Omega Point/2 Bulletin Board System for OS/2

User Guide and Reference Manual

October 19, 1990

ExcelSoft Software
1100 Indian Trail Road #1725
Norcross, GA 30093

24 High Point Road
Lincroft, NJ 07738

BIX: chrisb MCI Mail: chrisb CompuServe: 74766,1034

Preface

Thank you for purchasing the Omega Point/2 Bulletin Board System for OS/2.

The Omega Point/2 BBS program embodies several innovations to the concept of "BBS Software". We choose OS/2 as our development platform because it offers power and flexibility far greater then that offered by MS-DOS, and more tailored to the needs of a BBS system then other multitasking operating systems, such as UNIX. Omega Point/2 can meet the needs of virtually any BBS System, from the single line message base, to a full blown 48 line system with several add-on programs and customized features.

The system represents approximately 32,000 lines of 'C' code. It was written entirely from scratch and specifically tailored to take advantage of OS/2's advanced features.

We hope your experience as a BBS System Operator is an enjoyable and profitable one.

Christopher A. Boaro
ExcelSoft Software
Norcross, Georgia
October 19, 1990

IMPORTANT - READ CAREFULLY BEFORE OPENING. By opening this sealed package, you indicate your acceptance of the following ExcelSoft License Agreement.

ExcelSoft Software License Agreement:

This is a legal agreement between you, the end user, and ExcelSoft. By opening this sealed package, you are agreeing to be bound by the terms of this agreement. If you do not agree to the terms of this agreement, promptly return the unopened package and any accompanying items for a full refund.

ExcelSoft Software License

1. GRANT OF LICENSE. ExcelSoft Software grants to you the right to use one copy of the enclosed software program (the "SOFTWARE") on a single computer (i.e., with a single CPU). You may not network the SOFTWARE or otherwise use it on more then one computer at the same time.

2. COPYRIGHT. The SOFTWARE is owned by ExcelSoft Software and is protected by the United States copyright laws and international treaty provisions. Therefore, you must treat the SOFTWARE like any other copyrighted material (e.g., a book or musical recording) except that you may either (a) make a reasonable number of copies of the SOFTWARE solely for backup purposes or (b) transfer the SOFTWARE to a single hard disk provided the original and any other copies are kept solely for backup or archival purposes. You may not copy the written materials accompanying the software.

3. OTHER RESTRICTIONS. You may not rent or lease the SOFTWARE, but you may transfer the SOFTWARE and accompanying written materials on a permanent basis provided you retain no copies and the recipient agrees to the terms of this Agreement. You may not

reverse engineer, docompile, or disassemble the SOFTWARE.

4. DUAL MEDIA SOFTWARE. If this SOFTWARE package contains both 3.5 inch and 5.25 inch disks, you may use only the disks appropriate for your single-user computer. You may not use the other disks on another computer or loan, rent, lease, or transfer them to another user except as part of the permanent transfer (as provided above) of all SOFTWARE and written materials.

LIMITED WARRANTY

LIMITED WARRANTY. ExcelSoft Software warrants that the SOFTWARE will perform substantially in accordance with the accompanying written materials for a period of 90 days from the date of purchase. Some states do not allow the limitations on the duration of an implied warranty, so the above may not apply to you.

CUSTOMER REMEDIES. ExcelSoft Software's entire liability and your exclusive remedy shall be, at ExcelSoft Software's option, either (a) return of the price paid or (b) repair or replacement of the SOFTWARE that does not meet this Limited Warranty and which is returned to ExcelSoft Software with a copy of your receipt.

NO OTHER WARRANTIES. ExcelSoft Software disclaims all other warranties, either express or implied, including, but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to the SOFTWARE and accompanying written materials. This limited warranty gives you specific legal rights. You may have others, which vary from state to state.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. In no event shall ExcelSoft Software or its suppliers be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use of or inability to use this ExcelSoft Software product, even if ExcelSoft Software has been advised of the possibility of such damages. Because some states do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

This Agreement is governed by the laws of the State of Georgia.

Should you have any questions concerning this Agreement, or if you wish to contact ExcelSoft Software for any reason, please write: ExcelSoft Software Customer Service, 1100 Indian Trail Drive, #1725, Norcross, GA 30093.

