNIX Data

All data following a header is UNIX message data. The data can be of any form - though it is usually linefeed-separated ASCII text. If received from a Uniplex system, the data may contain a Uniplex message header (see the section Mail From Other Uniplex Systems, below). The data ends at either:

- End of file
- The next UNIX header trigger line.

Note that this means that, with the normal configuration, the only *pure* data *umd_runix* will not support in message data is of the form \n**From** (using C notation). It is for this reason that most UNIX **mail** programs always prefix the character **>** to any message line starting with the word **From**.

However, since *umd_runix* recognizes a Uniplex-style message (see the section Mail From Other Uniplex Systems, below), then if the first non-blank line of a non-Uniplex message starts with the characters **%UNIPLEX** or **FROM** it will not be treated as message data.

A simple UNIX mailbox might contain the following (trigger lines highlighted in bold):

```
From helen Fri May 20 20:17 EDT 1988
To:  peter
Subject:  Standard UNIX mail (via MAILX)

This is the first line of my message. MAILX has added the previous header and a following blank
line.

From james Fri May 20 22:01 EDT 1988
Some mail systems make life very awkward by not adding any obvious separator between the header and
the message data. The line above, starting "Some mail systems", is the beginning of message data
simply because its first word is not a recognized header keyword.
```

UNIX Header Keywords

UNIX mail systems use a wide variety of header keywords. These are described to *umd_runix* in the *umail.rc* file.

Other than a FROM_TRIGGER, keywords are only recognized at the beginning of a line and if followed by a *keyword delimiter character*.

The keywords fall into 5 categories, depending on whether *umd_runix* uses information from the associated keyword header line:

Keyword	Explanation
FROM_TRIGGER	Defines possible header trigger words (see the section UNIX Header, above). Each word specified is only recognized at the beginning of a line and if followed by a blank character (all the other keywords can also be followed by a colon). *Note: You can use the F option of the MODE keyword to override a preceding FROM_TRIGGER line (usually "From") with a "From:" line in the header.*

Keyword	Explanation
HEADER_FROM	Alternate sources of information of sender and sending system if the sender on the FROM_TRIGGER line is a specified DISPATCHER (see From Formats).
HEADER_SUBJECT	Gives mail subject text - only used if no Uniplex message follows.
HEADER_INFO	General information lines. If any of these keywords are found in a UNIX header and the message does not contain a Uniplex message, then the entire header line (and any continuation line) is transferred into the Uniplex message.
HEADER_IGNORE	All header lines for these keywords are ignored. They are defined solely to enable *umd_runix* to recognize the set of lines that constitute a UNIX header. *Note: With MODE=B, the HEADER_IGNORE keywords define lines that can occur after the blank line and before a Uniplex Message Header.*

Mail From Other Uniplex Systems

If the DATA part of a UNIX mailbox message is in the form of a message from a Uniplex system, **umd_runix** uses the Uniplex-style header in preference to the UNIX header detail.

In this case, the UNIX message Data has the format:

- *Optional blank lines*
- Uniplex Message Header (up to 18 lines; 'n' options are omitted)
- Uniplex Message Data (may be followed by optional ATTACH file data)
- Uniplex Message Footer
- *Potential additional data (discarded)*

More details of each of these are in the sections that follow.

Uniplex Message Header

This consists of up to 18 lines followed by one blank line. Each of the lines up to the blank line contains a keyword followed by optional qualifiers. The keyword field is always padded out with trailing spaces to 15 characters:

Line	Keyword	Notes	Qualifier (if any)
1	%UNIPLEX		- none -
2	%TO	1	List of addressees, as specified by the sender.
3	%CC	1	List of carbon copies, as specified by the sender.
4	%BCC	1	As %CC, for blind carbon copies.
5	%MAILBOX		Name of recipient's Uniplex mailbox. If omitted, or the recipient has no such mailbox, this field is retained for display only.
6	%FROM		Sender's name (see the later section From Formats).
7	%SYSTEM		Sending system name (see the later section From Formats).
8	%SUBJECT		Subject text.

Line	Keyword	Notes	Qualifier (if any)
9	%ATTACH		Sender's names for any attached file(s). (See the later section Attached Files.)
10	%PRIORITY		ASCII digit indicating senders priority. Defaults to **2**.
11	%VERIFY	2	Indicates whether the sender asked for automatic verification when the recipient reads the mail using usmail.
12	%CONFIRM	2	Indicates whether the sender asked for the recipient to explicitly confirm receipt from the usmail Read Incoming Mail menu.
13	%REGISTERED	2	Indicates whether the sender asked for message receipt verification.
14	%ENCODE	2	Indicates whether the message data following this header is encrypted by Uniplex.
15	%PASSWORD	2	Indicates whether the message is password protected.
16	%DATE		Date sent in form DD/MM/YY HH:MM, regardless of senders configuration of Uniplex.
17	%REFERENCE		Integer number.
18	%ADDRESS	1	Reserved.

Notes :

1 Information is carried into the recipients' Uniplex mailbox for display only (using EXPAND option).

*2 The keyword option can be either a lower-case letter **y** (yes) or **n** (no). If the **n** option is selected, the line is omitted from the message header by default.*

Uniplex Message Data and Footer

Any data may follow the header, including attached file contents (see Attached Files). The end of this data is marked by a footer consisting of a string of 7 characters: "**\n%UEND\n**" (using C notation). Any data, following this marker, up to the next UNIX header, or end-of-file, is discarded by *umd_runix*.

From Formats (Sender's Name and System)

The format of a UNIX mail *from* line is:

keyword[*delim* ...] *address* [*text* ...] [*date-time*] [*text* ...]

where:

Keyword	Explanation
keyword	Either the FROM_TRIGGER keyword, or a HEADER_FROM keyword - see UNIX Header, earlier (usually **From** or **>From**).
delim	The optional keyword delimiter. There must be at least one (normally a blank).
address	The sender's name, consisting of any string of non-blank characters.
text	Any text, which will usually be discarded by *umd_runix*.
date-time	The date and time sent (see Date Formats). Only used if present and there is no Uniplex header %DATE field in the message.

Sender Name

When determining who sent a message, *umd_runix* uses the first *sender name* it finds, looking for it in the following order:

Which header	Keyword line	Notes
Uniplex	**%FROM**	3
UNIX	FROM_TRIGGER *address*	1,3
UNIX	HEADER_FROM *address*	1,3

Notes:

*1 If the address found on a FROM_TRIGGER line is specified in the DISPATCHER statement in umail.rc (**uucp** for instance) and there is also a HEADER_FROM line, then the sender name is taken from the HEADER_FROM line instead.*

2 *When processing messages from a DISPATCHER, if any text in a HEADER_FROM line starts with the word(s) specified in the DISPATCH_SYSTEM statement in umail.rc (usually **remote from**), then the following word is assumed to be the sending system name (unless pre-empted by a %SYSTEM in a Uniplex header).*

3 *If the senders name contains the special character **!** or **@**, then UMAIL assumes that this is a UNIX mail address understood by the System mail server and will always attempt to send return messages via the UMAIL system named **mail** in umail.rc.*

4 *If MODE string contains a character F then an address found on a From: line, in subsequent header information for this letter, will override that found in the FROM_TRIGGER line. The FROM_TRIGGER line however, still triggers the start of a message.*

If no name is found, then the name UNKNOWN is used.

Sending System Name

When determining the sending *system name*, which may be used to construct return addresses, *umd_runix* uses the first name it finds, looking for it in the following order:

Which header	Source	Notes (above)
Uniplex	**%SYSTEM** line	
UNIX	HEADER_FROM line, DISPATCH_SYSTEM argument	1,2,3

If no name is found, then the local system name is used.

Date Formats

The program *umd_runix* recognizes date/time information in UNIX message headers as a series of strings, which can be separated by any number of blank characters or commas, in one of the formats:

Date/Time formats	Examples
[*day*] *dd month year time* [*zone*]	24 May 89 14:15 24 May 89 14:15 EDT Fri 5 Aug 89 09:33:49 MDT Fri, 5 Aug 89 03:29:31 EDT Thu, 1 Sep 89 13:27:53 edt Thu, 19 May 89 13:54:43-0000 Tue, 2 Aug 89 10:07:31 +0200 Tue, 30 Aug 89 21:55:50 edt
[*day*] *month dd time* [*zone*] *year*	Jan 31 12:30 1989 Sun, Jan 31 12:30 PDT 1989 Jan 31 12:30 89 May 28 13:11:24 1989

where:

day Word whose first 3 characters are, in any case, one of: SUN, MON, TUE, WED, THU, FRI, SAT

dd One or two digit number in the range 1-31

month Word whose first 3 characters are, in any case, one of: JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC

year Four digit number starting **19** or **20**, or two digit number in the range 00-99. In the latter case, numbers in the range 70-99 represent 1970-1999, other numbers represent 2000-2069.

time String of the form: *hh*:*mm* [:*ss* [-*nnn*]] giving hours, minutes, seconds (optionally) with optional integer suffix.

zone Any string whose first character is non-numeric. Examples: `EST`, `Z`, `+0100`.

If no sending date is found, then **1 Jan 1970** is used.

Attached Files

The program *umd_runix* recognizes the passing of one or more files *attached* to Uniplex mail message. Files are *attached* by embedding their data following a prefix line. **umd_runix** looks for one such prefix line for each file name mentioned in the **%ATTACH** line. Note that these file names only relate to the file on the sending system, and are not recreated on the receiving one.

The prefixes are:

Version	Prefix (C notation)	Size (characters)
Current	"\n**%UATTACH**\n"	10
Version 5	"\n**ATTACH** " *filename*"\n"	10, plus filename

Note that, due to this prefix logic, and the message triggers, Uniplex may not handle attached files, or include message data, that contain any of the strings:

String	Notes
"\n**From** "	Default UNIX Header trigger. Recognized if entered anywhere outside the UNIX header.
"\n**%UNIPLEX**"	Uniplex trigger. Only recognized if immediately following a UNIX header.
"\n**FROM**"	Uniplex Version 5 trigger. Only recognized if immediately following a UNIX header.
"\n**%UEND**\n"	Uniplex Data delimiter. Only recognized following a Uniplex header.
"**%UATTACH**\n"	See above. Only recognized following a Uniplex header that specifies attached file(s).
"\n**ATTACH** " *filename*"\n"	See above. Only recognized following a Uniplex header that specifies attached file(s).

Sample UNIX Mailbox Files

The following are examples of possible messages that the program will understand. Each mailbox is divided by a line for clarity.

```
From helen Fri May 20 20:17 EDT 1989
To: peter
Subject: Standard UNIX mail (via MAILX)

This is message text
text ...

From uucp Fri May 20 21:51 EDT 1989
>From uucp Fri May 20 18:43 BST 1989 remote from HBX20
uuxqt cmd (rmail peterb ) status (DENIED)

From uucp Fri May 20 21:57 EDT 1989
>From peterb Fri May 20 18:48 BST 1989 remote from HBX20

%UNIPLEX
%TO                    3068>peterb
%FROM                  peterb
%SYSTEM                HBX20xx
%SUBJECT               This has SAVED text (for remail)
%PRIORITY              1
%VERIFY                y
%CONFIRM               y
%REGISTERED            y
%DATE                  20/05/89 18:48
%REFERENCE             8
sdfdsf
dsf
dsfsd
dsf
%UEND
```

```
From 3068!peterb Tue May 24 14:15:56 1989
Received: by uniplex.UUCP (5.52/4.7)
            id AA20985; Tue, 24 May 89 14:15:51-0000
Date: Tue, 24 May 89 14:15:51-0000
From: 3068!peterb
Apparently-To: peterb

Mail poll - please ignore
```

```
From peterb Thu Jun  2 13:18 BST 1989
FROM peterb
SYSTEM UNDEFINED
TO peterb
SUBJECT Test from 5.04. Attached file
REF 2
ATTACH /etc/group /etc/group.old
COPY pauldw

Message text line 1

Last line in message pad

ATTACH /etc/group
root::0:root
other::1:root
sys::2:root,bin,sys,adm
bin::3:root,bin,daemon
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root

ATTACH /etc/group.old
root::0:root
other::1:root
sys::2:root,bin,sys,adm
bin::3:root,bin,daemon
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root
support::10:
```

```
From uucp Tue May 24 18:58 EDT 1989
>From peterb Tue May 24 15:45 BST 1989 remote from HBX20
```

```
%UNIPLEX
%TO                     3068>peterb
%FROM                   peterb
%SYSTEM                 HBX20xx
%SUBJECT                Test from XPS /etc/passwd
%ATTACH                 /etc/passwd
%PRIORITY               2
%DATE                   24/05/89 15:45
%REFERENCE              15

This is memo pad text

Last line

%UATTACH
root:DIp5ACxQxqFJY:0:0:0000-Admin(0000):/:/bin/sh
lp:x:71:2:LP 0000-Admin(0000):/usr/spool/lp:/bin/sh
daemon:x:1:12:0000-Admin(0000):/:/bin/sh
bin:x:2:2:0000-Admin(0000):/bin:/bin/sh
who::2:2::/:/bin/who
uucp:x:5:5:0000-uucp(0000):/usr/lib/uucp:/bin/sh
nuucp::6:5:0000-uucp(0000):/usr/spool/uucppublic:/usr/lib/uucp/uux
diag:x:7:2:DIAGNOSTICS:/usr/diag:/bin/sh
%UEND

From 3068!HBX20!peterb Thu May 19 13:54:51 1989
Received: by uniplex.UUCP (5.52/4.7)
            id AA09357; Thu, 19 May 89 13:54:43-0000
Date: Thu, 19 May 89 13:54:43-0000
From: 3068!HBX20!peterb
To: 3068!uniplex!peterb, 3068!uniplex!andrea, 3068!uniplex!andreaa

Subject: Sending mail between Honeywell&Pyramid
              (blank header line above; this is still UNIX header)
Status: RO

%UNIPLEX
%TO                     3068!uniplex!andrea
%CC                     3068!uniplex!andrea, 3068!uniplex!peterb
%FROM                   peterb
%SYSTEM                 TESTxx
%SUBJECT                Sending mail between Honeywell&Pyramid
%PRIORITY               2
%DATE                   19/05/89 13:49
%REFERENCE              2
```

```
Andrea A:

If you want to send mail (including attached files, maybe) from the
xxxxxxxxx to yourself on the Pyramid, use the following address:

xxxxxxx
xxxxxx

%UEND
```

```
From uucp Tue May 24 21:32 EDT 1989
>From peterb Tue May 24 18:22 BST 1989 remote from HBX20

%UNIPLEX
%TO                     3068>peterb
%FROM                   peterb
%SYSTEM                 HBX20
%SUBJECT                Two attached files
%ATTACH                 /etc/group /etc/group.old
%PRIORITY               2
%DATE                   24/05/89 18:22
%REFERENCE              16

This has 2 files attached
(/etc/group twice)
_# lines=6 file=two.attached attlines=12
Last memo line

%UATTACH
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
mail::6:root
rje::8:rje,shqer
daemon::12:root,daemon
easyusr::37:
informix::300:
support::10:
```

```
 %UATTACH
 root::0:root
 other::1:
 bin::2:root,bin,daemon
 sys::3:root,bin,sis,adm
 adm::4:root,adm,daemon
 mail::6:root

 rye::8:rje,shire
 daemon::12:root,daemon
 easyusr::37:
 informix::300:
 support::10:
 %UEND
```

```
 From SMTP daemon Wed Jun  8 05:13 EDT 1989
 >From peterb@sysA Wed Jun  8 09:04 BST 1989

 Sample AIX-style SMTP message
```

```
 From SMTP daemon Wed Jun  8 05:13 EDT 1989
 >From SMTP peterb@blue Wed Jun  8 09:04 BST 1989

 Another sample AIX-style SMTP message

From rgb Thu Aug 24 14:12:18 1989
Date: Thu, 24 Aug 89 14:12:16-0000
From: rgb (R Graeme Burnett)
To: rgb

To:          rgb
Subject: test

%UNIPLEX
%TO                   fred>rgb
%FROM                 rgb
%SYSTEM               florence
%SUBJECT              test
%ATTACH               f
%PRIORITY             2
%VERIFY               y
%DATE                 24/08/89 14:12
%REFERENCE            1852

See Binary file attached
```

```
%UATTACH
%BINARY
:lk\‘!p!Qn[#>"r6idEE!<dBC"L=^(J-+q"5X%r95X7L7j5:Mg+CIAV0EUH’5X7L7N$&/:
$mG*c"!JfhkXY^[E$6q;kYON04p$fH^^M@T,S:%_E$6q;kYsf44p$fH^^M@T,ROMO
xBINARY End
%UEND
```

```
From rgb Thu Aug 24 14:27:03 1989
Date: Thu, 24 Aug 89 14:27:02-0000
From: rgb (R Graeme Burnett)
To: bob

To:          bob
Subject: test esc

%UNIPLEX
%TO                   brains>bob
%FROM                 rgb
%SYSTEM               florence
%SUBJECT              test esc
%PRIORITY             2
%VERIFY               y
%DATE                 24/08/89 14:26
%REFERENCE            1853

Hello there Bob,

\211 Here is the example of the escape file you wanted.

\211 As you can see, all 8-bit characters are converted to octal esc.
%UEND
```

Enabling the Mail Maintenance Program

The program umd_clean performs the following mail maintenance tasks:

- Sends notification messages that are due
- Sends messages specified for the future that are due
- Cleans out checked statuses

The Mail maintenance program is set to run automatically at one minute past midnight (00:01) every day by default. You can make it run at other times of the day using the Enable External Mailbox form (see the Uniplex Installation Guide for details) by setting **Mailbox driver** to **umd_clean** and leaving **Mailbox** blank.

Address Limitations

The maximum number of characters in a full Uniplex address is:

System name (optional)	15
Separator (>) (optional)	1
Username	511
Separator (<) (optional)	1
Mailbox name (optional)	250

These are the maximum sizes for internal, fully expanded addresses.

For Mail Send Forms, the limits for the "To", "CC", and "BCC" fields are 200 characters. This maximum is not set by default, the 200 character values can be set by exporting a non-blank value in the environment variable **Umailaddr**. For example:

Umailaddr=yes; export Umailaddr; usmail

For Mail Send Forms, the limit for the "Mailbox" field is 15 characters.

Chapter 10
Configuring Other Uniplex Advanced Office Applications

THIS PAGE INTENTIONALLY LEFT BLANK

Notification Messages ("Alarms") and UCLOCK

The program **uclock** checks the Uniplex alarms file (UAP/diary/diary.alarms) used by both Electronic Mail and Time Manager. The alarms file contains details of messages which need to be sent to users at specific times and binaries which need to be run at specified times. If action needs to be taken for any of these occurrences, uclock takes the appropriate action. The possible actions depend on whether the message recipient is using a Uniplex process, is using a non-Uniplex process, or is logged out.

The actions vary as follows:

IN UNIPLEX	An appropriate message is added to the file $Utemp/WPMSOtty*xxx* where *xxx* represents the terminal number and $Utemp is the TEMP directory specified in *uniplex.sys*.
IN OTHER	A beep plus the message is sent to the recipient's terminal.
LOGGED OUT	If the recipient requested that messages be saved when logged out, the message will be retained in the alarms file. Otherwise no message will be sent.

Once running, uclock locks the file UAP/diary/.uclock.lock to indicate that uclock is currently running and that no other uclock should be enabled. The location of the lock file may be altered by setting the environment variable **Uclock** to indicate the directory in which the lock is placed.

uclock is normally only activated on running an office application. Only one occurrence of uclock should be activated on a given system (the locking above ensures this). Modules initiate uclock by executing the shell script **ustartclock** (located in UAP/cmds).

uclock is always run as a background process, and can be started manually by entering:

uclock *n* &

where *n* indicates the time interval in minutes that the alarms file will be checked.

If **uclock** is not running and PATH is reset before "setuid root" programs can invoke other programs, the user may see uclock-related error messages on their screen. To ensure that this does not happen, do either of the following:

- Always start uclock from system startup scripts. For example:

```
/usr/bin/uniplex -run ustartclock
```

- Add a script called **ustartclock** into **/usr/bin** (or another directory that is on the privileged user PATH) containing the above script.

Starting uclock when Character Client Services are Run

When using the onGO Character Clients, **uclock** is started by running "uniplex.rc" (or "ongo.rc" if "ongo.link" has been run); it is actually started by the script UAP/NVO/bin/ustartagent which is run from the CSB, and means that UCLOCK will run as user "uniplex".

This differs from non-onGO AOS, where, unless other system configuration has been made (for example, to start UCLOCK from the system re-boot scripts), it will be started by the first user to run UBS's Mail or Time Manager.

Turn Off uclock

Individual users can enable or disable the delivery of messages using the "Enable Alarms to Screen" and "Disable Alarms to Screen" options on the Time Manager Admin menu. These simply run the commands:

uclock off
uclock on

This does not affect the other functions of uclock, nor does uclock need to be running for this action to be taken. The user's username is registered in a stop file (UAP/diary/diary.alarms.x) which is checked before an alarm or message is sent to that user.

