UNICOM 3.0

Help Index

Before Starting

Starting UNICOM

Setting Up UNICOM

Connecting to a Remote Computer

Using the Dialing Directory

Screen Editing

- Erasing the Terminal
- Erasing the Scroll Buffer
- Erasing Selected Text
- Copying Selected Text to the Clipboard
- Copying the Window to the Clipboard
- Copying the Scroll Buffer to the Clipboard
- Pasting Clipboard Text to a Remote Computer
- Sending Selected Text
- Selecting All Text
- Searching the Screen Buffer

Printing Operations

- Printer Logging
- Printing the Terminal Screen
- Printing the Screen Buffer
- Configuring the Active Printer

Transferring Files

File Transfer Protocols

XMODEM Checksum, CRC and 1K)
YMODEM (Batch, G)
ZMODEM
Resuming an Aborted ZMODEM Transfer
Kermit
CompuServe B and Quick B
ASCII

WinScript Command Language

Host Mode

Remote User Operation

Advanced Operation

License

UNICOM is not and has never been public domain software, nor is it free software. UNICOM is copyrighted by David Gan and any unauthorized use or use inconsistent with the terms of this license is an infringement of the copyright.

UNICOM may be used without registration on a 21-day trial basis for the purpose of determining whether UNICOM is suitable for your needs. The use of UNICOM, except for the initial 21-day trial, requires registration. Beyond the 21-day trial period, the use of unlicensed copies of UNICOM by any person, business, corporation, government agency or any other entity is strictly prohibited. After the 21-day trial period, you must either register the software or erase it.

A single user license permits a user to use UNICOM only on a single computer. A purchaser of a single user license may use the program on different computers, but may not use the program on more than one computer at the same time.

No one may modify or patch the UNICOM executable files in any way, including but not limited to decompiling, disassembling, or otherwise reverse engineering the program.

A limited license is granted to copy and distribute UNICOM only for the trial use of others, subject to the above limitations, and also the following:

1) UNICOM must be copied in unmodified form, complete with all accompanying files, documentation, order forms and this license information.

2) UNICOM may not be distributed in conjunction with any other product without a specific license to do so from Data Graphics.

3) No fee, charge, or other compensation may be requested or accepted, except as authorized below:

A) Operators of electronic bulletin board systems (sysops) may make UNICOM available for downloading only as long as the above conditions are met. An overall or time-dependent charge for the use of the bulletin board system is permitted as long as there is not a specific charge for the download of UNICOM.

B) Vendors of user-supported or shareware software may distribute only the most current shareware version of UNICOM without specific permission, subject to the above conditions. It is the responsibility of the vendor to obtain the latest shareware version. Vendors may charge a disk duplication and handling fee, which may not exceed eight dollars.

LIMITED WARRANTY

Data Graphics warrants for a period of 60 days from the date of original delivery from Data Graphics that the program will perform in substantial compliance with the documentation supplied with the software product. During the warranty period, if the software does not perform as warranted, the user's exclusive remedy shall be to send all copies of the software and documentation to Data Graphics, which shall, at its sole option, either refund the license fee or repair or replace the software.

EXCEPT AS PROVIDED ABOVE, THIS PRODUCT IS SUPPLIED ON AN "AS IS BASIS" AND DATA GRAPHICS DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE PRODUCT. SHOULD THE PROGRAM PROVE DEFECTIVE, THE USER ASSUMES THE RISK OF PAYING THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION AND ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES. IN NO EVENT WILL DATA GRAPHICS BE LIABLE FOR ANY SPECIAL OR CONSEQUENTIAL, INDIRECT OR INCIDENTAL DAMAGES OR LOST PROFITS (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR BUSINESS INTERRUPTION OR LOSS OF BUSINESS INFORMATION) ARISING OUT OF THE USE OR THE INABILITY TO USE THIS PRODUCT, EVEN IF DATA GRAPHICS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. DATA GRAPHICS LIABILITY TO THE USER HEREUNDER, IF ANY, SHALL IN NO EVENT EXCEED THE PRICE PAID FOR THE LICENSE TO USE THE PROGRAM, REGARDLESS OF THE FORM OF THE CLAIM.

Use of this product for any period of time constitutes your acceptance of this agreement and subjects you to its contents.

U.S. GOVERNMENT RESTRICTED RIGHTS

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013. Contractor/manufacturer is Data Graphics, P.O. Box 58517 Renton, WA 98058 (206)432-1201

Introduction

UNICOM is a complete data communications package especially designed for users of the Microsoft Windows 3 operating environment.

All program features have been designed to operate within the Windows multitasking environment. That is, you can instruct UNICOM to perform a task then switch to another Window application while UNICOM does it's job.

Built-in are many advanced features only found in software costing hundreds of dollars more. In UNICOM you will find:

Many Popular File Transfer Protocols:
Xmodem, Ymodem, Zmodem and their variations, Kermit, CompuServe B, QuickB and ASCII.
Of these, CompuServe B, Quick B and Zmodem protocols have been implemented for automatic downloading.

Color Terminal Emulation with special attribute support. Emulations include: DEC **VT102 (ANSI)**, **DEC VT52, ANSI-BBS and TTY**. Screen fonts, colors and attributes are user selectable. Fonts installed by type managers can be readily used.

A Scrollback Buffer that can hold up to **1500 lines** of information is provided. This buffer can be integrated into the window display or remain hidden until it is needed.

WinScript Windows Script Language. 200 statements and commands are included to provide you with the ability to create complex applications for use with UNICOM.

Script Recording Capability can be used to create command files automatically as you interact with a remote computer.

Script Scheduling allows up to eight WinScript command files to be executed at specific days and times. Programming the scheduler is just like programming your VCR.

Convenient Screen Toolbar gives you quick access to many of UNICOM's menu selected features.

Easy screen editing. Highlight text on the screen using the mouse. You may then copy this selected text to the clipboard, erase it, mark as a file or send it back to the host.

An Expanded Dialing Directory will hold an unlimited number of entries, maintain statistics, program settings and a dialing string for each entry. A directory editor is provided for maintaining the directory. Only disk space limits the number of directories that can be stored and retrieved.

Expanded Host Mode allows remote access to the files in your computer or to DOS.

User ID's, passwords, access levels, login path and time limits may be assigned to each user.
The sysop may monitor interaction, log events, operate in dial-back mode and chat with a remote user.

Hardware and Software Requirements

To effectively use UNICOM 3.0, you will need:

* Microsoft Windows Version 3.
* A personal computer equipped to run Microsoft Windows 3 **efficiently****:
  - An 80386 DX CPU (or higher is recommended, but a fast 286 may suffice)
  - 2 mb of physical ram available to Windows (after smartdrive and ramdrive)
  - A hard disk (28ms or faster) with disk caching software.
  - A Monitor and Video Adapter operating in an efficient video mode: Even with the fastest video board, 256 color display modes may produce a slow scroll or screen response. Sixteen color modes are recommended since performance is usually much faster.

  Very high resolution modes (above 640x480) can also slow performance. Any 'standard' video mode (VGA, EGA) is recommended.
* A Hayes Compatible Modem.
* A Serial Communications Port (if using an external modem).

**UNICOM is a real time application, that is, it must be able to respond to time critical events. If Windows runs 'slowly' on your 286 or 386sx, it is a sign that UNICOM may not be able to operate efficiently, especially when using baud rates above 2400. Loss of data and/or unreliable operation may result if your computer is poorly equipped to run Microsoft Windows.

Users with High Speed Modems (above 2400 bps) will need:
* An 80386 DX CPU (or higher) to deliver the extra performance necessary to run Windows while performing high speed transfers.

Configuring Your Communications Port

IMPORTANT!

The Microsoft communication port driver (comm.drv) requires that **your communications port be set to operate using a unique interrupt (IRQ)**. Unfortunately, COM1 and COM3 usually share IRQ 4 and COM 2 and COM 4 usually share IRQ 3. If your communication port is set to an IRQ used by another device, the Microsoft driver may lock when you try to access the port.

Communication port assignments and IRQs are usually jumper selectable on your computer motherboard, plug-in serial interface card or plug-in internal modem card. Consult your hardware reference manual for information on how to configure these devices.

If your computer is not equipped with COM3 or COM4, the above conflict may not be a problem. You should be aware that other devices may exist in your computer that can be set to operate at a conflicting IRQ. Some devices include : a bus mouse interface card and a sound board.

Version 3.0 of Windows will not allow reassignment of communication port IRQ's to anything other than IRQ 3 or IRQ 4 for AT class machines. This means that at most TWO serial ports can be enabled in your computer at any time. DOS comm packages are not affected by this limitation, only Window applications which use the Microsoft serial communication driver (such as Terminal and UNICOM).

The following table lists the default base port and interrupt levels for COM1 - COM4.

Device	Base Port	Default IRQ
COM1	3F8	IRQ 4 - conflicts with COM3 if it exists
COM2	2F8	IRQ 3 - conflicts with COM4 if it exists
COM3	3E8	IRQ 4 - conflicts with COM1 if it exists
COM4	2E8	IRQ 3 - conflicts with COM2 if it exists

To configure your communications port:

1) Determine which communication port you intend to use:(COM1, COM2, COM3 or COM4)
Configure your serial port to the base port and IRQ for the desired COMx device. This involves setting jumpers or switches on the device. Consult your hardware reference manual.
Ensure that this port is configured to a unique and appropriate IRQ.

Windows versions above 3.0 may support reassignment of IRQ's (and Base Ports) for each COMx device using the control panel. For AT class machines, this will allow Window applications access to all 4 communications ports . Ensure that each port is physically set to a unique IRQ, then use control panel to designate the IRQ for each port.

Installing UNICOM

The UNICOM 3.0 distribution disk should contain the following files:

File	Description
UNICOM.EXE	The UNICOM executable program.
UNICOM.DIR	A sample dialing directory.
UNICOM.KEY	A sample keyboard macro file
UNICOM.CFG	A default program configuration file.
UNICOM.MNU	A sample utility menu configuration file.
UNICOM.HLP	UNICOM 'Windows' Help File.
UNICOM.FON	VT-102 special fonts.
UNICOM.SND	A sound file used to generate notification music
UC3READ.ME	Program release notes.
UC3ORDER.WRI	Product order form in Windows Write format.
UC3ORDER.TXT	Product order form in ASCII text file format
GDI.SCR	Example script demonstrating graphics.
CMPUSRV.SCR	Example CompuServe login script.
MCI.SCR	Example MCI login script.
	Other example script files may be included.

Software Installation Procedure

Copy the UNICOM distribution any subdirectory on your disk drive.

Install UNICOM into a program group within Program Manager:

1) Activate Program Manager and highlight the program group where UNICOM is to be stored.
2) From the Program Manager file menu, select 'New..."
3) A New Program Object Window appears prompting you for the object type. Select the Program Item button.
4) A window appears prompting you for the program item properties.
   a) In the description field enter "UNICOM"
   b) In the command line field, enter the complete pathname of the installed UNICOM executable. For example, if you stored UNICOM in your windows directory the entry would look like: C:\WINDOWS\UNICOM.EXE

5) Installation is complete.

Starting UNICOM

UNICOM may be activated in a number of ways as shown below:

From a RUN command line in Program Manager or File Manager enter:

UNICOM [*configfile.cfg*][*scriptfile.scr*]

From Program Manager as an installed program item in a program group:

Double Click on the UNICOM Icon

From File Manager

a) Double click on a UNICOM script file, -OR-
b) Double click on a UNICOM config file, -OR-
c) Double click on the executable file UNICOM.EXE

When UNICOM is activated for the first time, a file path setup window will appear (shown in Figure 1) prompting you to enter a UNICOM files, upload and download directory.

(see manual)
Figure 1. File Path Setup Window

The files directory should be set to the drive and directory where UNICOM has been installed. The download directory should be set to the drive and directory where files received from data transfers are to be stored. The upload directory should be set to the drive and directory where UNICOM will first look to locate files for upload selection.

Enter the pathnames into the edit fields within the Window. Paths defined here are valid only for the current UNICOM session. To make the paths permanent, activate the SAVE SETUP option from within the SETUP menu. Paths are stored in your Windows WIN.INI file.

An error message will be displayed if any of the path fields contain an invalid directory or if UNICOM could not locate its executable file in the directory specified in the files directory field.

At the start of each UNICOM session, a configuration file will be accessed (from the UNICOM files directory) to determine what communication port will be used and other operating parameters. This configuration file will default to UNICOM.CFG if no file was specified when invoking UNICOM. If UNICOM cannot locate any configuration file, the port will default to COM2, 2400 baud, No Parity, 8 data bits and 1 stop bit.

Should a communication port fail to open, UNICOM will display a message box to indicate the failure. The port configuration window will then be displayed automatically. At this point, a valid communication port should be selected. If no communication ports can be opened and you're sure the port is there, the port IRQ may be set to conflict with another device or port.

When a communication port is successfully opened, UNICOM will try to initialize the modem if the port was configured for a modem connection. Should the message "Modem Not Responding" appear, UNICOM failed to obtain an 'OK' response from the modem. Make sure the communications port and modem are configured properly. Ensure that the modem is set to return VERBOSE responses and that it is not already 'online' connected to a remote computer.

Screen Regions

The top buttons	Description
	Erases selected screen text or the whole terminal screen.
	Search for text in the scroll buffer.
	Transmit selected screen text back to the host.
	Download a file into your computer.
	Upload selected files to a remote computer.
	Access the Dialing Directory.
	Access the Manual Phone Dialer.
	Instruct the modem to hang up.
	Activates the Comm Port configuration window.
	Activates the Terminal Setup window.
	Displays the Modem Setup window.
	Displays the Keyboard Macro Editor.
	Displays the Zmodem Setup window.
	Toggle Host Mode operation on or off.
	Toggle Chat Mode operation on or off.
	Mark a file on the screen for download.
	Answer an incoming data call
	Terminal auto wrap mode toggle.
	Terminal local echo toggle.
	Terminal CR->CRLF toggle.
	Terminal erase on backspace toggle.

Status Line

Program messages, communication settings, terminal settings and menu definitions are displayed on this line located at the bottom of the UNICOM window. If the Dialing Directory is visible, status line messages will be displayed in it's window caption.

Scroll bars

Vertical and horizontal scroll bars allow the user to position the UNICOM screen using a mouse (or via the control menu). The vertical scroll bar allows viewing of the scroll back buffer. The horizontal scroll bar is useful for viewing any columns beyond the right edge of the window.

User Keys

Any of the twenty two keyboard keys may be user defined as hot keys, macros or for activating script command files. The keyboard macro editor is used to define the meaning and screen button label of the twenty two keys which include: F1-F12 and the keypad keys.

Terminal Screen

Characters sent from a remote host are displayed on a 24 line terminal screen. If your window is sized too small to view all the lines, the scroll bar must be used to scroll the terminal top into view. If your window is sized to support more than 24 lines, the scroll buffer can be set to occupy these extra lines starting at the top of your screen.

Scroll Buffer

When your terminal scrolls, lines which scroll of the top are placed in the scroll buffer for later review. This buffer is configurable in size from 80 to 1500 lines. The bottom of the scroll buffer can be integrated into the UNICOM window (on top of the terminal) or hidden from view.

Modem Initialization

UNICOM will automatically send a modem initialization string to your Hayes compatible modem at the start of each session if the connection type is set to 'modem' in UNICOM's communication port setup screen.

Auto Start Script Execution

UNICOM can execute a predefined WinScript command file at the start of each program session To enable this feature, enter the file name of the auto-start script into the corresponding edit box within the general setup window. This window is activated by selecting General from the setup menu.

To disable auto-start script operation, activate the general setup window and remove the filename from the auto-start edit box.

Positioning the Startup Window

The position of and dimensions of UNICOM's startup window is determined by the Startup Window setting in the General Setup window. Three selections are available as described below:

Normal: UNICOM reads its stored window position and dimension from your win.ini file. This information is saved during a Save Setup operation. UNICOM records the position and dimension of the window at the time of the last Save Setup. The next activation (in this mode) will cause UNICOM to examine win.ini for the position and size information. If found, the window will be positioned and sized accordingly. If this info is not found, Windows will determine the position and dimension of the startup window.

Full Screen: The window is zoomed to occupy the entire screen.

Iconic: UNICOM is minimized to an iconic state.

Communication Port Settings

The physical communication link is described to UNICOM using the setup window below in Figure 2. To activate this window, select Comm Port from the setup menu.

For most uses, your communication settings will be Word:8, Parity: NONE and Stop: One
-OR-
Word:7, Parity: EVEN and Stop: One.

To establish successful communication with a remote computer, the following settings must exactly match those of the remote computer: Baud Rate, Parity, Stop, and Word.

If any of these settings are improperly set, the Windows port driver will detect a communication error. UNICOM will report these errors as PORT STATE messages on the status line at the bottom of the window.

(see manual)
Figure 2. Communication Port Setup Window

Select the desired communication settings from this window using a mouse or keyboard.

COM1 through COM4 are shown as available options. If your computer does not support a particular port, an error message will be displayed if an attempt is made to configure it. If another device (or port) is assigned the same IRQ of device you select, the Microsoft Port driver may lock up and appear to freeze UNICOM.

Configuration Option Descriptions:

Port: Specifies the DOS name of your communications port. (COM1 - COM4)

Baud Rate: Port Operating Speed (bits transmitted per second) (300 bps to 128kb)

Parity: Specifies the character parity for the currently selected port. NONE means no parity bit is provided. EVEN, ODD, MARK, SPACE specify that parity will be set as follows: NONE is the usual setting but EVEN is commonly used.

EVEN	Parity bit set to provide an even number of set bits.
ODD	Parity bit set to provide an odd number of set bits.
MARK	Parity bit always set.
SPACE	Parity bit always clear.

Stop: 1 or 2 Stop Synchronization Bits, 1 is the usual setting.

Word: Defines the number of data bits that make up the character size.
Eight bit words are commonly used with no parity.
Seven bit words are commonly used with even parity.

Handshake: Is a means by which your computer (and the remote host) will control incoming and outgoing data. Some modems require a handshake to avoid losing data. Handshakes may be performed using hardware (RS-232 pins) or via software using special ASCII control characters. UNICOM provides selection of the following handshake types as supported by the Windows comm port driver:

Hardware: specifies that RS-232 pin 4: Request to Send (RTS) performs receive

flow control and pin 5: clear to send (CTS) for transmit flow control. RTS will be dropped when the receive queue is full and raised otherwise. Character transmission will be suspended when CTS is dropped by the external device and resumed when it is raised.

None: specifies no handshake. A software specific handshake is up to the application program (such as an XMODEM protocol transfer) driving each end of the communication link.

Xon/Xoff: interprets DC1 (CTL Q) and DC3 (CTL S) characters as special flow control characters. When UNICOM receives a DC3 (Xoff), it will suspend any transmission until a DC1 (Xon) is encountered. Likewise, when UNICOM's receive buffer is full, a DC3 (Xoff) is transmitted to the remote computer to cause it to suspend (provided the remote recognizes XON/XOFF) transmission. UNICOM resumes the suspended remote transmission (when ready) by transmitting a DC1.

Connection:

Instructs UNICOM to treat the remote connection as a modem or a direct computer-to-computer link. If set to MODEM, UNICOM will transmit an init string at the start of each program session and when invoking host mode. A modem reset command will be issued upon terminating UNICOM.

Disable Error Report:

Controls the reporting of hardware detected communication errors from the communication port driver being used with Windows.

Parity: When selected, disables reporting of parity errors detected in received characters.

Framing: When selected, disables reporting of improperly synchronized transmissions due to poor line quality or mismatched communication settings.

Overrun: When selected, disables reporting of a UNICOM transmit or receive buffer overflow conditions.

Select the port and the desired characteristics from the above options and press the CONFIGURE button to activate the port. To restore the port settings to the original configuration (as stored in the program configuration file), press the DEFAULT button then press CONFIGURE.
Baud rates up to 19,200 are supported by the Windows 3.0 port driver. Special drivers and or hardware are required to use baud rates above 19,200.

NOTE: The Windows 3.0 port driver may become unreliable when operating at speeds above 9600. If UNICOM complains of CRC errors during file transfers above 9600 when you switch to another application that references a drive, the port driver may be at fault. If your version of Windows does not correct this problem, contact Microsoft at (206)637-7098. If no replacement driver (comm.drv) is available, 3rd party communication port drivers are available.

Terminal Settings

UNICOM's terminal setup window lets you control all aspects of your terminal screen. Terminal type, fonts, colors, attributes, scroll buffer, terminal modes, terminal width and cursor type can be configured by selecting terminal from the setup menu. The terminal setup window will be displayed as shown below in Figure 3.

.

(see manual)

Figure 3. Terminal Setup Window

Options provided in the terminal setup window are described as follows:

Terminal Type:

DEC VT102 -(ANSI) :	Emulates an ANSI compatible terminal including VT-100. Supports special DEC character sets and double size chars.
DEC VT52:	Emulates a DEC VT52 terminal.
ANSI-BBS:	Provides an ANSI emulation compatible with that used in dial-up bulletin board systems. Supports color.
TTY:	No emulation, responds to ASCII control codes for cursor movement, line control and character display.

Terminal Modes:

Newline: This option will automatically generate a linefeed upon receipt of a carriage return. If characters seem to wrap around on a single line with your particular host, enable this option. If lines always appear double-spaced, disable this option.

Local Echo: Some hosts do not echo characters back when typed from the keyboard. Half duplex systems typically operate this way (such as GEnie). Enable this option to instruct UNICOM to echo characters to the screen as they are typed from your keyboard. Likewise, should characters appear double on your screen, disable this option.

Autowrap: Some remote hosts do not position the cursor to the start of the next line after reaching the end of the current line. If enabled, this option will instruct UNICOM to move the cursor to the 1st column of the next line after the end of line is reached.

Erase on Backspace: Once enabled, backspace characters received will be translated into BS-SPACE-BS to erase the character on your screen. This translation is normally performed by the remote host. If characters are not erased using backspace with your particular host, enable this option. This option has no effect if the backspace key is defined as a delete key (except for tty operation).

Backspace Key is DEL: Controls the meaning of your backspace key. If checked, the key will operate as a delete key. When unchecked, the backspace key operates normally.

Scroll Buffer:

Lines edit box: The size and positioning of the scroll buffer is selected here. Enter the desired size (in lines) of the scroll buffer in the edit box. The scroll buffer

can be sized from a minimum of 80 to a maximum of 1500 lines. Values entered that are outside this range will be reset to the closest limit.

Visible @ Top, if selected, the scroll buffer will be visibly integrated into the UNICOM window above the terminal screen. The bottom of the scroll buffer will occupy as many lines over 24 that can be displayed in the window. This will allow your video hardware to display the maximum number of lines possible.

Some remote hosts do not make clean use of your scroll buffer. Instead, the scroll buffer may appear to contain useless junk. The scroll buffer can be hidden from view : just un-check the Visible @ Top option. This dedicates the entire window real estate to the 24 line terminal screen. The scroll buffer may still be accessed using the vertical scroll bar or paging options in the control menu.

Columns:

80 or 132 column widths are supported. This option will not automatically select an appropriate font size to allow viewing of the entire line. The user selected font (and size) will determine if the entire line can be viewed. In any case, the horizontal scroll bar will let you position the screen to scroll any obscured columns into view.

Terminal Font

All font facenames are enumerated for selection in the listbox on the left. These fonts are all available fonts installed by Windows, type managers or Window applications. UNICOM loads its own fonts prefixed with VT100 you may access like any other font. However, some of these fonts contain only special characters and symbols required by the VT102 emulation.

NOTE: The facename of the font also determines what character set is used. The OEM (IBM) character set contains block drawing and line drawing characters in the high order ASCII range. The ANSI character set contains special symbols, accented characters and empty blocks in the high order ASCII range.

The Terminal font should be normally be used (in the U.S.) since many hosts and BBS computers use the line drawing characters to format their displays.
All other fonts use the ANSI character set.

See Appendix H for a description of the ASCII and OEM character sets.

WxH All font sizes for the selected facename are enumerated in this listbox. As you select a different font name, this listbox is automatically updated to reflect the available sizes.

Override

Width An override width edit box is provided to allow you request a variation in font width. Entering an override width does not guarantee success. The resulting font size will be determined by Windows (not UNICOM).

Height An override height edit box allows you to request a variation in the font height. Like above, entering an override height will be considered (not guaranteed) by Windows when creating the font

Screen Color

Foreground: A rectangular window displays the currently selected text color to be used when UNICOM displays text to the screen. A horizontal scroll bar is provided for user selection of all supported text colors.

Background: A rectangular window displays the currently selected background color to be used when UNICOM displays text to the screen. A horizontal scroll bar is provided for user selection of all supported text colors.