Omega Point/2 Bulletin Board System

Introduction

The Omega Point/2 BBS system can be used for a variety of purposes. It will serve equally well as a one line mail or message server, or as a large multiline pay-to-use entertainment, teleconference, and message system. Some of Omega Point/2's significant innovations include:

- Takes full advantage of OS/2's advanced features, such as threading, virtual memory, Dynamic Link Libraries, and multitasking.
- Teleconferencing with full paging and private messages, as well as unique features such as "anonymous teleconference".
- Full featured E-Mail and Message systems.
- File transfer section with file database and advanced search options. Standard file transfer protocols such as Xmodem, Ymodem, and Ymodem-G.

- Full support for modern high speed and error correcting modems.

- Complete toolkit to allow you to create your own add-on programs. These programs take full advantage of OS/2's DLLs to make add-on programs virtually transparent to the host system.

- Full Sysop control with access levels from 0-99 and 32 access keys. Virtually any command can be restricted based on the user's access level, access keys, or a combination of the two. Specific access rights for message and file system access can also be assigned.

- Complete Sysop monitoring capabilities - even when the user is executing an add-on program! There is also a wide variety of online reports available to the SysOp. In addition, data can be exported from the system into your favorite spreadsheet or database.

- Omega Point/2 is fast. It has been run successfully with as many as 12 lines on a simple 8mhz 286 based system.

Who needs Omega Point/2?

Omega Point/2 is designed to fit the needs of any conventional BBS system, and at the same time define new, nontraditional, uses for BBS software. It's ideal for the small business that needs a customer support or E-Mail system. Likewise, it provides a stable

foundation on which to build a multiline pay-to-use system. It can even be used in a contained office environment for electronic mail, teleconferencing, or even anonymous teleconferencing. Omega Point/2 is designed to be flexible enough to meet your needs, whatever they may be.

Chapter 1

Installation

This section outlines how to install Omega Point/2 on your system. Following these procedures will give you a ready to run, basic system. You may wish to customize your system by adding ports and modems, changing menu commands, and creating new message and file forums after your initial system is operational.

1.1 System Requirements

Installation requires a 286 or 386-based AT or PS/2 compatible, running OS/2 1.1 or later. OS/2 1.2 with an HPFS partition is required to use all of Omega Point/2's advanced features. Using OS/2 1.2 with HPFS is also highly recommended because of it's superior performance and reliability. Approximately 1 megabyte of disk space will be used for the system's programs. More will be needed for messages, download files, news items, etc.

1.2 Installation Procedure

1. Insert the distribution diskette in drive A:, then enter A:INSTALL from any OS/2 command prompt.

2. You will be asked to select a destination drive for the software, the drive letter may be from 'C' to 'Z'. The install program will create a directory off the root directory of the designated drive called "RUNBBS".

3. During installation, you have the option of initially setting up your system with either one working modem or none. You may add more modems by modifying the PORTS.BBS file (see appropriate section), but this will allow you to get started. If you choose "Yes" here, port 1 will be initialized for a Hayes compatible modem on COM1. Again, this may be easily changed but is provided here to allow you to get started quickly.

4. There is at least one DLL (Dynamic Link Library) file on the distribution diskette. This will, by default, be copied to the root directory of drive C:. If you wish the install program to copy it elsewhere, you may specify the

correct pathname on the INSTALL command line, or you may copy the files yourself after the installation is complete.

5. After all the files have been copied, the BBS will be started. When you see the initial status screen, you may enter ALT-U to login to the BBS as the local user. Use the username "SYSOP" with the password "SYSOP" to get started. You will want to change this password before you open your system for regular use.