More specific delivery options can be set using "uclock on X.U.C." syntax described in the appendix Program Usage and Invocation.

The default for users who have not used these commands is taken from the central UAP/diary/diary.rc file.

Disable Mail Notifications and Alarms

To disable mail notifications and alarms altogether, do one of the following:

- Change the UCLOCK_DEFAULT section in the central UAP/diary/diary.rc file to say **0.0.0**.

  This causes the following:

  1. No alarm messages are processed.
  2. No notification is given of mail received.

- Set both CHECKTIME and CHECKCHARS to 0 in uniplex.sys.

  This has the same effect, but just for users in Uniplex applications.

- Copy the shellscript ustartclock to a backup file and edit ustartclock so that it contains the single instruction: **exit 0**.

  This causes the following:

  1. No alarm messages are processed.
  2. No background processes are initiated that have been entered into the alarms file.
  3. No notification is given of mail received.
  4. No remote mail (or UNIX mail) is received unless umd_runix is initiated manually. See the section Picking Up Mail From Other Machines (umd_runix) in the chapter Configuring and Administering UBS Electronic Mail for details.

Timing of Alarm Messages

If a message is sent to a user, it is displayed immediately if the user is not using a Uniplex process. If the user is in Uniplex, the message is displayed according to the settings of CHECKTIME and CHECKCHARS in uniplex.sys.

When either of the above situations exist, Uniplex checks the message file that uclock has written out (i.e. $Utemp/WPMSOtty*xxx* as described above). If any messages exist, they are displayed on the screen.

Uniplex Mail Messages when Using 'su'

Uniplex's mechanism to deliver alarms and popup messages, for example, "You have new mail", can only locate a user by their original login name.

Thus, if user "jane" logs in and then uses the **su** command to operate as "vpsales", then the user operates with the mailboxes and default diary for "vpsales", but receives the popup messages and alarms for "jane", and not for "vpsales".

Message (Alarm) Delivery Problems

The following known problems connected to Operating System management or configuration, can result in users not receiving alarm messages:

o If the alarm scheduler (uclock) does not have write access to **tty** devices, then it is unable to deliver asterisk-boxed alarms to users not currently in a Uniplex application.

The two most common cures for this are:

- Make all tty devices write enabled. For example:

```
chmod +w /dev/tty*
```

- Run uclock as "root" from the system boot scripts. For example:

```
/usr/bin/uniplex -run ustartclock
```

o If the Uniplex temporary directory (/usr/spool/uniplex/tmp, by default) has the sticky-bit set on (to delete-protect files), then if Uniplex or the system crashes, old, undeleteable message files can be left behind which may affect delivery of messages to users in Uniplex applications.

As well as not using sticky-bit directories, the good housekeeping practice of removing all files and the uxwindows sub-directory on system boot (and at any other appropriate time when no-one is using Uniplex) minimizes this problem. For example:

```
cd /usr/spool/uniplex/tmp
rm -fr uxwindows
rm -f *
```

Or run the command:

```
/usr/UAP/cmds/housekeeping -ucleanup 0 -x
```

- If Uniplex Windows or the system crashes, then the X Display Message Delivery File ($Utemp/uxwindows/XM.$Uusername[0-9]) is not deleted and messages accumulate, but are not delivered (using default configuration).

  The best solution to this is to clear the $Utemp/uxwindows directory on system re-boot, as described in the previous known problem.

- If users have configured Uniplex to use different settings for its temporary directory (TEMP= string in uniplex.sys).

  For example, if uclock is running on behalf of a user with TEMP=/tmp/testing, whereas all other users use the default (TEMP=/usr/spool/uniplex/tmp), then uclock generates messages which most users never see.

- The UAP/pc database is corrupt (due to machine or application "crashes").

  To cure this, make sure that everyone logs out of Uniplex and then remove all files in this directory. This is also best done on a system reboot. For example:

  ```
  cd /usr/UAP/pc
  rm *
  ```

  Or run the command:

  ```
  /usr/UAP/cmds/housekeeping -ucleanup 0 -x
  ```

Configuring Time Manager

Time Manager Files

Note: This section refers only to the Uniplex Business Software version of Time Manager. For details of the onGO Character Client Time Manager, refer to the onGO Character Client user and administration guides.

Uniplex Time Manager System uses the following files:

Filename	Location	Contents
usdiary	UAP/bin	Program file for Time Manager. You cannot configure this file.
diary.alarms	UAP/diary	Stores alarm messages. You cannot configure this file.
diary.alarms.x	UAP/diary	Stores alarm information. You cannot configure this file.
diary.fn	UAP/diary	Stores softkey text. You can configure this file.
diary.help	UAP/diary	Stores Time Manager help text. You can configure this file.
diary.msg	UAP/diary	Stores Time Manager messages. You can configure this file.
diary.holiday	UAP/diary	Stores National and Special Holiday information, in addition to default personal working hours. You can configure this file.

In addition, three files for each calendar are stored in the subdirectory, diary. These are:

calendar.***dat***
calendar.***idx***
calendar.***env***

where *calendar* is the calendar name.

The file **UAP/diary/diary.rc** is the Time Manager command file, containing two main sections: #ACCESS and #UCLOCK_DEFAULT. In addition, diary.rc contains a section defining the layouts for printing each of the calendar plans available.

Note: In earlier releases, diary.rc also contained a #CONFIG section. This section is no longer required since the configurability it enabled is now contained in the DATEMODE in uniplex.sys. The #CONFIG section does, however, support a flag:

```
OR_SEARCH='c'
```

which is reserved for future use.

Set Default Access Permissions

The #ACCESS section determines the default access permissions all users will be given to a calendar. The calendar owner may re-define the permissions on their calendar with the Time Manager 'Set Access' menu option. The three entries in this section allow the specification of access rights for reading, writing or modifying a calendar. The three keywords (READ, WRITE and MODIFY) are each followed by one word specifying the required default access for the keyword.

READ access can be one of the following:

Access	Explanation
NONE	No read access - other users may neither select nor search the calendar.
SEARCH	Other users may not select the calendar to examine it but they may search for free time slots.
PUBLIC	Other users may select the calendar and examine the entries, but private text is not displayed.
PRIVATE	Other users may select the calendar and examine the entries, all text including private text being displayed.

WRITE access can be one of the following:

Access	Explanation
NONE	Other users are not able to add events to the calendar.
PUBLIC	Other users may add events to the calendar, but may not add private text.
PRIVATE	Other users may add full calendar entries for the calendar.

MODIFY access can be one of the following:

Access	Explanation
NONE	Other users may not change or delete entries in the calendar.
ALL	Other users may edit or delete entries.

If any of the entries are omitted, the relevant access type is set to NONE.

As shipped, the standard ACCESS section is configured as:

```
#ACCESS
READ=SEARCH
WRITE=PUBLIC
MODIFY=NONE
))
```

This allows users to search the calendars of other people and make bookings with the Conference Scheduling menu option, but does not allow them to select those calendars and read the events they contain.

Default Alarm Processing

The #UCLOCK_DEFAULT section determines the default alarm processing. There is just one line in this section containing an **X.U.C** value. See **uclock** in the appendix Program Usage and Invocation.

Define Calendar Plan Layout

The **diary.rc** file contains six sections which define the print layout for the six different calendar plan layouts available. The first three sections define the format of the daily plans; the last three sections define the format of the weekly and monthly plans.

Section	Information stored
DAILY1	Defines the page length and layout of headings for daily plans.
DAILY2	Defines the main table containing calendar events for the day.
DAILY3	Defines the Notes box which is usually printed at the bottom of the page.
WEEKLY1	Defines the page length and layout of headings for weekly plans.
WEEKLY2	Defines the format of the Day and Date.
WEEKLY3	Defines the format of the text.

The WEEKLY1 section, for example, is as follows:

```
#WEEKLY1
.L.............................................................J
.PL 64
.PT 15
.FN SMALL
.SP .6
.....L..................R..L.....................A...
....L.................R.L.....................................A...
.JN
))
```

Each section is made up of 2 rulers. The first ruler defines the format of the printed copy of the calendar plan. The second ruler defines the format for the screen display.

You can edit the length of the ruler. You must not change the ruler characters, or the order in which the ruler characters are displayed, as Uniplex refers to these when it executes the program. See the Word Processor chapter in the Uniplex II Plus User Guide for more details.

In addition, some of the sections include a selection of dot commands defining the page length, line spacing, pitch and font of the plan. You can edit these.

Holiday File

The file **UAP/diary/diary.holiday** contains details of the annual holidays and the working hours of the calendar user. This file comprises of two sections: **#HOLIDAYS** and **#AVAILABILITY**.

The #HOLIDAYS section contains a list of all the holidays in date order. The syntax of each line of this section is:

<date><flag> Name of holiday

The *date* should be in the same format as declared in the DATEFMT flag, as set in the file *uniplex.sys*.

The *flag* should either be blank or an asterisk (*). The asterisk flag is for those holidays which are not necessarily national holidays. Calendar bookings can be made for these days, though a warning message is issued. When printing out a diary, these holidays will be noted.

The *Name of holiday* should describe the holiday.

For example:

```
04/01/89        Good Friday
10/24/89*       Veterans Day
```

Note: The #HOLIDAY section cannot contain more than 96 lines.

Set Working Week

The #AVAILABILITY section details a typical working week, starting from a defined date.

For example:

```
#AVAILABILITY
01/01/89
09:00 17:30
09:00 17:30
WE
WE
09:00 17:30
09:00 17:30
09:00 17:30
))
```

The first line gives a pre-defined date. The next seven lines give the working hours for the week beginning on that date, ignoring national holidays. The number of days included in this section is variable, so for example, you can specify a shift system, whereby you work days one week and nights the next by specifying 14 entries instead of 7.

The above example indicates that 01/03/89 and 01/04/89 constitute the weekend and the work hours for the other days are from 9 in the morning to 5:30 in the evening. From this information, together with the list of holidays, the time manager can calculate the working hours for any subsequent date.

If your site uses a five and a half day working week, for example, with Saturday afternoon and Sunday off then the fourth line in the above file might be configured:

09:00 12:30

instead of "WE", which signifies weekend.

For a change in AVAILABILITY to take effect, delete the user(s) **$Uniplex/diary/diary/**<*username*>**.env** file and then re-invoke the Time Manager.

Create Group Calendars

Uniplex automatically creates a personal calendar for each individual user. All users can create other calendars. System Administrators may also create calendars to schedule the use of a shared facility or piece of equipment. The way you do this is the same as for creating additional calendars and is described in the Uniplex Advanced Office System User Guide.

Configuring Card Index

You can configure the Uniplex Card Index to suit the needs of a particular user, group of users or installation.

- You can design an index for a particular application. For example, you might design a Record Collection card index. You use an external schema to define the labels, valid data types and number of fields in the index. You can use the index, and view the labels, but not change any label details. Using this external schema facility, you can tailor up to 9 external card indexes for specific applications.

- You can fix the number of fields in an index. For example, you could tailor a phone list index to display two fields: name and number.

- You can load records from other indexes, the database, or external files into a card index. For example, you use a simple command to load the contents of a database table into a card index.

- You can search for a particular record from startup. For example, you might tailor a particular softkey option so that you can search the index for a particular record on startup.

Card Index files are contained in the directory **ucard**. This is made up of the following files:

File	Explanation
autodial.ttys	Defines the available paths to autodiallers for each user tty. See the section Auto-Dialler Configuration.
ucard.help	Contains Card Index help text.
ucard.sk	Contains compiled Card Index softkey information.
ucard.fn	Contains Card Index softkey information.
ucard.msg	Contains Card Index messages and external schemas.

Possible Definitions for External Schemas

You can tailor an external schema to define:

- Labels for the index.

- Valid data types for fields in the index.

- Fields to be linked or grouped when ucard searches the index.
- The display width of the card index.
- Whether fields are keyed and, if they are, the length of the key.
- Whether duplicate entries are allowed in a keyed field.
- Whether the index is edit protected.

Description of Available Schemas

You can design up to 9 external schemas. When you define menu options, you specify to invoke ucard with an external schema using a number between 1 and 9. The number corresponds to a series of numbered messages in the Card Index message file, **ucard.msg**. Twenty entries are reserved for each schema for ease of numbering, though a maximum of 17 entries can be made for each.

ucard -t1	uses the external schema defined in the message series 421 to 440.
ucard -t2	uses the external schema defined in the message series 441 to 460 (the Notepad schema).
ucard -t3	uses the external schema defined in the message series 461 to 480 (used for both the System Wide and the Personal Phones and Addresses schemas).
ucard -t4	uses the external schema defined in the message series 481 to 500 (the To-do List schema).
ucard -t5	uses the external schema defined in the message series 501 to 520 (the Project List schema).

You can define your own external schemas as follows:

ucard -t6	uses the external schema defined in the message series 521 to 540.
ucard -t7	uses the external schema defined in the message series 541 to 560.
ucard -t8	uses the external schema defined in the message series 561 to 580.
ucard -t9	uses the external schema defined in the message series 581 to 600.

External Schema Syntax

You define each external schema using a series of messages in the message file. Each message series must obey the following rules:

- The first line must be a number indicating the number of fields in the schema. It must be prefixed with a single quote.

  For example, to define external schema 3, to have 8 fields, enter:

  461 ’8

- The remaining messages in the series define each of the fields, one message is used for each field. Here is an example entry for a "Personal Organizer" card index:

```
461 ’8
462 ’1:To Do:chr:1,2,3,4:20:20:y:y::
463 ’2::chr
464 ’3::chr
465 ’4::chr
466 ’5:To Call:chr:5,6,7,8:20:20:y::
467 ’6::chr
468 ’7::chr
469 ’8::chr
```

Each message follows a strict syntax:

number:label:type:links:display_width:key_length:key:duplicates:
edit_protected:unused:unused:duplicate_check:display_field_length

Each element in the syntax is terminated with a colon. The following table explains these elements.

Element	Explanation
number	field number.
label	is the text to be displayed beside the field. For example, to display "To Do" beside field 1: `462 ’1:To Do:`

Element	Explanation
type	is the valid datatype in the field. The options are: **Option** — **Explanation** chr — Character type field num — Numeric type field dat — Date type field tel — Telephone type field For example: `462 '1:To Do:chr:` (For further information on valid field types, see the Uniplex II Plus User Guide.)
links	are the numbers of the fields you want grouped together when searching the index using the **find** menu option. When you search a Card Index using the find option, Uniplex finds the entry you made in one field in all the linked fields.The field numbers are separated by commas. For example: `462 '1:To Do:chr:1,2,3,4:`
display width	is the display width of the field when you see the summary list of information. The maximum width is 40 characters. For example: `462 '1:To Do:chr:1,2,3,4:20`
key length	is the length of the key field. By default this is 40 characters. If you have assigned a smaller key to a field, you can enter the width here.
key y¦n	is whether or not the field is keyed. The default is no. The valid entries are: y for yes n or nothing (::) for no For example: `462 '1:To Do:chr:1,2,3,4:20:20:y`

Element	Explanation
duplicates	is whether duplicate entries are allowed in the key field. The default is yes. The valid options are: y for yes n or nothing (::) for no For example: `462 ’1:To Do:chr:1,2,3,4:20:20:y:y`
edit protected	is whether the field is edit protected. The default is no. The options are: y for yes n or nothing (::) for no For example: `462 ’1:To Do:chr:1,2,3,4:20:20:y:y::`
duplicate check	is check for duplicate fields. The options are: y for yes n or nothing (::) for no For example: `523 ’2:Due-Dat:dat::10::y:y::::a’`
display field length	is whether display field length should be equal to display width as specified in schema (default=40 characters). The options are: y for yes n or nothing (::) for no For example: `523 ’2:Due-Dat:dat::10::y:y::::y’` The Due-Date field is of length 10.

Auto-Dialler Configuration

The file *UAP/autodial.ttys* defines the available paths to autodiallers for each user tty. It consists of a number of entries for the ttys on the system, using the syntax:

usertty dialertty dialprogram

where:

usertty is the full path of an operating system tty from which ucard can be invoked. The string **ANY** indicates any tty names not specified in this field.

dialertty is the full path of an operating system tty to dial out to, by a ucard user logged in to *usertty*. A hyphen (-) can be used to indicate that the autodialler for this user is attached to the printer port on the users terminal. The string **NO** indicates that the user on this tty is not allowed to use the auto-dial facility.

dialprogram is the name of the program which will perform the auto-dial. It must be the fully qualified pathname of the program.

For example:

```
/dev/tty10 /dev/tty23 /usr/UAP/unsupported/autodialler
ANY -
```

This will cause user on /dev/tty10 to dialup /dev/tty23 using the named program. No other users will be able to autodial out.

Configuring the Personal Organizer

The directory */UAP/outlines/ufill* contains the files:

easiletter1	*easimemo*
easiletter2	*message*

These files define the format of the mailmerge files for Easiletter, Easimemo and While you Were Out. Easiletter is split into two files, easiletter1, which contains the letter heading, and easiletter2, which contains the letter ending. You can configure these files to include the information you require.

The following is an example of the *easimemo* file:

```
.Vfrom
.Vtitle
.Vto
.Vto2
.Vcc
.Vcc2
.Vdate
.Vsubj
.HM2
.PL62
.FO3
L...........T.........T.......T.......T.......T.......T.......T....M
```

```
memo....memo....memo....memo....memo....memo....memo....memo....memo
```

```
L.......T.......T.......T.......T.......T.......T.......T.......T....M
                                        M E M O
L.......T.......T.......T.......T.......T.......T.......T.......T....R

      From:         _Dfrom
                    _Dtitle

      To:           _Dto
                    _Dto2

      CC:           _Dcc
                    _Dcc2

      Date:         _Ddate

      Subject:      _Dsubj
```

To edit any of these files, first copy make a copy of the directory *UAP/outlines/ufill* in your home UAP.

You can edit any part of the files, except the commands beginning with **.V**, using Uniplex or any other editor. For example, if you want the header margin for Easimemo to be ten lines, change the command **.HM2** to **.HM10** or if you want the title to be MEMORANDUM, simply delete the word **MEMO** and replace it with the word **MEMORANDUM**.

You can also delete commands. For example if you do not want the date to appear on your memo, simply delete the command **_Ddate**.

Note: You can delete existing commands but you cannot insert new ones.

When the file is as you require save and exit the file, then quit from and re-invoke Uniplex. When you use the application, your edited form will be printed instead of the default.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix A
Character Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Overview

Uniplex uses a character set which is a superset of the American Standard Code for Information Interchange (ASCII) set and which defines characters in the "8 bit" range (values 128 to 255) to provide more features such as international and line drawing characters.

This appendix contains:

- A table of the ASCII character set
- A table of the X/OPEN character set extensions

ASCII Character Set

The following table describes the ASCII character set. The way these characters are displayed on a terminal or printer depends on the use of the MAP and HIGHBIT statements in the associated Tcap or Pcap entry.

The table columns are:

Dec The decimal value of the character.

Hx The hexadecimal value of the character.

Oct The octal value of the character.

ASC The ASCII value of the character.

CTL The control sequence for the character (where applicable).

The delete character (decimal 127) is displayed on the terminal as a question mark, in stand out mode (usually reverse video), if the terminal is not configured with a MAP statement in *Tcap* for this character.

Dec	Hx	Oct	ASC	CTL	Dec	Hx	Oct	ASC	Dec	Hx	Oct	ASC	Dec	Hx	Oct	ASC
0	00	000	NUL		32	20	040	SP	64	40	100	@	96	60	140	‘
1	01	001	SOH	A	33	21	041	!	65	41	101	A	97	61	141	a
2	02	002	STX	B	34	22	042	"	66	42	102	B	98	62	142	b
3	03	003	ETX	C	35	23	043	#	67	43	103	C	99	63	143	c
4	04	004	EOT	D	36	24	044	$	68	44	104	D	100	64	144	d
5	05	005	ENQ	E	37	25	045	%	69	45	105	E	101	65	145	e
6	06	006	ACK	F	38	26	046	&	70	46	106	F	102	66	146	f
7	07	007	BEL	G	39	27	047	’	71	47	107	G	103	67	147	g
8	08	010	BS	H	40	28	050	(	72	48	110	H	104	68	150	h
9	09	011	HT	I	41	29	051	)	73	49	111	I	105	69	151	i
10	0A	012	LF	J	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	0B	013	VT	K	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	0C	014	FF	L	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	0D	015	CR	M	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	0E	016	SO	N	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	0F	017	SI	O	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE	P	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1	Q	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	R	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC2	S	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	T	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	U	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	V	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	W	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	X	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM	Y	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB	Z	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC	[	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS	\	60	3C	074	<	92	5C	134	\	124	7C	174	¦
29	1D	035	GS	]	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS	^	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US	_	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

X/OPEN Character Set

This section describes characters with codes in the range 128 to 255, along with a default set of keystroke sequences you can use within Uniplex to generate these characters.