Foreground and background colors include:
Black, Red, Green , Blue, Cyan, Magenta, Yellow and White.

Char Spacing:

This option controls the spacing of proportional fonts such as roman and helv.
These fonts look nice but vary in their individual character widths. All known terminal emulations assume a fixed size font - the width (and height) of each character is equal. To use proportional fonts, each character must be positioned to a cell that can hold the widest character in the font. The resulting extra space is wasteful and it just doesn't look nice.

This character spacing option allows you to reduce this extra spacing at the risk of losing parts of wide characters such as uppercase W or X.

Normal: Characters are mapped to fixed size cells that can hold the widest character.
7/8: Characters are mapped to a fixed size cell sized 7/8 of the widest character.
3/4: Characters are mapped to a fixed size cell sized 3/4 of the widest character.
2/3 Characters are mapped to a fixed size cell sized 2/3 of the widest character.

This option has no affect on character placement for fixed sized fonts such as Terminal or Courier.

Cursor Blink

None: The cursor does not blink
100: Blinks every 100 msec.
200: ' ' 200 msec.
400: ' ' 400 msec.

Cursor Type

Underline: Choose
Vert Bar: your
Block: preference

It is a good idea to use blinking as the cursor may temporally disappear

when activating other windows.

Initial Font Attribute

Normal: Characters are displayed with no special attributes
Underline: All characters will be underlined, screen erases will initialize all rows to underlined spaces.
Italic: Characters are displayed in italics. This attribute is a substitute for blinking.
Bold: All characters will be displayed bold.

These attributes are normally controlled by the remote computer that supports ANSI emulation. UNICOM will display subsequent characters using the selected attribute, until this attribute is overridden by the remote. Normal is the suggested setting.

Modem Settings

UNICOM provides a modem setup window containing user selectable options for Hayes compatible modems. Select the MODEM option from the Setup menu. A Modem Setup window will appear (shown in Figure 4.) containing the current modem settings.

(see manual)
Figure 4. Modem Setup Window

The purpose of this window is to construct a modem init string that will be sent to the modem upon activating UNICOM or by pressing Accept. The modem setup window supports two types of init strings: User Entered and Selected. The two radio buttons located at the top of the window are used to determine which init string will be used by UNICOM.

When selected, the User Entered radio button will instruct UNICOM to transmit the modem init string defined in the edit box. If the Selected radio button is chosen, UNICOM will construct a modem init string from the menu selections in the Selected Init String section.

User Entered Init String:

An edit box is provided so that you may define your own modem init string. You must prefix the string with an AT. Though not visible, UNICOM will append a terminating carriage return to the end of the string placed here. No user entered control character prefixing is supported in this edit field.

Selected Init String:

A modem init string is constructed automatically based upon the configurable modem options contained in the Selected Init String section. These options are defined as follows:

Wait for dial tone: (2-255 seconds) DEFAULT = 2 determines the maximum time the modem will wait for a dial tone during dialing operations.

Wait for answer: (1-255 seconds) DEFAULT = 30 determines the time the modem will wait for an answer after dialing has commenced.

Dial Type: Tone or Pulse operation.

Speaker Control: Always OFF, ON for dialing or ON while the phone is off hook.

Auto Answer: ON or OFF.

Answer on ring [x]: If Auto Answer is enabled, the modem will pick up the phone on ring x (if x > 0).

Dialer Speed: Slow, Medium or Fast. This affects the dialing rate for tone operation only.

Call Waiting Protection: ON or OFF.
When enabled, this feature will prevent the modem from breaking a phone connection because of a call waiting 'click' associated with incoming calls. The loss of carrier time is extended to 10 seconds to prevent the modem from hanging up during this type of interruption. This method does not

instruct the phone system to block waiting calls.

Many local phone systems will allow you to dial *70 to block call waiting when dialing. The prefix/suffix dialing feature may be used for this purpose. See the Dialing Directory section for information on how to use prefix dialing.

For more detailed information regarding these (and other) modem settings, refer to your modem reference manual.

Hayes compatible modems may differ in modem responses when attempting a connection or hanging up by dropping the RS-232 data terminal ready signal.

A modem-specific setup window has been provided to describe responses and timing behavior that can vary from one Hayes compatible brand of modem to another.

To activate the modem-specific setup window, press the MORE pushbutton from the modem setup window. Figure 5.illustrates the Modem Specific Setup Window.

(see manual)
Figure 5. Modem Specific Setup Window

Connect String: This field should contain your modems response upon making a successful connection. When dialing, UNICOM examines modem responses to determine the result. The typical default string is uppercase CONNECT for most Hayes compatible modems. Some modems respond with CARRIER followed by the connect baud rate. If UNICOM displays the message 'Connection Established' on the status line when using a program dialer, you can be sure the connect string is set properly.

No Connect Responses: Enter the possible responses produced by your modem that indicate unsuccessful dialing. If UNICOM encounters one of these strings during dialing, the specific response will be reported to the user. The most common responses are listed in the illustration above. Consult your modem reference manual for these response strings.

Hang Up String: Should UNICOM fail to hang up by dropping DTR, it will perform a software hang up procedure. This involves sending the escape to command character sequence '+++' to bring the modem into command mode. Once in command mode, the modem is instructed to hang up using the string defined in this field.

Escape Guard Time: (0.5,1.0,1.5 Sec) This is the amount of time UNICOM will delay before and after sending the modem attention '+++' sequence to bring the modem into command mode during a software hangup attempt.

Response to DTR drop: Modems typically produce a response string once a connection is dropped for reasons that include loss of DTR. UNICOM drops DTR (RS-232 pin 20) for hang up operations and watches for the response defined here to determine if the attempt was successful.

To allow UNICOM to hang up quickly using the DTR drop method, you must provide this hardware signal to your modem using an RS-232 cable that supports pin 20. The modem must also be commanded to drop the line upon loss of DTR. This command is typically provided from the modem init string which is loaded at program initialization. Consult your modem reference for the particular modem command.

Entering a value in this field is not necessary if your modem cable provides Data Carrier

Detect (DCD or RLSD). UNICOM will watch for this line to transition after dropping DTR. If this line changes state, UNICOM will consider the hangup successful. If no DTR drop response string or DCD transition is encountered after dropping DTR, UNICOM will perform a software hangup procedure.

Command Speed: (Slow,Med,Fast) Some Hayes compatible modems become confused when commands arrive too quickly to the modem. This option controls the amount of time to delay per character when commands are issued to the modem. A Fast setting means no character delay. Medium introduces a 30 msec delay and Slow introduces a 60 msec delay. For most modems, the Command Speed can be set to Fast.

Keyboard Settings

The meaning of your keyboard function and keypad keys are user definable. Function keys F1 through F12 and the keypad keys may be defined as keyboard macros, program hot keys or for launching script command files. To activate the keyboard macro editor, select Keyboard from the setup menu. The keyboard editor window will appear as shown in Figure 6.

Keyboard macros are simply character definitions that UNICOM will type for you. Program hot keys are nothing more that predefined UNICOM menu picks, that when activated, perform the same action as if you selected the menu pick with the mouse or keyboard. A key may also be defined to launch a script command language file.

The entry you provide in definition field for each key will determine if the key is a macro, hot key or script launcher.

The following codes are used to define the key (^^ codes must appear in the first and second columns):

Script Launcher ^^*Sscriptfile.ext*
Executes the script file specified in the *scriptfile.ext* argument. The file is expected to reside in the UNICOM files directory. Example: ^^Sunicom.scr - activates unicom.scr.
No space is allowed between ^^S and the filename. The filename must not contain a drive and/or directory.

Hot Key ^^*colrow*
col is the position of the UNICOM menu. 1 = File, 2=Edit,... 8 = Utility.
row is the position within the selected menu that is to be activated.
No space is allowed between *col* and *row*.

Examples:
^^11 selects UNICOM's file logging feature in the file menu.
^^42 selects UNICOM's Transfer Upload menu pick.

For menu picks that are in positions greater than 9, position 10 or greater must be designated with reference to the characters that follow 9 in the ASCII character set. Position 10 = ':', 11 = ';', 12 = '<' and so on. Please refer to Appendix C.

Macro Any text and/or control characters may be defined up to a maximum size of 80.

Certain keys (marked with an asterisk *) have special meaning when using VT102 or VT52 emulations. You may override this special meaning by leaving the checkbox at the lower right of the Keyboard Macro editor unchecked. Once unchecked, these special keys operate with the definitions you provide.

If the checkbox is set, F1-F4 operate as VT102 PF1 through PF4 keys. The arrow keys are used for cursor positioning.

(see manual)
Figure 6. Keyboard Macro Editor

To define a key, place the keystrokes into the definition editbox. Control characters can be inserted into macros and are denoted with the ^ character prefix. For example: ^C will output a control-C (ASCII 03). Control characters may be mixed with printable ASCII characters. Each macro is limited to a maximum of 80 characters. For a complete list of all possible prefix character combinations see Appendix I.

The label field for each key will be displayed in the corresponding screen button to identify the key. The key may be activated by pressing the key itself or by activating the screen button with your mouse. Screen buttons containing user defined labels assigned to each function key are displayed at the bottom of the screen above the status line. To toggle display of these button on or off, select the User Keys item from the Control Menu.

Hot keys may be used to 'launch' utility applications stored in the Utility menu. Just add the desired applications to the Utility menu and note the position of the applications within the menu. The first position in the Utility menu is reserved for passing parameters.

As an example, a hot key to activate the first application stored in the Utility menu is: ^^82. ^^83 activates the second, ^^84 - the third and so on.

Figure 7 illustrates the use of the user defined screen buttons. The two rows of buttons contain key
labels defined using the Keyboard Macro Editor.

(see manual)
Figure 7. Macro Screen Buttons

Host Mode Settings

Host mode allows remote access to your computer similar to that of a mini BBS. At a minimum, UNICOM requires that you establish a user id and password for each remote user. To activate the Host setup window, select Host from the Setup Menu. The Host Setup window will be displayed as shown below in Figure 8.

(see manual)
Figure 8. Host Setup Window

There are two types of settings : User and System. The User Maintenance section is used to maintain information about each remote user who will be allowed to login to your computer. All other settings are system settings that control how host mode behaves for all users.

Host System Settings

Host Identification

String (80 chars max): This field contains the string that identifies your system to a remote user who is attempting to login. It is displayed when UNICOM detects a CONNECT response from the modem (if one is used). If the connection is set to computer in the comm port setup window, this message is displayed after the remote user enters 2 consecutive carriage returns. This string indicates the start of the login process.

Greeting File: This file contains text information that will be transmitted to the remote user once a connection has been established but before a user logs in. This file may contain embedded escape codes to format the remote terminal screen. At each screenful of text (23 lines), the remote user is prompted: More? (Y/n).

A blank entry or invalid filename in this field will disable this option.

UNICOM will look for this file to be located in the UNICOM Files Directory as defined in the File Path Setup Window.

Bulletin File: This file is transmitted to the remote user after each successful user login. At each screenful of text, the remote user is prompted: More? (Y/n).

A blank entry or invalid filename in this field will disable this option. UNICOM will look for this file to be located in the UNICOM Files Directory as defined in the File Path Setup Window.

Menu Filename: UNICOM provides a default remote user menu. You may define your own menu and cause UNICOM to display it to the remote user. The menu can be created using a text editor. Special control characters may be embedded in the file.

A blank entry or invalid filename in this field will cause UNICOM to display a default menu.

UNICOM will look for this file to be located in the UNICOM Files Directory as defined in the File Path Setup Window.

Help Filename: A help option exists on the default menu presented to the remote user. When the user selects the help option, UNICOM will transmit the file named in this field to the user. A default help file is not provided with UNICOM. The help file must reside in the UNICOM files directory.

Monitor Mode: Allows viewing of the remote users session from the host display. The sysop may use the host keyboard to interact with the host menu. All characters type by the remote user (including passwords) can be viewed.

No Activity Check: Automatically logs out a remote user who does not respond to an input prompt for approximately 5 minutes. In all cases, the user will receive a lack of activity warning if no response has been received to a prompt after 45 seconds.

Log User Activity: If enabled, UNICOM will log all user actions to the event file defined in the general setup window. User responses to the log in process, responses to menu picks, file and directory activity will be logged.

Dial Back Mode: This mode will allow a remote user to log in then UNICOM hangs up and calls the user back within 60 seconds. Each user record includes a phone number which is used to dial the user back. Dial back mode (once enabled) will be in effect for all users.

Time Limits: Remote users are limited in amount of time they may be logged in to your computer. Remote users are classified by access level (described in the next section). Level 3 is the most restrictive and Level 1 is the least restrictive access level.

Enter a time limit in minutes for each access level. A value of 0 will prohibit all users with the particular access level from logging in. A limit of 45 minutes is a suggested value.

Host User Settings

The user maintenance section of the host setup window lets you maintain information for each user authorized to access your computer through host mode. The listbox visible in this section contains the User IDs of all authorized users. By highlighting an entry in this listbox with a mouse or keyboard, existing entries may be changed or removed. New user records are created using the Add button.

Pushbutton	
Add	Used to create a new user record. Displays the Host User Setup window shown in Figure 9 below.
Delete	Removes the user record associated with the User ID selected in the listbox.
Change	Activates the Host User Setup window and fills in all fields with the user information of the User ID selected in the listbox.

(see manual)
Figure 9. Host User Setup window.

The Host User Setup window is displayed by selecting Add or Change from the user maintenance section of the Host Setup window. At a minimum, you must enter a User ID, Password and Login Drive & Path. The Name and Address fields are not used by UNICOM

and are for your reference only.

The telephone field is required if you intend to operate in dial back mode. UNICOM dials the number exactly as stored in this field. No modem commands are allowed in this field.

Each Host User field and option is defined as follows:

User ID : Identifies the remote user. UNICOM prompts the remote user for this name during login. Once logged in, the user id is displayed on UNICOM's status line to identify the current user.

Password: Validates the user attempting to log in with a particular User ID.

User Information

Name: The real name of the user who is allowed remote access. UNICOM does not use this field. It can be used anyway you choose.

Telephone: The telephone number of the remote user. If operating in dialback mode, a modem must exist and be set to answer mode at this phone number.

Address: The street address of the remote user can go here. Like the Name field, UNICOM does not use this field. It can be used anyway you choose.

Login Drive & Path: This field determines the initial drive and directory which the remote user will access once logged in. Users with access level 3 will be confined to this path.

Access

Level 1: Full access allowed. Can shell to DOS if running in 386 enhanced mode.

Level 2: Partial access. Same as Level 1, but user cannot shell to DOS.

Level 3: Limited access. Cannot upload, shell to DOS, or change directories.

File Paths

UNICOM must be told where its operating files are stored, where files received from data transfers are to be stored and where to look for files for upload selection. These three file paths are user selectable and altered by selecting File Paths from the Setup Menu.

A window will appear as shown in Figure 10. An edit box is provided for the following paths:

UNICOM Files Directory — Holds the executable UNICOM program, configuration files, script files, host mode support files, dialing directories, key files, menu files and other support files.

Download File Directory — UNICOM stores files received from data transfers into this directory if a full pathname was not specified in the transfer. Full pathnames override this path.

Upload File Directory — When UNICOM prompts you for upload file selections, files contained this directory will be offered. See Figure 11. The script command SendFile will examine this directory for files specified in its argument list if the arguments do not contain a full pathname.

(see manual)

Figure 10. File Path Setup Window

All path information is stored in your Windows WIN.INI file under [UNICOM].
If a nonexistent path is entered in any of the fields, or if UNICOM.EXE does not reside in the path specified in the UNICOM files directory edit box, an error message box will be displayed.

Make sure all paths entered in these fields are valid. UNICOM cannot operate properly if any of these fields are incorrect. A batch upload selection window is shown below. The files displayed in the directory listbox are those in a user selected Upload Directory named c:\win30

(see manual)

Figure 11. File Upload Selection Window

ASCII Transfer Settings

The ASCII transfer setup is divided into operating parameters for uploading and downloading operations. To access the ASCII transfer options, select the ASCII Xfer option from the setup menu. A setup window will appear as shown in Figure 12.

(see manual)
Figure 12. ASCII File Transfer Setup Window

ASCII Upload Parameters

Echo Locally: If enabled, the file data being transferred will be echoed to your screen.

Pace Character: [0-99] The pace character is the numeric value of an ASCII character that is transmitted by the remote host receiving the file. This character is interpreted by UNICOM as 'send the next line'. UNICOM will wait for the remote to sent this character for each line transmitted.

Char Pacing: [0-999] Represents a delay time (in milliseconds) between transmission of each character to the remote host computer. Setting this value to zero, disables any time delay. A zero value also greatly increases speed.

Line Pacing: [0-999] Represents the time (in 1/10 seconds) to delay after the transmission of each line or carriage return. A zero value in this field disables line pacing.

CR Translation: [None, Strip or Add LF] Carriage return translation can be used to strip carriage returns or insert linefeeds (after carriage returns) for the file being transmitted. Selecting none disables any translation.

LF Translation: [None, Strip or ADD CR] Linefeed translation will strip linefeeds or add carriage returns after linefeeds to the file being transmitted. Selecting none disables any translation.

ASCII Download Parameters

CR translation and LF translation as described above will filter and control these characters received during ASCII file downloads from remote host computers. The selection and definition (as described above) for downloading is the same as for uploading.

When downloading using ASCII, UNICOM will automatically end the transfer if a control-Z character is encountered.

Kermit Transfer Setup

The Kermit is a configurable protocol and you may not want to change the settings shown in Figure 13 unless you are an advanced user. Assuming you are, here are the field definitions:

Max Packet Size: This is the maximum length for outbound packets, regardless of what was negotiated with the other Kermit. Normally, you would change this field (from the default) only to send shorter packets than the other Kermit requests, because you know something the other Kermit doesn't know, e.g. there's a device on the communication path with small buffers.

Timeout: This can be used to adjust the normal Kermit timeout parameter for both local and remote systems. Timeout will occur if a packet is not received after the number of seconds specified in this field.

# of pad chars: This value controls the number of pad chars to be requested form the remote Kermit to precede each packet it sends. Padding is not usually required but may be necessary to keep some intervening communication happy.

Padding Char: Use the specified control character for interpacket padding. Some hosts may require padding characters (normally NULL or DEL) before a packet, and certain front ends or other communication equipment may need certain control characters to put them in the right mode. The number is the ASCII decimal value of the padding character, (0 - 31, or 127).

EOL Char: This field contains the ASCII value of the packet terminator to put on outbound packets. Normally a carriage return (13). Change this field if the other Kermit requires a nonstandard packet terminator.

Quote Char: This field contains the ASCII value of the character to be used to prefix control and other prefix characters. The only reason to change this would be for sending a very long file that contains many '#' characters (the normal control prefix) as data.

Port: (Switch to N-8-1 or No Switch) This option determines if UNICOM will automatically set the port for binary operation before Kermit is initiated. Selecting N-8-1 (the normal default) will allow Kermit to transfer binary data. No Switch should be used if the remote Kermit does not switch automatically to 8 data bits, No parity and 1 stop bit.

NOTE: UNICOM's implementation of Kermit does not support transfer of 8 bit data through 7 bit links.

Much overhead is built into the design of Kermit. It's performance in UNICOM is limited to under 700 cps even when operating at the fastest baud rate possible. If you need performance, Zmodem and Ymodem G are recommended.

The fixed attribute definitions are not described here. Refer to the Kermit Users Guide from Columbia University.

General Setup Window

The General Setup window provides many user selectable options that affect program operation. This window allows you decide how UNICOM is to behave during many program procedures.

To activate the General Setup window, select General from the setup menu. A window will appear containing the current option settings as shown in Figure 14 below.

Figure 14 (see manual)

Definitions of General Setup Options

UNICOM Startup Window:

The startup window options control the appearance of the UNICOM window upon program activation.

Normal: UNICOM will position and size its window on the screen according to where it was at the time of the last Save Setup. The values are stored in WIN.INI.

Full Screen: will zoom the UNICOM window to occupy the entire screen.

Iconic UNICOM will be activated in iconic form, the UNICOM icon will be displayed at the bottom of the screen.

User Keys: Controls the display of the user defined function key buttons at the bottom of the screen at program activation.

Scroll Bars: When enabled, UNICOM will display both horizontal and vertical scroll bars at each program activation.

Logo: Controls the display of UNICOM's opening logo. Disabling the option speeds program startup time. Unregistered users cannot disable this option.

Verification Prompts: If set, UNICOM will display a message box prompting the user to acknowledge end of file transfers, program termination and modem hangup operations.

Log Events: to File Controls recording of events such as dialing, hanging up, executing scripts , file transfers and other program activities. Each event is time-stamped. Events are written to the file whose name you provide in the edit box. This file is assumed to reside in the UNICOM files directory. The drive and directory is not required in the filename.

Auto Minimize on File Transfers: When checked, UNICOM will automatically iconize itself at the start of every file transfer. This can be useful for clearing the screen quickly so you may resume operating another windows application. UNICOM will pop back up to the screen after the transfer completes.

Auto Minimize on Repeat Dialing: Enable this feature to quickly remove UNICOM from the screen when batch dialing systems that are typically busy. UNICOM will pop back up to the screen when a connection has been established.

Notification Beeps are enabled or disabled with this option. Notification beeps occur at end of file transfers and upon successful dialing. The type of notification beep is determined by the file: UNICOM.SND which contains musical note and duration values.

Log Filter Terminal escape codes may be filtered out of any file or printer logging operation by enabling this feature. When disabled, no filtering is performed - incoming characters are logged exactly as received. This has no effect when using the TTY terminal.

Default File Transfer Protocol: Choose the protocol to be selected within the upload and download protocol selection window when transferring files.

Auto File Downloading

Zmodem: Controls detection of a Zmodem init packet. UNICOM will initiate a Zmodem download automatically at the request of the remote computer. Enabling the option will free you from manually selecting Transfer - Download, then Zmodem anytime you wish to receive a file.

CompuServe B & Quick B Controls automatic detection of CompuServe B and Quick B file transfers from CompuServe. If enabled, UNICOM will automatically begin download **and upload** operations at the request of CompuServe. Automatic uploading is supported only for these CompuServe protocols.

Dialing Directory File: Enter the name of the default dialing directory to be loaded each time you activate the Dialing Directory.

Script Editor: The filename of the script language editor of your choice should be entered here. UNICOM activates this editor when the Edit, Edit Last or Create items are selected from the script menu. If this field is empty, UNICOM will activate Notepad by default.

AutoStart Script File: A script filename entered in this edit box will automatically execute upon each initial activation of UNICOM. A blank entry or invalid filename in this field will disable the autostart feature. Script command files must be located in the directory defined by the UNICOM files path.

Keyboard Macro File: The filename of the default keyboard macro file should be entered here. The keyboard macro file defines the meaning of the keyboard function keys either as macros or program Hot Keys.

Log File: UNICOM will default to the filename entered here when file logging is activated. This name is entered in the filename editbox that is displayed when File Log is selected from the file menu.

Utility Menu Settings

(see manual)
Figure 15. Utility Menu Setup

This setup screen allows configuration of the Utility Menu with application entries of your choice. These applications are then listed by name for quick activation either from a menu selection or with a Hot Key definition. With this configuration screen, you may add many commonly used applications for a quick 'Launch' by UNICOM.

To operate this screen, just use the directory listbox to navigate across drives and directories to make your selections. Selected programs are stored in the right listbox shown in Figure 15 by highlighting the desired application then pressing ADD. This file selection listbox is very similar to the batch upload file selection listbox used for file transfers.

Applications names may be removed by first highlighting the desired entry in the Selected Applications listbox then press the delete button.

Once you have selected all the desired applications, press Ok to instruct UNICOM to configure the Utility Menu. UNICOM stores the complete path for the application in memory. If the application cannot be found (or for any other activation error) when it is selected from the menu, UNICOM will automatically display this setup window.

Applications may be 'Launched' with the press of a function key by defining a Hot Key for the particular entry in the Utility Menu. For information on setting up hot keys, see the previous section on Keyboard Macros.

Zmodem Transfer Setup

(see manual)
Figure 16. Zmodem Transfer Settings

UNICOM provides a Zmodem setup window (Figure 16) for advanced users of this protocol. If the setup screen seems confusing to you, don't worry, just select the Defaults push button to ensure correct operation. Advanced Zmodem users may wish to use some of the options provided by the design of this protocol. File management options allow examination of an existing file size and length before a transfer will occur. Other options control the amount of feedback during the transfer. Lot's of feedback could be useful for determining the source of problem transfers. The default is minimum feedback since the additional reports can be quite confusing if you're not a Zmodem expert.