6. Finally, you should be sure that your CONFIG.SYS file has the following options configured correctly: IOPL should be equal to YES (IOPL=YES), memory swapping and moving should be turned on (MEMMAN=SWAP,MOVE), threads should be equal to at least 128 (THREADS=128), and you should have a serial device driver loaded. If you are using the standard OS/2 drivers on an AT compatible machine, use the command: DEVICE=C:\OS2\COM1.SYS. If you are working on an IBM PS/2 with IBM OS/2, use the command: DEVICE=C:\OS2\COM2.SYS. You must set the TimeZone envirtonment variable in order for Omega Point/2 to keep proper time. Do this by including the line: TZ='timezone', where 'timezone' is the three letter timezone abbreviation followed by a number indicating the hours difference between you and GMT. For example, 'PST8' would be used for Pacific Standard Time. More information can be found on this environment variable in your OS/2 Manual. Finally, if you are using a 3rd party communications board, such as a DigiBoard, consult the manual for that product. Omega Point/2 requires no special execution path or data path.

1.3 Running the Program

After installation, the program will be automatically loaded for you. Subsequently, you may load the program by first going to the drive the program is installed on (C:, D:, E:, etc), changing directories to RUNBBS, and entering the command "DOBBS".

Chapter 2

Guided Tour

This chapter will quickly acquaint you with the features, layout, and functionality of Omega Point/2. While reading this chapter, it is suggested that you follow along on your own system. Each of the topics discussed here will be covered in greater detail later.

2.1 The System Operator Display

When you first start your system, you're presented with the main program screen, or the System Operator Display. The system operator is also known as the "SysOp" for short. This display presents several important pieces of information to the operator. First, it describes the status of each port; the line number, who is logged in, and the user's location or the port's status. Below that is a short "audit trail". As the BBS is used, this window will display various types of information. For example, when a user pages the SysOp, a line to that effect will be displayed. If there were any errors starting the BBS or initializing ports, that too will be displayed here. To the right of the Audit window, is a short help window. The help window indicates several of the keys that can be used to operate the system. For a complete list of those commands, press your F1 key. A window with the full list of console commands will be displayed. To clear this window, press your ENTER key.

Notice that the command to log onto the system is ALT-U. Hold down the ALT key, and then press the U key. The screen will clear, you will be presented with a skeleton INIT.MSG file, and prompted to enter a username. When the system is first installed, a single account is set up and activated. This account has the username SYSOP, so enter that at the prompt. Next you are prompted for a password. The initial password for the SYSOP account is, oddly enough, SYSOP. After entering that, the login procedure will continue by telling you that you are the first caller, that you have no messages waiting. Next, the main menu will be displayed followed by the main menu prompt.

2.2 Selecting a System Name

The first thing you will want to do is choose a name for your system. To enter your own name for the BBS, enter SETUP at the main menu prompt. Only those users with SysOp level access may execute this function. There are several such "protected" commands.

When you select SETUP, the screen clears and a list of various configurable options are displayed. To change the name of your BBS, select option number 1. You will be prompted for a new name; enter it now. Notice that by pressing 'N' you can cycle through several screens of configuration data.

On the first screen, you may also wish to change the SysOp name and the system phone number. When you are satisfied with the results, enter Q to quit back to the main menu.

2.3 Teleconferencing

Remote BBS teleconferencing is fast becoming extremely popular, and teleconferencing is a favorite feature on most multiline bulletin boards. To enter the teleconferencing section, enter 'T' at the main menu. The screen clears and a short menu of available commands is presented. Following this is a short status display showing the room number, it's name, and the other users in the room (right now it should say that you are the only one present).

All commands in the teleconferencing section are preceded by the "/" character. This helps the program distinguish between commands and regular text. Start things off by simply typing "hello". Your screen will show:

[Sysop] hello

You can also send a private message to yourself by entering "/p sysop hello!". On your screen, you will see:

[Sysop PRIVATE] hello!

The teleconference section is partitioned into several sections. Each section is called a "room". There are as many "rooms" as there are ports available on your system. If you have a 2 line version of the software, you will have 3 "rooms" available. To go to another room, enter "/j 2". This command tells the software

that you wish to join room 2.

Teleconferencing will be covered in detail in a later section of this manual. For now, enter the command "/Q" to quit teleconferencing and return to the main menu.

2.4 Personal Information

A record is maintained for each user on your system. This record contains such commonly used information as the users name, his password, his age, etc. You can modify your own personal information by entering 'Y' at the main menu.