Remember that the display of any of these characters on a terminal or printer will depend on the use of MAP and HIGHBIT statements in the associated Tcap or Pcap entry. See the chapters Configuring Printers and Configuring Terminals in the Device Configuration Guide for more details.

This character set is called the X/OPEN Character Set because the codes from 160 to 255 are taken from the encoding which was recommended for internal use by the X/Open Portability Guide Issue 3. This encoding was originally called ECMA-94, and is now known as ISO 8859/1 or ISO Latin 1. Some of the codes from 128 to 159 are used for Uniplex-specific purposes.

Note: Non-English translations of Uniplex may use MAP statements in Tcap, Pcap and uniplex.cmd to map the characters differently, for example using a different ISO character set.

The table columns are:

Dec	The decimal value of the character.
Hex	The hexadecimal value of the character.
Oct	The octal value of the character.
Chr/Char	The printed image of the character.
Suffix	The one or two characters to type following the X/OPEN leadin key (defined in *uniplex.cmd*) to generate the desired character. Note that, if your keyboard has special keys that correspond to any of these characters, for example an é key, these should be MAPped directly to the character.
¢	The character written to the terminal by Uniplex, in stand out mode (usually reverse video), if the terminal is not configured using MAP and HIGHBIT statements in *Tcap*.
Description	Description of the character.

The following table summarises the characters in the range 128 to 255.

Note: The characters printed in the first Char column are not defined by Uniplex and may vary from printer to printer.

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
128	80	200		160	A0	240		192	C0	300	À	224	E0	340	à
129	81	201		161	A1	241	¡	193	C1	301	Á	225	E1	341	á
130	82	202		162	A2	242	¢	194	C2	302	Â	226	E2	342	â
131	83	203		163	A3	243	£	195	C3	303	Ã	227	E3	343	ã
132	84	204		164	A4	244	¤	196	C4	304	Ä	228	E4	344	ä
133	85	205		165	A5	245	¥	197	C5	305	Å	229	E5	345	å
134	86	206	*	166	A6	246	|	198	C6	306	Æ	230	E6	346	æ
135	87	207		167	A7	247	§	199	C7	307	Ç	231	E7	347	ç
136	88	210		168	A8	250	¨	200	C8	310	È	232	E8	350	è
137	89	211		169	A9	251	©	201	C9	311	É	233	E9	351	é
138	8A	212		170	AA	252	ª	202	CA	312	Ê	234	EA	352	ê
139	8B	213		171	AB	253	«	203	CB	313	Ë	235	EB	353	ë
140	8C	214		172	AC	254	¬	204	CC	314	Ì	236	EC	354	ì
141	8D	215		173	AD	255	-	205	CD	315	Í	237	ED	355	í
142	8E	216		174	AE	256	®	206	CE	316	Î	238	EE	356	î
143	8F	217		175	AF	257	¯	207	CF	317	Ï	239	EF	357	ï
144	90	220		176	B0	260	°	208	D0	320	Ð	240	F0	360	ð
145	91	221		177	B1	261	±	209	D1	321	Ñ	241	F1	361	ñ
146	92	222		178	B2	262	²	210	D2	322	Ò	242	F2	362	ò
147	93	223		179	B3	263	³	211	D3	323	Ó	243	F3	363	ó
148	94	224		180	B4	264	´	212	D4	324	Ô	244	F4	364	ô
149	95	225		181	B5	265	µ	213	D5	325	Õ	245	F5	365	õ
150	96	226		182	B6	266	¶	214	D6	326	Ö	246	F6	366	ö
151	97	227		183	B7	267	·	215	D7	327	×	247	F7	367	÷
152	98	230		184	B8	270	¸	216	D8	330	Ø	248	F8	370	ø
153	99	231	▷	185	B9	271	¹	217	D9	331	Ù	249	F9	371	ù
154	9A	232	◁	186	BA	272	º	218	DA	332	Ú	250	FA	372	ú
155	9B	233	■	187	BB	273	»	219	DB	333	Û	251	FB	373	û
156	9C	234	□	188	BC	274	¼	220	DC	334	Ü	252	FC	374	ü
157	9D	235	●	189	BD	275	½	221	DD	335	Y	253	FD	375	ý
158	9E	236	○	190	BE	276	¾	222	DE	336	Þ	254	FE	376	þ
159	9F	237		191	BF	277	¿	223	DF	337	ß	255	FF	377	ÿ

ISO 8859/1 only defines *8-bit* characters in the range 160-255. It assumes that the low-order 32 characters (128-159) are surrogates for the corresponding 7-bit character. Uniplex uses some of these codes for its own purposes. Uniplex's on-screen representation of these 32 characters is:

Dec	Suffix	¢	UNIPLEX usage[1], where relevant
128	= E	e	Euro sign (default Pcap MAP is 'EUR')
129		.	Abbreviation period *[10]*
130		?	
131		_	EM space *[13]*
132		.	Decimal dot *[12]*
133		_	Thin space *[15]*
134		^	Footnote marker *[9]*
135		?	
136		?	
137		>	Hard tab *[3]*
138		/	Hard line break *[4]*
139		-	EM dash *[14]*
140		?	
141		<	Preferred hard return[2] *[5]*
142		?	
143		?	
144		?	
145		?	
146		*	Remark marker *[6]*
147		?	
148		*	Contents marker *[7]*
149		?	
150		.	Leader dot *[11]*
151		?	
152		.	Index separator character *[8]*
153		?	
154		?	
155		?	
156		?	
157		?	
158		?	
159		?	

The following table describes the ISO 8859/1 characters and the default set of keystroke sequences you can use within Uniplex to generate them.

Dec	Chr	Suffix	¢	Description
160		SPACE	_	Hard Space (NBSP)
161	¡	!	!	Open shriek
162	¢	¦ c	C	Cent
163	£	- L	#	Sterling
164	¤	* X	O	Sputnik/Currency
165	¥	= Y	Y	Yen
166	¦	¦	¦	Vertical Bar
167	§	S S	$	Section (s’s)
168	¨	" "	~	Umlaut
169	©	c O	C	Copyright
170	ª	_ a	a	Feminine ordinal
171	«	<	<	Double Guillemot
172	¬	- !	~	Logical NOT
173	-	- -	-	Soft hyphen (SHY)
174	®	r O	R	Registered
175	~	~ ~	~	Tilde
176	°	* *	o	Ring
177	±	+ -	+	Plus/minus
178	²	2 ^	2	2 superscript
179	³	3 ^	3	3 superscript
180	´	’ ’	’	Acute
181	µ	u	u	Mu
182	¶	¦ P	P	Para. (pilcrow)
183	·	.	.	Center dot
184	¸	, ,	,	Cedilla
185	¹	1 ^	1	1 superscript
186	º	_ o	0	Masculine ordinal
187	»	>	>	Double Guillemot
188	¼	1 4	4	1/4
189	½	1 2	2	1/2
190	¾	3 4	4	3/4
191	¿	?	?	Open query

Dec	Chr	Suffix	¢	Description
192	À	‘ A	A	A grave
193	Á	’ A	A	A acute
194	Â	^ A	A	A c/flex
195	Ã	~ A	A	A tilde
196	Ä	" A	A	A umlaut
197	Å	* A	A	A ring
198	Æ	A E	A	AE dipthong
199	Ç	, C	C	C cedilla
200	È	‘ E	E	E grave
201	É	’ E	E	E acute
202	Ê	^ E	E	E c/flex
203	Ë	" E	E	E umlaut
204	Ì	‘ I	I	I grave
205	Í	’ I	I	I acute
206	Î	^ I	I	I c/flex
207	Ï	" I	I	I umlaut
208	Ð	- D	D	ETH
209	Ñ	~ N	N	N tilde
210	Ò	‘ O	O	O grave
211	Ó	’ O	O	O acute
212	Ô	^ O	O	O c/flex
213	Õ	~ O	O	O tilde
214	Ö	" O	O	O umlaut
215	×	x	x	Multiply sign
216	Ø	/ O	0	O stroke
217	Ù	‘ U	U	U grave
218	Ú	’ U	U	U acute
219	Û	^ U	U	U c/flex
220	Ü	" U	U	U umlaut
221	Y	’ Y	Y	Y acute
222	Þ	P	P	THORN
223	ß	B	B	Sharp s (German)

Dec	Chr	Suffix	¢	Description
224	à	‘ a	a	a grave
225	á	’ a	a	a acute
226	â	^ a	a	a c/flex
227	ã	~ a	a	a tilde
228	ä	" a	a	a umlaut
229	å	* a	a	a ring
230	æ	a e	a	ae dipthong
231	ç	, c	c	c cedilla
232	è	‘ e	e	e grave
233	é	’ e	e	e acute
234	ê	^ e	e	e c/flex
235	ë	" e	e	e umlaut
236	ì	‘ i	i	i grave
237	í	’ i	i	i acute
238	î	^ i	i	i c/flex
239	ï	" i	i	i umlaut
240	ð	- d	d	eth
241	ñ	~ n	n	n tilde
242	ò	‘ o	o	o grave
243	ó	’ o	o	o acute
244	ô	^ o	o	o c/flex
245	õ	~ o	o	o tilde
246	ö	" o	o	o umlaut
247	÷	: -	/	Divide sign
248	ø	/ o	0	o stroke
249	ù	‘ u	u	u grave
250	ú	’ u	u	u acute
251	û	^ u	u	u c/flex
252	ü	" u	u	u umlaut
253	ý	’ y	y	y acute
254	þ	p	p	thorn
255	ÿ	" y	y	y umlaut

In addition to the X/OPEN defined character set, the following characters in the *8-bit* range are understood, by default, by Uniplex (they are defined in the #COMMANDS section of *uniplex.cmd* and so they are understood whether or not the **include=#XOPEN-MAPS** statement is enabled in the terminal's *uniplex.cmd* entry):

Dec	Default sequence	¢	Description
126*	ESC RETURN	~	Explicit hard return
128	ESC **% E**	e	Euro sign
129*	ESC **% .**	.	Abbreviation period
134*	ESC ^	^	Footnote marker
137*	ESC TAB	>	Hard tab
138*	ESC **/**	/	Hard line break
160*	ESC SPACE		Hard space
163	ESC **% #**	#	Pound sign
173*	ESC **% -**	-	Soft hyphen

* Decimal values defined in HARDCHARS string in *uniplex.sys*

Notes:

1 Uniplex usage is dictated by the HARDCHARS array in *uniplex.sys*.

The first two characters in HARDCHARS are HARD SPACE and SOFT HYPHEN. The position of the remainder is given in brackets in the table above.

For example: *[3]* means that HARD TAB is the third character in HARDCHARS

2 HARD RETURN is set, by default, to 126 (the tilde character - ~) to allow compatibility with documents created with vanilla Uniplex 5.xx. The value 141 is the recommended setting if an *8-bit* value is required.

Appendix B
Program Usage and Invocation

THIS PAGE INTENTIONALLY LEFT BLANK

Overview

This chapter lists the programs that make up Uniplex. Unless otherwise stated, all programs are supplied in the *UAP/bin* directory, the exceptions being those programs implemented as shell scripts, which are supplied in the *UAP/cmds* directory. Neither of these directories should be set in users' PATHs since this operation, and others, is performed by the front-end script (*/usr/bin/uniplex*, by default).

Each program is described as follows:

Name

Specifies the name of the program with a brief description of its function.

Synopsis

Specifies the syntax to use when invoking the program.

Description

Describes the function of the program.

Special Files

Describes any configuration files relevant to this program/module only, not described elsewhere in this or other documentation.

Files for the onGO Character Client

For details of files supplied for the onGO Character Clients, see the onGO Administration Supplement.

Uniplex Programs

Name

aid.ufillp - backend for ufill template printing operations

Synopsis

aid.ufillp *mode form*
aid.ufillp -V

Description

Called by ufill, when in formfill mode, to perform printing operations.

Options:

mode	specifies what to print, as:
DF	for data and form
D	for data only
F	for form only
form	is the template name
-V	display version number and exit.

Notes

Implemented as a shell script.

Name

bcheck - database checking utility

Synopsis

bcheck [*options*] *isamfiles*

Description

bcheck checks the integrity of the ISAM files. It can rebuild damaged ISAM files. It compares the index file (.idx) to the data file (.dat). If they are not consistent, bcheck prompts whether to delete and rebuild the corrupted files.

Options:

-i	check only the index file
-l	(lowercase 'L') list the entries in b-trees
-n	respond negatively to every question
-y	respond positively to every question

Notes

This program is installed within the **UAP/informix** directory hierarchy.

Name

chkfile - checks existence, validity and permissions of files

Synopsis

chkfile [*options***]** *listfile*

Description

Checks the existence, validity and permissions of files against the list contained in the specified *listfile*. The *listfile* must contain a list of file and directory names with the file ownerships and permissions that these files should have.

Files are validated by producing a checksum of the file and comparing it against the checksum specified in the specified *listfile*.

The *listfile* may be split into several Uniplex-type sections to enable partial processing of a list. If a section is not specified with the -m option, chkfile processes all of the sections within the *listfile*.

Each line of *listfile* is treated as a separate entry. It must be of the form:

name mode [*owner* [*group* [*checksum*]]] [***comment*]

where:

name	is the file/directory name.
mode	is the access mode needed by the file (octal number).
owner	is who the file should be owned by (optional). This may be either a number (userid) or a name. In the latter case the userid will be derived from the name and an error reported if the user does not exist. Specify **-** to indicate a missing field.
group	is which group the file should be in (optional). Specify **-** to indicate a missing field.
checksum	is a group of check digits that you can use to check for corruption (optional).

***** *comment* specifies that the remainder of the line is a comment.

Options:

-V display version number and exit.

-m *name* process only the files in the specified section *name*, where *name* is a Uniplex section starting #*section_name* and ending with)). Up to 20 sections may be processed by specifying multiple [-m *name*] options.

-s if a file's permissions are not correct they are corrected. Without this option chkfile will simply report errors and not change anything.

-c do not issue warnings if files are found which are not required.

-p check the checksum of all files for which a checksum is given, reporting any errors found.

The exit status of chkfile is usually determined by errors resulting from the system calls stat, chmod and chown.

The return codes are:

0 the checks were successful or errors were overridden with -c or -s.

non-**0** there is an error.

Notes

The checksum algorithm is very simple, and is defeated by duplicating any alterations to the file without altering its size.

Name

dictitod - dictionary conversion program

Synopsis

dictitod *input_file* [*output_file*]

where:

input_file	is the name of the file to be converted
output_file	(optional) is the name to be given to the file produced by the conversion

Description

This program is invoked during installation or upgrade procedures only.

It is used to convert main dictionary files from machine-independent to machine-dependent format.

The suffixes *.lex* and *.indep* are used to identify machine-dependent and machine-independent files, respectively.

If no output file name is specified, the file produced by the conversion is given the same name as the input file, with the suffix replaced by the appropriate one.

Name

gd_fileout - graphics metafile handler

Synopsis

gd_fileout [-V]

Description

Filter for handling output files from Presentation Editor. It takes RGIP commands from stdin, prepares handler information and outputs RGIP commands to stdout. It does not take any flags, other than the standard Uniplex -V flag to display the version number and exit.

Name

gd_matrix - filter for graphics printing

Synopsis

gd_matrix [-V] -p *gcap_section* **[-i] [-m]**

Description

Filter for graphics printing. It takes the RGIP commands from stdin and outputs graphics to stdout.

Options:

-V	display version number and exit.
-p *gcap_section*	use the Gcap *section* to determine the capabilities of the device.
-i	enable interaction mode. The filter will respond to certain RGIP commands sent on its standard input, with the requested information output on the stderr. For example: GetSurface, MouseOn and so on.
-m	enable manual mode (ie: not standalone). Alters the action of the filter so as not to automatically generate internal Open, Initialize and Close sequences.

See also

The description of the graphics device capabilities file, Gcap, in the Uniplex Device Configuration Guide.

Name

gd_plotter - filter for printing with plotters

Synopsis

gd_plotter [-V] -p *gcap_section* **[-i] [-m]**

Description

Filter for printing with plotters. It takes the RGIP commands from stdin and outputs graphics to stdout.

Options:

-V	display version number and exit.
-p *gcap_section*	use the Gcap *section* to determine the capabilities of the device.
-i	enable interaction mode. The filter will respond to certain RGIP commands sent on its standard input, with the requested information output on the stderr. For example: GetSurface, MouseOn and so on.
-m	enable manual mode (ie: not standalone). Alters the action of the filter so as not to automatically generate internal Open, Initialize and Close sequences.

See also

The description of the graphics device capabilities file, Gcap, in the Uniplex Device Configuration Guide.

Name

gd_pscript - filter for graphics printing to postscript devices

Synopsis

gd_pscript *[options]*

Description

A filter for printing Uniplex graphics on postscript devices. It downloads a series of procedures to the device and then relays the RGIP metafile. It takes the RGIP commands from stdin and outputs postscript to stdout. This filter relies on UAP/filters/PostScript which contains the procedures for RGIP graphics.

Options:

-V	display version number and exit.
-d *n*	set the height offset for the image to be *n*, where *n* is a number of dots at 300 dots per inch.
-l *n*	(lowercase 'L') set the left margin indent for the image to be *n*, where *n* is a number of dots at 300 dots per inch.
-h *n*	set the height of the image to be *n*, where *n* is a number of dots at 300 dots per inch.
-w *n*	set the width of the image to be *n*, where *n* is a number of dots at 300 dots per inch.
-r	set resolution. Not currently implemented.
-s	scale the image to the viewport. If this flag is not present, it defaults to make the image fit within the largest possible square within the given viewport.
-b	draw an outline box around the image.

-i enable interaction mode. The filter will respond to certain RGIP commands sent on its standard input, with the requested information output on the stderr. For example: GetSurface, MouseOn and so on.

-m enable manual mode (ie: not standalone). Alters the action of the filter so as not to automatically generate internal Open, Initialize and Close sequences.

Name

gd_tekconfig - graphics filter for Tektronix 4200 series terminals

Synopsis

gd_tekconfig -p *gcap_section* [*options*]

Description

A configurable filter for the Tektronix 4207 and 4208 series of terminals. It takes RGIP commands from stdin and outputs graphics to stdout. It is invoked with the -p flag and a *gcap_section*, specifying a valid section name from the graphics device capabilities file, Gcap. The program will exit if the -p flag is incorrect or absent.

Options:

-V display version number and exit.

-i enable interaction mode. The filter will respond to certain RGIP commands sent on its standard input, with the requested information output on the stderr. For example: GetSurface, MouseOn and so on.

-m enable manual mode (ie: not standalone). Alters the action of the filter so as not to automatically generate internal Open, Initialize and Close sequences.

See also

The description of the graphics device capabilities file, Gcap, in the Uniplex Device Configuration Guide.

Name

gd_tek4010 - generic filter for Tektronix 4010/14 emulation terminals

Synopsis

gd_tek4010 -p *gcap_section* [*options*]

Description

A generic filter for Tektronix 4010/14 emulation terminals. The filter is currently configured to drive the following terminals in their respective Tektronix emulation modes, for which Gcap sections are provided:

Wyse 99GT
Visual 550
Visual 603
Falco 5600

It is invoked with the -p flag and a *gcap_section* name, specifying a valid section name from the graphics device capabilities file, Gcap.

Options:

-V display version number and exit.

-i enable interaction mode. The filter will respond to certain RGIP commands sent on its standard input, with the requested information output on the stderr. For example: GetSurface, MouseOn and so on.

-m enable manual mode (ie: not standalone). Alters the action of the filter so as not to automatically generate internal Open, Initialize and Close sequences.

See also

The description of the graphics device capabilities file, Gcap, in the Uniplex Device Configuration Guide.

Name

gd_x11 - **Rgip** front-end widget

Synopsis

gd_x11 [**-V**] [**-m**] [**-w**] [**-i**] [**-vscrollbar**] [**-hscrollbar**] [**-noTitle**] [**-geom WxH+X+Y**] [**-pipe**] [**-xrm** "*resource_string*"] [**-buttonrows** *n*] [**-name** *string*]

Description

Displays a window capable of accepting and displaying **Rgip** commands. Also supports softkey buttons, mouse buttons and scroll bars.

Options:

-m enables manual mode, that is, turns off automatic Open, Initialize and Close sequences.

-w enables **Rgip** wide-line drawing.

-i turns on interactive mode.

-vscrollbar turns on vertical scroll bar.

-hscrollbar turns on horizontal scroll bar.

-noTitle suppresses title bar at top of application.

-geom alternative method of specifying window geometry to the standard resource ***geometry**.

-pipe work around for deficiencies in some UNIX kernels. (Try this if X returns Select errors).

-xrm allows an X resource string to be passed to the application, which overrides the same resource occurring in a file.

-buttonrows specifies the number of rows of softkey buttons which should appear along the bottom of the widget.

-name allows a name other than the application's to be used when searching for resources.

-V returns version number only.