NOTE: The Zmodem upload option: Unlink After Transmission (if set) will DELETE the file on your disk once it has been uploaded.

Character Translation Tables

UNICOM provides user control over the translation of incoming and outgoing characters with the use of translation tables. Character translation, if enabled, is performed in terminal mode only. Translation is disabled during Chat mode, script operations and modem operations. To activate the translation table setup window, select Translation from the setup menu. A window will appear as shown in Figure 17 below.

(see manual)
Figure 17. Translation Table Setup Window.

A foreign language user may wish to map a special ANSI character the same character in the IBM PC extended character set. To accomplish this, a user would assign a new value for the desired characters in the Outgoing character table.

Saving All Settings

All program settings listed in the Setup menu (including terminal font selections, keyboard definitions, scheduler settings and prefix/suffix info) may be saved to configuration files and loaded automatically for your next UNICOM session. To save all currently defined settings, select SAVE SETUP from the Setup menu.

UNICOM will update the configuration file which UNICOM was initially activated with. If UNICOM was activated without a configuration file argument, UNICOM.CFG will be used. The configuration file will be created if it does not already exist or cannot be found in the defined UNICOM files path. File path settings are written to the Windows WIN.INI file. Keyboard macro definitions are written to the file currently listed in the General Setup Window. UNICOM.KEY is the default key filename.

A special configuration file named UNICOM2.CFG can be used to configure additional UNICOM instances. For example, selecting Spawn UNICOM from the files menu activates another copy of UNICOM. This new copy, or instance will need to be configured for a port different from that of the instance that created it.

You may activate additional instances of UNICOM with a configuration file parameter using an external application. Additional UNICOM instances activated from within UNICOM or externally will automatically look for a configuration file named UNICOM2.CFG. If this file cannot be found, previously described defaults apply. If these defaults fail, the new UNICOM instance will activate the communication port setup window as a last resort.

Connecting to a Remote Computer

Introduction

UNICOM provides your computer with ability to communicate with another using a serial transmission link. The physical connection is typically a direct computer to computer RS-232 cable link or a modem/telephone link:

Computer to Computer Link: You'll need a null modem RS-232 cable (that reverses pin 2 & 3 and 4 & 5) with the appropriate gender that mates your serial DB-25 or DB-9 connector to the remote DB-25 or DB-9 connector. The length of the cable should not exceed 50 feet.

Modem / Telephone: A Hayes compatible modem connects to your computer port via RS-232 cable. Cable pins 2,3,1,7 and 20 are required as a minimum. If you intend to use hardware handshaking, pins 4 and 5 are required.

Connect the modem to the phone system. If the phone line supports call waiting, 'clicks' associated with an incoming call may interrupt (and drop) an active transmission link. For information about solving this problem: See Modem Settings in the section titled Setting Up UNICOM.

UNICOM is most commonly used to connect to other computers with the use of a modem and telephone line Most modems are "Hayes Compatible', that is, they recognize an established set of commands that include dialing, hanging up and configuring various options supported by the modem.

Two dialing methods are provided in this software to allow you to easily connect to another remote computer using a modem.

Manual Dialing: UNICOM will prompt the user for a phone number then command the modem to dial the number.

Directory Assisted Dialing: A directory of remote hosts is maintained by the user. Any directory entry is easily dialed by just double-clicking on the entry with a mouse. Entries must be first added to the directory before they can be dialed from the directory.

Using the Phone Dialer

The phone dialer is useful for manually dialing a single phone number. To activate the phone dialer, select the telephone icon on the tool bar or select the Dial option from the phone menu.
A window will appear as shown below in Figure 18.

(see manual)
Figure 18. Phone dialer window

Enter the desired number then press dial. The Dial button turns into an abort button when dialing has been initiated. If the modem reports that a connection has been made, UNICOM destroys this window automatically. The exit button may be pressed at any time to remove the window from the screen (dialing will not be interrupted if in progress).

Answering an Incoming Data Call

There are two methods by which to answer an incoming data call.

1) Set your modem to auto answer by sending an init string to the modem. This is done from the Modem Setup Window. The modem will answer all calls after the set number of rings until another init string is sent to disable the option.

2) Command the modem to immediately answer an incoming call. This method is useful if your data call arrived before you had time to set the modem to auto-answer.

UNICOM supports both methods, the second method is initiated by selecting Answer Now! from the phone menu or by selecting the telephone pickup icon from the toolbar. UNICOM will send an ATA command to the modem causing it to immediately answer the incoming call.

Sending a 'BREAK'

Some remote host computers require the user to set the line to a break state in order to signal an event (such as aborting a display operation). You may instruct UNICOM to send this signal by selecting 'Break' from the Phone menu. The communication line will enter a break condition for 350 milliseconds.

Hanging Up

You may command the modem to hang up the phone by selecting the HANG UP option from the Control menu. UNICOM will attempt to hang up the line by dropping the data terminal ready line (DTR) to cause the modem to drop the line. UNICOM watches for a modem response string to determine if the operation was successful. The specific modem response must have been stored in the modem specific setup window in the Response to DTR drop field. UNICOM will also monitor the Data Carrier Detect (DCD) line for a state transition. If DCD transitions or the modem response to DTR drop was encountered, UNICOM will consider the hangup operation successful.

Should the modem fail to hang up using the hardware method described above, a software hangup sequence will be initiated. The Hayes attention sequence ('+++') is transmitted to the modem in order to place it into command mode. Once in command mode, the modem is commanded to hang up using the hang up string defined in the Modem Specific Setup window.

The message 'MODEM READY' should appear indicating that the operation completed successfully. Should the message 'MODEM NOT RESPONDING' appear on the status line, the modem may still be online. If necessary, invoke the HANG UP command again.

Using The Dialing Directory

The dialing directory is a useful tool for automating the task of connecting to many different host computers. After adding a host to the directory, dialing becomes easy - just double-click on the desired directory entry with your mouse. UNICOM will automatically dial the remote host and configure itself to the various settings provided in the directory entry. A directory editor is easily accessed from the directory. An example directory is shown below in Figure 19.

Many different directories may be created and loaded. The number of entries in a directory is limited only by the amount of available memory in you computer.

(see manual)
Figure 19. Dialing Directory

Each directory entry contains a number of fields as described below. The horizontal scrollbar below the listbox will allow you to scroll obscured fields (right of the script-file column) into view. The vertical scrollbar is used to scroll hidden entries into view.

Directory Field	**Description**
System Name: (22 chars max)	Identifies the remote host system
Number: (15 chars max)	Telephone number (The dial string can be used to extend this)
Duplex	(Full or Half) Normally set to Full.
Baud:	(300,1200,2400,4800,9600,19200 or 38400)
P (arity):	E(ven) or O(dd)
D (ata bits):	7 or 8
S (top bits):	1 or 2 - Normally 1
Script-File:	This field contains the name of script file in the UNICOM files directory to be executed upon successful dialing. If the directory entry does not include a phone number, UNICOM will not attempt to dial but will immediately execute any script file defined here.
Dialing String:	(24 chars max) Contains a modem dialing command and can be used to extend a phone number beyond the 15 character limit. This field allows you to use your own modem dial command and include specific modem settings that may be necessary to dial the remote host. The phone number listed above is appended to this string. If blank, UNICOM provides it's own dial string. A typical dial string is ATDT. No control prefixing is supported here.
Password:	(16 chars max) Displays the host password during dialing as a reminder for logging in.

Terminal: (VT102, ANSI-BBS, VT52 or TTY) Desired emulation for use with the remote host.

Protocol: Default file transfer protocol for the remote system.

Last-on-Date/Time: Time stamp of last access to the host system (Maintained by UNICOM)

Sessions: Count of the total number of access attempts to this host (Maintained by UNICOM)

Maintaining the Directory

The directory maintenance section of dialing directory lets you: add, change, delete and locate entries in the current directory. Directories may be loaded and saved by name. The pushbuttons contained in the directory maintenance section are describe as follows:

Pushbutton	**Description**
Find:	Activates a search window to allow the user to locate a directory entry using a specified pattern. An edit box is displayed to accept the pattern. All directory fields are searched. The first directory entry containing a pattern match will be highlighted and scrolled into view.
Open:	A directory file selection window is activated.
Add:	Activates the directory editor.
Change:	Activates the directory editor with the current directly selection.
Delete:	Removes all highlighted entries from the dialing directory listbox.
Save:	Displays a file save window for saving the current directory.

Adding a Directory Entry

Adding entries to the directory is made simple with the use of the directory editor. To activate the directory editor, select the ADD button from the directory. The editor window will appear as shown in Figure 20. After creating the host entry, select the ADD button. This record will be added to the directory listbox containing the current directory.

(see manual)
Figure 20. Dialing Directory Editor

Deleting a Directory Entry

To remove an entry from the dialing directory, scroll the entry into view (if necessary) and highlight your selection with the mouse or keyboard. Press the DELETE button to remove the entry. Multiple entries may be removed in a single delete operation. Just highlight all the desired entries then press DELETE. To restore the directory to its original contents, just exit the dialing directory without saving your changes. Any deleted entries will re-appear the next time the dialing directory is displayed.

Changing a Directory Entry

To edit an existing entry, highlight the desired directory listbox entry and press CHANGE. The directory editor will appear displaying the settings for the selected entry. Make any necessary changes then press Change from within the directory editor. The entry in the directory listbox will be immediately updated. To make changes permanent, the directory must be saved (see below).

Saving the Directory

Changes to the directory can be made permanent by selecting SAVE from the dialing directory window. A file save window will appear prompting you for the name of the file to receive the current directory. The edit box in this window will be preset to that of the current directory filename by default.

UNICOM will prompt you to save the directory if you attempt to exit the directory without saving your changes.

The directory window is automatically closed upon successful dialing. If this occurs, any unsaved changes will be written to the current directory file.

Opening a Directory

Many dialing directories may be maintained and stored on disk for retrieval into the directory display. To load a UNICOM dialing directory, select the OPEN option from within the directory maintenance section. A file selection listbox will appear. Once a valid directory file has been selected, the directory listbox will be updated with the file contents.

You may begin dialing or editing operations with any directory file once it has been loaded.

Searching the Directory

The current dialing directory may be search for a specific pattern in any field. To activate the directory search window, press the Find button. An edit box is displayed within the window (see Figure 21) for input of a search string. Check the Match Case option for a case sensitive search.

If a directory entry contains a matching pattern, the entry will be highlighted and, if necessary, scrolled into view.

(see manual)
Figure 21 Directory Search Window

Dialing from the Directory

To connect to a system listed in the dialing directory, highlight the target system and press DIAL. Dialing may be also be performed with a double mouse click on the listbox entry. UNICOM sets the communication parameters to that of the target system **BEFORE** dialing is attempted. Please note: In order for your modem to receive commands properly, the communication parameters must be as follows:

BAUD	**Word Size**	**Parity**	**Stop Bits**
0 - 300	7 or 8	Even	1 or 2
	7 or 8	Odd	1 or 2
	7 or 8	None	1 or 2
1200 or	7	Even	1 or 2
greater	7	Odd	1 or 2
	8	None	1 or 2

Should you attempt to dial a system with communication settings different from above, or if the modem does not support the baud rate listed in the directory entry, the modem may not receive commands properly and a PORT STATE message could appear.

The PORT STATE message is displayed on the status line along with an error code any time an error occurs during communication. Reporting of PORT STATE Parity, Framing and Overrun messages may be enabled or disabled from the user selectable options in the Comm Port setup window.

For a complete list of these error codes and their meaning, see Appendix E.

Dialing a Remote Computer

When the DIAL button is pressed after making a directory selection, the communication port is configured using the port parameters listed in the directory entry. The modem is then commanded to dial using the phone number selected from the directory.

A dialing message will appear on the status line indicating that UNICOM has entered the dialing state. This message also displays host specific information. If the dialing directory is displayed, it's window title will receive the same messages shown on the status line.

Once connected to the remote system, UNICOM checks to see if a script file has been defined in the directory to automate the login process. This script file must be located in the defined UNICOM files directory. If found, UNICOM begins processing the script file until it successfully completes or is aborted by selecting STOP from the Script menu.

If dialing is initiated without a number listed in the directory entry, UNICOM will check the entry for a script file. If one is found, UNICOM will close the directory and immediately execute the script file.

Directory Options

Redial Delay: This checkbox and editbox option: Redial with ____ second delay controls the amount of time UNICOM will pause when redialing a number due to a no connect response. The edit box ___ is provided to let you enter the desired amount of delay. If this is enabled during batch dialing, the specified delay will be introduced before dialing the next number.

Set port to Modem Connect Speed: UNICOM sets the communication port to the baud rate specified in the directory entry before dialing. If this feature is set, UNICOM watches for the modem CONNECT speed report then sets the computer port baud rate to match. This feature is useful for modems that cannot lock the baud rate between the computer and modem. This feature eliminates the need to manually change the port speed if the remote answered at a different baud rate than expected.

View Modem Response To Commands: If set, UNICOM will echo all modem responses produced by the modem during dialing directly to the screen.

Sort Directory: Controls the ordering of the directory entries. If enabled, entries are displayed alphabetically. New entries are added in sorted order. If disabled, new entries are added at the top of the directory listbox.

Apply to All Determines if prefix / suffix dialing will be used for all directory entries. See below:

Dialing Using a Prefix / Suffix

Prefix and suffix dialing is useful for special dialing operations like credit card dialing or dialing to 'get out' to a real telephone system. Normally UNICOM commands the modem to dial a number from the directory using one dial command. When using Prefix / Suffix dialing, UNICOM commands the modem to dial up to three times.

If the system being dialed is designated for prefix dialing (* in column1 of the name field), UNICOM will transmit a dialing string followed by the user defined prefix string. UNICOM appends a semicolon to the dialing command to instruct the modem to return to command mode. Normally embedded in the prefix string are commas (1-3) which allow a minimum of 2 seconds of delay time before dialing again. Some modems support a W command which will allow you to eliminate the need for commas since the modem can wait until a second dial tone is detected.

After the delay, UNICOM commands the modem to dial a second time using the phone number for the entry being dialed. If a suffix will be dialed, the phone number should contain a modem command to allow for any necessary delay (like 3rd dial tone). If no suffix is defined, UNICOM issues no further dialing commands and the wait for a connection begins.

If a suffix is defined, UNICOM appends a semicolon to the end of the dialing command for the phone number to return the modem to command mode. Once in command mode, the modem is commanded to dial using the suffix and a wait for a connection begins.

Long Distance Services

UNICOM may be used to dial systems using most long distance services. Long distance services require that you perform the following:

1) Dial the local long distance access number & wait for a tone
2) Touch tone the desired long distance number.
3) Wait for another tone
3) Touch tone your secret access code
4) Wait for a connection

This procedure may be accomplished with the use of UNICOM's dialing prefix/suffix capability. To setup this special dialing feature: identify the directory entry to be dialed for special dialing. Enter or edit a system entry in the Dialing Directory and place an asterisk in column one of the name field. UNICOM will recognize the entry for special dialing when it is selected.

Create the dialing prefix and suffix. In this case, the prefix should contain the local long distance access number to be dialed and some trailing modem pause command characters for an additional connection wait. The suffix will contain some leading modem pause command characters since the phone number is limited in length. The remaining suffix will contain the secret access code to authorize your use of the service.

To create the dialing prefix and suffix, select the EDIT button from within the dialing directory.

Figure 22 shows a pop up window that will appear displaying the currently defined dialing prefix and suffix. To make changes, place your desired strings into the corresponding edit boxes and press Ok. The stored dialing prefix and suffix will be made permanent once the

system configuration is saved by selecting Save Setup from the setup menu.

(see manual)
Figure 22. Modem Dialing Prefix/Suffix Edit Window

The prefix string in the Window above contains the local access number for MCI. Trailing commas instruct the modem to wait additional seconds for the connection to be made. UNICOM appends a semicolon to the prefix to command the modem to re-enter command mode. This allows full use of the modems command buffer which typically is limited to 40 characters - not enough to hold the prefix, phone number and suffix for one-shot dialing.

In other words, when using special dialing, UNICOM commands the modem to dial up to three times. Once for the prefix, once for the phone number and once for the suffix (if defined). To use this feature, your modem must have the capability to reenter command mode after a dialing command. Most Hayes compatible modems support the semicolon to return to command mode when dialing.

Dialing may be performed with just a prefix, in which case UNICOM will not append a semicolon to the phone number. It is not possible to dial the number and suffix less the prefix.

Exiting the Dialing Directory

To exit the dialing directory, press the EXIT button or the ESC key. If any changes were made to the dialing directory, you will be prompted to save them. The dialing directory window is automatically closed upon a successful connection to a remote system.

Automatic Redialing

Some remote systems (such as bulletin boards) may require numerous dialing attempts in order to get through. The automatic redialing feature can be used for this purpose.

When enabled, UNICOM will re-dial until a connection is established or until redialing is disabled. To enable or disable this feature, set or clear the checkbox labeled 'Redial with __ second delay' as shown in Figure 23. This checkbox is located below the DIAL button from within the dialing directory.

Once set, redialing will occur whenever the modem returns no connect responses as defined in the No Connect fields within the Modem Specific Setup Window.

Aborting a Call in Progress

To ABORT a call initiated from the dialing directory, press the ABORT button from the dialing directory or press the ESC key repeatedly. Since the escape key is also used to close windows (such as the dialing directory) that may be visible, UNICOM will not recognize the ESC key as an abort until these windows have been closed.

Batch Dialing

A batch dialing feature has been included for dialing within the dialing directory. This feature is useful when trying to connect with one of any number of typically 'busy' remote systems (such as bulletin boards). Batch dialing will terminate if one of the systems being dialed answers or after the last system has been dialed. The batch operation may be repeated if no connection could be established after dialing all the specified numbers by selecting the redial checkbox.

Selecting systems to be dialed in batch can be accomplished as follows:

(see manual)
Figure 23. Batch Dialing Selections

For Keyboard Users
To make a batch selection, hold down the CTRL key and press the UP or Down Arrow key to move to the system to be selected. Select and highlight the entry by holding down the SHIFT key and pressing the SPACEBAR. Repeat these steps to make more selections. Figure 23 above illustrates (highlighted) batch dialing selections.

For Mouse Users
Scroll the listbox entry into view using the scrollbar and position the mouse to the desired entry. Hold down the SHIFT key and press the LEFT mouse button to high-light the entry. Repeat this step to select additional systems to be dialed. When all the systems have been selected, begin dialing by activating the DIAL button with the mouse, or entering ALT D using the keyboard.

Screen Editing

UNICOM release 3.0 allows you to edit the screen buffer directly with your mouse. Text may be highlighted using the mouse then erased, copied to the clipboard, transmitted back to the host or marked as a download file.

To highlight a screen selection for editing, move the cursor the desired starting position. Hold down the left mouse button (the cursor changes to a hand) and move the cursor to the ending position. Let up on the left mouse button and select an editing operation from the edit menu.

Erasing the Terminal

To clear the terminal screen, select ERASE TERMINAL from the Edit menu. The cursor will move to the first row and column of the active terminal screen. If the cursor should disappear after clearing the screen, the 1st row of the terminal screen may be located above the top of the window. Should this happen, use the scroll bar to bring the top line into view. This command does not erase the contents of the terminal scroll back buffer. If characters remain on the screen after an erase, they belong to the scroll back buffer.

Erasing the Scroll Buffer

The scroll back buffer is user selectable in size and can hold a maximum of 1500 lines of text including the 24 lines of the active terminal screen. To erase the entire scroll buffer contents, select Erase Buffer from the edit menu.

Erasing Selected Text

Highlight a selected area on the screen then select Erase from the edit menu. The highlighted rows and columns will be cleared using the current background color. Once a selection has been erased, there is no way to recover the erased area.

Copying Selected Text to the Clipboard

You can copy screen text to the Clipboard then paste this text into other applications. Select the COPY option from the Edit menu. The entire terminal screen (excluding the scroll back buffer) will be copied, including any rows and columns obscured by a small sized window.

Copying the Window to the Clipboard

The entire window contents may be copied to the clipboard as text by selecting Copy Window from the Edit menu. Text obscured by any side of the window will not be copied. This is a wysiwyg option.

Copying the Scroll Buffer to the Clipboard

The contents of the entire scroll buffer may be copied to the Clipboard as text. Once on the Clipboard, this text may be saved to a file using the Clipboard file save option. The text may also be copied to any other application that supports a paste option.

Pasting Clipboard Text to a Remote Computer

Clipboard text may be pasted (transmitted) to the remote host computer. This operation is equivalent to uploading an ASCII file. An ASCII file can be copied to the Clipboard using a program such as Notepad. The text can then be pasted (uploaded) to the remote host by selecting PASTE from the Edit menu. The file transfer information window will appear once pasting has begun. A moving bar graph gives a visual readout as to the number of bytes remaining to be transferred at any given time.

Sending Selected Text

Selected portions of your screen may be transmitted back to the remote computer. Just highlight the desired area using the mouse then select Send from the edit menu. Selecting Send with CR will append a carriage return to the transmitted text.

Selecting All Text

The entire scroll buffer can be selected (highlighted) at once by activating the Select All menu option from the Edit menu. The text can then be copied to the clipboard, transmitted to the remote or erased.

Searching the Screen Buffer

The scroll buffer may be searched for the occurrence of a user entered pattern. Select the Find option from the Edit menu. A window will appear as shown in Figure 24 to prompt the user for a pattern. Enter the pattern into the edit box. If the search is to be case sensitive, select the Match Case checkbox. Press Begin to initiate a top to bottom search. Subsequent searches may be performed by selecting Find Next from the edit menu.

(see manual)
Figure 24 Search Buffer Window.

Printing Operations

Printer Logging

Incoming characters may be echoed directly to a printer. To activate printer logging, select the Printer Logging toggle from the files menu. A checkmark will appear next to the menu item to show that it is active. The printer used by this feature is the current (active) printer defined by configuration settings in your WIN.INI file as set using the Windows Control Panel.

Printer logging and file logging may operate simultaneously. UNICOM logs to the printer on a per page basis. The printer will produce output only when logging has exceeded the current printer page size.
To disable printer logging, again, select the Printer Logging toggle from the files menu.

UNICOM provides a log filter that when enabled, filters terminal escape sequences when logging to a printer or file. Select General from the Setup Menu to locate this option.

Printing the Terminal Screen

A UNICOM screen snapshot may be sent to the printer at any time by selecting PRINT from the File menu. A message box containing a CANCEL button will appear to inform you of the print operation. Press the CANCEL button to abort printing.

Printing the Screen Buffer

The entire contents of the terminal scroll back buffer can be printed by selecting the PRINT BUFFER option from the files menu. Blank lines are not filtered out when printing. The buffer print is a snapshot in time - what you see at the instant the print was initiated is what you get when printing is finished. Any updates to the terminal or scroll back buffer are ignored during printing.

Configuring the Active Printer

To view or change the current printer settings, activate the Printer Setup option from within UNICOM's file menu. A printer setup window will appear that will allow you to configure specific options for your particular printer.

Figure 25 illustrates the setup window obtained for a HP LaserJet Series III printer.

(see manual)
Figure 25. Example Printer Setup Window

Transferring Files

Introduction

File exchange between your computer and another is what UNICOM readily provides. Many popular file transfer protocols are built-in and fully implemented for background operation, they include: X-Y-ZMODEM, Kermit, CompuServe B , Quick B and ASCII.

Before a file exchange can take place, you must decide on the particular protocol to use between both computers. If the remote computer does not support any of the protocols provided with your communication package, binary file transfers cannot take place. Fortunately, UNICOM provides the most popular and widely used protocols.

Those who are performance minded should use ZMODEM or YMODEM - G protocols. ZMODEM offers the best all around performance and reliability. Users with high speed error correcting modems can use YMODEM-G to obtain very impressive throughput.

Once you have determined which transfer protocol to use with the remote computer, the next step is to instruct both computers to begin the transfer. The computer transmitting the file needs to be instructed to begin. The computer receiving the file usually needs to be instructed to receive. If this process is not completed in a timely manner, one or both computers will 'time out'. Some communication packages (like UNICOM) can detect the download operation and start automatically when ZMODEM
or CompuServe protocols are used.

Downloading Files

To download a file into your computer, instruct the remote system to send the desired file(s) then select the DOWNLOAD FILE option from the Control menu. The PgDn key (if not macro defined) may also be used. UNICOM will then prompt you to select a protocol from window as shown in Figure 26.

If automatic downloading is enabled for Zmodem or CompuServe B /Quick B, UNICOM will detect the transfer request and automatically start the download on your end. Otherwise, UNICOM will display the protocol selection window below.