When you do this, a list of several data items will be presented. Enter the number corresponding to the item you wish to change. This would be a good time to change the SYSOP password. Since the SysOp has a tremendous amount of power on the system, it's very important to prevent others from using that account -- and "sysop" is a pretty easy password to guess.

From this prompt, you may also select 'F' to set certain personal preference toggles. You may, for instance, choose to disable the display of line numbers when editing messages, or turn on your "Big Ben Chimer", which will notify you of the time and your time logged onto the system every hour on the half hour.

When you have changed the appropriate fields, enter 'Q' to return to the main menu.

2.5 Electronic Mail

Early on in the history of computer communications, it was realized that communicating electronically was very efficient. No more telephone tag and you're free to hold your conversations at whatever time of day suites you. To enter the Omega Point/2 Electronic Mail system, enter 'E' at the main menu prompt.

As with most other sections of the system, you are presented with a menu describing the commands available. After that, you are told that your mailbox is empty. Let's begin by writing ourselves a message. Enter 'W' at the E-Mail prompt. You are first prompted for the name of the recipient. Enter "SYSOP" here. Next, you are prompted for a message subject. Enter "TEST". You are next prompted to begin entering your message. Enter a line or two. To finish your message, enter a period (.) as the first character on

a line by itself. You will be prompted for various editing commands, enter 'S' to save. [Note, you could also have entered ".S" in the first part of an empty line to quickly save the message]. And lastly, you will be prompted for any special processing you wish for this message. Later, you will learn how to attach a separate file to the message or send a "Carbon Copy" to other users. For now, simply press ENTER to send the message.

Back at the E-Mail menu, enter 'B' to browse your mail. One line representing the message you just sent yourself will be displayed. You may now read that message by entering 'R' at the E-Mail prompt. Press ENTER to start reading at the first message.

The text you entered above will be displayed. You may opt to reply to your own message by entering 'R' here.

When you have finished, enter 'Q' to return to the main menu.

2.6 Public Messages

Public messages are similar in nature to Electronic Mail, but are generally posted for public consumption, for anyone to view, hence the term "bulletin board". To enter the message section, enter 'M' at the main menu.

When installed, the system sets up a certain number of message bases, also called "forums". Your system may have up to 100 forums. The SysOp may add or delete forums as needed.

Start by posting a message. At the message prompt, enter the command "W". Follow the same procedure used in the E-Mail section to enter and save your message. Now, to read the message, enter "RF" (for Read Forward). Press ENTER when prompted for a message number. The message you just posted will be displayed. There are several options available for reading messages, for example, you could enter "RR" to Read Reverse, or "RN" to Read New messages posted.

Since you are the SysOp, you have powers beyond those of mere mortals. For example, at the message prompt, enter EDIT. This will display a list of configuration information for this particular forum. From here, you can set access levels needed to gain entry or perform certain actions such as read or post messages. You can also toggle on and off the file library or message section of this forum. Let's create a new forum now. Enter 'Q' to return to the main message menu.

To create a new forum number 23, first attach to that forum. Notice that even though the forum is not yet active, you may still enter -- since you're the SysOp. To do so, enter 'C' at the message menu, then "23" when prompted for a forum number. When you arrive in the forum, you'll notice that it does not yet have a name. Enter "EDIT" to bring up the configuration information. Start by selecting option number 1 and choosing a new title. Next, toggle options number 7 and 8 to turn on the message base and the file library for this forum. Finally, enter 'Q'. And behold, you have just created a new Forum.

There are many options for customizing and using the message forums. For now, enter 'Q' to quit back to the main menu.

2.7 File Libraries

The personal computer BBS was first created by Ward Christianson to enable himself and a colleague to exchange program files. This was the first "file system". To use the Omega Point/2 file system, enter 'F' at the main menu.

The "feel" of the file system is similar to that of the message forums. For example, you can use the "C" command to change to a new file library. You can also enter "EDIT" at the file library prompt. This will allow you to do various maintenance on the system.

You may also use "S" or "B" to scan or browse files. Or "P" to set various search and list preferences.

There are few options available in this section for us to cover now, since your system is new and has no download files to view or search. A more in depth discussion will follow later. For now, enter 'Q' to once again return to the main menu.