Name

get.frontend - utility returns name of front-end program

Synopsis

get.frontend [**program** ¦ **directory**]

Description

If invoked with the argument **program**, get.frontend returns the name of the front-end program. If invoked with the argument **directory**, get.frontend returns the pathname or directory containing the front-end program.

Name

getSnames - returns a list of section names

Synopsis

getSnames { **-Pcap** ¦ **-Gcap** ¦ *file* }

Description

This script returns a list of section names (including synonyms) from named file(s):

-Pcap	returns list of names the same for central (only) Pcap file, removing non-printer names.
-Gcap	does same for plotter names from central Gcap.

Name

housekeeping - removes outdated files

Synopsis

housekeeping [-t] [-q] [-v] [{-data¦-trash} [NN]] [-ucleanup [NN]] [-x] [-l]]
or
housekeeping -V

Description

Removes old Uniplex trash and/or temporary files.

Options:

-data [*NN*] removes Uniplex trash files and "Alarms" older than *NN* days. *NN* can be in the range 1-999 and defaults to 15.

-trash [*NN*] Same as -data; argument preserved for backward compatibility.

-ucleanup [*NN*] removes Uniplex temporary files older than *NN* days. *NN* can be in the range 0-999 and defaults to 1; a value of 0 forces all files to be cleaned regardless of age.

The files that are candidates for removal are:

- All files under the Uniplex temporary directory (**TEMP** in uniplex.sys).
- All process-control files (**$Unode/UAP/pc/***).
- If the onGO Character Clients are installed on a client-only system, all files under the onGO **USERTMP** and **SYSTMP/SHARED** directories.

Note that, since they are created in the TEMP directory, this operation will remove **ufmscan** logfiles created with a **-l** argument and no filename.

-x only used in conjunction with '-ucleanup'. The '-ucleanup' operation normally tries not to remove control files for currently active Uniplex sessions. **-x** terminates active Uniplex sessions (except the invoker's session, if the script is run from within a Uniplex session) and removes their files provided the invoker of the script is "root".

-l (lowercase 'L') only used in conjunction with '-ucleanup'. The '-ucleanup' operation normally removes ALL files in the Uniplex temporary directory. However, if the Uniplex temporary directory is set to be the same as a standard UNIX one (if its name does not end with '/uniplex/tmp'), only files with names matching the known ones used by various parts of Uniplex are removed. **-l** forces this 'known files' removal in all cases.

-q ('quiet') no processing messages displayed, other than for errors.

-v ('verbose') message displayed for each operation.

-t ('test') echo, and do not execute, all file removal and process termination commands.

-V display version number and exit.

Examples

```
# Clear all Uniplex work files. This performs the same as the
# "ucleanup" in earlier releases.
# This could be called from /etc/rc re-boot scripts:
housekeeping -x -ucleanup 0

# CRONTAB entry to clear various files every night:
2 0 * * * /usr/UAP/cmds/housekeeping -data 7 -ucleanup
```

Notes

This script locates the Uniplex installation if required, and so can be called without any special variables preset.

It is called from the main menu Delete Old Trash Files and Clean the Uniplex Environment options and by the File Manager file system scanner (ufmscan).

It should normally be run by "root", otherwise it will be unable to remove some files, and cannot verify whether processes are active.

Name

hyusbuild - hyphenation exception file update program

Synopsis

hyusbuild [*options*] *file*

Description

Updates hyphenation exception file(s) with details contained in the specified *file*.

Options:

-c	create a new user exception file. The language of the file is specified by the argument to the **-d** option described below.
-l	(lowercase 'L') list the current contents of the language exception file, as specified by the argument to the **-d** option described below.
-d *language_code*	specify the language of the hyphenation exception file. Where *language_code* is a recognized Uniplex code. For details of valid codes see the later program specification of uspell. Use this option in conjunction with the **-c** and **-l** options described above.
-V	display version number and exit.

Name

install - Uniplex Business Software installation program

Synopsis

install

Description

Uniplex Business Software installation program.

See the Installation Guide for program usage and details.

Name

is_graph - RGIP file verification program

Synopsis

is_graph [*file*][**-V**]

Description

Script to verify whether or not the filename passed to it is a valid RGIP file. This will be decided by searching for the string %RGIP_METAFILE on line 1 of the file.

Exit Status Message:

0	is a graph
1	is not a graph

Name

keygen - Uniplex Key generator

Synopsis

keygen [*options*]

Description

Updates the Uniplex license key file "uniplex.key".

Options:

-V display version number and exit.

-l (lowercase 'L') (for testing only) use **$HOME/UAP/uniplex.key**.

-p list the applications and specify whether KEYED or DEMO versions are installed.

-u '*company name*' **-o** *license* **-i** *first_sequence* **-j** *second_sequence*

update license key file with the company name and license keys.

-U display the following information based on BASE, WP and UW keys.

Number of Uniplex users = nnn
Number of Uniplex Windows users = nnn

-T display the Language, Version, Category (user level), Module Code, Stock (key) Number and number of users for each valid key in the license key file. See UNote 018 for a detailed description of these codes.

-N display the Company Name.

-n display the Company Name and main key number.

If none of the above options is specified, the key file is updated interactively.

Name

make.mmerge - standard letter creation program

Synopsis

make.mmerge *filename* [**-V**]

Description

Creates a standard letter *filename*, if one does not already exist.

Option:

-V display version number and exit.

Name

ongo.link - links up UAP/NVO to a fuller NVO area

Synopsis

ongo.link [**-f**|**-F**] [**-v**] **NVO** *path* **UAP** *path*
ongo.link -already_linked

For example:

ongo.link /work/NVO /uniplex/current/UAP

Description

*Note: This program is located in the directory **UAP/install.cmds** and communicates with the invoker in the English language only.*

This program is run to link up UAP/NVO to a fuller NVO area installed before or after Uniplex and containing any or all of:

UOS
onGO Office

After Uniplex and onGO (Motif Clients or UOS) have both been installed, this script must be run to ensure that both products are using the same set of onGO services (CSB, notifications agents, and so on). The main operations carried out are:

- If the $NVO area contains an older version of onGO than the one on which Uniplex V8 is built, update its core services with the later versions. If it is not possible to automatically determine which is the older version, the user is asked whether to move each required file. The **-f** (force) flag can be used to suppress this prompt and always update $NVO from the files and programs supplied with Uniplex.

- Add files to the $NVO area that are used only by Uniplex.

- Symbolically link UAP/NVO to the real $NVO area.

- If AOS is installed, update $NVO/STATIC/C/csb/csb.cfg with calls to start the Uniplex/onGO notifications agent, unagent.

- If AOS is installed, disable uniplex.rc because all services should now be started with ongo.rc.

Options

-f Suppress prompt when non-standard versions of programs found, and always use the Uniplex-supplied ones.

-F Don't check program versions; always use the the Uniplex-supplied ones.

-v "View mode" - don't move or update any files, just report actions that would be performed if no **-v** flag given.

-already_linked
Used by the install program to ask if $Uredirect is already linked to a $NVO area.

Notes:

1 If either Uniplex or the full $NVO area are upgraded to a later release, ongo.link must be re-run to ensure that the appropriate files are present in $NVO.

2 ongo.link will replace any files in $NVO/bin that it thinks are of the wrong version, but will make backup copies of any other files in $NVO that it changes. Thus, the $NVO/bin area should be backed up before running ongo.link.

Name

onlinedoc - gives access to the on-line documentation

Synopsis

onlinedoc *options*

Description

This script prepares file lists and runs **ufill** screens to let the user access on-line documentation.

Options:

-V	display version number and exit.
-menu_call	script called to show the main on-line documentation form.
-list	used when in **-menu_call** mode to generate file lists.

Name

ped - presentation editor (graphics)

Synopsis

ped [*options*] [*file*]

Description

The presentation editor program, draws simple graphics images, or enhances graphs created using Presentation Graphics (uchart). The optional *file* specifies an existing RGIP metafile to use on invocation.

Options:

-V	display version number and exit.
-B *n*	set the screen background color index to *n*. The default is 5 (BLUE).
-H *n*	set the screen highlight color index to *n*. The default is 3 (GREEN).
-P *n*	set the screenmenu choice box color to *n*. The default is 2 (YELLOW).
-S *file*	set an alternative softkey file.
-T *term*	override the standard terminal type selection selection mechanism and use *term* instead.
-I *n*	(uppercase 'i') set the background color index to *n*. The default is 3 (GREEN).

Name

pfilter - pre-processor for Uniplex print program

Description

A pre-processor for the Uniplex print program, uprop. It filters Uniplex documents, ensuring that the documents contain all the Uniplex controls necessary for printing by uprop. Specifically, this filter checks for and corrects:

Hard returns
Rulers
Page breaks
Print-time merge commands

It passes the filtered text, a line at a time, to uprop. It is automatically called each time the print program (uprop), is invoked.

See also

The print program uprop.

Name

popup - popup window program

Synopsis

popup *n*¦*name* [*arguments*]

Description

Runs miscellaneous Uniplex utilities within a defined window area.

It can be invoked with either a number *n* or a *name* as an argument, as follows:

1 or **calc**	display the calculator
2 or **clock**	display the clock
3 or **decide**	display the decision maker
4 or **life**	run life program
5 or **cal**	display the calendar
6 or **logo**	display the logo
7 or **sketch**	run the sketchpad program
8 or **tz**	display timezones
9 or **usketch**	display the sketchpad
10 or **clipboard**	display the clipboards.
11 or **ptos**	print a document to the screen

Options:

Depending on which name or number is specified, options can be specified as follows:

popup 3¦decide [*filename*]

Where *filename* contains an alternative set of messages that can be used for decision maker. The file format is described below.

popup 9¦usketch [*filename*]

Where *filename* is the input filename.

popup 11¦ptos *fname* [**-r**] [**-w**] [**-p**] [**-S** *file*]

Where:

fname is the file to use as input.

-r specifies to remove the file after viewing.

-w prevents the program looking for a character of value 3 in the file, to indicate that *fname* is complete.

-S *file* specifies use of an alternate softkey file.

-p indicates that the Print to Screen is to link to WP in the Process table and thus permit process switching between ptos and its related WP.

By default, 'popup ptos' expects the *fname* file to have been created by **uprop**, with positioning data prepended to each text line.

To display a file not prepared in this way, preset the environment variable **Unowpdptrack** to any value and use the **-w** flag. For example:

```
Unowpdptrack=no popup ptos -w plain.text.file
```

Special Files

The popup files are contained in the directory *UAP/popup*. There is a small command file **popup.rc** which presently is little used and not configurable. The user can configure the files to use in association with popup timezones and the popup decision maker.

Popup timezones are not configured in the menu system, but may be executed with the command:

popup tz

This will display 6 separate clocks on the screen, showing the times of 6 different cities. Each clock is continuously updated until the quit command is given. The names of the cities used, and their time relative to the user's own time are configured in the file **timezone**.

This file is currently configured as follows:

LONDON : 0, 0
PARIS : +1, 1
SYDNEY : +9, 9
DALLAS : -6, -6
NEW YORK : -5, -5
LOS ANGELES : -8, -8

The syntax for each of the 6 lines is:

name : <*Time difference shown on screen*>, <*Time difference*>

Thus the second clock displayed on the screen will be labelled "PARIS" with a second label "+1" displayed. The time shown on the clock will be one hour later than the user's own time. Use negative numbers for times earlier than local time.

The popup decision maker can also be configured. Users can create their own files to run with this product. To run the decision maker use the command:

popup decide [*filename*]

If a *filename* is not given, the default decision maker will be executed. Otherwise the format of the file should be as follows:

#Title 1
#Title 2
Text 1
Text 2
Text 3
Text 4
Text 5
Text 6
))

The text described here will be displayed in the decision maker as follows:

Title 1 *Title 2*			
Text 1		*Text 2*	
Text 3		*Text 4*	
Text 5		*Text 6*	

The titles should be a maximum of 29 characters and the text a maximum of 10 characters.

Name

pprint - print program

Synopsis

pprint [**-d**] [**-e**¦**-f**¦**-F**¦**-s**] [**-u**] [**-w**¦**-r**¦**-m**[**1**¦**2**]] *file1* [*file2...*]

Description

Prints the specified *file* or files.

Options:

-d	delete source file on completion.
-e	easi-print, use default style and printer.
-f	print using form, showing a fixed message in the "Name of document(s)" field.
-F	print using form, showing the filenames in the "Name of document(s)" field.
-s	print to screen, using default printer and style info.
-u	use $Ufile as location reference, if set.
-w	files are WP documents.
-r	files are Graphics (RGIP).
-m	files are from Mail. For historic reasons, an optional integer, ’1’ or ’2’, may follow th **-m**. Where, ’1’ is ignored and ’2’ is treated as **-f**.

Without arguments, default is easi-print, WP document without deletion.

Name

scrprint - used by Print-to-Screen processing

Synopsis

scrprint *filename*

Description

Utility script used by Print-to-Screen processing.

Name

setservers - porting/integration utility

Synopsis

setservers [-**X11.R3**¦-**X11.R4**] [*file* ...]

Description

Updates specified Uniplex Windows server file(s) to the correct backslash syntax, if necessary. If no file(s) specified, processes all files found in HOME and node ($Uredirect) UAP/XW/servers directories.

By default, reads the port control file (`.installation/install.info`) to decide which version of X11 Uniplex was ported to, and hence how many backslashes to use. The port control file information can be overridden using the argument **-X11.R3** or **-X11.R4**.

Note: Update is in place. So only run this if all files are backed up or otherwise recoverable.

For example:

```
cd $Uniplex
chmod +w XW/servers/*
install.cmds/setservers -X11.R3
chmod 444 XW/servers/*
```

Name

skcomp - uniplex softkey file compiler

Synopsis

skcomp [*options*] *file*

Description

Compiles uniplex softkey files. It takes the argument *file* (usually a '.fn' file) and produces compiled output which is used at runtime. Output files are placed relative to compiled files; they are named with a suffix '.sk'. This suffix replaces any '.fn' suffix to *file*.

For example: **skcomp uniplex.fn** creates **uniplex.sk**
whereas: **skcomp testfile** creates **testfile.sk**

Options:

-V	display version number and exit.
-o *file*	define output file name. The default is to create a new file with the suffix '.sk'.
-v	verbose output. Reports successful compilation.

Name

skcompall - softkey file compilation script

Synopsis

skcompall *UAP_directory_name*

Description

This is a script to help administrators re-compile all "real" softkey files. It runs **skcomp** on all normal mainline (rather than #included) *.fn* files.

Example

```
cd $Uredirect/UAP
skcompall .
```

Name

ssd - Spreadsheet PSF dump utility

Synopsis

ssd [**-V**] [**-l** *row,col*] [**-f**] *save_file*

Description

Spreadsheet PSF dump utility.

Options:

-V	print the version number and exit.
-l *row,col*	(lowercase 'L') print detail of cell stored for row,col for example ssd -l 27,4 t1.ss - cell at D27.
-f	report the format of the file with one of the following 3 messages (always in English language) on stdout: *filename* is NOT a spreadsheet save file *filename* is a PORTABLE FORMAT save file *filename* is a MACHINE DEPENDENT save file
save_file	the name of a portable save file. If a non-PSF file is specified, ssd simply says so and exits.

The output of ssd is a listing of the save file in text format.

Name

syscomp - uniplex command file compiler

Synopsis

syscomp -s [**-i** *file*] [**-o** *file*] [**-l**¦**-c**] [**-v**] [**-Q**] [**-h**]
syscomp [**-l**¦**-c**] [**-v**] [**-Q**] [**-h**] [*term*¦**ALL**]
syscomp -V

Description

Compiles Uniplex terminal, Word Processor and system files.

Options:

-s	compile the **uniplex.sys** file into **system.comp**.
-i *file*	input file to use in place of **uniplex.sys**.
-o *file*	output file to use in place of **system.comp**.
-l	(lowercase 'L') reserved.
-c	use only system files.
-v	verbose output, displaying full pathnames of input files and output files.
-Q	suppress all messages.
-h	output files are placed in users **$HOME/UAP** directory.
term	compile the entry for the named terminal into **termdef**/*term* and **commands**/*term* (defaults to user's TERM value).
ALL	compile all terminals in **Tcap**.
-V	display version number and exit.

Name

thesitod - thesaurus conversion program

Synopsis

thesitod *input_file* [*output_file*]

where:

input_file	is the name of the file to be converted.
output_file	(optional) is the name to be given to the file produced by the conversion.

Description

This program is invoked during installation or upgrade procedures only.

It is used to convert thesaurus files from machine-independent to machine-dependent format.

The suffixes *.lex* and *.indep* are used to identify machine-dependent and machine-independent files, respectively.

If no output filename is specified, the file produced by the conversion is given the same name as the input file, with the suffix replaced by the appropriate one.

Name

tiff2rgp - TIFF to Uniplex RGIP converter

Synopsis

tiff2rgp [*options*]

Description

Converts one or more TIFF files into one RGIP image.

Options:

-V	display version number and exit.
-o *file*	name of file to contain RGIP output.
-i *file*	name(s) of TIFF files to read. Multiple input files are allowed. Flags -X, -Y, -W, -H and -S can be used to qualify each input file to dictate the size and relative position within the RGIP image.
-X *x_offset*	horizontal RGIP offset for the current TIFF file. *x_offset* can be from 0 to 10000 and defaults to 0.
-Y *y_offset*	vertical RGIP offset for the current TIFF file. *y_offset* can be from 0 to 10000 and defaults to 0.
-W *width*	width that the current TIFF file is to take in the RGIP image. *width* can be from 0 to 10000 and defaults to 10000-*x_offset*.
-H *height*	height that the current TIFF file is to take in the RGIP image. *height* can be from 0 to 10000 and defaults to 10000-*y_offset*.
-S	stretch the TIFF image non-proportionally so that it fills the RGIP area. If this flag is not present, the image will retain its original aspect ratio.

Example

```
tiff2rgp -i leftprint.tif -W 5000 -i rightprint.tif -X 5000 -o side-by-side.gr
```

Notes

There are many extensions to the TIFF format, and not all TIFF files can be converted. The compression types of the baseline TIFF standard are supported, and the converter is capable of converting monochrome and greyscale logos and signatures.

Name

ucalc - spreadsheet

Synopsis

ucalc [*options*]

Description

The uniplex runtime spreadsheet program.

Options:

-V	display version number and exit.
-ucalc	override the ‘Ucalcmode’ environment variable to invoke in ucalc emulation mode.
-issi	override the ’Ucalcmode’ environment variable to invoke issi emulation mode (default).
-F *macro*	run softkey macro on startup. If used with a filename, the **-F** ’*macro*’ must precede the filename. For example: `ucalc -F "';calc':F11:F62:'*'" dbupdate.ss`
-M *macro*	same as -F.
-u *file*	*USE* filename.
file	*GET* filename (equivalent to Uniplex edit).
-l *n*	(lowercase ’L’) Set default page length to *n* -override ucdefs/issidefs.
-A	reserved.

-C *n*	set maximum columns to *n* (100-4999) - override ucdefs/issidefs.
-R *n*	set maximum rows to *n* (100-9999) - override ucdefs/issidefs.
-S *file*	Invoke with the softkey file *file*
-T*[n] file*	reserved.
-b	reserved.
-f	force STDROUNDING=off mode. For more details, see the section Define the Spreadsheet Defaults in the chapter Configuring Uniplex II Plus Applications.
-n	do not use scroll bars under Uniplex Windows.
-p *process*	run alternative database back-end instead of "usql". The process should not be named "usql". If used, the spreadsheet should not contain "database" commands. Instead, all "pipe" statements are written to the named process. Ucalc will wait for data being returned, assuming that two consecutive newline characters indicate end-of-response.
-t *file*	reserved.
-w *n*	set default page width to *n* - override ucdefs/issidefs.

See also

The spreadsheet command file compiler, ucbld, and the Configuring Uniplex II Plus Applications.

Name

ucard - card index

Synopsis

ucard [*options*] {**-d**¦**-c**} *index*

Description

The Card Index program. It takes the -d argument with a specified *name* as the card index to use on invocation. Ucard takes the -c argument with a specified *name* as the name of an index to create on invocation. In addition, ucard uses the -t*n* argument, where *n* is a message number from the ucard message file, specifying an external schema.

Options:

-n*n* create the card index with *n* fields (used with the -c option).

-a invoke the card index in ADD mode.

-l*file* (lowercase 'L') Load the records contained in the file into the card index.

The file should be in standard Uniplex "cut format" (tab-separated fields, one record per line).

-*n* invoke the card index and assign field *n* to be the startup search field.

-V display version number and exit.

-1:*search_string* ... **-17**:*search_string*

initial search list. When loading the card index, display only cards that match the *search_string* for the relevant field. For example -1:johnnie specifies to display cards only where the first field contains johnnie.

-c *index* create *index* if it does not exist.

-t *n* create the card index with external schema number *n* (used only with -c).

-d *index*	use card index *index*.
-D	dump all records in address format to stdout.
-P *dict*	use *dict* as the phonetic dictionary for soundex matching.
-S *file*	use *file* as alternative softkeys.