(see manual)
Figure 26. Download Protocol Selection Window

You may choose from XMODEM, YMODEM, ZMODEM, Kermit, CompuServe B, Quick B or ASCII protocols. After a selection has been made, the UNICOM will initiate the download. If ASCII was selected, UNICOM will prompt you for a filename in which to store the file.

The download filename is obtained by the remote host for YMODEM, ZMODEM, KERMIT and CompuServe B & QuickB protocols. UNICOM automatically scans the scroll buffer to obtain a download filename when using XMODEM. If no valid filename is found, UNICOM will prompt you for the name.

Throughout the course of the file transfer, an information window is displayed so that you may easily monitor the transfer operation. This window (shown in Figure 27 below) provides the following information: the name of the file, number of bytes transferred, current block number, error count, estimated transfer time, estimated remaining transfer time, elapsed time, characters per second (CPS), % efficiency and any messages generated from the use of the selected protocol.

(see manual)
Figure 27. File Transfer Information Window

A graphical bar display gives a visual report regarding the amount of data transferred at any time. For uploading, the bar moves down on a scale that reflects the bytes remaining to be transferred.

Downloading causes the bar to move up on a scale indicating the number of bytes received. To abort a transfer in progress, mouse users may select the ABORT button from the information window. Keyboard users must press the ESC key or hit the space bar.

The efficiency report is based on the baud rate of your communications port. If the modem to modem baud rate is lower than your computer to modem baud rate, an unusually low efficiency report will result which may fool you to believe that performance is bad when it is not. If this is the case, use the CPS report to measure performance.

Uploading Files

To upload a file to the remote system, instruct the remote computer to receive a file from you. Initiate the file upload on your computer by selecting UPLOAD FILE from the transfer menu. A protocol selection window will appear as shown in Figure 28.

(see manual)
Figure 28. Upload Protocol Selection Window

The PgUp key (if not macro defined) or the UPLOAD screen button may also be used. After selecting an upload protocol, an upload file selection window will appear to allow you to search your disk for file(s) to be transferred. The file(s) to be uploaded may be entered by name or selected from the listbox containing directory entries. The file selection window shown in Figure 29 will be displayed for non-batch upload protocols.

(see manual)
Figure 29. Non-Batch File Upload Selection Window

The Upload Path determines the default directory for making upload file selections.

The upload file directory listbox may be set to display files from a different drive. Just scroll the drive letter into view within the list box and double click on the entry. To change directories, double click on any directory entry displayed.

For ZMODEM or YMODEM file transfers, a Batch Upload File Selection box will appear containing two listboxes as shown below in Figure 30.

(see manual)
Figure 30. Batch Upload File Selection Window

The listbox on the left displays the current directory of files from which to select. The listbox on the right contains the selected files for transfer. Batch selections are made by double clicking the mouse on a selected file. Keyboard users must highlight the selection and use the tab key to activate the ADD button.

Once this is done, the file is added to the right listbox containing selected files. After making the file selection(s), press GO!. The transfer will begin and an information window will appear for transfer monitoring.

Marking a File for Download

UNICOM provides a file marker feature that frees you from remembering the filenames for the files you wish to download. As a BBS displays its list of available files, highlight the filename of a desired file with the mouse, then select Mark File from the transfer menu or select the file mark icon from the toolbar. After marking the 1st file, a File Marker window will be displayed.

The file marker window is titled "Marked Files". In this window you may :

1) Delete the currently highlighted filename selection by pressing the Del button.

2) Transmit the currently highlighted filename to the remote computer. - or-

3) Close the window - thus losing all marked file selections by pressing Exit.

Figure 31 illustrates a common interaction with a BBS requesting filenames for downloading.

(see manual)
Figure 31. File Marking

In the window above, 5 files were selected using the file marker feature. The BBS is requesting filenames. Filenames are transmitted (and emptied) from the filemarker listbox to the BBS one at a time as you press the Send button. When the filemarker listbox is completely emptied, the window is automatically closed. Note: The author lost his license # when creating this example. (UNLICENSED EVALUATION). Don't let this happen to you.

Automatic Downloading

The typical downloading procedure begins by instructing the remote computer to send a file. Then on your end, you command the communications software to begin receiving after 1st selecting the appropriate protocol. This second step can be avoided by using the automatic download feature provided with UNICOM..

UNICOM supports automatic downloading when using ZMODEM, CompuServe B or Quick B protocols. This option is enabled or disabled from the General Setup window.

When enabled, UNICOM continuously monitors all incoming characters for special signatures that indicate the start of a file transfer. These signatures are unique and can identify the protocol type. Not all file transfer protocols are designed to provide this capability.

When UNICOM encounters a Zmodem or a CompuServe signature, downloading is automatically initiated.

File Transfer Protocols

Many popular file transfer protocols have been implemented in UNICOM for full background operation:

XMODEM

XMODEM is a block-oriented error checking protocol introduced to the public domain by Ward Christensen. It is widely used by many electronic bulletin board systems. XMODEM transfers a single file at a time. The protocol uses a checksum or cyclic redundancy check (CRC) for error checking. XMODEM can handle text or binary files with over 99% accuracy. UNICOM provides three common variations of XMODEM: XMODEM Checksum, XMODEM CRC and XMODEM 1K(old YMODEM).

YMODEM BATCH

The YMODEM Batch protocol is an extension to the XMODEM/CRC protocol that permits transmission of full pathnames, file length, file date, and other attribute information. The design approach of the YMODEM Batch protocol is to use the normal routines for sending and receiving XMODEM blocks in a layered fashion similar to packet switching methods.

YMODEM G

Developing technology is providing phone line data transmission at ever higher speeds using very specialized techniques. These high speed modems, as well as session protocols, provide high speed, nearly error free communications at the expense of considerably increased delay time.

This delay time is moderate compared to human interactions, but it cripples the throughput of most error correcting protocols.

YMODEM G has proven effective under these circumstances. YMODEM G is driven by the receiver, which initiates the batch transfer by transmitting a G instead of C. When the sender recognizes the G, it bypasses the usual wait for an ACK to each transmitted block, sending succeeding blocks at full speed, subject to XOFF/XON or other flow control exerted by the medium.

The sender expects an initial G to initiate the transmission of a particular file, and also expects an ACK on the EOT sent at the end of each file This synchronization allows the receiver time to open and close files as necessary.

If an error is detected in a YMODEM G transfer, the receiver aborts the transfer with the multiple CAN abort sequence. The ZMODEM protocol should be used in applications that require both streaming throughput and error recovery.

ZMODEM

ZMODEM is a second generation streaming protocol for text and binary file transmission between applications running on microcomputers and mainframes. Zmodem is designed for optimum performance with minimum degradation caused by delays introduced by packed switched networks and timesharing systems.

ZMODEM accommodates network and timesharing system delays by continuously transmitting data unless the receiver interrupts the sender to request retransmission of garbled data. ZMODEM in effect uses the entire file as a window. Using the entire file as a window simplifies buffer management, avoiding the window overrun failure modes that affect other windowing protocols.

Resuming an Aborted Zmodem Transfer

UNICOM supports the ZMODEM Crash Recovery feature so aborted transfers may resume at the point of interruption. When ZMODEM Resume is specified by the receiver on the next transfer attempt, the receiver compares the size of the interrupted file to that of the sender. If the sending file is longer, the receiver instructs the sender to resume transmission at the appropriate offset and appends the incoming data to the existing local file.

Kermit

Kermit is a packet-oriented protocol developed at Columbia University and is available on many computer systems. UNICOM supports only the basic implementation of the Kermit protocol. The following Kermit options are fixed in this release of UNICOM.

No 8 bit Prefixing - (means : no 8 bit data thru 7 bit links)
One char checksum
No repeat prefix

CompuServe B and Quick B

CompuServe B is similar to XMODEM in the send/check/reply design but is a host-controlled protocol. The host (CIS) always tells the remote what to do next, no matter what the direction of transfer. Each packet (block) of the transfer contains a header describing the contents of the packet, either information, data or control.

UNICOM will automatically step up to QUICK B (QB) when requested to do so by CompuServe. QB is a thoughtful extension of B. The extensions include two new types of packets and acknowledgment windowing for drastically improved bandwidth. QB adds CRC-type checksumming capability to the B arithmetic checksum for improved error detection, and extends packet size to 2K (although the current size used is 1K packets) or reduce packet size to 256 bytes.

ASCII

ASCII is a very basic method of data transfer. No error detection is performed and the file should be free of non-printable characters other than carriage returns or linefeeds. Some host systems require XON/XOFF flow control to be used for handshaking purposes during ASCII transfers. If the remote host computer requires XON/XOFF flow control, enable this handshake option in the Comm Port setup window.

Using External Protocols

External protocols may be activated by executing a specially constructed UNICOM script file. Though their use is **not recommended** (or guaranteed), external protocols will require UNICOM to perform the following:

1) Release control of the communications port.
2) Activate the external protocol using specific parameters.
3) Go to sleep.
4) Wake up and re-connect to the communications port when the external protocol has completed.

The script file to accomplish these steps (as described) may resemble the following example:

```
INPUTSTRING FILE       "Enter a Download Filename"
INPUTSTRING EXTERN  "Enter the External Protocol"
PORT NONE              (UNICOM gives up current port)
RUN   COMMAND.COM EXTERN "download command" FILE "port cfg"
SHOWWINDOW         ("UNICOM 2.0",         HIDE)

WHILE NOT FOUND     ;(Wait till the protocol is done)
   FINDWINDOW ("Command")
ENDWHILE
                           ;(Re-connect and display UNICOM)
PORT "LASTDEVICE;LASTBAUD;LASTPARITY;LASTWORD..."
SHOWWINDOW "UNICOM 2.0" SHOW
EXIT
```

The above example is presented as a suggestion for constructing script files to support external protocols. It should not be considered as a fully functional model.

External protocols written for DOS may behave poorly in the Windows environment. Multitasking performance can be drastically reduced when executing DOS (external) applications using Windows 3.

Use of external protocols with UNICOM is possible, but not recommended.

Section 9
WinScript Command Language

Introduction to WinScript

WinScript is a Windows script language that gives you the ability develop custom applications for use with UNICOM. A rich set of functions have been provided that support communication, file management, window control, modem control, system interface, windows graphics and more.

WinScript Language Elements

WinScript provides **Conditional Expressions** that allow you to make program decisions to alter program control.

```
WHILE (expression is true)
        ; do these commands
ENDWHILE

IF (expression is true )
        ; do these commands if true
ELSE
        ; do these commands if not true
ENDIF

SWITCH (expression)          ; The switch expression is compared to each case
        CASE expression1     ; if this case matches control is passed to the next line
                ; do these commands
        ENDCASE
        CASE expression2
                ; do these commands
        ENDCASE
DEFAULT                      ; this is optional
        ; execute if no cases expressions match the switch expression
ENDSWITCH
```

Program Control can be directed unconditionally using **GOSUB** and **GOTO**

Types and Operators

Types

There are a few data types in WinScript

string	a variable length collection of characters no larger than 255 in length
integer	a 16bit signed value
word	a 16bit unsigned value
long	a 32 bit signed value
handle	same as word
bool	a 16 bit value TRUE is represented as 1, FALSE as 0.

Program variables can be alphanumeric up to 31 characters long
Functions returning values can be passed as function arguments.

Constant expressions are allowed:

Strings should be enclosed in double quotes "i am a string"
integers can be used just as they are, HOWEVER
Negative numbers MUST be enclosed in parenthesis (-1234).
UNICOM 2.0 flags are supported, they are
SUCCESS, WAITFOR, FOUND and CONNECTED

Arithmetic Operators

The binary arithmetic operators are +, - ,* , / and modulus %

Relational Operators

>, >=, <, <=

Equality Operators

== , !=

Bitwise Logical Operators

&	bitwise AND
|	bitwise OR
<<	left shift
>>	right shift
~	one's complement

Operator Precedence

( )	Parenthesis	**HIGHEST**
! ~	(bitwise not)	
* / %	(mult,div,mod)	
+ -	(add , sub)	
<< >>	(bitshift)	
== !=	(equality)	
|	(bitwise or)	
&&	(logical and)	
||	(logical or)	
=	(assignment)	**LOWEST**

Executing WinScript Command Files

WinScript command files are typically executed manually by selecting Execute from the Script menu.
A script file selection window will appear and display all script files found in the UNICOM files directory as shown in Figure 32 below.

Figure 32. Script Selection Window.

To activate a script, enter the filename in the editbox above and select accept, or just double-click using the mouse on the filename within the listbox.

Scripts may also be executed in the following ways:

Assigning a script to a function key: Individual scripts may be assigned to a particular function, that when pressed, begin execution. See the Keyboard Settings topic in the UNICOM setup section for information on assigning scripts to function keys.

User activated from the Dialing directory: Script filenames may be entered in the dialing directory and activated without causing UNICOM to dial. If there is no number defined in the phone number field, UNICOM will immediately execute any script whose filename is listed in the entry.

Automatic execution upon successful dialing: UNICOM provides a script field to be associated with each host system defined within the dialing directory. If a filename is provided in this field, the script will automatically execute upon successful dialing from the directory.

AutoStart Script execution: The general setup window contains an AutoStart Script file edit box. If a name is defined here, UNICOM will automatically execute the script upon each program activation.

Script Scheduling: Up to 8 different scripts may be programmed to execute at specific days and times using the Scheduler feature provided with UNICOM. See the section on Script Scheduling for information on how to use this feature.

Script Execution from a Script: A script language file may be executed from another script file with the use of the Execute script command. See the Script Statement and Commands topic.

Debugging a WinScript Program

A trace mode has been provide to let you view each program line as it is being executed. Select trace from UNICOM's script menu then execute your program. The TRACE script command will allow you to control this option from with your script application.

Each command will be displayed to UNICOM's status line as it is being executed. A 1/4 second delay is introduced for each command to allow the user time to view the status line.

WinScript Command Language Definitions

Command arguments contained in [brackets] are optional; all others are required. Vertical bars are used to separate all supported argument values. Conditional flag(s) affected by execution of a command are listed in the 'Returns:' description.

Script Recording

UNICOM provides a script recorder that will automatically construct a script command file according to your interaction with a remote host computer. The script recorder may be started manually by user selection or automatically upon connecting to a remote host after dialing. These activation methods are described as follows:

Manual Activation: UNICOM can initiate script recording 'on the fly' whenever you select Record Now from the Script menu. Script recording using the manual method will create a script named 'RECORD.SCR". This name cannot be changed and any previous contents of this file will be lost.

Automatic Activation: When Record on Dial is enabled from the Script menu, UNICOM will enter record mode immediately upon successful dialing when using the dialing directory. The script filename listed in directory entry being dialed will be recorded (written) to. UNICOM displays a warning when dialing using this mode to warn the user that the script file will be overwritten (if it exists).

The resulting script constructed using this method may contain commands generated by modem connect responses. If this happens, edit the resulting file with a text editor to remove any unwanted commands.

The Script Scheduler

Script scheduling is a means by which UNICOM can (on its own) execute script command files at predetermined days and times. Programming the script scheduler is much like programming recording times on your own video tape recorder.

Eight events can be scheduled. An event is defined as a UNICOM WinScript command language file that will begin execution on a specific day and time. To activate the Script Scheduler configuration window, select Scheduler from the script menu. The configuration window will appear as shown in Figure 33.

(see manual)
Figure 33. Script Command File Scheduler

The Script Scheduler configuration options and controls are defined as follows:

Event Enable: A checkbox is associated with each event 1-8 that can be defined. A check indicates that UNICOM will watch for the event once the timer is armed. No check indicates that the event has already been performed or the event will not be scheduled.

Time Setting: Select the hours (0-23) and minutes (0-59) in the day for which the event will take place. The accuracy of the time setting is plus or minus one minute of the designated time.

Day: Specify the day of the event: Sunday through Saturday.

Script Filename: Enter the UNICOM WinScript command file that will be activated for this event being defined. The script file must be located in the UNICOM files directory.

Repeat Event: When checked, events that have been performed will be re-scheduled for the following week at the same day and time.

Event Select: This button is used to select the event to be edited. Pressing Event Select will cause the next event to be displayed in the Scheduler window. After reaching the last event, the next press of the Event Select button will cause the first event to be displayed.

Arm Timer: Once all events have been defined and ready to be scheduled, press the ARM TIMER button. UNICOM will return to its previous operation and keep watch for the day and time for the events that have been defined. In order for UNICOM to watch for a given event, the event enable checkbox must have been selected in the configuration window.

When the day and time has been reached for an event, UNICOM checks to see that it is in terminal operation mode. If not, UNICOM will assume that it is executing another event or operating in some other special mode. The event that was triggered will be placed in a wait state for a maximum of 15 minutes. Should UNICOM return to terminal mode within that amount of time, the event will be processed. If not, the event will be lost.

Disarm Timer: When this button is pressed, any previously scheduled events are cancelled. Pressing the CANCEL button will have the same effect. If the Scheduler configuration window was activated by mistake, to re-activate the scheduled events, press the ARM TIMER button again.

The scheduler setup information can be saved and restored across UNICOM sessions.

Select Save Setup from the setup menu. If the Scheduler is armed when the setup is saved, UNICOM will arm the scheduler at the start of each UNICOM session. To disable scheduler arming at each session startup, Save your setup after first disarming the scheduler.

Remote Access Using Host Mode

Your computer may be accessed remotely with the use of UNICOM's host mode. Host Mode operates very similar to that of a mini bulletin board system. In order to use Host mode it must first be setup with user information and various system settings. See Host Mode Setup in UNICOM's setup section for information on how to configure host mode.
Host mode provides a remote user with the following capabilities.

1) Remote initiated file transfers: Uploading and Downloading with XMODEM, ZMODEM and Kermit.
2) Directory operations: Change directory, List Directory
3) Operator Paging: The sysop can drop into chat mode to talk to the user.
4) Shell to DOS: With the proper access level, a remote user can access the DOS command line.
5) File Display to Screen: A remote user may display the contents of an ASCII file.

Activating Host Mode

Host mode can be activated manually from a menu (or toolbar) selection or automatically using a WinScript command file. Once activated, UNICOM checks its connection type (as set in the Comm Port setup window) to determine if remote access is to be accomplished through a modem or a direct connection.

If the connection type is set to modem, UNICOM will initialize the modem to answer on the number of rings specified in the selected init string section of the modem setup window. This value will be used even if UNICOM is set to operate with the user entered init string. If a modem connect response is encountered, the remote user is prompted to log in. When the user logs out or when host mode is deactivated, UNICOM will command the modem to hang up.

If the connection type is set to computer, UNICOM monitors the line for two consecutive user entered carriage returns. After detecting these carriage returns, UNICOM will initiate the login process.

After the user is logged in, a menu will be presented to the user as displayed in the screen snapshot in Figure 34 below.

(see manual)

Figure 34. UNICOM Operating in Host Mode.

If configured for monitor mode, the host display will reflect the remote users screen. You can type at the keyboard to make selections for a remote user. Characters typed at the host keyboard will override any remote input.

Remote User Operation

Introduction

Once your computer has been configured and set for host mode operation, a remote user may gain access using an assigned userid and password. The level of user access is determined by the access level assigned to the user id.

One of three access levels must be assigned to every host mode user.

Level 1	Full access allowed. Can shell to DOS if host is running in 386 mode.
Level 2	Partial access. Same as Level 1, but user cannot shell to DOS.
Level 3	Limited access. Cannot upload, shell to DOS, or change directories.

A time limit is associated with each access level and is set in the host setup window.

Shell to DOS

Users with level 1 access may shell to DOS if the host is running in 386 enhanced mode. The remote user must type 'EXIT' at the DOS prompt when finished using the shell. There is no guaranteed return to UNICOM's host menu since the modem may drop the connection after the user types 'exit' . If the modem is set to drop the line at the loss of DTR, typing exit at the DOS prompt will cause DTR to drop for a moment before UNICOM regains control. In any event, once a user shells to DOS, UNICOM logs the user out but does not hang up. If exiting DOS does not drop the connection, UNICOM prompt the user to log in.

Chatting with the Sysop

A remote user may request a conversation with anyone who might be at the computer running UNICOM in host mode. When the user selects P(age) Operator, a notification beep will sound. The operator (sysop) can then activate chat mode to converse with the remote user. To terminate this operation, the sysop exits chat mode which will transfer the remote user back to the host command menu.

File Transfers

Files transfers (uploading and downloading) may be initiated by the remote user. Uploading is not allowed for users with access level 3. Three protocols are available, XMODEM, ZMODEM and KERMIT. A remote user initiates a transfer by selecting '(U)pload) or (D)ownload' from the menu. The user will be prompted to select a protocol as follows:

Select Transfer Protocol
(X)modem CRC (Z)modem (K)ermit (A)bort >

For uploading using Xmodem, the remote user will be prompted for the name of the file. Zmodem and Kermit will receive this information automatically. After instructing the UNICOM host to receive a file, the remote user must start the transmission on that end.

For downloading, the remote user will be prompted for a protocol and for the name of the file UNICOM is to transmit. Once activated, UNICOM will display the message: "Start your receiver now". The remote user then initiates the download on that end. If the remote user is using a communication package supporting auto Zmodem downloading (like UNICOM), it will begin downloading automatically.

After the file transfer is complete, UNICOM returns the remote user to the command menu.

Changing Directories

Users with access levels of 1 or 2 may move between any subdirectory on any drive on the UNICOM host computer. When 'C' is selected at the menu prompt, UNICOM will prompt the remote user for a new directory as shown below in the following example:

[60 min left] d:\>change Directory
New Directory>c:\util

Typing Files

A remote user may examine the contents of ASCII text files by selecting '(T)ype' from the command menu. UNICOM will prompt the user for the name of the file. At each screenful of text, UNICOM will prompt: More? (Y/n). After the file is transmitted (or if the operation is aborted) UNICOM returns the user to the command menu.

Displaying Directory Contents

All remote users may examine the contents of the directory which they are currently logged into. Selecting '(L)ist Directory instructs UNICOM to transmit name, size, date and time information for each file in the directory. The following example illustrates the directory format.

```
[60 min left] c:\>list Directory
COMMAND.COM        47845  4/ 9/91          5: 0: 0
SSTBIO.SYS         18640  6/11/90          9: 0:14
SSTDRIVE.SYS7733   6/11/90          9: 0:14
SSTSETUP.EXE       185724           5/ 1/91 14: 4:16
BUFFERS.COM7615    5/ 1/9   14: 4: 2
```

After each screenful of text the remote user will be prompted for More (Y/n?).

Using Chat Mode

UNICOM's chat mode allows you to easily communicate with a person on the other end of the communications link with your keyboard. To activate chat mode, select Chat from the control menu or activate the Chat Lizard icon from the toolbar. Be careful, the Chat Lizard looks happy enough, but is easily angered when someone pushes his button.

When chat mode is activated, a scrollable input window is created at the bottom of the terminal screen. Input your message into this window one line at a time. Use the backspace key to edit the line before it is sent. To transmit the current line, press the Enter or Return key on your keyboard. After the line is transmitted, it is moved up into the scroll area of that window.

Characters received from the remote user will be displayed on the terminal screen above the chat input window. Figure 35 (below) illustrates the use of chat mode.

(see manual)
Figure 35. Chat Mode Operation

To exit Chat Mode, select the Chat Mode option from the Control Menu.

Event Logging

An event recording capability is provided to allow monitoring of program operation and host user interaction. Events are categorized as either program events or host events.

Program events include: Modem Hangup, Modem Initialization, Modem Dialing, Modem Dialing Abort, Script Execution, File Uploading, and File Downloading.

Host events include: File Display, User ID Timeout, Valid Password Entered, Invalid Password Entered, Hangup Performed, User Requested Help, User Examined Directory, User Changed Directory, User Uploaded a File, User Downloaded a File, Operator was Paged, Userid Entered.

All events are recorded with a date and time stamp.

Logging of program events is enabled or disabled using the General setup window. Placing a check in the checkbox: "Log events to file" will activate logging to the file named in the edit box following this control.

Logging of host events is controlled by the host setup window. If checked, the checkbox labeled "Log Events" will enable recording of host events to the event file listed in the general setup window.

Events are recorded to the event file in ASCII text format. The date and time information is written followed by the event on the following line. Example event entries are shown below:

```
Thu Aug 08 00:39:48 1991
File Download chess101.zip
Thu Aug 08 14:19:32 1991
File Upload c:\winword\info.wri
```

File Logging

Incoming screen characters may be captured to a log file. Unwanted terminal escape characters and control characters may be filtered using the Log filter option in the general setup window.. To activate file logging, select the File Log option from the Files menu or activate the Log File button at the bottom of the display. A window will appear to prompt you for the log filename as shown in Figure 36.