See also

The chapter Configuring Other Uniplex Advanced Office Applications.

Name

ucbld - spreadsheet command file compiler

Synopsis

ucbld [**-V**] [*source-file*] [**-o** *compile-file*]

Description

The spreadsheet command file compiler. It takes the source contained in the specified *source-file* (defaults to **ucdefs**) and compiles it into the runtime command file, used by the spreadsheet program, ucalc.

Options:

-V	display version number and exit.
-o *compile-file*	named *compile-file* for compiled output. By default, the source file issidefs produces issi.rc; ucdefs produces ucalc.rc.
source-file	specify file to contain compiled output. By default invoking ucbld with the **ucdefs** source-file will produce a compile file named **ucalc.rc**, invoking ucbld with the **issidefs** source-file produces a compile file named **issi.rc**

For example, if you change the source definitions file, issidefs, recompile it by entering the command:

ucbld issidefs

Name

ucdiary - onGO Character Time Manager/Diary Client program

Synopsis

ucdiary [*options*]

Description

Note: This program is no longer supplied with UBS.

Invokes Character Client Time Manager.

Options:

-V	display version number and exit.
-S *softkey_file*	use named softkey file instead of the default.
-x *screen_name*	starts processing at named screen. Refer to **ucmail** description.
-c *name*	invoke with named calendar selected rather than the current user's personal calendar.
-H	loads ucdiary with the *Monthly View* showing; exits on QUIT from here, rather than returning to the menu. This is the same as the -x H option to **usdiary**, which is not supported by **ucdiary** since it conflicts with the -x *screen_name* option. However, the front-end script (cmds/udiary) maps -x H to -H.
-D [*level*]	turn on additional logging. Refer to **ucmail** description.

-D (*continued*) Other **usdiary** flags implemented using -x are:

Flag	UCDIARY equivalent
-x A	-x CTEvent
-x B	-x CTAddEditCopyEvent
-x L	-x CTSchedule

Note that arguments -a, -r and -x E are processed by the front-end script UAP/cmds/udiary which always calls **usdiary** to process them. This script also implements flags listed above.

Environment Variables

As **ucmail**, supports the U_dirpagesize and U_DEBUG variables. Refer to **ucmail** description.

Name

uchart - runtime presentation graphics program

Synopsis

uchart [*options*] [*file*]

Description

The runtime presentation graphics program. The optional *file* specifies an existing RGIP metafile to use on invocation.

Options:

-b *n1 n2* define the graph box size, where *n1* defines the width of the box and *n2* defines the box height. The default is 60 (characters) * 20 (lines). This flag is used in conjunction with -Oc for output to the Word Processor and Spreadsheet on character screens.

-c enable 8 color lines on line graphs.

-I *x*[*n*] (uppcase 'i') input data contained in clipboard number *n* (if *n* is not specified, the default 0 is used). The letter *x* defines the datatype as follows:

- **d** data table only
- **s** data series labels only
- **g** group labels only
- **a** all, data and labels
- **t** complete template

-O *x*[*n*] output data to go to clipboard *n* (if *n* is not specified, the default 0 is used) whenever the graph is saved. The letter *x* defines the output as follows:

- **d** data table only
- **s** data series labels only
- **g** group labels only
- **a** all, data and labels
- **t** template
- **c** a character image of the graph and a .GR command.

-m *file*	invoke with the named section of *UAP/uniplex.menu*.
-s *file*	invoke with the named softkey file.
-V	display version number and exit.
-Xs	run in split screen mode.

Name

uclock - alarm and message utility

Synopsis

uclock [**-T** *filename*] *x* **¦** *y***s ¦** *y***S**
uclock [*options*]
uclock on [**X.U.C**] **¦ off**

Description

Checks for alarms and messages which may have been set using the Uniplex mail and time manager programs. Normally, uclock is set to run silently in background.

The first form of the command is the normal background invocation. The alarms are checked every *x* minutes, *y* seconds, or every minute if no *x* or *y* argument is given. 0 is used to check alarms on startup only. If there are no alarms or messages to process for 5 successive intervals, uclock checks the pc/MASTER file (as for the **-c** option). Also, when delivering alarms for a user, uclock automatically corrects any pc/MASTER slots for that user that are incorrectly marked active ('A').

The second form of the command is used for housekeeping.

The third form of the command is used to enable/disable alarms for a current user.

Options:

-c	check all active ('A') slots in the pc/MASTER file. If any of them are for no-longer-active Uniplex sessions, remove any associated pc/*Uppid* file and mark the MASTER entry as dead ('X').
-d *n*	delete all alarms more than *n* days old (default is 30 days).
-l [*username*]	(lowercase 'L') display all alarms, or display all alarms for a given user.
-**r** *username*	remove all alarms for a given user.
-**s**	list default **X.U.C** value, and all "stop" file entries.

-T *filename* write debugging data to the named file.

-**u** list all users and the devices they are on.

on the uclock **on** argument can be followed by 3 dot-separated numbers allowing detailed control of message delivery:

X specifies whether messages are to be delivered to Uniplex Windows displays (UXW):

0 - no (never)
1 - yes (always)

U specifies whether messages are to be delivered to a Uniplex application on a character terminal (or emulator):

0 - no (never); Only valid if **uclock on 0.0.0**
1 - yes (always)
2 - only if no message could be delivered to a Uniplex Windows Display.

C Specifies whether messages are to be delivered to logged in terminals in asterisked boxes:

0 - no (never)
2 - only if a message could not be delivered to a Uniplex Windows Display.
4 - only if a message could not be delivered to a Uniplex application on a character terminal (or emulator).
6 - only if a message could not be delivered on either a Uniplex Windows Display or a character terminal (or emulator).

uclock on with no **X.U.C** argument is equivalent to **uclock on 1.2.6**.

uclock off is the equivalent of **uclock on 0.0.0**.

Files

uclock runs independently of any terminal, therefore messages written by any process it spawns are, by default, written to a file named $Utemp/uclock.$$ (using Bourne shell expansion). The name and use of this file are specified as message 224 in the file UAP/diary/diary.msg.

diary.rc - the UCLOCK_DEFAULT section specifies the default **X.U.C** value.

diary.alarms.x - "stop" file used to record "uclock off" and "uclock on" requests.

Use of **uclock on** or **uclock off** to set the default **X.U.C** value removes any reference to the current user from the diary.alarms.x file.

The entries in the diary.alarms.x file are fixed-length, 20 byte, records of the form:

username: 1 to 8 character login name, or, if record is spare,
either: the 8 characters `"!deleted"`
or: a single binary zero character.

newline: 1 ASCII newline character.

padding: binary 0 characters padding the record to the last byte.

flag: 20th byte is a flag. Prior to Version 7.02, always binary 0. From 7.02 contains flag bits matching the arguments to "uclock on X.U.C" values. For example:

0x00 - Alarms off (**uclock off**)
0x2C - Old style alarms on (**uclock on 1.1.4**)

uclock Limitations

uclock can deliver alarms for up to 1024 logged-in users. Up to 256 of these can be in the "stop" file.

See also

The chapter Configuring Other Uniplex Advanced Office Applications.

Name

ucmail - onGO Character Client Mail program

Synopsis

ucmail [*options*]

Description

Note: This program is no longer supplied with UBS.

Invokes Character Client Mail.

Options:

-V display version number and exit.

-S *softkey_file* use named softkey file instead of the default.

-x *screen_name* start processing at named screen, rather than normal default. As with all the other *alternate entry point* command line arguments, on QUIT/EXIT from the named screen the entire application exits. Reserved for use by Uniplex. Screen names depend on application, only certain screens can be invoked in this way. No attempt to validate the name is made by the application, other than that it is more than one character long (to catch **usdiary**-style -x *menucode* type calls). For example, run the *Directory Manager* by invoking:

```
ucmail -x CdDirManage
```

-f*filename ...* load the Send Form with the specified file(s) already attached as the first body part. For compatibility with **usmail**, the first/only *filename* may be a separate argument or part of the same argument.

For example:

```
ucmail  -f$Uclipboard
ucmail  -f  $Uclipboard
```

Once loaded, you can use F3 to edit the first body part file.

-F*filename ...*	same as **-f** except that the named file(s) are assumed to be *attached files* only. Use F3 to create a new first body part as a cover note.
-i [*trayname*]	start **ucmail** *from the top* but at the list of mail trays. That is, if the user QUITs from these, return to the main menu as usual. If *trayname* is specified and there are messages in the tray, start with the message list instead. A hyphen can be used as a *trayname* to represent the default tray.
-I [*trayname*]	(uppercase 'i') same as **-i** except that when the user QUITs from whichever list **ucmail** started with (mail trays or messages), the user exits from **ucmail** rather than returning to the Mail menu. Also, if *trayname* is specified and that mail tray has no messages, a dialog is presented and pressing any key exits **ucmail**.
-s	start **ucmail** as if *Send a Letter* had been selected from the main menu. Exit at the end of the send operation.
-D [*level*]	*level* is an integer (0-2) requesting debugging of this level and greater. The default is 0 (least available). Note that some levels are only available to developers. Output is to the *filename* specified in the U_DEBUG variable, or stdout if this is not set.

Environment Variables

As well as the generic set of Uniplex environment variables (Uproc, UTEST, etc), **ucmail** recognizes the following:

- UCMAIL_cache=*NN* sets the ucmail default list cache size to *NN*. If this variable is omitted or *NN* is not an integer in the range 1-10000 the default is set to optimize the current specified screen size.

- U_dirpagesize=*NN* sets the number of entries to fetch from the directory server to be held in memory cache at any one time. *NN* is any integer greater than zero. The default is 50.

- U_DEBUG=*file* turns on the writing of debugging information to the named file. This pre-defined information is useful for problem solving.

Name

ucuser - determine onGO Character Client users

Synopsis

ucuser [*username*]

Description

Utility script used to determine whether a named user is configured to use the onGO Character Clients.

Using the onGO-USERS "database", this script returns an appropriate exit code to indicate the status of the named user. If a username is not passed on the command line, reference will be to the environment variables **Uusername LOGNAME** and **USER** (order relevant) until a suitable value is found.

Exit codes are:

0 user is onGO Character Client user
1 user is not onGO Character Client user
9 Invocation error

Name

uc_extern - external browser utility

Synopsis

uc_extern -V
uc_extern -subcall *args*
uc_extern -selftest
uc_extern -**create** *toolid dir file*
uc_extern -**edit** *toolid dir file*
uc_extern -**browse** *toolid dir file filetype*
uc_extern -**open** *toolid dir file filetype*
uc_extern -tables

Description

Shell script used by **ucmail** and **ufilemgr** for two different but interlinked functions. These are, generation of data for **ucmail** control lists and invocation of external browsers and editors (-create, -edit, -open, or -browse). This is implemented as a Bourne shell script, to allow per-site extension (for example, support for other browsers or invocation controls). Read the detailed comments at the end of the script if you want to extend or modify it.

Exit Code

The exit code is set to the exit code of the invoked browser/editor. Invocation is normally synchronous and until the browser/editor exits, **ucmail** remains hidden in the process table.

Uniplex browsers, when run under Uniplex Windows, may invoke an independent (asynchronous) browse window. Editors will always be invoked synchronously.

Options:

-V	display version number and exit. Any invocation errors are reported in English only.
-subcall	used by uc_extern to recall itself to simplify processing.
-selftest	designed for use during porting if there are any problems invoking external browsers or editors. It tests the major SHELL and UNIX logic used and reports whether any of it is missing or faulty.

toolid the editor/browser tool identifier.

dir the name of a directory where *file* is (or is to be created). This will not normally be the same directory as the current working one. For example, most applications need to be passed the pathname as *dir*/*file*.

file the name of a read-only temporary file to browse. This may be a conversion from some other type.

filetype the OpenMail Type Code (*OMTC*) of the file to be browsed. On **-open** calls, the special value **0** is used by **ufilemgr** in conjunction with a fixed *toolid* of **1** to indicate that the Uniplex Word Processor is to be invoked with a text stream created by converting a non-Uniplex file.

-tables the uc_extern -tables output contains various categories of information. Each line consists of space separated words in one of the forms shown below. Lines of the same category are not necessarily grouped together.

N *OMTC*

This is data for the *Never Browse* list and *OMTC* is the type code that will never be browsed.

I *OMTC*

This is the data for the *Internal Browser Formats* list. Some of these types may be treated specially and so would never be presented in the browser as plain messages.

X *OMTC*

This is the data for the *Never Browse Internally* list. No longer generated (this was only used by the Version 8 onGO Character Client Mail program, **ucmail**).

B *OMTC toolid*

This is the data for the *Directly Browsable Externally* list. *OMTC* is the type code and *toolid* is the tool identifier. If there is more than one tool for a given OMTC, **ucmail** will run the first one returned.

T *toolid edit_types*

This lists all editors and browsers that are accessible to the current user. All tools are assumed to be browsers, but some may be editors as well. *Toolid* is an integer tool identifier.

If this tool is only a browser, *edit_types* is the character B Otherwise it is a comma separated list (no spaces) of the OMTCs that this editor can create, or a hyphen (-), to indicate that the editor can create a huge variety of filetypes.

When used as an editor, if *edit_types* contains a single OMTC, **ucmail** will use this as its file type, otherwise it will (re)auto-type the edited/created file.

The presence of a tool in this table indicates that the necessary program is an executable file in the current PATH. Should uc_extern fail to run the program (for example, out of processes) the user will see some form of Operating System message flash on the screen.

END

The last line of output, which informs **ucmail** that this is the end of the table.

Listed overleaf is a typical sample output:

Sample Output	Notes
	Never Browse:
`N 0`	Binary
`N 1277`	Voice message file
`N 1455`	Package
`N 1786`	IBM laser printable file
`N -101`	OpenMail folder
	Internal Browser Formats:
`I 1166`	Distribution list
`I 1167`	Text
`I 1168`	ARPA Message header
`I 1169`	ARPA Distribution list
`I 1590`	UNIPLEX II PLUS
`I 1527`	Non-delivery record
`I 1528`	Mail trace record
	Directly Browsable Externally:
`B 1637 2`	UCALC list file
`B 1639 2`	UCALC PSF file
`B 1638 2`	UCALC old-style save format
`B 1590 1`	Uniplex II Plus (WP)
`B 1167 1`	Text
`B 1167 101`	Text
	Tools:
`T 1 1167,1590`	Uniplex II Plus Word Processor
`T 2 B`	Uniplex II Plus Spreadsheet
`T 101 1167`	VI
`END`	End of table marker

Environment Variables

The calling application must preset the **Ufilter** variable from Tcap information. A non-null value in this variable informs **uc_extern** whether (and how) a character terminal can run the high-resolution graphics browser. **uc_extern** also uses the **Uxwindows** variable to determine whether the terminal is running under Uniplex Windows.

Name

udas - document agent service

Synopsis

udas *options input* [*output*]

Description

Identifies the type of a file, converts it to another type, or reports information about available conversions. *input* is the name of the input file, and *output* is the name of the output file (if required by the operation).

Options:

-**h**	print usage string.
-**H**	print usage string.
-V	display version number and exit.
-**n**	AMS slave (not used by UBS).
-**u**	set UBS (not client/server) mode.
-**d**[*level*]	set debugging mode.
-**S** *name*	requests a service by name.
-**C**	requests a format conversion.
-**F** *filecode*	indicates format of the source file. This option is not mandatory, as the DAS should recognize the file. However you can specify this option if you want to ensure that a particular source file type is used. The filecode can be looked up in the udastc.msg file.
-**T** *filecode*	indicates required output format. The filecode can be looked up in the udastc.msg file.

-**i**	attempts to recognize the filetype of the specified file.
-**I**	(uppercase 'i') as -i, but with more verbose report.
-**R** *filename*	indicates the configuration file to use. By default the DAS uses **$NVO/STATIC/C/SHARED/nvo.cfg**. If your installation is not standard or if you are using a temporary or personal configuration file, specify the name of an alternative file to be used. This file must be in the same directory as nvo.cfg.
-**L**	lists the configuration and checks that tools are available on your computer (according to the PATH). Exits when complete.
-**c**	list possible conversions.

Example

To convert a file from text to a Uniplex Word Processor document, use the command:

```
udas -u -C -F 1167 -T 1590 fred.txt fred.wp
```

Notes

This program is installed in **UAP/NVO/bin**.

Name

udate - display and format the date

Synopsis

udate [**-Q** *99999* ¦ **-d** *date* ¦ **-D** *dd/mm/*[*yy*]*yy*] [**-q** *9999* ¦ **-t** *hh*:*mm*] [**-i**[-]*999*] [**+***format*]

Description

Displays a date. The date and time can be specified by the following optional arguments, otherwise they are as defined by the operating system **date** command:

-Q *99999* — Date is this number of days since 1900 (the internal notation used by the Mailstore and Spreadsheet).

-d *date* — Date is as specified, in the format defined by DATEFMT in uniplex.sys.

-D *dd/mm/*[*yy*]*yy* — Date, in fixed format. A two-digit *yy* is interpreted according to DATEMODE in uniplex.sys.

-q *9999* — Time is this number of seconds since midnight (the internal notation used by the Mailstore).

-t *hh*:*mm* — Time in hours and minutes (the number of seconds is always zero).

-i [-]*999* — Use a date *999* days after (or before if **-i-***999*) the one that would be used if no **-i** specified.

For instance, when run on the 1st of January 2000, the command:

```
udate -i-1 +%d/%m/%y
```

will show

```
31/12/99
```

If the date is specified (**-Q**, **-d** or **-D**) but not the time (**-q** or **-t**), midnight (cf: **-t 00:00**) is assumed. If the time but not the date is specified, todays date is assumed (cf: **-D 'udate +%d/%m/%Y'**).

An optional *format* changes the way the date is displayed. This *format* can contain any text, and may include the following descriptors:

%a abbreviated weekday (Sun to Sat)
%b weekday name in full
%d day of month (01 to 31)
%D date as mm/dd/yy
%e day of month without zero padding (ie: 1st shows as "1", not "01" as it does with %d)
%E date in entry format (DATEFMT in uniplex.sys)
%f month number without zero padding
%h abbreviated month (Jan to Dec)
%H hour (00 to 23)
%k month name in full
%j Julian date (001 to 366)
%m month of year (01 to 12)
%M minute (00 to 59)
%n inserts a newline character
%o suffixed day of month (1*st*, 2*nd*, etc.)
%O reserved (for a Printed Date format, DATEPRTFMT, in uniplex.sys)
%q number of seconds since midnight (value used in Uniplex Mailstore's internal integer time fields)
%Q number of days since 1900 (value used in Uniplex Mailstore's internal integer date fields, and in the Spreadsheet)
%r time in AM/PM notation
%S second (00 to 59)
%t inserts a tab character
%T time as HH:MM:SS
%V week of the year (01 to 53). Weeks start on a Monday. Week 01 of a year is the week containing the first Thursday of the year, and it may include days of the previous year. The week before week 01 is the last week (52 or 53) of the previous year, even if it contains days of the new year. (See the ISO 8601:1988 standard.)
%w day of week (Sunday = 0)
%y last 2 digits of the year
19%y same as %Y (this is provided so that old scripts will continue to work in the year 2000)
%Y year with century (4 digits)
%z abbreviated time zone identifier
%% print a percentage (%) character

Note: %V and %Y mimic the X/OPEN standard "date" command.

The text for day and month names is taken from the file UAP/cmds/text/shell.msg.

Example

udate -Q 35739 "+%d/%h/%y"

The above displays **06/Nov/97**

udate '+%d %h 19%y'

The above displays **21 Jan 2001** if that is the current date.

Name

udiary - Time Manager invocation script

Synopsis

udiary [*options*]

Description

This is a shell script (in UAP/cmds) which invokes either **ucdiary** or **usdiary** (renamed from **udiary** in Version 7), depending on whether the user is registered as an onGO Character Client user or not.

Name

udif_uii - Visicalc™ Dif to Uniplex listfile converter

Synopsis

udif_uii [-V] < *input* **>** *output*

Note: Visicalc™ is a trademark of Visicorp, Inc.

Description

Converts Visicalc™ Dif (Data Interchange Format) files to Uniplex listfile format. This program requires *input* to be a Dif file, *output* is the name of a valid output file.

Options:

-V display version number and exit.

Name

ufbld - customized forms compiler

Synopsis

ufbld *database_name input output error*

Description

The customized forms compiler for the Uniplex database forms package. This program takes the following arguments:

database_name	the name of the database.
input	the name of the source input form.
output	the name of the compiled output form.
error	the name of the file in which to store a log of compilation errors.

See also

The runtime database forms program, uform.

Name

ufile, **ucp**, **umk**, **umv**, **urm** - filing utilities

Description

Performs basic filing actions, ensuring that a file's associated File Manager Index information is maintained as appropriate.

The following utilities are linked to ufile:

ucp *file1 file2* — copy *file1* to *file2*.

umk *file type title_text* — creates the named file with associated Index entry. Type should normally be an integer filetype number (from **NVO.subset/STATIC/C/udas/udastc.msg**). It can also be a string, which is interpreted as:

WPDOC	Uniplex Word Processor file (filetype 1590).
SSDAT	Uniplex PSF Spreadsheet file (filetype 1639).
SSUSE	Uniplex Spreadsheet "use" file (filetype 1637).
GRDAT or **PEDAT**	Uniplex RGIP Graphics file (filetype 1636).
other	Any other value is treated as BINARY (filetype 0).

umv *file1 file2*	move *file1* to *file2*.
umv *dir1 dir2*	rename the specified directory. The directory and the new name must both be within the same parent directory.
urm *files*	remove the named *files*.

Note: ***ufile*** *uses* ***ufilemgr*** *to perform the Index update. If run within a Uniplex session that is already running* ***ufilemgr****, that* ***ufilemgr*** *is used. However, if no* ***ufilemgr*** *is running, one is run up and then closed down for each* ***ufile*** *operation. See also the* ***-i*** *argument to* ***ufmscan*** *for a more efficient way of updating the Index.*

Name

ufilemgr - File Manager application

Synopsis

ufilemgr [**-D** *debug_level*] [**-S** *softkey_file*] [**-p** *IPC_request_file*] [**-x** *screen_name*] [**-q** *[SQL_logfile]*]

Description

This program is auto-invoked by other Uniplex applications when any File Management action is required. The invocation process ensures one copy of the program is shared by all applications within one Uniplex terminal session. The program silently exits when the master process for the session exits.

Options:

-S *softkey_file*
: use named softkey file instead of the default one.

-x *screen_name*
: start processing at named screen rather than normal default.

  Reserved for use by Uniplex. Only certain screens can be invoked in this way. No attempt to validate the name is made, other than that it is more than one character long. Attempts to use non-supported names can result in serious application failures (for example, core dump). **ufilemgr** exits on QUIT/EXIT from the named screen.

-p *IPC_request_file*
: the name of temporary file containing application request. Should only be used on first invocation from a Uniplex application.

-q *[SQL_logfile]*
: log all SQL used to perform database actions. The log includes comments (in usql style) describing what action is being performed. If the log file is not specified, the output goes to the standard output stream, otherwise the output is appended to the specified file.

-D*debug_level*

define the levels of debug logging. The level can be one of:

0 reports all unexpected/serious errors encountered by the application (expands on the information already displayed to the user).

1 produces more information about the cause of the unexpected error.

2 this requires the binary/application to be compiled with DEBUG flag. Reports all of level 1 as well as trace information providing historical record of the running application at key points.

If the environment variable **U_DEBUG** is defined, then all log output is appended to the file named in **U_DEBUG**. If it is not defined, the output is sent to stdout.

If **U_DEBUG** is defined, but no **-D** parameter is given on the command line, the application behaves as if **-D0** was supplied as an argument.

-V report version and exit.

Name

ufilepps - File Manager Index Path Prefix Synonyms management

Synopsis

ufilepps **-c** [**-n** *name*] [*current-path*] *new-path*
ufilepps **-l [-p** *path*] [**-n** *name*]
ufilepps **-V**

Description

Path Prefix Synonyms are multiple UNIX names for the same directory. These exist for one of two reasons:

- A symbolic link to a directory, or
- A shared file system with different mount points from different host systems.

The **ufilepps** utility allows these ambiguous names to be described, so that files underneath such directories have only one Index entry. Shared file system support is provided, but is only of use if you use a networked database for the Index, instead of the default one.

Options:

-c create a synonym *new_path* for the specified *current_path*. If *current_path* is not specified, a separate new entry is added to the Index database for the *new_path*. Both paths are in the form [*hostname***:**]*full_pathname*, where *hostname***:** is optional and specifies the machine on which the *full_pathname* is valid. The *hostname* string is not checked, but can be the reserved string **ALL**, which shows that the specified *full_pathname* is valid on all machines. If *hostname* is not specified, the *full_pathname* is taken to be valid only on the current machine.

-n *name* allows the user to specify a meaningful name for the specified path and all subsequent equivalents. This name overrides any previous name defined for the path and equivalents.

-l (lowercase 'L') lists all synonyms for the specified path or synonym name (specified by the **-p** or **-n** option). The name specified by **-n** must be an exact match for the entry in the synonyms table. The path specified by **-p** is the same format as described above and must be an exact match for a paths table entry. Only one of **-n** or **-p** can be specified. If neither **-n** nor **-p** are specified, all paths and synonyms are listed.

-V report version and exit.

Name

ufill - runtime formfill and screen builder program

Synopsis

ufill [*options***] [***file***] [***arg1 arg2 arg3...***] [-F** *filename***]**

Description

The runtime formfill and screen builder program. It is invoked with the -t option to specify a formfill (office forms package); without the -t ufill defaults to the screen builder (screen based front-end to operating system commands). The optional *file*, specifies an existing source file to access on invocation.

You can also invoke ufill with specific arguments, *arg1 arg2 arg3 .....*. These arguments will then be passed through the specified *file*, to be used as variables in the file, referenced as **$(1)**, **$(2)**, **$(3)** and so on.

For example, if you invoked ufill by entering:

```
ufill -t find invoice
```

invoice would be the first argument. You could then pass this as a default value into one of the fields in the specified file, find. For example:

```
document = type char, default = $(1);
```

Options:

-l	(lowercase 'L') use formfills/screens in the local UAP directory. If this flag is not specified, ufill uses the central UAP directory.
-t	invoke in formfill mode. If this flag is not specified, ufill is invoked in screen builder mode.
-m *name*	startup using the section *name* from the *uniplex.menu* file.
-x *x*	invoke and auto-execute action *x* where *x* is one of the ufill menu actions.
-S *file*	invoke using the named softkey *file*.

-V display version number and exit.

-F *filename* allows you to specify an alternative to the uniplex.menu file in the UAP directory.

See also

The Uniplex Form-Building Tools guide, Formfill and Screen Builder chapters, for details on how to build formfills/screens.

Name

ufilltext - strip text to fit a UFILL auto-expand list

Synopsis

ufilltext [**-lines** *NN*] *file*

Description

This script is a filter used to strip a text stream, down to something that can be comfortably displayed in a UFILL auto-expand list. The output is written to stdout.

Options:

file	name of file to process.
-lines *NN*	indicates that *file* is not a Uniplex Word Processor document, and contains *NN* lines of plain ascii text.

Name

ufmbuild - create and update the File Manager Index database

Synopsis

ufmbuild [**-v**] [**-p**¦**-u**¦**-V**]

Description

This script is called by the installation program to create the File Manager Index database and add some basic entries to it.

It can also be used to add entries to the database from the old "Folios" database.

In both cases, it uses **ufmscan** to do the updates.

Options:

-v verbose mode. Run **ufmscan** in verbose mode to display all update activity.

-p do not build the database, just add entries for standard templates.

-u update the database from the old "Folios" data.

-V Report version and exit.

Notes

This script is installed in **UAP/install.cmds**.

Name

ufmscan - File Manager Index integrity management utility

Synopsis

ufmscan [*options*]

Description

Ensures that the information about documents and folders in the File Manager's index database agrees with the information about the actual files and directories in the UNIX filesystem. It automatically checks the index database integrity and corrects any inconsistencies - purging redundant paths and disconnected index records.

Options:

-a scan the database for all folders which have the "auto-index" attribute set, creating new database entries for any files or folders within those folders which do not already have entries.

-c cleanup the Uniplex environment (just invokes the command **housekeeping -ucleanup**).

-d auto-type documents. Every file with no document type in a folder with the appropriate property set will be auto-typed. This can be a time consuming process. If this flag is not specified, such files have their type set to the type of their parent folders' default template, if available, or BINARY.

-f *foldername*
create an index entry for the specified folder (if it does not already exist), set the auto-index attribute of that folder to show that all files and folders should be auto-indexed, then scan the specified folder and all sub-folders, adding index entries for all files and folders.

-i [*data_file*]

uses data in the named file (defaults to stdin) to create File Manager Index entries.

This operation is designed to be used when upgrading from the earlier "folios" system, and the data must be in the "10-lines per record" format generated by the command **ufomanager -d**. The fields used from the *data_file* are:

Line	Used in File Manager Index for:
1	Filename
2	Title
3	Full pathname of parent folder
4	Not used (owner). The owner of the entry created in the index is set to the current user; this will be automatically corrected on next run of **ufmscan**.
5	Not used (category)
6	Filetype (using same rules as for the **umk** command).
7	Not used (created date+time)
8	Not used (revised date+time)
9+10	Keywords. Each word becomes a separate Index keyword.

The data supplied is not validated. For example, if the strings in lines 1 and 3 are not potential UNIX filenames, the Index may be corrupted. The filesystem is NOT checked when creating Index entries with the **-i** option, and **ufmscan** should normally be re-run after this operation.

-l [*logfile*]

log the actions carried out in the specified file. By default, this will be **$Utemp/ufmscan/UFM_LOG.***day_number*, where *day_number* is the day of the week (1-7 dependant on the UNIX implementation). If **-l** is not specified, the log output is sent to the standard output stream.

-n reports, but does not perform, any "real" updates to the File Manager database (to perform this processing, some temporary records will be added and then removed from the database) or the UNIX file system. Produces the log of actions and any other logs requested (see **-l** and **-q** above).

-o override the setting of the **native** field in the paths table of the File Manager database, so that entries are deleted from the database if they have no matching entry in the UNIX file system (see also **-x**). Only relevant in conjunction with **-r**.

-p *foldername*
same as **-f**, but processes a single directory level only, which must already exist in the index. The scanner will only process files and folders in the named folder, and will not examine the contents of any sub-folders.

-q [*SQL_logfile*]
log all SQL used, as for **ufilemgr**. See **ufilemgr** for details.

-r scan the database, removing all entries for UNIX files and folders which have been deleted (unless they have their "archive" attribute enabled or the **native** field of the paths table entry indicates that these deletes should not be performed; see also: **-x**, below).

-t clean out old trash files (just invokes the command **housekeeping -trash**).

-v verbose mode: report activity, rather than only errors. Use of **-D**, **-l** or **-n** will also set verbose mode.

-x retain information about files and folders that were found to be missing by a previous scanner run.

The scanner uses the folder type and the *pseudo folder* **index-lost&found** (note: there is no leading "/" on this folder name to distinguish it from real ones) with machine name **:DUMMY** to manage information in the index database for "lost" files and folders as follows:

If the scanner fails to find a folder whose **native** field is set to "UNIX DIRECTORY", rather than "MOUNTABLE UNIX DIRECTORY", it changes the **native** field to INVALID. If it finds a folder whose **native** field is already set to INVALID, but which (now) exists, it resets the field to "UNIX DIRECTORY". However, if such an INVALID folder (still) does not exist, it is moved to the pseudo folder (by prefixing its pathname with **index-lost&found/**). The **-x** flag causes such folders to be marked as "MOUNTABLE UNIX DIRECTORY" instead.

If the scanner finds an indexed file which has no entry for its parent directory in the **paths** table (this should not happen!) it moves it into the pseudo folder.

Unless the **-x** flag is given, all entries in the pseudo folder are removed at the beginning of a scanner run. Thus, in normal operation, an active file is only finally removed from the index on the third (sic) scanner run following it being orphaned.

Note that, if the scanner needs to move a file or folder into the pseudo folder but one of the same (full path) name already exists there, the existing one remains, and the newly "lost" one remains orphaned.

-D [*debug_level*]
same as for **ufilemgr**. See **ufilemgr** for details.

-V report version and exit.

The full functionality of the program is only available when run by the "root" user. If not run by "root" the following restrictions apply:

- **-c** and **-t** flags are ignored

- **-a** and **-r** can only be used if either **-f** or **-p** is given.

- only files that belong to the current user and which are contained in directories belonging to him/her are examined.

- the **-x** flag is irrelevant. When run by non-root users, missing folders have their **native** field set to INVALID, but INVALID folders can only be removed by a root user.

*Note: As with the File Manger, folders marked as not accessible (**H** in the UAP/ufilemgr/folder.ctl file) are not processed by **ufmscan**.*

Name

ufomanager - manager for now-retired Office Filing (folios) system

Synopsis

ufomanager -d [*extractfile*]

Description

This program was at the heart of the "folios" system that pre-dated the File Manager's Index database. It is no longer used, except during upgrade from a pre-File Manager version of Uniplex, when it is called to dump all records to *extractfile* (which defaults to stdout). Each record is dumped as ten new-line separated ASCII text lines:

Line	Contains
1	Document name
2	Title text
3	Folder
4	Owner
5	Category text
6	File type code
7	Created date/time as two integer numbers (first is days since 1900 and second is seconds since midnight) followed by a string showing the date and time in human readable format.
8	Revised date/time in same format as created date/time.
9	Keywords text, line 1
10	Keywords text, line 2

Sample output:

```
one.and.only.file
This is the only file I own
/users2/guest/test
test

WPDOC
33840 21240 25/08/92 05:54
33840 21240 25/08/92 05:54
KEYWORDS LINE ONE
REMAINING KEYWORDS ON LINE TWO
```

Name

uform - runtime database forms program

Synopsis

uform [*options*]

Description

The runtime database forms program. It provides a friendly forms based front-end to the Uniplex database.

Options:

-V	display version number and exit.
-d *name*	invoke with database *name*.
-t *name*	invoke with table *name.*
-f *name*	invoke with form *name*.
-m *name*	invoke using the section *name* from the menu file.
-x *x*	invoke and auto-execute menu action *x*.
-c *file*	invoke and use the named cut and paste file (clipboard).
-S *file*	invoke with the named softkey *file*
-xc	execute the form selected with the -f flag.

See also

The customized forms compiler, ufbld.

Name

ugetenv - Uniplex Windows program

Synopsis

ugetenv *variable ...*

Description

This is a program used by Uniplex Windows to return values from system.comp (uniplex.sys).

For example:

ugetenv Utemp
ugetenv Utrash Uusername

Name

ugraph - character graphics filter

Synopsis

ugraph [*options*] [*file*]

Description

The character graphics filter. The optional *file* specifies a graph metafile to use as input on invocation.

Options:

-V display version number and exit.

-E *code* used to invoke ugraph from within the spreadsheet. The *code* is appended to error messages returned by ugraph to the spreadsheet. The spreadsheet uses -E1 to define the **Ctrl a** character as its error code.

-O invoke and do not scan the metafile for % characters at the start of each line in the template section.

Name

uhyph - word hyphenation program

Synopsis

uhyph [*options*]

Description

Word hyphenation program. uhyph reads each word from standard input and displays one or more tab separated words showing the possible hyphenation positions for each word.

If a word contains a soft hyphen, then uhyph removes the hyphens and displays the word followed by a tab separated list of alternate hyphenation options.

If a word contains a hard hyphen, then uhyph will not display any alternatives.

As part of the input for uhyph you can include the command **.UP1** *language_code*, where *language_code* is a recognized Uniplex code (see the later program specification of uspell for details of valid codes). This will change the language dictionary used to check the hyphenation, to the one you specify, for all the words after the command.

Options:

-d *language_code*	use the language dictionary specified. See the later program specification of uspell for details of valid codes. If this option is not present, or if the number used does not appear in the above table, then uhyph assumes American English by default.

-p *directory_path*	look for the language rule files in the given directory, in preference to the standard Uniplex administration directory. If any of the files (?.hyp, ?.exc, ?.lex) are not found, then the standard directory is searched. *directory_path* must be a full pathname, that is it must not contain ../ or ./ sequences.
-V	display version number and exit.

Name

uidmappper - convert application code to name and vice versa

Synopsis

uidmapper [*options*]

Options:

-i *XX*	report application name for the application code XX.
-a *application_name*	report the application code for the named application.
-l	(lowercase 'L') list main application names, with the current $Uidcode one (if any) first.
-V	report version number and exit.

Examples

```
uidmapper -i SS
uidmapper -a 'Screen Builder'
```

Name

uiface - Datalink program

Synopsis

uiface

Description

Shell script which understands the Datalink dbs/interfaces and dbs/interfaces.idx file structures and reports details from them.

See the comments in the script for invocation details.

Name

uimap - IMAP server

Synopsis

uimap [*options*]
uimap.sh

Description

This is a server for the Internet Message Access Protocol -Version 4. When an IMAP client connects to the UNIX system, the Internet daemon (inetd) executes the shell script **uimap.sh.** This then executes the **uimap** process, which validates the UNIX userid and password, and works in conjunction with the binary **umailexec** to allow access to the Uniplex mailboxes of that user.

Options:

-V display version number and exit.

The environment variable UIMAP_LOG can be set TRUE in **uimap.sh** to create log files for debugging.

Files Used

/usr/spool/umail/uimap/imaplog.*process_id* - optional logging information.

Name

uindex - index the named configuration file

Synopsis

uindex [*options*] *file*

Description

Enhances performance when Uniplex is accessing configurable text files. The program scans the named text *file* and generates an index at the top of the file. The index contains an entry and a unique number for each section header in the file. Programs which access these files then use the number to locate the section more quickly and efficiently. This program is used primarily with help files, Pcap, Fcap and Gcap.

Options:

-V display version number and exit.

-r remove the index for the named *file*.

Name

uinfo - installation program utility

Synopsis

uinfo [-t]

Description

Part of the installation script that checks whether Uniplex uses TERMINFO or TERMCAP on the target machine. It returns either TERMINFO or TERMCAP accordingly.

Option:

-t terse format output. As used by the installation program.

Notes

This program, which is supplied in the *UAP/install.cmds* directory, must be invoked from within the main UAP directory with its PATH set to address the **syscomp** program and **Uredirect** set and exported.

Name

ulang - reports the language code

Synopsis

ulang [-v] [-d]

Description

This shell script reports the "language code" used to identify potential text supplied with an add-on module.

Exit code: 0 - language code found and written to stdout
1 - language code could not be determined

Option:

-v reports language information.

-d reports the dial code.

Examples

```
$ ulang
FR
$ ulang -v
Language (general/language):  french
Language code (uniplex.msg):  2
Dial code (uniplex.sys/SPELL):  033
```

Name

ults_uii - Lotus 1-2-3™ to Uniplex listfile converter

Synopsis

ults_uii [*options***]** *input output*

Description

The Lotus 1-2-3™ to Uniplex listfile converter. This program requires *input* to be a valid Lotus 1-2-3™ spreadsheet file; *output* is a file in Uniplex spreadsheet LIST/USE format.

Note: Lotus 1-2-3™ is a trademark of Lotus Development Corporation

Options:

-V	display version number and exit.
-l	(lowercase 'L') remove Lotus 1-2-3™ cell locking during conversion.

Name

umail - Electronic Mail invocation script

Synopsis

umail [*options***]**

Description

This shell script invokes either **ucmail** or **usmail** (renamed from **umail** in Version 7) depending on whether the user is registered as an onGO Character Client user or not.

Name

umailexec - electronic mail controller

Synopsis

umailexec [-V]

Description

The electronic mail controller. It is never called directly, but is invoked as a server by other Mail and Time Manager applications.

Option:

-V display version number and exit.

Files Used

/usr/spool/umail/LOGFILE - unexpected errors.

Unless specified otherwise, the temporary files used by all mail programs except **usmail** are located in $Utemp and $Utrash/umail, which are created if necessary.

Name

umd_clean - electronic mail administration program

Synopsis

umd_clean [*flags***]**

Description

The mail administration program. By default, it performs the following tasks:

- It scans the mail database for letters which have not been read within the period specified by the sender and notifies the sender that his mail has not been read.
- It processes delayed-send, that is future, mail.
- It cleans outgoing mailboxes, removing non-saved messages that are older than the value of DAYS, in umail.rc.

It also has options for deleting old incoming mail.

umd_clean uses the mail backend, umailexec, to post notifications of unread mail to the sender. It is automatically set up as an alarm to run once a day.

Options:

-V	print version number and exit.
-D	turn on debug mode.
-N	no action. Just report what would be done if **-N** not present, using hard-wired English language messages.
-L	use $HOME/UAP/umail/umail.rc not central one.
-A	reserved (memory logging in development versions).
-d	process "D" directory(s) (non saved letters).
-f	process "F" directory(s) (future deliveries).

-i	process "I" (uppercase "i") directory(s) (incoming mail), deleting read messages older than the currently selected time (DAYS, or **-Oi** flag).
-iu	same as **-i** but deletes unread messages too.
-n	generate Unread Letter notifications.
-s	process "S" directory(s) (saved letters).
-t	cleanup trashcan mailbox(es).
-u <*user*>	only process mailbox(es) for specified user. Multiple **-u** arguments can be used to specify more than one user.
-x <*user*>	process mailbox(es) for all users except the specified user. Multiple **-x** arguments can be used to specify more than one user.
-m <*mbox*>	process specified mailbox only, rather than all. The name is the user-visible mailbox name. For example: "general" or "personal".
-Ot <*date*>	override the value for TRASHDAYS in umail.rc. <date> can be either a number of days (as in TRASHDAYS) or a specific date in uniplex.sys DATEFMT. For example, `umd_clean -Ot 12/10/92`.
-Oi <*date*>	use this number of days or date, instead of DAYS value from umail.rc, when deleting incoming mail.
-Oo <*date*>	as for -Oi, but for outgoing mail.