(see manual)
Figure 36. Log Filename Window

UNICOM will enter the name of a default log file as listed in the general setup window. Enter a valid filename then press Ok. Should you enter a filename of an existing file, UNICOM will ask if you wish to append to this file. A NO response will abort the file log request. File logging is disabled should the program leave terminal mode (enter host mode, for example) and will resume upon return.

Using the Utility Menu

The Utility Menu is an application starter containing application names configured by the user. Once the menu is configured with application entries , UNICOM can 'Launch' them quickly and easily.

UNICOM may come configured with an example utility menu containing programs that do not exist on your computer. When using the utility menu for the first time, remove any existing utility entries using the utility menu editor from the setup menu. This editor lets you add your own applications to the menu by navigating through your disk drive. For information on configuring the utility menu, see the utility topic in the setup section.

Once you have configured the utility menu with application programs they may be 'Launched' directly from the menu or from a hot key.

A program parameter line can be constructed using the Set Param entry in the Utility Menu. Activating this menu item will produce the following screen:

(see manual)

Enter the parameter to be passed into the edit box above. It will be passed to the next application to be launched. The Window Activation option controls the startup appearance of the application window. A Normal selection will let Windows determine the size of the window. Zoom causes the application to start up in full screen mode. Minimize will iconize applications upon startup. These Window activation options are relevant to Windows applications only.

Should an error occur in the activation of any application listed in the Utility Menu, UNICOM will display the Utility Menu Setup Window.

Operating Multiple UNICOM Instances

Multiple UNICOM applications may operate concurrently in the Windows multitasking environment. Running additional instances of UNICOM will allow you to communicate with multiple remote computers at the same time.

Additional UNICOM instances may be activated as described in the section 'Starting UNICOM'. Each new instance should be invoked with a configuration file that will access an unused communication port that uses a unique IRQ. If the new instance is Spawned from a previous UNICOM instance, the new instance will refer to a configuration named UNICOM2.CFG.

Hardware and communication driver support will limit the number of usable UNICOM instances. In most cases this number will be two since the Windows 3.0 port driver only allows 2 serial ports to be configured for use. This does not apply to EISA or Microchannel machines.

When driver support arrives to let the user operate each serial port on a unique IRQ, a minimum of 4 UNICOM instances may be readily used- provided the serial port hardware will allow IRQ reassignment.

Using High Speed Modems

UNICOM can be used with high speed modems to deliver transfer rates above 1500 cps when using a computer equipped to run Windows 'efficiently'. The baud rate between UNICOM and a high speed modem is usually fixed at the highest link speed. Error correcting modems usually negotiate their own link speed (which should be lower than the PC to MODEM baud rate). This configuration allows for increased throughput for any modem compression method being used.

When operating with the Windows port driver at higher baud rates, please note:

A) The Windows port driver becomes unreliable at baud rates above 9600. If you switch between applications that access a drive during a UNICOM file transfer above 9600 baud, CRC errors may result. At this point you may wish to contact Microsoft if this has not been corrected in your version of Windows (above 3.0). If they can't help, 3rd party replacement drivers are available.

B) Your computer must be equipped to run Windows efficiently. That is, you will need at least a 20mhz 386 DX with at least 2mb of physical ram available to Windows (after smartdrive and ramdrive) and a fast hard disk (28ms or less).

Many 286 or 386 SX machines do not have the extra horsepower required to run UNICOM at the higher baud rates. Even on a fast machine, if there is not enough memory available to Windows, excessive swapping may result. Too much swapping could interfere with UNICOM's ability to empty it's receive buffer before it overflows when performing file transfers with streaming protocols.

UNICOM requires no special configuration to support high speed modems. As mentioned, if your modem supports a fixed baud rate, use the highest baud rate supported by UNICOM and the modem. Dialing directory entries are limited to 38.4kb. The baud rate should be fixed at 19.2kb or 38.4kb.

Some modem manufacturers recommend using hardware handshaking at high baud rates. If this is the case, set UNICOM to hardware handshake in the Comm Port setup window after configuring your modem with the appropriate setting. Refer to your modem reference manual for more information.

Changing the Sound of UNICOM's Notification Beep

If the notification beep option is enabled in the General Setup window, UNICOM will play a musical sequence of notes after each file transfer is complete or upon successful dialing. These notes are stored a text file named UNICOM.SND. If this file is not found, UNICOM will default to a predefined sequence of notes. You may change this music to any desired sequence by editing the file UNICOM.SND. This file is in ASCII text format and it contains 2 values per line separated by 1 comma.

The first value is a note value from 1 to 84 (seven possible octives). The second value following the comma is the reciprocal of the duration of the the note. 1 is a full note, 2 is a half note, 4 is a quarter note and so on.

Below is the contents of an example UNICOM.SND file.

```
54,8
54,8
54,8
59,8
54,4
```

Any number of notes may be placed in the file as long as it is formatted similar to the above example.

Command Summary

File

Log File	- Toggle File Logging ON or OFF
Log Printer	- Toggle Printer Logging On or Off
Spawn UNICOM	- Activate another UNICOM Instance
Print Screen	- Send a Terminal Snapshot to the Printer
Print Buffer	- Print a Scroll Buffer Snapshot
Printer Setup	- Configure the active printer
Control Panel	- Activate the Windows Control Panel
Run	- Run Another Windows (or Dos) application
Exit	- Exit the program
About	- About UNICOM

Edit

Copy	Copy selected screen text to the Clipboard
Copy Window	- Copy only viewable screen text to the Clipboard
Copy Buffer	- Copy the entire scroll buffer to the Clipboard
Paste	- Transmit any Clipboard text to the remote computer
Send	- Transmit any selected screen text back to the remote computer
Send with CR	- Transmit any selected screen text back to the remote with a carriage return
Select All	- Select (highlight) all text in the scroll buffer
Erase	- Erase any selected text from the scroll buffer or screen
Erase Terminal	- Erase the terminal screen (rows 1 - 24)
Erase Buffer	- Erase the contents of the scroll back buffer
Find	- Display a search window for locating text in the scroll buffer
Find Next	- Repeat the last Find operation
Log File	- View or Edit a file captured via File logging
Event File	- View or Edit a UNICOM event file.

Setup

Comm Port	- Select and Configure a Communications Port
Terminal	- Change the Terminal Settings
Modem	- Set Special Features in your Hayes Modem
Keyboard	- Define the meaning of your Keyboard keys
Host	- Set Host Mode Operating Parameters
File Path	- Define the UNICOM and UP/Download filepaths
ASCII Xfer	- Set ASCII Transfer Parameters
Kermit	- Set Kermit Transfer Options
General	- Set start window size, editor, auto script
Utility	- Configure the Utility Applications Menu
Zmodem	- Set Zmodem transfer options.
Translation	- Activates the Translation Table Editor
Save Setup	- Save the Program Configuration

Transfer

Download File	- Receive a File from the Remote System
Upload File	- Send a File to the Remote System
Mark File	- Mark a filename on the screen for downloading

Command Summary (continued)

Phone

Directory	- Display the Dialing Directory
Dial	- Display the Manual Phone Dialer
Hang Up	- Instruct the modem to hang up
Send Break	- Set the line to a break state for 350 ms.
Answer Now!	- Instruct the modem to answer an incoming data call

Control

Host Mode	- Toggle Host Mode ON or OFF
Chat Mode	- Toggle Chat Operation ON or OFF
Timer Mode	- Toggle display of elapsed or current time
User Keys	- Toggle Display of User defined screen buttons
Scroll Bars	- Toggle Display of Horizontal and Vertical Scroll Bars
Scroll Back	- Scroll the screen back 1 page
Scroll Forward	- Scroll the screen forward 1 page
Scroll Left	- Scroll the screen left 1 column
Scroll Right	- Scroll the screen right 1 column

Script

Execute	- Execute a WinScript Command File
Stop	- Stop Execution of Currently Executing Script
Trace	- Toggle echo of Script File Commands
Scheduler	- Display Script Scheduler Configuration Window
Record on Dial	- Automatic Script Recording on Dialing from the Directory
Record Now	- Activates Script Recording
Edit	- Activate Script Editor with Selected Script File
Edit Last	- Activate Script Editor with Last Accessed Script
Create	- Activate Script Editor with no passed filename

Utility

Set Param	- Set the application parameter line.

Help

Index	- Activate Help for UNICOM
Read Me	- Display UNICOM Release Notes

DEC VT102 Functions

unsupported features are marked with an asterisk (*):

Escape Seq	Mnemonic	Action
ESC D	IND	Index, moves cursor down one line, can scroll
ESC E	NEL	Move cursor to start of line below, can scroll
ESC H	HTS	Set one horizontal tab at current position
ESC M	RI	Reverse Index, cursor up one line, can scroll
ESC Z	DECID	Identify terminal (response is ESC [? 6 c)
ESC c	RIS	Reset terminal to initial state
ESC =	DECKPAM	Enter keypad application mode
ESC >	DECKNPNM	Enter keypad numeric mode
ESC 7	DECSC	Save cursor position and attributes
ESC 8	DECRC	Restore cursor from previously saved position
ESC # 3	DECDHL	Double height and width line, top half
ESC # 4	DECDHL	Double height and width line, bottom half
ESC # 5	DECSWL	Single height and width line
ESC # 6	DECDWL	Double width single height line
ESC # 8	DECALN	Test screen alignment, fill screen with E's
ESC [Pn @	ICH	ANSI insert Pn spaces at and after cursor
ESC [Pn A	CUU	Cursor up Pn lines, does not scroll
ESC [Pn B	CUD	Cursor down Pn lines, does not scroll
ESC [Pn C	CUF	Cursor forward, stays on same line
ESC [Pn D	CUB	Cursor backward, stays on same line
ESC [Pn; Pn H	CUP	Set cursor to row, column (same as HVP)
ESC [Ps J	ED	Erase in display: 0 = cursor to end of screen, inclusive 1 = start of screen to cursor, inclusive 2 = entire screen, reset lines to single width, cursor does not move.
ESC [Ps K	EL	Erase in line: 0 = cursor to end of line, inclusive 1 = start of line to cursor, inclusive 2 = entire line, cursor does not move
ESC [Pn L	IL	Insert Pn lines preceding current line.
ESC [Pn M	DL	Delete Pn lines from current downward, incl.
ESC [Pn P	DCH	Delete Pn chars from cursor to left, incl.
ESC [Pn; Pn R	CPR	Cursor report (row, column), sent by terminal
ESC [Pn c	DA	Device attributes (reports ESC [? 6 c)
ESC [Pn; Pn f	HVP	Set cursor to row, column (same as CUP)
ESC [Ps g	TBC	Tabs clear, 0 = at this position, 3 = all
ESC [4 h	IRM	Insert mode on
ESC [20 h	LNM	Set newline mode (cr => cr/lf)
ESC [4 l	IRM	Replacement mode on
ESC [20 l	LNM	Reset newline mode (cr => cr)
ESC [? Ps;...;Ps h	SM	Set mode, see table below
ESC [? Ps;...;Ps l	RM	Reset mode, see table below

Ps	Mnemonic	Mode	Set (h)	Reset (l)		
0	error (ignored)					
1	DECCKM	cursor keys	application	cursor/numeric		
2	DECANM	ANSI/VT52	ANSI/VT102	VT52		
3	DECCOLM	Columns	+132 col		80 col	
4	DECSCLM	*Scrolling	smooth		jump	
5	DECSCNM	Screen	reverse video	normal		
6	DECOM	Origin	relative	absolute		
7	DECAWM	Autowrap	on		off	
8	DECARM	*Autorepeat	on	off		
9	DECINLM	*Interlace		on		off
18	DECPFF	*Printer termination character, use FF if set				
19	DECPEX	*Printer extent,set=screen,off=scrolling region				

DEC VT102 Functions (continued)

ESC [Pn i	MC	*Printer controls (Media Copy)
0		Print whole Screen

4		Exit printer controller (transparent print)
5		Enter printer controller (transparent print)
ESC [? Pn i	MC	*Printer controls (Media Copy)
1		Print line containing cursor
4		Exit auto print (stop echoing to printer)
5		Enter autoprint (echo screen chars to printer)
ESC [Ps;...;Ps m	SGR	Select graphic rendition
		0 = all attributes off (#'s 1, 4, 5, 7)
		1 = bold, intensify foreground
		4 = underscore (reverse video on IBM CGA)
		5 = blink
		7 = reverse video
		non-DEC extensions: 30-37 = foreground color = 30 + colors
		40-47 = background color = 40 + colors
ESC [Ps n	DSR	Device Status Report.
		Response from VT100: 0=ready, 3=malfunction.
		Command to VT100: 5=report status with DSR,
		6=report cursor position using CPR sequence.
ESC [Ps;...;Ps q	DECLL	*Load LEDs, Ps = 0 means clear LED #1-4
		Ps = 1,2,3,4 sets LED # 1,2,3,4 on status line.
ESC [Pn; Pn r	DECSTBM	Set top and bottom scrolling margins, resp. ESC [r resets margin to full screen.
ESC [sol x	DECREQTPARM	Request terminal parameters, see table below
ESC [sol; par; nbits; xspeed; rspeed; clkmul; flags x		
	DECREPTPARM	*Reports terminal parameters
		sol = 0 request; terminal can send unsolicited reports - supported as sol = 1 below.
		sol = 1, request; term reports only on request
		sol = 2, this is a report (DECREPTPARM)
		sol = 3, terminal reporting only on request
		par = 1 none, 2 space, 3 mark, 4 odd, 5 even
		nbits = 1 (8 bits/char), 2 (7 bits/char)
		xspeed,rspeed = transmit & receive speed index
		0,8,16,24,32,40,48,56,64,72,80,88,96,104,112,120,128 correspond to speeds of
		50,75,110,134.5,150,200,300,600,1200,1800,2000,2400,3600,4800,9600,19200,
		and 38400 baud. clkmul = 1 (clock rate multiplier is 16)
		flags = 0-15 (Setup Block #5), always 0 here
ESC [2; Ps y	DECST	*Confidence tests - not supported
ESC (A	SCS	*G0 points to UK symbols SCS = Select character sets
ESC) A	SCS	*G1 points to UK symbols
ESC (B	SCS	G0 points to ASCII symbols
ESC) B	SCS	G1 points to ASCII symbols
ESC (0	SCS	G0 points to special (line drawing) graphics
ESC) 0	SCS	G1 points to special (line drawing) graphics
ESC (1	SCS	G0 points to alt char ROM - national symbols
ESC) 1	SCS	G1 points to alt char ROM - national symbols
ESC (2	SCS	G0 points to alt graphics ROM - as ESC (0
ESC) 2	SCS	G1 points to alt graphics ROM - as ESC) 0
^E	ENQ	*Answerback message (not supported)
^G	BELL	Sound VT102 beep
^H	BS	Backspace, move cursor left one character
^I	HT	Horizontal tab, move cursor to next tabstop
^J	LF	Linefeed, move cursor down one line
^K	VT	Vertical Tab, treated as a line feed
^L	FF	Formfeed, treated as a line feed
^M	CR	Carriage return, move cursor to col 1
^N	SO	Select usage of G1 character set
^O	SI	Select usage of G0 character set
^X	CAN	Cancel escape sequence in progress
^Z	SUB	Treated as a CAN

Other extensions:

ESC [25; Pc f	VT52/VT100 move cursor to 25th line.
ESC [25; Pc H	VT52/VT100 move cursor to 25th line.

DEC VT52 Terminal Functions

Escape sequence	Description of action

ESC A	Cursor up
ESC B	Cursor down
ESC C	Cursor right
ESC D	Cursor left
ESC F	Enter graphics mode
ESC G	Exit graphics mode
ESC H	Cursor home
ESC I	Reverse line feed
ESC J	Erase to end of screen
ESC K	Erase to end of line
ESC V	Print cursor line
ESC X	*Exit Printer Controller mode, transparent print
ESC Y row column	Direct cursor address, offset from space
ESC W	*Enter Printer Controller mode,transparent print
ESC Z	Identify (response is ESC / Z)
ESC ^ (caret)	Enter autoprint mode (printer echoes screen)
ESC _ (underscore)	Exit autoprint mode
ESC]	Print Screen
ESC =	Enter alternate keypad mode
ESC >	Exit alternate keypad mode
ESC <	Enter ANSI mode (changes to VT102)

Terminal Function Key Mapping

Key	ANSI Keypad Numeric Mode	ANSI Keypad Appl Mode	VT52 Keypad Numeric Mode	VT52 Keypad Appl Mode
0	0	ESCOp	0	ESC?p
1	1	ESCOq	1	ESC?q
2	2	ESCOr	2	ESC?r
3	3	ESCOs	3	ESC?s
4	4	ESCOt	4	ESC?t
5	5	ESCOu	5	ESC?u
6	6	ESCOv	6	ESC?v
7	7	ESCOw	7	ESC?w
8	8	ESCOx	8	ESC?x
9	9	ESCOy	9	ESC?y
PF1	ESCOP	ESCOP	ESCP	ESCP
PF2	ESCOQ	ESCOQ	ESCQ	ESCQ
PF3	ESCOR	ESCOR	ESCR	ESCR
PF4	ESCOS	ESCOS	ESCS	ESCS
-	-	ESCOm	-	ESC?
,	,	ESCOl	,	ESC?
.	.	ESCOn	.	ESC?n
+	+	ESCOM	+	ESC?M

ASCII Table

DEC	HEX	CHR	DEC	HEX	CHR	DEC	HEX	CHR	DEC	HEX	CHR
00	00	NUL	32	20	SP	64	40	@	96	60	`
01	01	SOH	33	21	!	65	41	A	97	61	a
02	02	STX	34	22	"	66	42	B	98	62	b
03	03	ETX	35	23	#	67	43	C	99	63	c
04	04	EOT	36	24	$	68	44	D	100	64	d
05	05	ENQ	37	25	%	69	45	E	101	65	e
06	06	ACK	38	26	&	70	46	F	102	66	f
07	07	BEL	39	27	'	71	47	G	103	67	g
08	08	BS	40	28	(	72	48	H	104	68	h
09	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	XON	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	XOFF	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	DEL

WinScript Error Messages

Message	Description
Unexpected Argument:	Wrong argument type
Too Many Nested Switch Stmts:	Switch stack overflow
Too Many EndSwitch Stmts:	Switch stack underflow
Too Many Nested While Stmts:	While stack overflow
Too Many EndWhile Stmts:	While stack underflow
Too Many Gosub without Return:	Return stack overflow
Too Many Return Stmts:	Return stack underflow
Gosub/Goto Label Not Found:	UNICOM could not find the specified label
Current line illegal to previous line:	Control flow is not permitted in the language.
Line to Complex to evaluate:	The expression needs too much memory for UNICOM to evaluate it. Simplify the expression.
Execution Stack Underflow:	Language use error.
Script File Access Error:	The script file could not be found.
Missing EndIf:	EndIf was expected but not found
Missing EndSwitch:	EndSwitch was expected but not found
Missing EndCase:	EndCase was expected but not found
Missing EndWhile:	EndWhile was expected but not found
Unexpected Case:	Case encountered before Switch
Unexpected EndCase:	EndCase encountered before Case
Unexpected EndWhile:	EndWhile encountered before While
Invalid Number of Arguments:	Too little or too many arguments are provided

Here is a list of system related errors. If you encounter any of these errors, contact UNICOM technical support:

Memory Lock Failure	Contact UNICOM technical support
Invalid Memory Pointer	""
Variable Access Failed	""
Unable to Store Variable	""
Unable to Allocate Variable	""
Unable to Update Variable	""
File Positioning Error	""

Communication Error Codes

Communication errors are detected by your communication port hardware the reported to UNICOM from the Windows communication port driver. UNICOM displays these reports in the form of port state messages. The following list describes the port state error codes.

Code	Description
DATA-OVERRUN or RX-OVERRUN	Character received before previous char could be read from the hardware. This suggests that UNICOM is not receiving enough processor time to empty its receive buffer. Also, UNICOM could be set to a wrong handshake setting in its comm port setup window. If you're operating a 286 above 4800 baud, it is a sign that you need a 386.
RX-PARITY	The hardware detects a parity error. Recheck the communication settings. Most common settings are 8 bits, No Parity, 1 stop bit -OR- 7 bits, Even Parity or 1 stop bit.
FRAMING	The hardware detects a framing error. The possible reasons include: 1) UNICOM or the remote host is operating at a different communication settings. 2) If you cannot dial or transfer files and this error is persistent, your serial device may be set to an IRQ that conflicts with another device. 3) Noise was introduced into the link due to hanging up or by a bad phone connection.
BREAK	The hardware detects a break condition.

The following is a list of uncommon errors. If encountered, access the comm port setup window and re-open (initialize) your port then contact UNICOM tech support.

CTSTO	Clear-to-send timeout - hardware handshake error.
DSRTO	Data-set-read timeout - hardware handshake error.
RLSTDO	Receive-line-signal-detect-(data carrier detect DCD) timeout
TX FULL	UNICOM's transmit buffer cannot be sent - .

Questions and Answers

Here is a list of commonly asked questions. If you are experiencing problems with UNICOM, check this list before contacting Data Graphics.

Question: Some rows at the top of the window do not erase when I clear the screen. Can I get rid of these rows?
Answer

Question : I want to set my keyboard to start a program when a particular function key is pressed. How do I set it up?
Answer

Question: UNICOM deleted the file I just uploaded from my disk. What happened?
Answer

Question: My communication ports do not work with UNICOM. What do I do?
Answer

Question: How do I get UNICOM to record a log on script? I don't feel ready to write one myself. **Answer**

Question: When I connect or try to log in, strange characters appear. How come?
Answer

Question: When connected to my favorite BBS, special characters are displayed instead of the normal
line drawing characters. What can I do?
Answer

Question: What File Transfer Protocol should I be using ?
Answer

Question: What Terminal Emulation should I use ?
Answer

Question: The font I am using appears to be spaced to widely apart between each character. Is something set wrong?
Answer

Question: UNICOM is not operating at the performance level I expect with my particular modem. What is wrong ?
Answer

Question: My screen saver does not appear to function correctly with UNICOM or UNICOM does not perform well with my screen saver.
Answer

Question: Where can I find the latest version of UNICOM
Answer

Answer : *Yes. If The bottom of the scroll buffer is being displayed at the top of the terminal screen, the Visible @ Top option is likely enabled in the terminal setup window. Unchecking this option causes UNICOM to dedicate the entire window to the 24 line terminal screen. Leave this option off if you do not wish to view the scroll buffer with the terminal screen*.

Answer : *Use the utility menu setup window to add your desired applications to the utility menu. UNICOM comes configured with example utility menu entries - they should be deleted in the utility menu setup window. After adding your applications to the utility menu, note the position of the application names in the utility menu. Applications will be positioned starting at 2.*

Use the keyboard macro editor (from the setup menu) to define a hot key to activate the utility menu pick that holds the application name. Enter the hot key sequence into the definition field of the key you have chosen: (Example: ^^8x , where x = the position in the menu of your application.

Answer: *The ZMODEM upload option "Unlink after transmission" is likely set. Activate the Zmodem setup window and uncheck this option. This is the only UNICOM feature that could cause this problem*.

Answer: *The windows port driver requires that your port be configured to a unique IRQ setting. DOS applications do not have this restriction. COM1 & COM3 are usually set to conflict at IRQ 4, COM2 and COM 4 are usually set to conflict at IRQ 3. If you do not have a conflict and COM3 or COM4 is not recognized, your bios may not be informing DOS as to your port hardware configuration. Use a port finder utility before starting Windows so your ports will be recognized. If you have two ports that conflict, one of them must be physically disabled or changed to an IRQ that does not conflict with another device. Watch out for BUS Mouse cards or any device that can operate at IRQ 3 or 4.*

Answer: *Two script menu options provide two ways to activate script recording. Record on dial will let you add a script filename to a directory entry to be dialed. When the entry is dialed, UNICOM will enter script record mode upon a successful connection. The resulting script will be stored in the script filename listed for the entry. The second method allows you to start and stop script recording at anytime. UNICOM will send all output to a file named 'RECORD.SCR'.*

Answer: *A mismatch may exist in the communication settings between UNICOM and the remote computer. Activate the Comm Port setup window and doublecheck the settings. The most common communication settings are 7 data bits, 1 stop bit and Even parity -OR- 8 data bits, 1 stop bit and No parity. When dialing from the directory, UNICOM configures your port to the settings listed for the entry being dialed. If the remote modem answers at a different baud rate, UNICOM must be set to the correct baud rate if your modem does not support a fixed baud rate between the PC and the modem.*
UNICOM can be set to automatically switch to the modem link speed using the Set Port to Modem Connect Speed option in the dialing directory.