Setting no flags is the equivalent to: `umd_clean -d -f -n -t`

Example

```
umd_clean -u sms -u ajk -Ot 0 -Oi 0 -Oo 0
```

Name

umd_runix - remote mail controller

Synopsis

umd_runix [*options***]** *UNIX_mailbox_file...*

Description

umd_runix reads each UNIX file you specify. If the UNIX file name is also the name of a user on this system, in */etc/passwd*, it is assumed to be a UNIX mailbox for that user; all other *UNIX_mailbox_file* names are ignored. If the file contains any data, it is opened and *umd_runix* extracts all valid messages from it, moving their data to the corresponding user's Uniplex mailbox, in *UAP/umail/text/user_name*.

Each UNIX mailbox is *locked* while a copy is taken of the contents, then unlocked while umd_runix processes the copied file.

Options:

-V	display version number and exit.
-D	produce debugging information for each *UNIX_mailbox_file*.
-DV	more verbose version of -D.
-L	by default umd_runix will not use any information in a local umail.rc file. This option forces a local umail.rc file to be used.

See also

The chapter Configuring and Administering Uniplex Business Software Electronic Mail.

Files Used

/usr/spool/umail/LOGFILE.log	-	fatal errors.
/usr/spool/umail/UMAILLOG.*process_id*	-	errors.
/usr/spool/umail/*username*.err	-	errors.
/usr/spool/umail/*username*.log	-	optional logging information.

Name

umerge - mailshot program

Synopsis

umerge [*options***]** *letter_file* < *input* **>** *output*

Description

The mailshot program. The specified *letter_file* is a Uniplex mailshot template. The *input* can be any data in Uniplex paste or record file format. (Omit *input* if umerge is invoked with the interactive -i option.)

Options:

-i	invoke in interactive input mode.
-q	invoke in quiet mode, suppress banners.
-p	invoke and suppress page breaks in output.
-f '*string***'**	invoke and set the field delimiter to *string*.
-r '*string***'**	invoke and set the record delimiter to *string*.
-s *n*	invoke and start processing the input at record *n*.
-e *n*	invoke and end processing the input at record *n*.
-V	display version number and exit.

Name

umsg - message display program

Synopsis

umsg [*options*] [*text*] ...

Description

The message display program. Allows you to display text or existing messages as part of a shell script.

Options:

-n	display all text on the same line. If this flag is not included, each message or text argument is displayed on a separate line.
-m*n*	display message number *n* from the file *UAP/cmds/text/shell.msg* or the file specified with an **-a** option.
-a*dirname/filename*	use the message file **UAP/***dirname/filename***.msg** for all subsequent **-m***n* options. Note that there must be no space between the **-a** and *dirname/filename*, and also that the suffix **.msg** is automatically added to the specified filename.
-M*n*	display message number *n* from the file *UAP/install.cmds/text/install.msg*.
-f *filename*	print the message(s) to the named file, without the screen controls. If *filename* is - (hyphen), the messages will be printed to stdout.
-N	similar to **-n** except that a newline is not written at the end of the text.

-i *input_file*	display the Uniplex document *input_file*, which should not be more than one page long. Text will be shown with effects where appropriate. If the filename is "-" then the data will be taken from the standard input.
-l*x*	(lowercase 'L') display line number *x* from the input file. This line should *not* be effected.
text	any other text is just displayed.

Name

unagent - onGO Notifications Delivery Manager program

Synopsis

unagent [*options*]

Description

Note: This program is no longer supplied with UBS.

Activates onGO Notifications Delivery Manager.

Options:

-V	display version number and exit.
-a	takes alarms from UNSERVER for users logged out. The default is to leave alarms with UNSERVER until a user is seen to have logged in.
-m	multiple Mail Alarms per user. The default (as with **umailexec**) allows only one outstanding mail alarm per user and discards all other alarms.
-D [*level*]	turn on additional logging. Refer to **ucmail** description. Differs from ucmail in that the default level is 2 and the range of levels is 0-9.
-x	ignore settings in diary.alarms.x. Default is not to take alarms for users who have turned Uniplex alarms off (uclock off).

Notes

unagent is normally started using the *$NVO/bin/ustartagent* shell script, invoked from the Connection Service Broker (CSB). It uses the file UAP/diary/.unag.lock to ensure only one unagent runs per machine (see **uclock**'s use of UAP/diary/.uclock.lock).

Environment Variables

Like **ucmail**, supports the U_DEBUG variable. Refer to **ucmail** description.

Name

uniplex - word processor and menu driver

Synopsis

uniplex [*options***] [***document***]**

Description

The word processor and menu driver for the entire product. Without any arguments, this will invoke the main (or startup) menu for the product. The optional *document* specifies a document to edit or create on invocation.

Options:

-c	invoke the word processor and create a new document, if necessary.
-e	invoke the word processor and display the edit a document screen.
-q	invoke and suppress banners (can be combined with **-m** option).
-V	display version number and exit.
-F '*macro***'**	invoke Uniplex and execute the *macro* every time a document is created or edited. Where *macro* is a keystroke command taken from *uniplex.cmd*. For example, to go to the top of a specific document and delete a line each time you edit the document, you enter: **uniplex -F 'F029:F006'** *document* where **F029** is the keystroke command to go to the top of the document and **F006** is the keystroke command to delete a whole line.

-L	override the setting of the Word Processor "Generate Style Information" working option, forcing it to "No". The information can still be generated using the Command Menu Print->Format option, but the Options->Mode menu will not turn the automatic generation back on.
-l	(lowercase 'L') use message 700 as menu title in demo mode.
-m *menuname*	invoke uniplex and display the menu specified.
-P *prod_code*	allow WP to assume the alias of another application for process switching purposes. *prod_code* should normally be one of the recognised 2-character application codes (WP, SS, OF, EM, etc) but can also be a text string up to 16 characters long.
-S *file*	invoke with the named softkey file.
-s *file*	invoke using the compiled *uniplex.sys* information from the named *file* instead of from *system.comp*.
-a	Automode - disables read-ahead optimization.
-n	Reserved (used in earlier versions to request that saved documents are not folioed).

Notes

This program is actually front-ended by a small shell program, usually called 'uniplex' as well, which performs important start-up functions. See the Uniplex Environment chapter for details of the front-end program.

Name

uniplex.first - processing the first time Uniplex is run

Synopsis

uniplex.first *version* [*options*]

Description

This script is called by **uniplex.start** when it has determined that the user has not run this version of Uniplex before.

uniplex.first is responsible for displaying the "Welcome to Version X.XX of Uniplex" messages and the optional warning of any local files that might conflict with new central UAP ones.

Name

uniplex.rc - start onGO services on a Character Client-only system

Description

Note: This command is no longer supplied with UBS.

This is a copy of the **ongo.rc** file, auto-created from the standard onGO version. It is used to start onGO services on a Character Client-only system. It differs from **ongo.rc** as follows:

- It sets NVO using the command *uniplex -W*.
- It forces *client* and *-nomail* operation.
- It does not call $NVO/STATIC/C/installed/environment, but sets the following variables directly:

```
LANG=C
PATH=$NVO/bin:$PATH
```

The **uniplex.rc** command should be used instead of **ongo.rc** on a Character Client only system. On a system with other onGO components installed, on which **ongo.link** has been run, only **ongo.rc** should be used to start services.

Name

uniplex.start - script called by front-end

Synopsis

uniplex.start

Description

Script called by main front-end (for example, **/usr/bin/uniplex**) to perform the majority of front-end startup processing.

See the section, The Uniplex Startup Script, in The Uniplex Environment chapter for more information.

Name

uniplex.stop - script called when MASTER process exits

Synopsis

uniplex.stop [*sessionid***]**

Description

New script called when MASTER process exits. Calls **UAP/termreset/$TERM** and deregisters an onGO Character Client session, if required.

Where:

sessionid	is the session identifier used by any onGO Character Clients called during this Version 8 session.

Name

upath - UAP file locator

Synopsis

upath [-a] [-g] [-p] [-U] [-V] *target1 [target2] ... [targetn]*

Description

Searches the UAP directory structure, in the order $Ulocal -$Unode - $Uniplex, for the file or files named in the *target* parameter/s.

The result returned is the pathname/s of the first occurrence of the target/s.

Options:

-g	search central UAP areas only; i.e. not $Ulocal.
-a	search all UAP areas.
-p	search UAP areas using the environment variable $Upath rather than $Ulocal, $Unode and $Uniplex.
-U	give usage of this program.
-V	display version number and exit.

Name

upcs - process switch information reporter

Synopsis

upcs [*options*]

Description

Lists process switcher information. By default, lists the master table entries and the process table for the current session.

Options:

-V	display version number and exit.
n	list detail for master table and process table *n*.
-a	list detail for master table and all live process tables.

Name

uphonelist - for use by Phone/Information List popup

Synopsis

uphonelist [*options*]

Description

This script generates information for use by the **Phone/Information List** popup desk option.

Options:

-f *file* use file(s) other than the default, $Unode/PhoneList and $HOME/UAP/PhoneList.

-s do not sort output.

Name

uplot - bit-map graphics plotter

Synopsis

uplot *metafile*[**,***name*] [*options*] **-f** *filter_flags*

Description

The bit-map graphics plotter. This program is called by uprop, the Spreadsheet and the Word Processor to plot graphs. The source for the graph is the graphics *metafile* created by the user in the uchart system. If the metafile contains multiple RGIP (Redwood Graphics Interface Protocol) templates, an optional *name* can be included, specifying the *#section_name* of the required template.

The specified *filter flags* are those passed to the filter, to enable plotting on the specified device.

Options:

-b	plot with a surrounding box.
-c	enable 8 color lines on line graphs.
-d *n*	y offset of viewport for printing.
-E	do not erase display of graph.
-f	pass following flags directly to filter.
-h *n*	height of viewport for printing.
-ic	interactive synchronization flag. 'c' is flushed on stdout after plot.
-l *n*	(lowercase 'L') x offset of viewport for printing.
-m	mask open flag. Invoke filter with -m flag and generate own explicit Open, Initialize and Close sequences.

-p *filtername*	used by uprop when calling uplot for printing. The format is: *filtername* **-p** *Gcap_section_name* For example: -p "gd_matrix -p hp_jet+.MEDIUM"
-o *filename*	output RGIP primitives and Template to named metafile.
-q	quiet mode (default is user interactive mode).
-s	scale viewport flag. If this flag is set, the viewport is scaled to the box size. If it is not set, the default is a square viewport. Note: If the box is not square, a pie chart will be ellipsoid shape rather than circular.
-t	plot graph from Template in metafile. (The default is to plot the graph from RGIP postscript sequences in the metafile.)
-V	display the version number and exit.
-w *n*	width of viewport for printing.

Name

uprop - runtime print program

Synopsis

uprop [*options*] [*file*]

Description

The runtime print program. Output from the program is usually piped to the system spooler, or redirected to a named device.

Options:

-A *n*	set width *n* for an indentation in 10 pitch for index or table of contents.
-a	print alternate pages.
-B	use Word Processor page breaks.
-b *n*	set number of pages from alternate bin.
-C	guess centring around (C) ruler character.
-c *n*	set *n* number of copies.
-cm *n*	set *n* number of copies, without using auto-copy features.
-D '*string***'**	set date to be included in document, where *string* is the date in the format you require.
-d '*string***'**	set default font name, where *string* is a font name defined in Pcap.
-E *n*	stop printing on page *n*.
-Es *n*	stop printing on section *n*.
-e	do not expand tabs.
-F0	scale rulers to page width.
-F *n*	set rulers column width to *n* number of 1/12000 units per inch.

-Fp0	same as -F0.
-Fp *n*	set rulers to pitch *n*.
-Fx	set rulers to the pitch of default fonts.
-f *n*	set form length to *n* lines.
-G '*file***'**	set real file and path name (to be used for deriving relative path names).
-g	do not extend graphics inside margins.
-H '*string***'**	set the selection of pages to print. The specified *string* is a selection of page numbers separated by commas. The '-' character is used to indicate a range, as part of the selection.
-h	reset headers and footers for each file.
-i	enables interactive page control.
-Ii	(uppercase 'i') create index.
-It	(uppercase 'i') create table of contents.
-j	justify at print time.
-K	auto-hyphenate.
-k	allow overlaying of text on graphics.
-L	treat blank lines as normal lines.
-M *n*	offset index by *n* number of spaces.
-m *n*	left margin offset of *n* spaces.
-mB *n*	set bottom margin of *n* lines, for each page specified by the **-b** flag.
-mb *n*	set bottom margin of *n* lines, for subsequent pages.
-mg *n*	set gutter margin of *n*.
-ml *n*	set left margin of *n*.
-mr *n*	set right margin of *n*.

-mT *n*	set top margin of *n* lines, for each page specified by the **-b** flag.
-mt *n*	set top margin of *n* lines, for subsequent pages.
-N '*label***'**	create an index or contents using entries with the named *label*.
-n	output carriage returns with newlines.
-O	force all rulers to be the same length as the longest ruler in the file.
-o	check for widows and orphans.
-P '*name***'**	set paper size to *name,* where *name* is a paper size defined in Fcap.
-p '*name***'**	use the printer section *name*, from the printer capabilities file, Pcap.
-Q	quiet, do not print error messages.
-q	do not call uplot.
-R	reformat at print time.
-r	reset page number for each file.
-S *n*	start printing on page *n*.
-Ss *n*	start printing on section number *n*.
-s '*string***'**	set default reference field to *string,* for index or contents.
-T	do not guess tabs.
-t	print to terminal.
-u	ignore format details in Pre-Styled documents.
-V	display version number and exit.
-W	create the index or contents without leader effect, that is with a ragged edge.
-w *n*	set width of index to *n* characters, in 10 pitch.

-x reserved.

-y uprint emulation mode.

-z page range numbers are physical pages.

-Z page range numbers are as close an approximation as possible to Word Processor page numbers.

Note

You can use the **.UP 0** command to replace *pfilter* with an alternative print program. You enter the command at the beginning of a file in the format:

.UP 0 *program_name*

where *program_name* is the name of the print program to replace *pfilter*.

If you repeat a uprop flag in a single invocation, only the last occurrence is used. For example, if you enter the following line, uprop specifies the paper size to be 8x11 not A4:

```
uprop -e -n -PA4 -j -m10 -mr4 -P8x11 my.file
```

Name

uprtcmd - print interface manager

Synopsis

uprtcmd [*options***]**

Description

Manages the information held by Uniplex describing the printers and print styles available to the user when printing. It can be invoked in many different modes to retrieve and store new printer/style information as required by the print interface.

The user is able to refer to printers and print styles by straightforward names. The information required to allow Uniplex to direct output to the necessary device or spooler and to produce the required output result in the style desired, is maintained by this program. It can be supplied on request in almost any form (usually lists of options for scrolling form fields) with sensible default values supplied.

The program itself has very little knowledge of what the information it maintains is and what it is used for.

Three types of files are used to hold the information: the printer details file, *printers*; and both global and local styles details files, *styles*.

- *printers* - Printer details are held in this system-wide printer definitions file (this should only be modified by the system administrator).

  The file is located in UAP/PRINT.

  The name of this file may be changed using the environment variable Uprinters. For example, the interfaces to the plotter definitions file, UAP/PRINT/plotters, presets Uprinters to the word 'plotters' before calling UPRTCMD.

- System-wide *styles* - Global style details are held in this file. The styles held in this system-wide file are available to all users, and are the standard defaults.

  The file is located in UAP/PRINT.

- Local *styles* - Local style details are held in this file for each user. Details of the current default printer and style for each Uniplex product area is held in a user local file with a set of standard defaults in a central file.

  The file is held in $HOME/UAP/PRINT.

Options:

Note: In the following options product_code is the code for the Uniplex application you require. For example, WP for Word Processor or SS for Spreadsheet.

To set or store printing information supplied by the printing interface:

-n '*availability*' '*style*' '*uprop*' '*ufill*'

Store the style flags.

Where *availability* is either SYSTEM or LOCAL, *style* is the name of the style to be created, *uprop* are the flags that produce the style, *ufill* are the ufill flags that reproduce the style on the style description form.

-r '*availability*' '*style*' '*uprop*' '*ufill*'

Amend an existing style. Set the style name and flags.

Where *availability* is either SYSTEM or LOCAL, *style* is the name of the style to be modified, *uprop* are the flags that produce the style, *ufill* are the ufill flags that reproduce the style on the style description form.

-e '*product_code*' '*printer*' '*style*'

Set the default style and printer for a product area.

Where *product_code* is the code, *style* is the name of style and *printer* is the name of the printer.

-b ['*availability*'] '*printer*' '*Pcap_entry*' '*device_suffix*' '*list_of_users*'

Store the printer, Pcap entry and suffix and applicable users. Where availability is either SYSTEM or NETWORK, the default is SYSTEM. You can enter the string up to 20 times.

-k '*string***'** Remove a print style. The argument is the name of the style.

To retrieve stored information, and pass it back to the printing interface:

-p ['*availability*' '*string*']

Return all printer names. The *string* is the required product code and availability is either SYSTEM or NETWORK. The default is all.

-s ['*string*'] return all style names (local and global). The *string* is the required product code.

-d ['*string*'] return the name of the Pcap entry to use and the name of the device to which output is to be sent. The *string* is the printer name.

-f '*string*' return the uprop flags. The *string* is the style name.

-u '*string*' return the ufill flags. The *string* is the style name.

-g ['*string*'] return default printer and style name. The *string* is the product code.

-a ['*availability*'] return all printer details. The *availability* is either SYSTEM or NETWORK, the default is SYSTEM.

-c ['*string*'] return the uprop flags, and the name of the Pcap entry and device to use for the default printer and style name. The *string* is the required product code.

-i*filename* if *filename* is a Pre-Styled document, information from its format details will be used in preference to the user defaults. This can be used in conjunction with the following arguments:

-c, **-d**, **-f**, **-g**, **-p**, **-s**, **-B**, **-P** and **-S**.

-m [’*availability*’] ’*printer*’ ’*Pcap_entry*’ ’*device_suffix*’ ’*list_of_users*’

Add or modify a single printer definition. The *availability* is either SYSTEM or NETWORK. The default is SYSTEM.

-B [’*availability*’] ’*printer_name*’

Return full details of named printer. The *availability* is either SYSTEM or NETWORK, the default is SYSTEM.

-x [’*availability*’] ’*printer_name*’

Delete the entry for the named printer. The *availability* is either SYSTEM or NETWORK, the default is SYSTEM.

-S [’string’] return default style name.

-P [’string’] return default printer name.

-[0-9] get style name from clipboard (0-9).

-V display version number and exit.

Note:

The NETWORK availability reported by some options is no longer used and will be removed in some future release.

Name

ureport - report writer

Synopsis

ureport [*options***]** *report_template* **<** *input* **>** *output*

Description

The Uniplex report writer. The specified *input* can be any file containing data in Uniplex paste or record file format. The specified *output* is a file for the ureport output.

Options:

-V	display version number and exit.
-q	quiet, suppress banners and error messages.
-f '*string***'**	set field delimiter in the input file to be the specified *string*.
-r '*string***'**	set record delimiter in the input file to be the specified *string*.
-s *n*	start processing at record *n*.
-e *n*	end processing at record *n*.
-b *n*	specify size of sort buffer to be *n* (default is 8192 bytes).

Special Files

The report writer can be configured by editing the command file, *UAP/ureport/ureport.rc*. This is an ASCII text file and contains the keywords associated with the report writer. Each keyword is contained on a separate line and the keywords are *position* dependent. The report writer will take the first non-blank line after the header KEYWORDS to be the first keyword. Each of the keywords can be translated, but the keywords must remain in the same order.

See also

The report writer menu program, urmenu.

Name

urmenu - report writer menu program

Synopsis

urmenu [*options*]

Description

The Report Writer menu program.

Options:

-d *database* specify a database to select.

-S *softkey* specify a file containing an alternative set of softkeys.

-m *menufile* specify a section within *UAP/uniplex.menu* containing an alternative menu.

-V display version number and exit.

-r *reportname* specify a report to use.

-x *n* auto-execute action *n*, where *n* is one of the options listed below:

Letter	Details
t	Prompt for report names
q	List reports; pick and point to edit
p	List reports; pick and point to delete
r	List reports; pick and point to copy
o	List reports; pick and point to rename
s	List reports
x	Run report named in -r option

See also

ureport.

Name

usbld - screen and formfill builder

Synopsis

usbld [-l] [-t] [-n] *input output [error]*

Description

The screen and formfill builder. This program requires *input* to be the name of a source screen, in the *UAP/ufill/usrc* directory; *output* is the name of the compiled output file, in the *UAP/ufill/ubld* directory; *error* is the name of the error log.