Answer: *Most fonts use the ANSI character set which contains many symbolic characters in the high order ASCII range. The terminal font is the only font that contains line drawing and special characters normally used in the U.S. When you change fonts, you also can character sets.*
Foreign language users may need to create a translation table to filter incoming or outgoing characters.

Answer: *If your remote host supports Zmodem, use it. It provides the best error detection, correction and transfer throughput of all the protocols. For speed demons with error correcting modems, Ymodem-G is a good choice. On CompuServe, the CompuServe protocols are recommended as they are implemented for auto uploading and auto downloading in UNICOM.*

Answer: *Most Emulations are designed to expect fixed sized fonts. Windows provides proportional size fonts. UNICOM allows these fonts to be used but must map each character into a fixed size cell. By default, the size of this cell is set to the width of the widest character in the proportional set. Hence the appearance of extra spacing - since most characters can fit in a much smaller cell. The character spacing option in the terminal setup window lets you reduce the default cell width at the risk of chopping off part of wide character.*

Answer: *ANSI-BBS is recommended for most bulletin boards. VT-102 or VT-52 is recommended for Mini's that support DEC VT100 terminals. TTY is recommended if none of the above applies.*

Answer: Even if you have a regular modem, see 'Using High Speed Modems' in the Advanced Operation Section.

Answer: *UNICOM has not been tested with screen savers. If your computer is equipped to run Windows efficiently and screen savers are a problem, the screen saver application may be poorly developed for the Windows environment. Intensive visual effects of screen savers are likely accomplished at the expense of other applications processing time. Use of screen savers is not recommended during file transfers.*

Answer: The latest version of UNICOM may always be found on any Windows forum on CompuServe.

Product Support

UNICOM product support is available for registered users only.

For information or technical support, contact UNICOM Technical Support at (206) 432-1201 during technical support hours: 7p.m. to 9p.m. Pacific time Monday through Thursday.

CompuServe users may direct electronic mail to Data Graphics at 71631,464. Internet users can direct email to Data Graphics using the following internet address: 71631.464@compuserve.com. Registered users may fax their questions to directly to the author at (206)432-8673.

Windows Character Sets

(see manual)
OEM Character Set (used by terminal font)

(see manual)
ANSI Character Set

Prefix Character Combinations

Control character prefixing allows unprintable characters to be represented when defining keyboard macros using the Keyboard macro editor. The following table illustrates control prefixing for all supported control characters.

Control Char DEC	HEX		Control Prefix Combination
00	00	NUL	^@
01	01	SOH	^A
02	02	STX	^B
03	03	ETX	^C
04	04	EOT	^D
05	05	ENQ	^E
06	06	ACK	^F
07	07	BEL	^G
08	08	BS	^H
09	09	HT	^I
10	0A	LF	^J
11	0B	VT	^K
12	0C	FF	^L
13	0D	CR	^M
14	0E	SO	^N
15	0F	SI	^O
16	10	DLE	^P
17	11	XON	^Q
18	12	DC2	^R
19	13	XOFF	^S
20	14	DC4	^T
21	15	NAK	^U
22	16	SYN	^V
23	17	ETB	^W
24	18	CAN	^X
25	19	EM	^Y
26	1A	SUB	^Z
27	1B	ESC	^[
28	1C	FS	^\
29	1D	GS	^]
30	1E	RS	not supported
31	1F	US	^_
127	7F	DEL	^?

WinScript Language Statements and Commands

ModemSetRegister
MoveWindow
OutpW
Outp
Port
Quit
RETURN
Rmdir
Rotl
Rotr
Run
Sendfile
Send
ShowWindow
Snapshot
SoundPlayNote
SplitPath
StatusMessage
StrCat
StrChecksum
Strcmpi
Strcmp
Strcpy
StrCRC16
StrCtlToPrefix
StrIsAlphaNum
StrIsAlpha
StrIsGraph
StrIsHex
StrIsLower
StrIsNumeric
StrIsPunct
StrIsSpace
StrIsUpper
StrLen
StrPrefixToCtl
StrRev
StrStripCtl
Strstr
Strtok
StrToLower
StrToUpper
SWITCH
TitleWindow
Trace
Transmit
VarIsDefined
WaitFor
WHILE
WinEnableInput
WinExitWindows
WinFlashWindow
WinGetActiveWindow
WinGetDesktop
WinGetFlags

WinGetFreeSpace
WinGetScrollPos
WinGetScrollRange
WinGetSystemDir
WinGetVersion
WinGetWindowHeight
WinGetWindowsDir
WinGetWindowWidth
WinIsIconic
WinIsVisible
WinIsZoomed
WinReadIni
WinSetScrollPos
WinWriteIni

Abs

int Abs(int expression)

Description:
Returns:
abs returns the absolute value of an integer expression
Example:

```
i = abs((-10 ))
DisplayString(1,1,i)            ;At row 1 column 1, this program displays 10
```

Alarm

int Alarm ([int seconds])

Description:
This function produces a beep for the specified number of seconds.
If the seconds parameter is omitted, the duration will default to 1 second.
Returns:
The duration (in seconds) of the resulting alarm
Example:

```
Alarm(4)        ; sound beep for 4 seconds
```

Assign

any_type Assign(string variablename, any_type data)

Description:
Assign stores the contents of the data argument into a program variable identified by the variablename string.
Returns:
This function returns the value and type of the data argument.
Example:

```
Assign ( x, 32)                 ;creates integer variable x of value 32
Assign (szName,"Dave")          ;creates string variable szName of value "Dave"
```

atoi

int Atoi(string)

Description:
Returns:
Atoi converts a character string representation of a value into an integer value.
Example:

```
szNumericString = "123456"          ; create a numeric string
x = atoi(szNumericString)           ; convert it to an integer x
x = x + 1                           ; operate
DisplayString(10,10,x)              ; Display the integer variable contents
```

This program displays the contents of integer x at row 10, col 10 123457

average

int Average(int expr1[,int expr2,...])

Description:
Returns:
Average accepts a variable number of integer expressions and returns the average value to the nearest integer.
Example:
nAve = Average(3, 5,7, 1, nValue * 4)

break

bool Break([int msec])

Description:
Break places the communication line into a break state for a number of milliseconds specified in the argument. If omitted, the duration will default to 350 msec.
Returns:
TRUE always
Example:

```
Break(700)                    ; places the line into a break state for 700 msec
```

CASE

See **Switch**

Chdir

int Chdir (string pathname)

Description:
Chdir changes the current directory to the drive and path defined in the pathname argument.
Returns:
0 if successful, -1 if not.

Example:

```
if (chdir("c:\windows") ==   0)
        StatusMessage("We changed the default directory")
endif
```

Clear

bool Clear([int foregroundcolor, int backgroundcolor])

Description:

Clear erases the terminal display and optionally sets the foreground and background screen colors. The following table lists the color values:

value	**color**
0	black
1	red
2	green
3	yellow
4	blue
5	magenta
6	cyan
7	white

Returns:

This function returns TRUE always.

Example:

```
; We prompt the user to enter the color numbers as strings

InputString(szBackgroundColor,"Enter a background color")
InputString(szForegroundColor,"Enter a foreground color")

; Then we convert the strings into integer numbers as required by the clear function

Clear(atoi(szForegroundColor),atoi(szBackgroundColor))
```

ClipGetStr

string ClipGetStr(string variablename)

Description:
ClipGetStr copies text from the clipboard into a program variable identified in the argument. If the variable exists, its contents are replaced. The variable is created if it did not previously exist.
Returns:
The clipboard text is returned if successful. If not, "function failed" is returned.
Example:
ClipGetStr(szClipText) ; stores clipboard text into szClipText string

ClipPutStr

bool ClipPutStr(string expression)

Description:
Copies text to the Windows Clipboard from the string argument.
Returns:
TRUE if successful, FALSE otherwise
Example:

```
InputString(szPutText,"Enter Text for the Clipboard")        ; collect the text

if (ClipPutStr(szPutText))                                   ; try to update the clipboard
        StatusMessage("Text Copied to the Clipboard")        ; it worked
else
        StatusMessage("Text Not Copied")                     ; it did'nt
endif
```

CommClearBreak

bool CommClearBreak(void)

Description:
This function restores character transmission and places the transmission line in a nonbreak state.
Returns:
TRUE if successful, FALSE if there is no current port
Example:

```
CommSetBreak()      ; set the line to a break state
TimeDelay(500)              ; wait half a second
CommClearBreak()    ; clear the break state
```

CommEscape

bool CommEscape(string szFunction)

Description:

This function directs the communication device to carry out an extended function specified by the szFunction parameter.

Parameter

szFunction : Specifies an extended function listed as follows:

"CLRDTR"	Clears the data terminal ready signal (pin 20)
"CLRRTS"	Clears the request to send signal
"RESETDEV"	Resets the device if possible
"SETDTR"	Sets the data terminal ready signal (pin 20)
"SETRTS"	Sets the request to send signal
"SETXOFF"	Port acts if as xoff char has been received
"SETXON"	Port acts if as xon char has been received

Returns:

TRUE if function is successful, FALSE in not.

Example:

```
CommEscape("RESETDEV")  ; resets the communication port
```

CommFlush

bool CommFlush(string szQueue)

Description:

This function removes characters in the receive or transmit queue as specified by the szQue parameter.

Parameter	Value	
szQueue	"RECEIVE"	Flush receive queue
	"TRANSMIT"	Flush transmit queue

Returns:

TRUE always

Example:

```
InputString(szQueToFlush,"Enter Receive or Transmit")
CommFlush(szQueToFlush);
```

CommGetError

int CommGetError(void)

Description:

If a communication error occurs using CommRead or CommWrite, Windows locks the port until the error is cleared by calling CommGetError.

Returns:

The code returned identifies the error associated with the last CommRead or CommWrite function and can be a combination of one or more of the values below.

Hex Value	**Meaning**
0000	No error
0001	Receive Queue Overflow
0002	Receive Overrun Error
0004	Receive Parity Error
0008	Receive Framing Error
0010	Break Detected
0020	CTS Timeout
0040	DSR Timeout
0080	RLSD Timeout (DCD)
0100	Transmit Queue Full
0200	LPT Error
0400	LPT Error
0800	LPT Error
1000	LPT Error
8000	Port Mode Not Supported

Example:

```
if (GetCommError() == 0)
  StatusMessage("No Errors Reported)
else
  StatusMessage("Port Error")
endif
```

CommGetPortID

string CommGetPortID(void)

Description:
Returns:
Returns the name of the currently open port.
Example:

```
szPort = CommGetPortID()
DisplayString(1,1,szPort)      ; displays the port in use at 1,1
```

CommGetString

string CommGetString(string defaultstring, int seconds, string echo)

Description:
Returns:

This function will accept a string entered from a user at the other end of the communication link.

Parameter	Meaning
defaultstring	CommGetString returns this string if the function times out.
seconds	time to wait for user to enter string
echo	A "NO" value will not echo the user typed characters back. The default action is to echo user type characters.

Example:

```
Send ("Enter your name^M")
if (CommGetString("no response",40,"YES")!="no response")
        Gosub ProcessName
endif
```

CommInQueCount

int CommInQueCount(void)

Description:
Returns:
This function returns the number of characters in the receive queue waiting to be read.

Example:

```
while (CommInQue() > 0)             ; Read until receive line is idle
    CommRead(szChar,1)
    if (CommGetError())             ; Unlock port if error occurs
        StatusMessage("Port Error")
    endif
endwhile
```

CommOutQueCount

int CommOutQueCount(void)

Description:
Returns:

This function returns the number of characters in the transmit buffer waiting to be sent.
It is useful for determining when the remote has received the data you transmit.

Example:

```
Send("Hello, did you get this message")
while (CommOutQueCount() > 0)
  StatusMessage("Transmitting Message...")
endwhile
  StatusMessage("Message Transmission Complete")
```

CommReadMatching

int CommReadMatching(int nSeconds, string szPattern 1 [,szPattern2 ,...])

Description:

This function will scan the input stream against the string pattern arguments for a duration specified in the nSeconds parameter. This function can support a maximum of 20 patterns.

Returns:

For a successful match, a pattern number is returned representing its argument position in the call. (1 - n).

If no pattern is matched during the specified time, 0 is returned.

Example:

```
if (CommReadMatching(60,"HI","Carla") == 1)
    StatusMessage("HI came in across the line")
endif
```

CommRead

int CommRead(string szBufferName, int nCount)

Description:

This function will request nCount characters from the receive queue and store them in the string buffer identified in the 1st parameter. If szBufferName is not an existing variable, it will be created.

Should a communication error occur when reading the port with this function, Windows will lock the port. CommGetError must be called to clear this lock condition.

Returns:

The number of characters actually read. On error, the result is negative - and the absolute value of the result is the number of characters read.
A return of zero indicates that the receive queue is empty.

Example:

```
while (CommRead(szMybuf,1) == 1)
        StatusMessage("Reading input")
endwhile
```

CommSetBreak

bool CommSetBreak(void)

Description:
This function places the communication port into a break condition.
Returns:
TRUE if successful, FALSE otherwise
Example:

```
CommSetBreak()      ; enter break state
Delay(1000)         ; wait 1 second
CommClearBreak();   ; remove break state
```

See Also

CommWriteDelay

int CommWriteDelay(int msec, string szLine)

Description:
This function will transmit the string specified in szLine to the remote computer. A delay of msec will occur between transmission of each character. If an error occurs, this function will automatically clear any lock condition

Returns:
The actual number of characters transmitted. A negative return indicates an error and the absolute value is the number of characters transmitted.

Example:
```
CommWriteDelay(200,"ATZ")          ; reset the modem
```

CommWrite

int CommWrite(string szLine)

Description:
CommWrite will transmit a string to the remote computer.
If a communication error occurs during transmission, Windows
will lock the communication port until it is cleared with a
call to CommGetError

Returns:
The actual number of characters written. If negative, an error occurred and the number
of characters transmitted is the absolute value of this return.
Example:

```
if (CommWrite("The Lazy dog beat up the brown fox")<0)
        StatusMessage("Communication Error")     ; complain
        CommGetError()                           ; unlock the port
endif
```

DEFAULT

See Switch

Delay

bool Delay(long lmsec)

Description:

This function will pause script execution for the number of milliseconds specified in the lmsec parameter. The receive buffer is not read during this operation and any characters arriving in the receive queue will be buffered. The receive buffer can hold up to 24k (12k in real mode) of data but can overflow if it not emptied fast enough with a read operation.

Returns:

TRUE always

Example:

```
Delay(40000)            ; pauses script execution for 40 seconds
```

Dial

bool Dial(string szNumber)

Description:
Instructs UNICOM to issue a modem dialing command with the passed string as the phone number.
Returns:
TRUE if dialing resulted in a connection as defined in the connect string in the modem setup window. FALSE is returned if the modem returns any of the no connect responses.
Example:

```
while (Dial("12345678") == 0)
        Delay(10000);
endwhile
StatusMessage("Connected!")
```

DisplayString

bool DisplayString(int row, int col, any_type data)

Description:
This function will display data at the specified row and column of the UNICOM terminal
screen. Any data type is allowed to be displayed since it will be automatically converted
to a string for you.

The terminal screen is restricted to rows 1 - 24. The scroll buffer cannot be accessed.
Returns:
TRUE if the function is successful, FALSE otherwise
Example:

```
nRow = 13
nCol= 5
DisplayString(nRow,nCol,"System Going Down")
```

ELSE

See IF

Emulate

bool Emulate(string szTermType)

Description:

Sets the operating terminal type to the type specified in the szTermType parameter. The screen is cleared and the cursor moved to the home position.

Valid Terminal Types are: "VT52", "VT102","ANSI-BBS","TTY"

Returns:

TRUE if the operation succeeded, FALSE otherwise.

Example:

```
if (!Emulate("UNSUPPORTED TERM"))
   Emulate("VT102")
endif
```

ENDCASE

See Switch

ENDIF

See IF

ENDSWITCH

See Switch

ENDWHILE

See WHILE

Execute

bool Execute(string szScriptName)

Description:
This command executes another WinScript command language program. Execution of
the program that calls this function is immediately halted. Control does not return.
Program variables remain intact.
Returns:
TRUE if the call succeeded, FALSE otherwise.
Example:

```
if (!Execute("step2.scr"))
        StatusMessage("We could not branch")
endif
```

Exit

bool Exit(void)

Description:
Terminates execution of the current WinScript Command File.
Returns:
TRUE always
Example:

```
if (x > 32)
   Exit()
else
   Goto MoreProcessing
endif
```

FileChmod

int FileChmod(string szPathname, string szMode)

Description:
FileChmod changes a file access permission setting.
Valid values for szMode are shown below:

"READ"	Read Only Access Allowed
"WRITE"	Write Only Access Allowed
"NORMAL"	Both Reading and Writing Are allowed

Returns:
0 if successful, otherwise -1 is returned.
Example:

```
if (FileChmod("data.xls","READ") == -1)
        StatusMessage("Error changing file mode")
endif
```

FileClose

int FileClose(int fh)

Description:
Closes a file opened for reading or writing.
Returns:
0 if successful, -1 otherwise.
Example:
See FileOpen

FileCopy

bool FileCopy(string szSourceFilename, string szDestFilename)

Description:
Copies an existing source file to a pathname defined in the szDestFilename argument.
Returns:
TRUE if successful, FALSE otherwise.
Example:

```
szSource = "c:\command.com"
szDest = "d:\command.com"
FileCopy(szSource,szDest)    ; copies command.com to drive d:
```

FileDelete

int FileDelete(string szFile)

Description:
Removes the file named in the szFile parameter.
szFile contains the complete pathname of the file to be removed.
Wild cards are not supported.
Returns:
0 if successful, -1 otherwise.
Example:

```
if (FileDelete("c:\mail\messages.dat") == 0)
        StatusMessage("Mail Removed")
endif
```

FileDisplayAnsi

bool FileDisplayAnsi(string szPathname)

Description:
Displays a file containing ANSI escape sequences directly to the terminal screen. UNICOM switches to ANSI mode automatically then returns to the previous emulation when display is complete.
Returns:
TRUE if successful, FALSE otherwise.
Example:
FileDisplay("c:\pictures\bbs.scr")

FileEOF

int FileEOF(int handle)

Description:
FileEOF tests whether a specific file associate with a specified handle is at end-of-file (EOF),
Returns:
1 if at EOF, -1 if the handle is invalid, 0 otherwise
Example:

```
while (!FileEOF(fh))                    ; Display the file line by line to the status window
    szLine = FileRead(fh)
    StatusMessage(szLine)
endwhile
```

FileLength

long FileLength(string szPathname)

Description:
Computes the file size in bytes of the file specified in the szPathname argument.
Returns:
The file size if successful, -1 otherwise.
Example:
lSize = FileLength("messages.dat")

FileMkTemp

string FileMkTemp(string Pattern)

Description:

FileMkTemp returns a unique filename based on the specified pattern. Pattern is a character string that specifies the form of a new filename. It consists of two starting letters such as TM, followed by six upper case X's; for example TMXXXXXX.

The X's are replaced with alphanumeric characters to create a filename not currently in use.

Returns:

The a string containing the unique filename if successful, otherwise "function failed" is returned.

Example:

szUniqueFile = FileMkTemp("A3XXXXXX")

FileOpen

int FileOpen(string Pathname, string OpenMode)

Description:

This function opens a file specified by the Pathname for reading or writing according to the OpenMode parameter.

Valid OpenMode strings are:

"READ"	Open File for reading only
"WRITE"	Open File for writing only, Creates file if it does not already exist.

Returns:

The file handle if successful, -1 if an error occurred.

Example:

```
fh = FileOpen("output.dat","READ") ; open a file for reading
if (fh < 0)                        ; complain if it didn't work
        StatusMessage("File Open Error")
else
        szBuf = FileRead(fh)
        FileClose(fh)
endif
```

FileRead

string FileRead(int fh)

Description:
Reads a file associated with a passed file handle fh. The file is read until a newline character is reached '\n' or the number of characters read equals 255. Control characters are not copied.

Returns:
The string read from the file if successful, otherwise "*EOF*" is returned.

Example:

```
szLine = FileRead(fh)
while (szLine != "*EOF*")
        StatusMessage(szLine)
        szLine = FileRead(fh)
endwhile
FileClose(fh)
```

FileRename

bool FileRename(string szOldname, string szNewName)

Description:
The szOldname parameter specifies the file to be renamed.
The szNewName parameter specifies the desired pathname for naming.
FileRename does not allow you to move between physical devices.
Returns:
TRUE if successful, FALSE otherwise.
Example:
FileRename("messages.new","messages.old")

FileWrite

int FileWrite(int fh, any_type data)

Description:
This function writes any data type to a file associated with fh that is opened for writing using FileOpen. This function will convert the data passed to a string before writing.
Returns:
The actual number of characters written to the file or -1 on error.
Example:

```
i = (36/6) * 10
fh = FileOpen("Numbers.out","WRITE")
if (FileWrite(fh,i) != (-1))
    StatusMessage("Data Written Ok")
endif
FileClose(fh)
```

FindFile

bool FindFile(string pathname)

Description:
This function examines the specified pathname for an existing file.
Returns:
TRUE if the file is found, FALSE otherwise.
Example:

```
if (FindFile("process.sem"))
        Execute("script2.scr")
endif
```

FindWindow

handle FindWindow(string windowtitle)

Description:
This function will obtain the handle of a top level window with a window title matching
the string passed in the argument. This command is normally used to test for existence of other Window applications. You can obtain the handle of other window applications for the purpose of drawing directly on their window using UNICOM gdi calls. (Do so at your own risk)
Returns:
The window handle if successful, 0 if the window is not found.
Example:

```
if (hWnd = FindWindow("Clock"))     ; let's paint on the clock
   GdiSetWindow(hWnd);
   Gosub PaintASmile
endif
```

Find

bool Find(string source, string pattern)

Description:
	Find examines a source string for the occurrence of a pattern;
Returns:
	TRUE if found, FALSE otherwise.
Example:
```
	szGander = "There once was a goose named gander"
	if (Find(szGander,"goose"))
		Gosub CookGoose
	endif
```

FlushComm

bool FlushComm(string szQueue)

See CommFlush it is identical.

GdiArc

bool GdiArc(X1, Y1, X2, Y2, X3, Y3, X4, Y4)

Description

This function draws an elliptical arc. The center of the arc is the center of the bounding rectangle specified by the points (X1, Y1) and (X2, Y2). The arc starts at the point (X3, Y3) and ends at the point (X4, Y4). The arc is drawn using the selected pen and moving in a counterclockwise direction. Since an arc does not define a closed area, it is not filled.

Parameter	Type / Description
X1	int Specifies the logical x-coordinate of the upper-left corner of the bounding rectangle.
Y1	int Specifies the logical y-coordinate of the upper-left corner of the bounding rectangle.
X2	int Specifies the logical x-coordinate of the lower-right corner of the bounding rectangle.
Y2	int Specifies the logical y-coordinate of the lower-right corner of the bounding rectangle.
X3	int Specifies the logical x-coordinate of the arc's starting point. This point does not have to lie exactly on the arc.
Y3	int Specifies the logical y-coordinate of the arc's starting point. This point does not have to lie exactly on the arc.
X4	int Specifies the logical x-coordinate of the arc's endpoint. This point does not have to lie exactly on the arc.
Y4	int Specifies the logical y-coordinate of the arc's endpoint. This point does not have to lie exactly on the arc.

Return Value

The return value specifies whether the arc is drawn.
It is nonzero if the arc is drawn. Otherwise, it is zero.

Comments: Width and Height absolute values cannot exceed 32767

Example:

```
i = 0
while (i<300)
  GdiArc(10+i,100,300+i,200,11+i,111,300+i,199)
  i= i+1
endwhile
```

GdiBitBlt

bool GdiBitBlt(X, Y, nWidth, nHeight, XSrc, YSrc)

Description:

This function moves a bitmap from an area of the GdiWindow starting at XSrc and YSrc. The X, Y, nWidth and nHeight parameters specify the origin, width, and height of the rectangle on the screen that is to be filled by the bitmap.

Parameter	Type/Description
X	int Specifies the logical x-coordinate of the upper-left corner of the destination rectangle.
Y	int Specifies the logical y-coordinate of the upper-left corner of the destination rectangle.
nWidth	int Specifies the width (in logical units) of the destination rectangle and source bitmap.
nHeight	int Specifies the height (in logical units) of the destination rectangle and source bitmap.
XSrc	int Specifies the logical x-coordinate of the upper-left corner of the source bitmap.
YSrc	int Specifies the logical y-coordinate of the upper-left corner of the source bitmap.

Return Value

The return value specifies whether the bitmap is drawn. It is nonzero if the bitmap is drawn. Otherwise, it is zero.