Options:

-l	(lowercase 'L') specify that the screen or formfill is for local UAP use. If this flag is not specified, systemwide usage is assumed.
-t	specify that the source screen is to be a formfill, rather than an operating system command front-end.
-n	specify that the program should affect the network layer only.

Example

```
cd $Uniplex/ufill/usrc
usbld print print
```

recreates $Uniplex/ufill/ubld/print.

Name

usdiary - time manager

Synopsis

usdiary [*options***] [-d** *file***]**

Description

The runtime time manager program. The optional *file* specifies an existing diary to access on invocation.

Option:

-m *name*	invoke the time manager, displaying the section *name* from the menu file.
-x *x*	invoke the time manager and auto-execute action *x*, where *x* is one of the time manager menu actions.
-l	(lowercase 'L') list today's events.
-a	add a driver entry, for example umd_runix.
-r	remove a driver entry.
-S *file*	use the named softkey file.
-d *name*	invoke the time manager with calendar *name*.
-V	display version number and exit.

See also

The chapter Configuring Other Uniplex Advanced Office Applications.

Files used

UAP/uniplex.alias	List of usernames and distribution lists.
UAP/diary/.	Configuration and control files.
UAP/diary/diary	Calendar database.

Name

usmail - runtime electronic mail program

Synopsis

usmail [*options***]**

Description

The front-end to the Uniplex mail system. It works in conjunction with the binary umailexec to send and receive mail as well as performing administrative tasks such as creating and deleting mailboxes and setting auto-replies.

If any of the -**b***x* options are present, -**f** must also be present, and the mail is sent without any screen coming up (no video or tty initialisation, and no access to the process table). Argument validation is as on the main screen (e.g. at least one main addressee required, subject only optional if the user's preference allows this, etc).

Exit code from a -**b***x* mode send:

0	All OK
1	Invalid or conflicting arguments (including unreadable *file*).
2	Invalid addressee
3	Send failed for some other reason.

Any non-0 exit code is accompanied by explanatory text on stderr.

Options:

-V	display version number and exit.
-m	send a letter (use Uniplex WP).
-s	send a letter (use mail memo pad).
-**e**	send a letter (Easi Send).

-f *file1 file2 ...* send the first file as the main message. If there are more than one, the second and subsequent ones become attachments (this is the same rule as for *ucmail*). Also a hyphen can be used as a filename to mean use stdin.

-**p** send file via request form.

-i read incoming mail.

-o read sent mail.

-l (lowercase 'L') check details of sent mail.

-a read archived mail.

-n *menu* start with named menu from uniplex.menu file.

-I (uppercase 'i') returns to the Electronic Mail menu when quitting from Incoming Mail.

-R set auto-reply.

-S unset auto-reply.

-F set auto-forward.

-F *file* send the specified file as an attachment.

-G unset auto-forward.

-**L** generate summary log.

-M create a mailbox.

-N remove mailbox.

-bt *to_name* Names of addressees as a comma-separated string.

-**bc** *to_name* Names of CC addressees as a comma-separated string.

-bb *to_name* Names of BCC addressees as a comma-separated string.

-bs *subject* Subject text.

-bm *mailbox* Destination mailbox.

-bos [y¦n] Save copy of mail? With this and the following yes/no arguments, the argument must be qualified with a single lowercase character, "y" or "n". The default is the same as for the sending user's main send form.

-boc [y¦n] Read verification ("certify") required?

-bor [y¦n] Delivered verification ("registered") required?

-bon *999* Send non-read notification after the specified number of days. Set to **0** to suppress non-read notifications. Default is as on the send form.

-bop *9* Set the priority to the specified number. Default is as on the send form.

-bod *dd/mm[/[yy]yy]*
Defer send until this date (must be in dd/mm/yy format, regardless of the format used on the send form).

See also

The electronic mail controller, umailexec and the chapter Configuring and Administering Uniplex Business Software Electronic Mail.

Files used

UAP/uniplex.alias	List of usernames and distribution lists.
UAP/umail/.	Configuration and control files.
UAP/umail/text	Mailstore.
UAP/diary/.	Notification delivery controls.

Name

usort - file sorter for eight-bit characters

Synopsis

usort [*options*] [**-t**[*x*]] [**+***pos1*] [*-pos2*] [**-p***n*] [**-s***m*] [**-o** *output*]
[*file ...*]

Description

A file sorter for eight-bit characters. Usort sorts the file(s) you specify according to the *weight* of the character, rather than its ASCII value. Weights for each character are held in the configuration file *sort.weights*. Uniplex looks for this file first in the directory *$HOME/UAP* and then in *$Uredirect/UAP*. If Uniplex cannot find the file in either of these directories a built-in default is used.

Options:

-u	suppress all but one in each set of equal lines.
-b	ignore leading blanks.
-d	dictionary order. Only letters and digits are significant.
-f	fold lower case letters into upper case.
-m	maintain blank lines in position. Blank lines are placed in the same position in the sorted result as they were in the input, even when the **-r** option is used.
-n	sort by arithmetic value. If several strings are numerically equal, sub-sort by text.
-r	reverse the sense of comparisons.
-t*x*	use *x* as a field separator character. If *x* is missing **\t** is taken to be the field separator. If this option is missing, all white space between non-whitespace characters is taken as one separator.
-p*n*	sort using field *n* as the primary key. This option is overridden by either **+pos** or **-pos**.

-s*m*	sort using field *m* as the secondary key. This option is overridden by either **+pos** or **-pos**.
-o *output*	direct output to the file *output*
+*pos1*	use *pos1* as the start of the primary key, unless this has previously been set, in which case it starts the secondary key. *pos1* has the format *n.m*, which means start the key at *n*+1st field and *m*+1st character. If *.m* is missing it is assumed to be zero.
-*pos2*	delimit the primary key, or secondary key if the primary was previously delimited. The search ends at the character immediately before *pos2*. *pos2* has the format *n.m*. If you do not enter this option the end of the string is assumed to end the key.

Name

uspell - spelling checker and dictionary updater

Synopsis

uspell [*options*] [*filename*]

Description

The spell checking, correcting and dictionary updating program. The optional *file* specifies an existing text file to use as input on invocation.

Options:

-local	perform dictionary functions on local files only.
-system	perform dictionary functions on system files only.
-net	reserved.
-D *language_dictionary*	use the language dictionary specified by the *language_dictionary*.
-d *language_code*	use the language dictionary specified by the *language_code*. If neither of these options are present, or if the number or name used does not appear in the Language table, then uspell assumes American English by Default. *Note: For a list of the language codes, see the table in the section Define the Spell Checking Keywords - SPELL in the chapter Configuring System Parameters.*

-p *directory_path*

look for the language rule files in the given directory, in preference to the standard Uniplex administration directory. If the necessary *.lex* file is not found, then the standard directory is searched.

Note: directory_path must be a full pathname, that is, it must not contain any ../ or ./ sequences.

-y

display a list of all available language dictionaries with their language code.

-L

display a list of all available language dictionaries.

-a

display a list of all supplementary dictionaries to which you have write permissions.

-r

display a list of all supplementary dictionaries to which you have read permissions.

-e

extra information protocol flag. This information, required by the Word Processor, is prefixed to the word and tab separated alternatives.

The format of this record is as follows:

<flag><type><skip><length><lang>

<flag> ::=':'
<type> ::='0' + <record type code>
<skip> ::=' ' + <lead-in count>
<length> ::=' ' + <word length>
<lang> ::= [0-9][0-9][0-9]

record type codes:

0 one alternative correcting case
1 up to 9 alternatives correcting spelling
2 one alternative giving information

-h	invoke uhyph. (See the section in this chapter on uhyph for more details.)
-l	(lowercase 'L') display the line number in the file of each misspelled word.
-m	report misspelled words without offering alternatives.
-s	use the default supplementary dictionaries.
-b '*dict1***' '***dict2***' '***dict3***'**	set default supplementary dictionaries. Where *dict1* is the first supplementary dictionary, *dict2* is the second supplementary dictionary and *dict3* is the third supplementary dictionary. If you only want to set one or two supplementary dictionaries enter **''** in place of the other dictionaries. For example: `uspell -b 've044d51.lex' '' ''`
-u *pathname*	update the supplementary dictionary specified by *pathname*. For example:

```
uspell -u /usr/UAP/dict/ve044d51.lex < /usr/paul/addtodict
```

Uniplex extracts the language code from the *pathname*, then uses it to match any words in the input file (addtodict). The specified supplementary dictionary is then updated with these matched words and Uniplex displays the number of updated words.

-U '*supp_dict***'**	as **-u**, except you need only specify the supplementary dictionary name. For example:

```
uspell -U legal.dict < /usr/paul/addtodict
```

-c '*supp_dict*'	create a supplementary dictionary with the name *supp_dict*. Where *supp_dict* is the text name for the dictionary. This flag must be used in conjunction with **-D** or **-d**, to specify the language, and **-local**, **-system** or **-net**, to specify its location.
-x1	display the first supplementary dictionary.
-x2	display the second supplementary dictionary.
-x3	display the third supplementary dictionary.
-X	display all three current supplementary dictionaries.
-V	display version number and exit.

Name

uspooler - manages the interface with the UNIX spoolers

Synopsis

uspooler [*options*]

Description

This script manages all the interfaces between Uniplex and the UNIX **lp** and **lpr** spoolers. See the script for details of its options. It also contains extensive documentation and self-test logic to allow it to be extended to support other spoolers if required.

Notes

The script is normally driven from the printer description entries in the *UAP/PRINT/printers* file. It looks for a command name there of either **lp** or **lpr** to decide what form of spooler is being used.

Where networked sites use a mixture of over-mapped versions of these commands, the command in the *printers* file must match the correct underlying command used to display print queue information.

For instance, on several Uniplex internal systems, both **lp** and **lpr** can be used to print jobs, and both **lpstat** and **lpq** will report print queues. However, the underlying network spooler is **lpr** and the **lpstat** command actually generates an **lpq** print queue report. In this case, if the **printers** file contains **lp** instead of **lpr** commands, the "Show Print Requests" menu calls cannot be used to cancel print jobs.

Also, for efficiency, on a site with multiple printers, the command in the **printers** file for the system default printer should give the default queue name. Otherwise, the "Show Print Requests" menu call will show requests for all printers, not the one selected in Uniplex, which can take a long time.

Name

usql - database query program

Synopsis

usql [*options***]**

Description

The database query program.

Options:

-b	suppress display of the banner on startup.
-c	clear the screen on startup.
-d *x*	send the specified character, *x*, after a query.
-E *n*	set *n* as the error prefix character, where *n* is the ASCII decimal for the character.
-f*n*	use the specified character, *n*, as format and field separator. Where *n* is the ASCII decimal for the character. Also causes usql to ignore any format commands, and suppresses the printing of the word 'null' for null fields.
-F*n*	use the specified character, *n*, as format and field separator. Where *n* is the ASCII decimal for the character. Use any format statements provided and enter null in any null fields.
-h *n*	specify number of saved history commands.
-i *file*	specify an initialization script *file*.
-I*'char'*	(uppercase 'i') set *'char'* to be the format input separator character, (for example: **-I '\t'**).
-l	(lowercase 'L') do not print blank lines after each response to an usql command.
-m *file*	specify a merge *file*.
-n	do not print the word "null" for null fields in the output of a select statement.

-o *file*	specify the redirection *file* for output.
-p '*string***'**	set the prompt to the specified *string*.
-q	quiet mode, suppress display of banners and prompts.
-u *file*	specify a use script *file*.
-V	display version number and exit.
-w *n*	set the maximum width of columns on output to *n* characters. If *n* is set to 0, there is no limit.
-x	use in conjunction with -f, this causes usql to behave as if a -F option were used.
-X	ignore errors in use files.

See also

The section Configuring Database Query in the chapter Configuring Uniplex II Plus Applications.

Name

ussto123 - convert Uniplex spreadsheets to Lotus 123 format

Synopsis

ussto123 [**-3**¦**-4**] [**-c** *N*] [**-u**] [**-t**] [**-q**] [**-n**] [**-V**] *input*.**ss** **-o** *output*.**wk***?*

Description

Converts Uniplex binary format worksheets into Lotus 123 .WK3 or optionally .WK4 format.

Input and output files must be named on the command line and the output file name should follow DOS naming convention including the extension. Any warnings and errors generated during the conversion process are written to the tty and to an error log file named '*output*.err'.

By default two output files are created. The worksheet file '*output*.wk3' as named on the command line and an associated format file '*output*.fm3'. When copying these files should be kept in the same directory, otherwise text format information will be lost. If the -4 flag is used to generate .WK4 format then there is no secondary file since all the format information resides in the .WK4 worksheet file.

Invocation flags modify the action of the converter as follows:

-4 Generate Lotus 123 .WK4 format worksheet.

-c *N* Convert Ucalc currency code *N* to the Lotus 123 currency format. By default currency code 6, UK Sterling, is converted. The code number refers to currency codes as defined in #FORMAT section of UAP/uc/issdefs. Refer to the chapter Configuring Uniplex II Plus Applications for more information.

-u Convert Ucalc unlocked cells to Lotus 123 unprotected cells. The default conversion protects cells since this is the Lotus 123 default.

-t Include row and column titles in the conversion if present. Row titles appear as text in column A and column titles as text in row 1. All other cells and referencing formulas are offset to accommodate the extra row and column.

-q Quiet mode, do not report conversion warnings to the tty.

-n Do not produce an error log file, even if there are any problems during the conversion.

-V Report the version number and exit.

-D Outputs debugging information to *output*.dbg.

Example

ussto123 -4 -t -c 7 budget.ss -o budget.wk4

Converts the Ucalc worksheet budget.ss to Lotus 123 budget.wk4. Any row and column titles are converted to text in column A and row 1. Any dollar formatted cells are converted to Lotus 123 currency format.

Name

ustartclock - start uclock program in background

Synopsis

ustartclock

Description

This program is always invoked by **usmail**, **usdiary** or **unagent** if, when they start up, they determine that the **uclock** program is not active. It simply starts **uclock** as a background process.

Notes

Implemented as a shell script.

Name

utemplate - removes text from a report template

Synopsis

utemplate < *report_template* **>** *new_template*

Description

Removes all text from a report template. It is useful for printing data only reports. It is called from the Formfill package for this purpose.

Name

uterm - text front-end widget

Synopsis

uterm [**-V**] [**-vscrollbar**] [**-hscrollbar**] [**-noTitle**] [**-geom WxH+X+Y**] [**-xrm** "*resource_string*"] [**-buttonrows** *n*] [**-name** *string*] [**-font***A fontname*]

Description

A window capable of displaying Uniplex text with effects, which can handle resizes, mouse buttons, scroll bars and softkey buttons.

Options:

-vscrollbar	Turns on vertical scroll bar.
-hscrollbar	Turns on horizontal scroll bar.
-noTitle	Suppresses title bar at top of application.
-geom	Alternative method of specifying window geometry to the standard resource ***geometry**.
-xrm	Allows an X resource string to be passed to the application, which overrides the same resource occurring in a file.
-buttonrows	Specifies the number of rows of softkey buttons which should appear along the bottom of the widget.
-name	Allows a name other than the application's to be used when searching for resources.
-V	Returns version number only.
-font*A fontname*	Specifies a named font for fontA, fontB, and so on.

Name

utm_admin - carries out various maintenance tasks for the Time Manager

Synopsis

utm_admin *options*

Description

Performs various Time Manager maintenance tasks.

Note: utm_admin must only be run by the System Administrator or by root.

Options:

Note: For options -u, -d and -p, if no diary-name is specified as an argument, all diaries are processed.

-V	display version and exit.
-u [*diary diary* ...]	reserved.
-d [*diary diary* ...]	delete expired entries from diaries. This is similar to the "Delete Calendar Entries" menu option to **usdiary** except that it does not recover disk space by reallocating the calendar file.
-p [*n*] [*diary diary* ...]	set retention period (used with delete). *n* is the retention period in days, the default is 60.
-r *dir*	reserved.

Example

utm_admin -d -p5

The above would cause all diaries to be searched for entries more than five days old. All such entries would be deleted.

Name

uxinvoke - connects uterm or gd_x11 to application

Synopsis

uxinvoke [**-name** *application*] [**-d** *directory*] [**-b** *button_number*] *uniplex_command_line* \
[**-** **-** *frontend_command_line*]

Or:

uxinvoke **-V**

Description

uxinvoke connects **uterm** or **gd_x11** to the requested Uniplex application. The front-end command, for example, **uterm -vscrollbar** may be placed on the **uxinvoke** command line, or **uxinvoke** can search in the file *UAP/XW/xrule* for the command to run.

Options:

-V	display version number and exit.
-name *string*	gives the Uniplex application an 'alias' which **uxinvoke** searches for in *UAP/XW/config/xrule*. It also passes this argument to the front-end, for example, **uterm -name Word_Processor**.
-d *path*	defines the path for **uxinvoke** to change directory to before running the requested application.
-b *number*	indicates the mouse button used. Allows **uxinvoke** to invoke the front-end process differently for different mouse actions.

Files Used

UAP/XW/config/xrule	defines which emulator to connect to an application. For details, see **uxinvoke** in the Uniplex Windows chapter of the Device Configuration Guide.
UAP/XW/config/xdirectory	optional file, used in earlier versions of Uniplex to define a directory to change to for the named application if the **-d** argument is not given. Although this file is still supported, it is no longer shipped as part of Uniplex.

Name

uxlaunch - places a command onto uxspawn's process queue

Synopsis

uxlaunch [*uxspawn_options*] **-exec** *command*

Or:

uxlaunch -V

Description

uxlaunch places a command onto **uxspawn**'s process queue, invokes **uxspawn** if it is not currently running, and notifies it of a pending command. It is called from the **uxuniplex** shell script.

Options:

-exec should be the last option passed to **uxlaunch**. The text following it is taken to be the required command to be executed by **uxspawn**.

-V display version number only.

Any arguments that **uxspawn** may make use of can be passed to it via **uxlaunch** by placing them before the **-exec** option.

For further details, refer to the section Startup of Uniplex Windows Applications in the chapter Configuring Uniplex Windows in the Uniplex Device Configuration Guide.

Name

uxmsg - places text in an information box

Synopsis

uxmsg [**-n**] [**-m***message_number*] [*text*] ...

Or:

uxmsg -V

Description

uxmsg provides a comparable functionality to **umsg** but places the resulting text in a Motif information box. Text may be entered as an argument to **uxmsg** by specifying a message number stored in the file *UAP/uxwindows/uxwindows.msg* or as a combination of these methods.

Options:

-n does not place a carriage return at the end of the following string.

-m a message number should immediately follow the **m** in this argument to indicate which message is to be loaded from the file *UAP/uxwindows/uxwindows.msg*.

Name

uxprint - program to recognise Uniplex file types and print accordingly

Synopsis

uxprint [*options***]** *filename ...*

Description

A program to recognise a given file as a Word Processor document, Spreadsheet or RGIP file. Uxprint will optionally print a document or file using appropriate easi-printing as defined in the user's UAP/PRINT/defaults file.

Options:

-V display version number and exit.

-P analyses *filename* and returns the product code for the file (for example, WP). As an option, the temporary file uxprint would use to print is also returned. Temporary files are used by uxprint to print spreadsheets. The file itself is not printed.

Name

uxspawn - Uniplex process spawner

Synopsis

uxspawn [*X_arguments*] **-l** *lockfile_path* **-q** *queue_path* **-t***named_pipe_path*

Or:

uxspawn -V

Description

uxspawn is the Uniplex process spawner used by Uniplex Windows. It is notified by **uxlaunch**, via a named pipe, of new commands placed on its process queue to be launched. It also handles the display of **uclock** messages in Motif information boxes and terminates its children when either Uniplex Windows or the X server is shut down.

Options:

X_arguments	because **uxspawn** uses standard X calls to load command lines, such arguments as **-font** and **-display** are supported.
-l	(lowercase 'L') the string following **-l** indicates the full path of a lock file used for communication between **uxlaunch** and **uxspawn**.
-q	the string following **-q** specifies a full path of the **uxspawn**'s process queue.
-t	the string following **-t** specifies the full path of the named pipe used by **uxlaunch** to communicate with **uxspawn**.
-V	display version number only.

See also

The section Running Uniplex Windows in the chapter Configuring Uniplex Windows of the Uniplex Device Configuration Guide.

Name

uxuniplex - link to Uniplex front-end script

Synopsis

uxuniplex uxcopyright

Or:

uxuniplex [**-AOS**] *uniplex_command_line*

Description

uxuniplex is a link to the standard **uniplex** front-end script.

Options:

uxcopyright	outputs the Uniplex copyright message.
-AOS	**uxuniplex** checks that AOS has been installed before invoking the required command. If it has not been installed, **uxuniplex** notifies the user via a Motif dialog box.

Name

wp.browse
wp.convert
wp.edit
wp.print

Description

Sample scripts to support Uniplex Mail/WordPerfect integration. For more details, see the keywords section in the chapter Configuring and Administering Uniplex Business Software Electronic Mail.