GdiCreateBrush

bool GdiCreateBrush(int red, int green, int blue)

Description:

This function creates a solid color brush according to the mix colors specified in the parameters. The color values can range from 0 (minimum color) to 255 (max). Any previously created brush is replaced. The new brush will be used for any Gdi drawing calls that utilize a brush.

Returns:

TRUE if brush is created, FALSE otherwise (old brush will be retained).

Example:

```
GdiCreateBrush(255,0,0)          ;create a red brush
GdiFloodFill(20,20,0,0,0)        ;fill with new brush
```

GdiCreatePen

bool GdiCreatePen(int nStyle, int nWidth, int red, int green, int blue)

Description:

This function will create a pen to be used for Gdi Drawing functions.

Valid Pen styles are:

0	Solid
1	Dash
2	Dash Dot
3	Dash Dot Dot
4	Null
5	Inside Frame

The pen color is mix from the red, green and blue parameters.
A zero parameter is no color while 255 is maximum color.
For white, rgb values would be 255,255,255

Returns:

TRUE if pen is created, FALSE otherwise

Example:

```
GdiCreatePen(0,1,0,255,0)   ; create green pen
GdiMoveTo(0,0)
GdiLineTo(200,200)          ; draws a green line
```

GdiEllipse

bool GdiEllipse(X1, Y1, X2, Y2)

Description:

This function draws an ellipse. The center of the ellipse is the center of the bounding rectangle specified by the X1, Y1, X2, and Y2 parameters. The ellipse border is drawn with the current pen, the interior is filled with the current brush.

If the bounding rectangle is empty, nothing is drawn.

Parameter	Type/Description
X1	int Specifies the logical x-coordinate of the upper-left corner of the bounding rectangle.
Y1	int Specifies the logical y-coordinate of the upper-left corner of the bounding rectangle.
X2	int Specifies the logical x-coordinate of the lower-right corner of the bounding rectangle.
Y2	int Specifies the logical y-coordinate of the lower-right corner of the bounding rectangle.

Returns

TRUE if the ellipse is drawn. FALSE otherwise.

Example

```
GdiEllipse(10,20,200,200)    ; draws an ellipse in a rectangle 190x180
```

GdiFloodFill

bool GdiFloodFill(X, Y, red,green,blue)

Description

This function fills an area of the window with the current brush. The area is assumed to be bounded by the color defined by the red, green and blue parameters.

The FloodFill function begins at the point specified by the X and Y parameters and continues in all directions to the color boundary.

Parameter	Type/Description
X	int Specifies the logical x-coordinate of the point where filling begins.
Y	int Specifies the logical y-coordinate of the point where filling begins.
red	int Red component of the color boundary
green	int Green component of the color boundary
blue	int Blue component of the color boundary

Returns

TRUE if the function is successful.
FALSE if the filling could not be completed, the given point has the boundary color specified by red-green-blue, or the point is outside the clipping region.

Example

```
GdiCreateBrush(0,0,255)      ; create blue brush
GdiFloodFill(10,10,0,0,0)    ; fill to black borders
```

GdiGetTextHeight

int GdiGetTextHeight(string text)

Description:
This function computes the string text height using the currently selected font.
Returns:
The string height in pixels. Zero is returned if an error occurred.
Example:

```
nHeight = GdiGetTextHeight("X")
```

GdiGetTextWidth

int GdiGetTextWidth(string text)

Description:
This function computes the string text width using the currently selected font.
Returns:
The string width in pixels. Zero is returned if an error occurred.
Example:
nWidth = GdiGetTextWidth("ABCDEFG")

GdiLineTo

bool GdiLineTo(X, Y)

Description

This function draws a line from the current position up to, but not including, the point specified by the X and Y parameters. The line is drawn with the selected pen. If no error occurs, the position is set to (X,Y).

Parameter	Type/Description
X	int Specifies the logical x-coordinate of the endpoint for the line.
Y	int Specifies the logical y-coordinate of the endpoint for the line.

Return Value

It is TRUE if the line is drawn. Otherwise, it is FALSE.

Example

```
i = 0
while (i<640)
        GdiMoveTo(0,0)
        GdiLineTo(i,300)
        i = i +2
endwhile
```

GdiMoveTo

bool GdiMoveTo(X, Y)

Description:
This function moves the current position to the point specified by the X and Y parameters.

Parameter	Type	Description
X	int	Specifies the logical x-coordinate of the new position.
Y	int	Specifies the logical y-coordinate of the new position.

This function has no output, but it affects other Gdi calls that use the current position.
Returns:
TRUE if successful, FALSE otherwise.
Example:

```
GdiMoveTo(100,100)
GdiLineTo(200,400)          ; draws a line from 100,100 to 200,400
```

GdiRectangle

bool GdiRectangle(X1, Y1, X2, Y2)

Description

This function draws a rectangle. The interior of the rectangle is filled by using the selected brush, and a border is drawn with the selected pen.

Parameter	Type/Description
X1	int Specifies the logical x-coordinate of the upper-left corner of the rectangle.
Y1	int Specifies the logical y-coordinate of the upper-left corner of the rectangle.
X2	int Specifies the logical x-coordinate of the lower-right corner of the rectangle.
Y2	int Specifies the logical y-coordinate of the lower-right corner of the rectangle.

Returns

TRUE if the rectangle is drawn. Otherwise, it is FALSE.

The current position is neither used or updated by this function.

Example:

Rectangle(0,0,200,200)

GdiRoundRect

bool GdiRoundRect(X1, Y1, X2, Y2, X3, Y3)

Description:

This function draws a rectangle with rounded corners. The interior of the rectangle is filled by using the selected brush, and a border is drawn with the selected pen.

Parameter	Type/Description
X1	int Specifies the logical x-coordinate of the upper-left corner of the rectangle.
Y1	int Specifies the logical y-coordinate of the upper-left corner of the rectangle.
X2	int Specifies the logical x-coordinate of the lower-right corner of the rectangle.
Y2	int Specifies the logical y-coordinate of the lower-right corner of the rectangle.
X3	int Specifies the width of the ellipse used to draw the rounded corners.
Y3	int Specifies the height of the ellipse used to draw the rounded corners.

Returns

TRUE if the rectangle is drawn, FALSE otherwise.

The current position is neither used nor updated by this function.

Example

GdiRoundRect(10,100,300,300,30,20)

GdiSelectFont

bool GdiSelectFont(string facename, int desiredwidth, int desiredheight)

Description:

This function to selects the default font to be used for GdiTextOut calls. The facename parameter describes the font family. This function will enumerate all sizes and select the best match to the desiredwidth and desiredheight parameters.

Any previously selected font is replaced.

Returns:

TRUE if successful, FALSE otherwise.

Example:

```
GdiSelectFont("Helv",8,12)   ; Sets the font to Helv but size is approximate
```

GdiSetBkColor

bool GdiSetBkColor(int red, int green, int blue)

Description:
This function sets the current background color to a color generated by the red, green and blue parameters or to the nearest physical color.

If the background mode is OPAQUE, the background color is used to fill the gaps between styled lines, gaps between hatched lines in brushes, and character cells. The background color is also used when converting bitmaps from color to monochrome and vice versa.

Returns:
TRUE if successful, FALSE otherwise

Example:

```
GdiSetBkColor(0,255,255)    ; sets background color to cyan
```

GdiSetBkMode

bool GdiSetBkMode(string bkmode)

Description:
This function sets the background mode used with text and line styles. The background mode defines whether or not existing background colors on the device surface will be removed before drawing text, hatched brushes, or any pen style that is not a solid line.

Parameter	Type/Description
bkmode	string Specifies the background mode. It can be either one of the following modes:

Value	Meaning
"OPAQUE"	Background is filled with the current background color before the text, hatched brush, or pen is drawn.
"TRANSPARENT"	Background remains untouched.

Returns
TRUE if successful, FALSE otherwise.
Example:

```
GdiSetBkMode("TRANSPARENT")
GdiTextOut(10,10,"Hello")     ; Background is untouched by Hello
```

GdiSetMapMode

bool GdiSetMapMode(string mapmode)

Description:

This function sets the mapping mode of the specified device context. The mapping mode defines the unit of measure used to transform logical units into device units, and also defines the orientation of the device's x- and y-axes. The mapping mode is used to convert logical coordinates into device coordinates.

Value	Meaning
"ANISOTROPIC"	Logical units are mapped to arbitrary units with arbitrarily scaled axes. The GdiSetWindowExt and GdiSetViewportExt functions must be used to specify the desired units, orientation, and scaling.
"HIENGLISH"	Each logical unit is mapped to 0.001 inch. Positive x is to the right; positive y is up.
"HIMETRIC"	Each logical unit is mapped to 0.01 millimeter. Positive x is to the right; positive y is up.
"ISOTROPIC"	Logical units are mapped to arbitrary units with equally scaled axes; that is, one unit along the x-axis is equal to one unit along the y-axis. The GdiSetWindowExt and GdiSetViewportExt functions must be used to specify the desired units and the orientation of the axes.
"LOENGLISH"	Each logical unit is mapped to 0.01 inch. Positive x is to the right; positive y is up.
"LOMETRIC"	Each logical unit is mapped to 0.1 millimeter. Positive x is to the right; positive y is up.
"TEXT"	Each logical unit is mapped to one device pixel. Positive x is to the right; positive y is down.
"TWIPS"	Each logical unit is mapped to one twentieth of a printer's point (1/1440 inch). Positive x is to the right; positive y is up.

Returns

TRUE if successful, FALSE otherwise.

Example:

```
GdiSetMapMode("TEXT")
```

GdiSetPixel

bool GdiSetPixel(int x, int y, int red, int green , int blue)

Description:

This function sets a point on the screen to the closest color approximated by the red, green and blue parameters.

The current drawing position is set to x, y after calling this function.

Returns:

TRUE if the function is successful.

Example:

```
i = 0
while (i<640)
        GdiSetPixel( i, 200)    ; draws a horizontal line across the screen
        i = i+1
endwhile
```

GdiSetTextColor

bool GdiSetTextColor(int red, int green, int blue)

Description:
This function sets the color of the text for GdiTextOut to the nearest approximation of the color mix of the red, green and blue parameters.
Returns:
TRUE if successful, FALSE otherwise.
Example:

```
GdiSetTextColor(255,0,0)
GdiTextOut(10,30,"These characters are RED")
```

GdiSetViewportExt

bool GdiSetViewportExt(X, Y)

Description

This function sets the x- and y-extents of the viewport in device units.
The logical window wX and wY extents are mapped to physical x,y device units.

When the following mapping modes are set, calls to the GdiSetWindowExt and GdiSetViewportExt functions are ignored:

HIENGLISH
HIMETRIC
LOENGLISH
LOMETRIC
TEXT
TWIPS

Returns:

TRUE if successful, FALSE otherwise

Example:

SetViewPortExt(2,1)

GdiSetViewPortOrg

bool GdiSetViewPortOrg(int x,int y)

Description:
This functions sets the Viewport origin to the position specified in the x and y parameters.(in device units). This moves the logical origin (0,0) to device coordinates x,y.

Returns:
TRUE if successful, FALSE otherwise.

Example:

```
szWindow = "UNICOM.3.0"                    ; sets logical origin specified device pts.
xClient = WinGetWindowWidth(FindWindow(szWindow));
yClient = WinGetWindowHeight(FindWindow(szWindow));
GdiSetViewPortOrg(xClient/2, yClient /2)    ; sets logical origin to middle
```

GdiSetWindowExt

bool GdiSetWindowExt(int x, int y)

Description:
This function determine how may logical x and y units map to physical device units established by the GdiSetViewPort function.
Returns:
TRUE if successful, FALSE otherwise
Example:

```
GdiSetWindowExt(2,1)                ; map 2 logical x units to 1 device unit
```

GdiSetWindowOrg

bool GdiSetWindowOrg(int x, int y)

Description:
This function allows you to move the logical coordinate axis. After this call, logical point (x,y) will be mapped to device point (0,0).
Returns:
TRUE if successful, FALSE otherwise
Example:
GdiSetWindowOrg(-nWindowWidth/2,-nWindowHeight/2) ; moves logical 0,0 to middle

GdiSetWindow

bool GdiSetWindow(handle hWnd)

Description:

This functions designates the window that is to receive graphical output from UNICOM Gdi function calls. The default window is the UNICOM screen. **This call lets you draw anywhere on any window**.

Use WinGetDesktop to draw on your Window background.
Use FindWindow to obtain the handle of other windows.

Returns:

TRUE if successful, FALSE otherwise.
If this function fails, output will default back to the UNICOM screen.

Example:

```
GdiSetWindow(WinGetDesktop())
GdiMoveTo(0,0)
GdiLineTo(600,300)          ; draws a line on the Desktop window
```

GdiTextOut

int GdiTextOut(int X, int Y, string text)

Description

This function writes a character string at position X,Y using the currently selected font. X,Y is the upper left position of the rectangle enclosing the string.

By default, the current position is not used or updated by this function.

Returns:

The number of characters displayed., Zero if an error occurred

Example:

```
GdiSelectFont("Roman",6,8);
GdiSetTextColor(0,255,0);
ndisplayed = GdiTextOut(10,100,"There are green roman characters")
```

GetCwd

string GetCwd(void)

Description:
This function returns the pathname of the current working directory.
Returns:
Pathname if successful, returns "function failed" on error.
Example:
szDirectory = GetCwd()

GetDateTime

string GetDateTime(void)

Description:
Returns:
This function returns a string containing the current date and time in the following format:

DAYOFWEEK MONTH DAYOFMONTH HH:MM:SS YYYY

Example

```
szDateTime = GetDateTime()
DisplayString(10,10,szDateTime)
```

The above code displays todays date at row 10, column 10 : Tue Jul 27 19:26:44 1991

GetDiskFree

long GetDiskFree()

Description:
Returns:
This function returns the number of bytes available on the currently selected drive.
-1 is returned if an error has occurred.
Example:
lBytesRemaining = GetDiskFree()

GetEnvironment

string GetEnvironment(string environmentvariable)

Description:
Returns:
Returns the string value associated with the environment variable associated with the passed string. On error , 'function failed' is returned.

Example:

```
szPath = GetEnvironment("PATH")   ; retrieve your current DOS Path
```

GetFile

bool GetFile(string protocol [,string filename])

Description:

This function initiates a file download with the specified protocol. The filename parameter is necessary for XMODEM type and ASCII protocols.

Protocol Values	Filename Required ?
"XMODEMCS"	yes
"XMODEMCRC"	yes
"XMODEM1K"	yes
"YMODEMBAT"	no
"YMODEMG"	no
"ZMODEM"	no
"B"	no
"QUICKB"	no
"ASCII"	yes

Returns

TRUE if the transfer was initiated, FALSE otherwise.
The outcome of the download can be obtained using the IF command with the SUCCESS conditional flag.

Example:

```
InputString(szFile,"Enter the download filename")
GetFile("XMODEMCRC",szFile)                    ; start the download
if (SUCCESS)
        StatusMessage("Download Completed Ok")
else
        StatusMessage("Download Failed")
endif
```

GOSUB

GOSUB label

Description:

This statement transfers execution to a program segment identified by 'label'. A label is any alphabetic name prefixed with a colon ':' that is positioned within the script file (typically in the 1st column).

Control executes until a RETURN statement is encountered, after which, execution resumes at the statement following the Gosub call.

Gosub calls may be nested no more than 10 deep.

Returns:

nothing

Example:

```
Gosub NYSE
Exit

:NYSE
DisplayString(10,1,"Hello")
return
```

GOTO

GOTO label

Description:
	This statement unconditionally transfers execution to a label specified in the argument.
	If you use this statement to jump out of While loops and Switches while deeply nested,
	additional nesting may be limited.
Returns:
	Nothing
Example:
	GoTo Done
:Done
	Exit

Hangup

bool Hangup(void)

Description:

Causes UNICOM to initiate a modem hangup operation. DTR (pin 20) is dropped to the modem. If data carrier detect transitions or the modem produces the defined response to DTR drop, the hardware drop is considered successful.

If none of the above responses are encountered, UNICOM instructs the modem to initiate a software hangup. The modem attention sequence is transmitted "+++" to place the modem in command mode. If an OK results, then ATH0 CR is transmitted to the modem. If an OK results again, the software hangup is considered successful

Returns:

TRUE if hardware or software hangup procedures succeeded.
FALSE if both procedures fail to produce the expected responses

Example:

```
if (Hangup())
        Exit
endif
```

Host

void Host()

Description:

Sets UNICOM into Host Mode operation. Script Execution is terminated at this call.

This command is useful for placing UNICOM into host mode at program start with the use of an auto start script file.

Returns:

Nothing

Example:

```
HOST
EXIT
```

IF

bool IF (expression)

Description:

The IF - ELSE statement can be used to make program decisions.

```
Syntax:         IF (expression)          ; statements must start on next line
                statement 1    ; if expression is TRUE, these
                statement 2    ; statements are executed
                statement N
            ELSE                ; This Else section is optional
                elsestatement1      ; if expression is FALSE these
                elsestatement2      ; these statements are executed
                elsestatementN
            ENDIF               ; This ENDIF is required to terminate an IF
```

An expression can be any arithmetic expression, function return value, variable,or program flag.

Program flags operate as documented in UNICOM 2.0 and are provided for compatibility.

They are: SUCCESS, CONNECTED, FOUND, and WAITFOR. The following flags indicate a response to a Message command : OK, CANCEL, ABORT, RETRY,IGNORE, YES,NO. In UNICOM 3.0, the SUCCESS flag is used to determine the result of a data transfer.

Returns:

TRUE if the expression evaluates TRUE, FALSE otherwise.

Example:

```
IF (x != 33)
    DisplayString(1,1,"X does not equal 33")
ENDIF
```

InputString

string InputString(string variablename, string windowtitle)

Description:

InputString is used to accept input from a user. A window containing an edit box is displayed for the user to enter text. An optional window title can be used to prompt for input.

When the user completes the text entry by pressing return. The text entered by the user is assigned to a string variable provided in the call. If the variable of the name specified does not exist, it is created.

The text input has a limit of 255 characters.

Returns:

A user entered string is returned from the call. If an error occurred, the message " function failed' is returned.

Example:

InputString(szUserInput,"Please Enter Your Name")

InpW

word InpW(int portid)

Description:
InpW is used to read an unsigned integer from a cpu port location passed in the argument.

portid is a valid cpu port in the range of 0 to 65535
Returns:
A word from the addressed port.
Example:
wSpeakerValue = inpW(97)

Inp

word inp(int portid)

Description:
inp is used to obtain a byte from the specified computer port.
Returns:
The result is stored in a word with the hi-order byte zeroed.
Example:
wSpeakerValue = inp(97)

Itoa

string ItoA(int nValue, string result, int nRadix)

Description:

This function converts an integer value to its character string representation.

nValue is the integer to be converted.
result is a variable to receive the string representation.
nRadix specifies the base value to be used for the conversion.

Returns:

The string representation if successful. Returns 'function failed' on error.

Example:

```
nInt = 4096
ItoA(nInt,szHex,16)
DisplayString(1,1,szHex)
```

The above code displays '1000' at row 1 column 1

KillWindow

bool KillWindow(string windowname)

Description:

This function will destroy any top level window that has a title matching the windowname passed in the argument call. This function sends a WM_DESTROY message to the window.

To guarantee that the window was destroyed, a FindWindow command must be used after the call to KillWindow.

Returns:

TRUE if the window was found. FALSE otherwise.

Example:

```
if (KillWindow("Clock"))
        StatusMessage("Time to get a new Clock")
endif
```

LoadKeys

bool LoadKeys(string szKeyFile)

Description:

This function loads the keyboard keys with the definitions stored in the specified file. The user screen buttons are updated immediately. If the terminal override checkbox is selected in the Keyboard Macro Editor, keys F1-F4 and the arrow keys will be reserved for use if VT102 or VT52 is the current emulation.

The key file is assumed to reside in the UNICOM files directory.

Returns:

TRUE if successful, FALSE otherwise.

Example:

LoadKeys("alternate.key")

LoadTranslationFile

bool LoadTranslationFile(string szTranslationFile)

Description:
Any file containing UNICOM character translation tables can be loaded into memory using this function. After loading, character translation takes effect immediately.
Returns:
TRUE if successful, FALSE otherwise.
Example:
LoadTranslationFile("SWEDEN.XLT") ; This function was suggested by my friends in Sweden

Locate

bool Locate(int row, int col)

Description:
Locate positions the terminal cursor to the row and column specified in the arguments.
Returns:
TRUE is successful, FALSE if an invalid number of arguments is encountered.
Example:
Locate(12,40) ; moves the cursor center screen

Log

bool Log(option[,filename])

Description:

File capture of incoming text is controlled using this function during script execution. The following options are supported:

OPEN -	Opens a file, creates one if necessary. Logging begins.
TRUNC -	Opens a file, truncates if it exists. Logging begins.
CLOSE -	Closes a log file.
SUSPEND -	Pauses file logging until a close or resume log command is given.
RESUME -	Resumes logging after a log suspend command.

The OPEN and TRUNC options require a filename.

Returns:

TRUE if the operation succeeded, FALSE otherwise.

Example:

```
Log("OPEN","myquotes.dat")
```

MakePath

string MakePath(string result, string drive, string dir, string filename, string ext)

Description:
MakePath builds a complete pathname from the drive, dir, filename and ext string arguments. The result is stored in the string variable in the first argument. The resulting string is also returned from the call.

Returns:
The complete pathname if successful. Returns "function failed" if an error occurred.

Example:
MakePath(szResult, "c:","\\windows","win","ini")

Max

int Max(int nValue1, int nValue2)

Description:
Returns:
Max compares two integer values and returns the largest.
Example:
nMax = Max(22,333)

Message

int Message(string szMessage [,string title[,string btnoptions[,string iconoptions]]])

Description:

Message will display a message in a Windows message box. An optional title can be displayed in the window caption area. Optional button styles and icons may be selected.

Button Options	Icon Options
OK	ICONHAND
ABORTRETRYIGNORE	ICONQUESTION
OKCANCEL	ICONEXCLAMATION
YESNO	ICONASTERISK
YESNOCANCEL	
RETRYCANCEL	

Response Codes	User Button Selected
1	OK
2	Cancel
3	Abort
4	Retry
5	Ignore
6	Yes
7	No

If optional parameters are omitted, the button style defaults to OK. The icon style defaults to no icon.

Returns:

An integer response code corresponding to the button selected by the user. For 2.0 compatibitly the appropriate program flag is set:
OK, CANCEL, ABORT, RETRY, IGNORE, YES, NO

Example:

```
nResponse = Message("Are you sure?","YESNO","ICONHAND")
if (nResponse == 6)
        StatusMessage("yep, they're sure")
endif
```

Min

int Min(int nValue1, int nValue2)

Description:
Min compares the to integer arguments and returns the smaller of the two values.
Returns:
The smaller value.
Example:

```
szNumber = "3456"
nSmaller = min(3457,atoi(szNumber))
```

Mkdir

int Mkdir(string pathname)

Description:
Mkdir creates the a subdirectory named in the pathname.
Returns:
Zero if successful, -1 if an error occurred.
Example:

```
if (!Mkdir("mydir"))
        StatusMessage("Directory Creation Succeeded")
endif
```

ModemCmdMode

bool ModemCmdMode(void)

Description:

This function causes the modem escape sequence "+++" to be transmitted to the modem in an attempt to obtain an OK response. If an OK is received, this function returns TRUE.

This function is best used during an active connection so the modem can be issued a command while online. Some modems are configured to hang up when the escape sequence is received. Check with your modem reference.

Returns:

TRUE if OK was received, FALSE otherwise.

Example:

```
if (ModemCmdMode())         ; get into command mode
  Send("ATS0=1^M")          ; turn auto answer mode on
  Delay(1000);              ; give modem a change to digest
  Send("ATO^M")     ; return online
endif
```

ModemDial

bool ModemDial(string phonenumber)

Description:
This function initiates a modem dialing sequence using the phone number passed in the argument.
Returns:
TRUE if the modem connected, FALSE otherwise.
For 2.0 compatibility, the CONNECTED flag affected.
Example:

```
if (ModemDial("123-456-7890"))
        Execute("logon.scr")
endif
```

ModemGetAttention

Description:

This command is useful to check if the modem is in command mode and listening. An AT^M is transmitted to the modem. If the modem responds with an OK, this function returns TRUE.

Returns:

TRUE if the modem is awake and in command mode, FALSE otherwise.

Example:

```
if (ModemGetAttention())
        ModemDial("345-6789")
endif
```

ModemGetRegister

int ModemGetRegister(int register)

Description:

This function queries the modem for the value of a specific modem register. The desired register number is contained in the argument.

Consult your modem reference manual for information about modem registers.

Returns:

The contents of the specified register. On error, Zero is returned.

Example:

```
if (ModemGetRegister(0) > 0)
        StatusMessage("Auto Answer is On")
endif
```

ModemHangUp

See HangUp

ModemInit

bool ModemInit(string initstring)

Description:
This function configures your modem with the init string provided in the argument.
Returns:
TRUE if successful, FALSE otherwise.
Example:

```
if (ModemInit("ATV1"))
        StatusMessage("The modem is set to verbose")
endif
```

ModemReset

bool ModemReset(void)

Description:
This function transmits a reset command "ATZ" to the modem and waits for an OK response.
Returns:
TRUE if an OK response was obtained, FALSE otherwise.
Example:
ModemReset()

ModemSetRegister

bool ModemSetRegister(int register, int value)

Description:
This function will set any modem register to a specific value.
The modem is assumed to be in command mode before the call.
Returns:
TRUE if successful, FALSE otherwise.
Example:

```
InputString(szRings,"Enter number of rings for autoanswer")
if (ModemSetRegister(0,atoi(szRings)))
        StatusMessage("Modem Set to Answer")
endif
```

MoveWindow

bool MoveWindow(string title, int x, int y, int width, int height)

Description:

This function will move and size any top level window that be identified by it's window title.

Parameter	Meaning
x	x coordinate where upper left window corner will be placed
y	y coordinate where upper left window corder will be placed
width	Window will be resized to width pixels.
height	Window height will be resized to height pixels.

Returns:

TRUE if successful, FALSE otherwise.

Example:

```
MoveWindow("UNICOM3.0",0,0,100,200)    ; moves to upper left 100x200
```

OutpW

bool OutpW(int port, int value)

Description:
OutpW outputs an integer value to the port specified in the argument.
Returns:
TRUE if successful, FALSE otherwise.
Example:
OutpW(97,31000)

Outp

bool Outp(int port, int value)

Description:
Outp will output the lo-order byte of the value parameter to the specified port.
Returns:
TRUE if successful, FALSE otherwise.
Example:
Outp(97,254) ;

Port

bool Port(string configstring)

Description:

Selects and configures a communication port. The configstring must obey the following format:

"device;baud;parity;wordsize;stopbits;duplex"

Field	**Supported Values**
device	NONE,COM1, COM2, COM3, COM4
baud	300,1200,2400,4800,9600,19200,38400
parity	N,E,O
wordsize	7,8
stopbits	1,2
duplex	FULL, HALF

All fields are required and must be separated by semicolons with no spaces.

Returns:

TRUE if successful, FALSE otherwise.

Example:

Port("COM4;2400;N;8;1;FULL")

Quit

bool Quit()

Description:
This function will terminate script execution and exit UNICOM with no questions asked.
Returns:
TRUE always.
Example:
QUIT()

RETURN

See GOSUB

Rmdir

int Rmdir(string pathname)

Description:
Rmdir will remove an empty subdirectory identified by the pathname parameter.
Returns:
Zero if successful, -1 if the directory could not be found or deleted.
Example:
Rmdir("temp")

Rotl

word Rotl(word value, int shift_value)

Description:
Rotl rotates the bits of a word value to the left the specified number of bits.
Returns:
The resulting word is returned. Zero is returned on error.
Example:

```
i = 1
while (i < 15)
        DisplayString(10,10,Rotl(1,i))
        i = i+1
endwhile
```

Rotr

word Rotr(word value, int shift_value)

Description
Rotr rotates the bits of a word value to the right the specified number of bits.
Returns:
The resulting word is returned. Zero is returned on error.

Example:

```
i = 1
while (i < 15)
        DisplayString(10,10,Rotr(1,i))
        i = i+1
endwhile
```

Run

bool Run(string programname[,string param1 [,string param N]])

Description:
Run will execute a Windows or DOS application after first constructing a command line from a number of argument strings. The combined argument string length is limited to 80 characters. If parameters are not accepted by the activated program, incorporate the program and parameters into one run command string. (for example: run ("notepad.exe mytext.doc"))

Returns:
TRUE if the application started, FALSE otherwise.

Example:
Run ("UNICOM.exe","startup.scr")
or Run ("UNICOM.exe startup.scr")

SendFile

bool SendFile(string protocol, string file1 [,file2[,file N]])

Description:

Initiates a file upload using the specified protocol.

Valid Protocols are:

	Multiple File Arguments
"XMODEMCS"	no
"XMODEMCRC"	no
"XMODEM1K"	no
"YMODEMBAT"	yes
"YMODEMG"	yes
"ZMODEM"	yes
"ASCII"	no
"B"	no
"QUICKB"	no

Returns:

TRUE if the transfer was initiated, FALSE otherwise.
The SUCCESS flag must be tested to determine the outcome of the transfer.

Example:

```
SendFile("ZMODEM","file1.ext","file2.ext")

if (SUCCESS)        StatusMessage("File Upload Complete")
endif
```

This function is available only to registered users.

Send

bool Send(string text)

Description:

Send will transmit text to the remote host computer. Control characters may be denoted in the string with a ^ prefix.

Control characters are unprintable characters in the ASCII range of 0 to 31.

Use the ^ char to prefix visible ASCII characters in the range of 64 to 95. (Refer to an ASCII table to locate the specific characters.) Prefixing characters in this range, will produce the corresponding unprintable character in the range of 0 to 31.

To send a delete character (255) use '^?'

Returns:

TRUE if successful, FALSE otherwise.

Example:

```
Send ("777777,8888^M")     ; sends the string with a trailing carriage return
```

ShowWindow

bool ShowWindow(string windowtitle, string showoption)

Description:

ShowWindow is used to hide or display any top level window that can be identified by its window title. Valid show options are:

"HIDE"	Iconize the window and activate another
"ZOOM"	Zoom the window to full screen
"MINIMIZE"	Display the window in iconic form
"SHOW"	(default) Activates and displays a window. Restores hidden or iconized windows to their original size and position.

Returns:

TRUE if successful, FALSE otherwise.

Example:

ShowWindow("UNICOM 3.0", "ZOOM")

Snapshot

bool SnapShot(void)

Description:
Sends the contents of UNICOM terminal rows 1 - 24 to the print manager for printing.
Returns:
TRUE always
Example:

```
if (Message("Want your picture taken") == 1)
        SnapShot()
endif
```

SoundPlayNote

bool SoundPlayNote(int nValue, int nLength, int nCdots)

Description:

Plays a note on the sound output device.

Parameter	Meaning
nValue	Specifies 1 of 84 possible notes (seven octaves). If nValue is zero, a rest is assumed.
nLength	Specifies the reciprocal of the duration of the note. For example, 1 specifies a whole note, 2 a half note, 4 a quarter note.
nCdots	Specifies the duration of the note in dots. The duration is equal to nLength x (nCdots x 3/2). If in doubt, use 0.

Returns

TRUE if successful, FALSE otherwise.

A TRUE result does not mean that a sound was produced. Let your ear determine the true return value.

NOTE: A Microsoft 3.0 bug exists in the sound driver and may reduce the duration of the note for whole and half notes.

Example

```
SoundPlayNote(40,4,0)          ; play a quarter note
```

SplitPath

bool SplitPath(string pathname, string drive, string subdir, string file, string ext)

Description:
SplitPath breaks down a complete pathname into a drive letter, subdirectory, filename and extension. The variables are created if they do not already exist.
Returns:
TRUE if successful. FALSE if not.
Example:
SplitPath("c:\\windows\\win.ini",szdrive,szsubdir,szfile,szext)

this function:
stores c: into szdrive
stores \windows\ into szsubdir
stores win into szfile
stores ini into szext

StatusMessage

bool StatusMessage(string szmessage)

Description:
Displays a string message on the UNICOM status line for 4 seconds.
Returns:
TRUE if successful, otherwise FALSE.
Example:
StatusMessage("This is a test message")

StrCat

string StrCat(string target, string source)

Description:

StrCat appends the source string to the target string.
If target is not a variable name, the source is appended to the target string and the result is returned by the call. If the target is a variable, the source is concatenated to the contents of the variable.

Returns:

The concatenated target string if successful. Returns 'failed function' on error.

Example:

szResponse = StrCat("Hello"," There")
szResponse now holds "Hello There"

StrChecksum

int StrChecksum(string source)

Description:
This function computes the numeric checksum of the argument string.
Returns:
The checksum if successful, or 0 on error.
Example:
i = StrChecksum("STRINGTOTEST")

Strcmpi

int Strcmpi (string string1, string string2)

Description:
String compares two strings without regard to case.
Returns:
< 0 string1 is less than string2
= 0 string 1is identical to string2
>0 string1 is greater than string2
Example:

```
if (!Strcmpi("unicom","UNICOM"))
        StatusMessage("Strings Match);
endif
```

The above code will display the message.

Strcmp

Description:
String compares two strings with regard to case.
Returns:
< 0 string1 is less than string2
= 0 string 1is identical to string2
>0 string1 is greater than string2

Example:

```
if (Strcmp("unicom","UNICOM"))
        StatusMessage("Strings don't Match");
endif
```

Strcpy

string Strcpy(string target, string source)

Description:
Strcpy copies the contents of one character string to another.

If the target string is not a variable name, the string value
will be used as the name by which to create the variable.

The source string will be stored in the target variable string.

Returns:
The source string if successful. Returns "function failed" on error.

Example:
```
Strcpy(szNewVar,"ABCDEFG")          ; Initializes a new string var named szNewVar
```

StrCRC16

word StrCRC16(string source)

Description:
This function computes the 16bit crc of the argument string.
Returns:
The 16 bit CRC if successful, Zero if the function fails.
Example:
wCrc = StrCRC16(szbuffer)

StrCtlToPrefix

string StrCtlToPrefix(string source)

Description:
Converts a string containing unprintable control characters to a string containing that represents control characters with a ^ prefix
^M represents ASCII 13, ^C represents ASCII 03
Returns:
The converted string if successful. 'function failed' is return on error.
Example:
szNoControls = StrCtlToPrefix(szControlString)

StrIsAlphaNum

bool StrIsAlphaNum(string source)

Description:
Returns:
Returns TRUE if the entire source argument string is in the range 'A'-'Z','a'-'z', or '0'-'9'
Otherwise, it is FALSE.
Example:

```
if (StrIsAlphaNum("ABCDEF0123456"))       ; expression is true
        x = 1                             ; executes this statement
else
        x = 0
endif
```

StrIsAlpha

bool StrIsAlpha(string source)

Description:
Returns:
Returns TRUE if the entire source argument string is in the range 'A'-'Z', 'a'-'z'.
Otherwise, it is FALSE.
Example:

```
if (StrIsAlpha("452"))     ; expression is false
        Exit;
endif
```

StrIsGraph

bool StrIsGraph(string source)

Description:
Returns:
StrIsGraph returns true if the entire string is in the printable ASCII range 33 to 126. The space character is excluded.
Example:

```
if (!StrIsGraph("      abcdefg"))
        StatusMessage("StrGraph returned false")
endif
; in the example above, the message will be displayed
```

StrIsHex

bool StrIsHex(string source)

Description:
Returns:
StrIsHex examines the source string for characters 'A'-'F', 'a'-'f', and '0'-'9'. if any other characters are present, this function returns FALSE, otherwise TRUE is returned.
Example:

```
InputString(szHex, "Enter a Hex Number")
if (!StrIsHex(szHex))
        StatusMessage("You did not enter a Hex number")
endif
```

StrIsLower

bool StrIsLower(string source)

Description:
Returns:
StrIsLower examines the source string and returns TRUE if the entire string is in the lowercase character range of 'a' - 'z'.
Example:

```
if (!StrIsLower("abcdefgH"))
        StatusMessage("StrIsLower is FALSE")        ; this line will be executed
endif
```

StrIsNumeric

bool StrIsNumeric(string source)

Description:
Returns:
StrIsNumeric examines the source string and returns TRUE if the entire string contains characters in the range of '0' - '9'.
Example:

```
i = 9432;
itoa (i,szNumber,10)
if (StrIsNumeric(szNumber))
        DisplayString("Number is valid")
endif
```

StrIsPunct

bool StrIsPunct(string source)

Description:
Returns:
StrIsPunct examines the source string and returns TRUE if the entire string contains punctuation characters such as a comma, semicolon or a period.
Example:

```
if (StrIsPunct("...."))
        StatusMessage("I see dots")
endif
```

StrIsSpace

bool StrIsSpace(string source)

Description:
Returns:

StrIsSpace examines the source string and returns TRUE if the entire string contains whitespace characters such as tabs, vertical tabs, linefeeds, formfeeds, carriage returns or spaces.

Example:

```
InputString(szUserInput,"Enter your phone number")
if (StrIsSpace(szUserInput))
        DisplayString("You did not enter anything")
endif
```

StrIsUpper

bool StrIsUpper(string source)

Description:
Returns:
StrIsUpper examines the source string and returns TRUE if the entire string is in the ASCII range of 'A' - 'Z'.
Example:

```
:getitright
        InputString(szPassword,"Enter your password in UPPERCASE")
        if (!StrIsUpper(szPassword))
                Goto getitright
        endif
```

StrLen

int StrLen(string source)

Description:
Returns:
Strlen returns the length of a character string. The terminating NULL in not counted.
Example:

```
DisplayString(10,10,strlen("123456"))          ; 6 is displayed at row 10, col 10
```

StrPrefixToCtl

Description:

StrPrefixToControl converts a character string containing control prefixes to a string containing the actual control characters.

Returns:

The converted string if successful. On error, 'function failed' is returned.

Example:

```
szCtlString = "^B^C^D^E"
szConverted = StrPrefixToCtl(szCtlString)
```

StrRev

string StrRev(string source)

Description:
Returns:
StrRev reverses the order of the characters within a string.
Example:

```
szString = "XYZ"
DisplayString(1,1,StrRev(szString))  ;ZYX is displayed at 1,1
```

StrStripCtl

string StrStripCtl(string source)

Description:
Returns:
StrStripCtl converts any control characters within the source string to blanks.
Example:

```
szString = "AB^CDEF^G"
szCtl = StrPrefixToCtl(szString)
szString = StrStripCtl(szCtl)  ;szString now holds "AB DEF "
```

Strstr

string StrStr(string source, string substring)

Description:
Returns:
StrStr returns a string pointing to the 1st occurrence of a substring within a source string. If the substring is not found, an empty string is returned.
Example:
szSubstring = StrStr("Program material","mat") ;szSubstring contains "material"

Strtok
string Strtok(string source, string delimiter)

Description:
Returns:
Strtok returns a string within the source string that is delimited by any character listed in the delimiter string. This function will return only the 1st occurrence. If the function is unsuccessful, an empty string is returned.
Example:

```
szToken = Strtok("Seattle, WA",",")          ;szToken contains "Seattle"
```

StrToLower

string StrToLower(string source)

Description:
StrToLower examines the source string and converts any uppercase characters to lowercase.
Returns:
The converted string. If an error occurs, 'function failed' is returned.
Example:

```
szString = "AbCdEfG"
szString = StrToLower(szString)          ;szString now holds "abcdefg"
```

StrToUpper

string StrToUpper(string source)

Description:
StrToUpper examines the source string and converts any lowercase characters to uppercase.
Returns:
The converted string. If an error occurs, 'function failed' is returned.
Example:

```
szString = "AbCdEfG"
szString = StrToUpper(szString)          ;szString now holds "ABCDEFG"
```

SWITCH

```
SWITCH (expression)
   CASE (expression 1)
   ;statements to execute if expression 1 matches
   ENDCASE
   CASE (expression 2)
   ;statements to execute if expression 2 matches
   ENDCASE
   CASE (expression N)
   ENDCASE
DEFAULT
   statements to execute if none of the above case expression match
ENDSWITCH
```

Description:

The SWITCH statement is decision maker that compares an expression to a number of CASE expressions. Control will pass the next line of a matching CASE expression.. If none of the CASE expressions match the SWITCH expression, control will pass to the next line following a DEFAULT line. If no DEFAULT is found , then control is passed to the line following an ENDSWITCH

Example:

```
nValue = 36/3

SWITCH ( nValue / 4)
        CASE 3
          StatusMessage("Control Will Pass Here")
        ENDCASE
        CASE 4
          StatusMessage("Control will not reach here")
        ENDCASE
ENDSWITCH
```

TitleWindow

bool TitleWindow(string oldtitle, string newtitle)

Description:
This function will locate a top level window with a caption matching the oldtitle parameter.
If a window is found, its caption is replaced with the string passed in the newtitle parameter.
Returns:
TRUE if successful, FALSE otherwise.
Example:
TitleWindow("Clock","Digital Clock")

Trace

bool Trace (string mode)

Description:

Trace is used to enable or disable viewing of script commands as they are executed. Valid modes are "on" or "off". Any unrecognized mode parameter will be interpreted as off.

Commands are viewed in the program status line as they are executed. Tracing adds a 3/4 second delay for each command so that it can be viewed.

Returns:

TRUE if tracing is enabled, FALSE otherwise.

Example:

```
Trace ("ON")   ;        Turns it on
Trace          ;        Turns it off
```

Transmit

See SEND it is identical

VarIsDefined

bool VarIsDefined(string name)

Description:
Returns:
This function returns TRUE if a variable exists by the name specified in the argument. If no variable exist by that name, FALSE is returned.
Example:

```
X = 1
if (VarIsDefined(X))
   X = X + 32
endif
```

WaitFor

bool Waitfor(string target[, int delayseconds])

Description:

Waitfor pauses script execution for the specified number of delay seconds while the port is scanned for an incoming string that matches the target. If the target string is found within the time specified, this function returns true. If the target string is not encountered during this time, Waitfor returns FALSE.

If the time parameter is omitted, then Waitfor will default to 30 seconds.

Returns:

TRUE if the target string was read from the port during the Waitfor time. FALSE otherwise.

Example:

```
if (Waitfor("Password",30))
        Send("secretcode^M")
endif
```

WHILE

```
WHILE (expression)
        statement 1
        statement 2
        statement N
ENDWHILE
```

Description:

The WHILE - ENDWHILE statements are used to define a block of program statements that will be executed while the WHILE expression evaluates TRUE.

An ENDWHILE is required to terminate each WHILE statement. Statements to be executed within the WHILE - ENDWHILE block, must not appear on the same line with the WHILE or ENDWHILE statements.

Returns:

Nothing

Example:

```
i = 0;
While (i < 640)
        LineTo(i,400)
        i = i + 1
EndWhile
```

WinEnableInput

bool WinEnableInput(int nEnableInput)

Description:

This function disables mouse and keyboard input. The input is saved if the nEnableInput parameter is 1 and discarded if it is 0.

Returns

The return value specifies whether mouse and keyboard input is disabled. It is TRUE if input was previously enabled. Otherwise, it is FALSE. The default return value is TRUE.

Example:

WinEnableInput(1)

WinExitWindows

bool WinExitWindows(void)

Description:
Initiates the standard Windows shutdown procedure. If all applications agree to terminate, control is returned to DOS.
Returns:
FALSE if at least one application refuses to terminate. No return is possible if all applications terminate.
Example:
WinExitWindows()

WinFlashWindow

bool WinFlashWindow(string windowtitle)

Description:
This function will flash the caption of the window identified by the windowtitle argument. Flashing changes the appearance of the caption from inactive to active and visa versa.
Returns:
TRUE if the window state was active before the call. FALSE if the window state was inactive before the call.
Example:
FlashWindow("Clock")

WinGetActiveWindow

handle WinGetActiveWindow(void)

Description:
Returns:
This function returns a handle to the Window that currently has the input focus.
Example:
hWnd = WinGetActiveWindow()

WinGetDesktop

handle WinGetDesktop(void)

Description:
Returns:
WinGetDesktop will return the window handle of your windows background.
Example:
See GdiSetWindow

WinGetFlags

word WinGetFlags(void)

Description:
Returns:

The function retrieves a word containing flags which specify the memory and cpu configuration on which Windows is running.

Bit Position	Meaning if Set
0x1	Windows is running in protected mode
0x2	The cpu is an 80286
0x4	The cpu is an 80386
0x8	The cpu is an 80486
0x10	Windows is running in standard mode.
0x20	Windows is running in 386 enhanced mode.
0x40	The cpu is an 8086
0x80	The cpu is an 80186
0x100	Windows is running in EMS largeframe
0x200	Windows is running in EMS smallframe.
0x400	A 80x87 Math co-processor is detected

Example:

```
if (WinGetFlags() & 1)
        StatusMessage("Windows is in protected mode")
endif
```

WinGetFreeSpace

Description:
Returns:
Returns the amount of free global heap memory above any EMS bank line. The return value is the number of K bytes available.
Example:

```
nBytesAvailable = WinGetFreeSpace()*1024
```

WinGetScrollPos

int WinGetScrollPos(string bar)

Description:
This function retrieves the current position of the UNICOM horizontal or vertical scroll bar. The bar parameter may contain "h" or "v".
Returns:
The current position of the bar specified.
Example:
nhPos = WinGetScrollPos("h")

WinGetScrollRange

int WinGetScrollRange(string bar)

Description:
Returns:
This function returns the maximum position of UNICOM's horizontal or vertical scroll bar. The minimum position is always 0. The bar parameter should specify "h" for the horizontal bar or "v" for the vertical bar.
Example:
nVertMax = WinGetScrollRange("v")

WinGetSystemDir

string WinGetSystemDir(void)

Description:
Returns:
This function obtains the complete pathname of the Windows system directory.
Example:
szSystemDir = WinGetSystemDir()

WinGetVersion

word WinGetVersion(void)

Description:
Returns:
WinGetVersion returns a word containing the current operating version of Windows. The low order byte contains the major version number. The hi-order byte contains the minor version.
Example

```
wVer = WinGetVersion()
itoa(wVer,szVersion,10)        ; convert to ascii
StrRev(szVersion)      ; place number in correct order.
```

WinGetWindowHeight

int WinGetWindowHeight(handle hWnd)

Description:
Returns:
This function obtains the height of client area of a window identified by its handle. A Zero is returned on error.
Example:
nDesktopHeight = WinGetWindowHeight(WinGetDesktop())

WinGetWindowsDir

string WinGetWindowsDir()

Description:
Returns:
The windows pathname is returned to the caller of this function.
Example:
DisplayString(1,1,GetWindowsDir())

WinGetWindowWidth

int WinGetWindowWidth(handle hWnd)
Description:
Returns:
This function obtains the width of client area of a window identified by its handle. A Zero is returned on error.
Example:
nDesktopWidth = WinGetWindowWidth(WinGetDesktop())

WinIsIconic

bool WinIsIconic(string windowtitle)

Description:
Returns:
Locates a top level window that has a caption title matching the windowtitle parameter. If the window is iconic (minimized) this function returns TRUE.
Example:

```
if (WinIsIconic("UNICOM 3.0"))
        ShowWindow("UNICOM 3.0","ZOOM")
endif
```

WinIsVisible

bool WinIsVisible(string windowtitle)

Description:
Returns:
This function locates a window with a caption matching the windowtitle parameter. If the Window has been made visible using the ShowWindow function this function returns TRUE even if it is obscured by another overlapping window.
Example:

```
if (WinIsVisible("UNICOM 3.0"))
        StatusMessage("UNICOM is on the screen")
endif
```

WinIsZoomed

bool WinIsZoomed(string windowtitle)

Description:
Returns:
WinIsZoomed locates a window containing a caption that matches the window title parameter. If a window is found, this function returns TRUE if it is zoomed to full screen.
Example:

```
if (WinIsZoomed("UNICOM 3.0"))
        ShowWindow("UNICOM 3.0","HIDE")
endif
```

WinReadIni

string WinReadIni(string application, string keyname)

Description:
Returns:
This function will retrieve key values stored in your win.ini file by application name.
If the application or key cannot be found, this function returns "function failed".
Example:
szBeepValue = WinReadIni("windows","beep")

WinSetScrollPos

int WinSetScrollPos(int newpos, string bar)

Description:
This function will scroll the UNICOM scroll buffer vertically or horizontally to a position defined in the newpos argument. The bar is selected in the second argument. Valid bar values are "h" for the horizontal bar and "v" for the vertical bar.

Returns:
The previous scroll position.

Example:

```
oldpos = WinSetScrollPos(1,"v")      ; scrolls the scroll buffer all the way back
WinSetScrollPos(oldpos,"v")  ; returns the bar to where it was
```

WinWriteIni

bool WinWriteIni(string application , string key, string value)

Description:
This function will store key values into your win.ini file by application name.
Returns:
TRUE if successful, FALSE otherwise.
Example:

```
if (WinWriteIni("windows","beep","no"))
        StatusMessage("No more window beeping")
endif
```