2.8 Other Commands

Several other commands are available at the main menu. For example, you can enter "WHO" to see who else is online now. You may enter "P" to enter your personal "Profile" (something of an electronic resume or biography). You may send a paged message to

yourself or another user with the "PAGE" command. For example, enter "page sysop hello!". You may enter "R" to view a list of the most recent callers. Take the time to experiment with these commands.

2.9 Sysop Only Commands

As has been stated before, the SysOp of the system has great and wondrous powers. Several options, besides the "SETUP" command, are available from the main menu.

Enter "CONSTAT" at the menu. This is a SysOp only report that displays the status of each line, as well as some system specific information.

Enter "1" at the menu to view a log of who has done what on the system recently.

You could enter "2" to forcefully disconnect an unruly user.

Use the "3" command to change the information record for a particular user. This is very similar to the user available "Y"

command, except that the SysOp may use it to change anyone's information. This command is also used to assign user access levels and/or access keys.

The "4" command deletes the logged information viewed with the "1" option.

"MESSDUMP" displays a summary report of the message system. It shows the number of messages in each forum, how much memory is being used for indexing, and which users are currently using the forum.

Omega Point/2 is very modular. Almost all the commands available are held together by the concept of "Menu Configuration Files". Any command in a Menu Configuration File (MCF) can be attached to an internal BBS function, another program, or even a specific function in a DLL. Most commands are internal BBS functions. For example, when you enter "WHO", the main menu MCF searches for the command "WHO" and finds that it is linked to the function USERSON.

Because of this modularity, you may opt to link or not to link any internal function to a menu item. You can even run functions that may not be linked to any particular menu option. For

example, by using the "RUN" command, you can specify any function you wish to run. Type "RUN USERSON" to bypass the "WHO" command and list the users on the system. Another useful function is "SysopRemSpy", which allows the SysOp to "emulate" a particular user. Once you have a caller on your system, enter "RUN SYSOPREMSPY" from the main menu, then type in the port number the user is logged onto. This provides basically the same functionality as using ALT-W from the console to watch a user, but SysopRemSpy can be used by sysops calling in remotely.

Chapter 3

Adding Ports and Modems

Omega Point/2 supports between 2 and 48 incoming lines, depending on your configuration. Incoming lines can be either direct connections (via "Null Modem" cables) or modem ports. Virtually all types of modems are supported. Ports may be configured individually, to allow for maximum flexibility. The configuration of specific ports is done by editing the file PORTS.BBS with any ASCII text editor (the OS/2 system editor is sufficient for this

task).

The PORTS.BBS file is composed of several lines. One line represents the settings for each port. Lines that begin with the '#' symbol are considered comments and are ignored. Each line of the PORTS.BBS file is composed of 25 fields. Fields are delimited by commas. Each field and it's options are discussed below:

Field 1: This field may hold either the text "ON" or "OFF". By setting this field to "ON" and starting the BBS, you are indicating that the incoming BBS line that corresponds to the line number in the file should be made active. Likewise, changing this field of "OFF" tells the program that you do not wish to receive users on that port.

Field 2: Field two holds the device name that is to be used for this port. Commonly, this value will be COM1, COM2, COM3, etc. If you are using the standard OS/2 serial device driver, you may use either COM1 or COM2 in this field. If you are using IBM OS/2 on an IBM PS/2 machine, you may use COM1, COM2 or COM3 (assuming you have hardware ports associated with those device drivers, of course). If you are using a 3rd party communications card, such as a DigiBoard product, consult the manual for that product.

Field 3: This field indicates the type of flow control that the program should use for this line. CTS/RTS flow control is used either on input or output, both, or neither. The value here may be either RECEIVE, TRANSMIT, NONE or BOTH. In most cases, you will want to use NONE for standard 2400 baud modems, and BOTH for high speed or error correcting modems, such as US Robotics HSTs. If you specify flow control, you will probably need to specify the correct option for your modem in the initialization string.

Field 4: Field four is used to indicate to the program that the link rate should be fixed, or variable to match the connect rate. For example, if you're using a high speed modem, you may wish to

always communicate with your modem at the maximum supported speed to insure you get the maximum throughput over the line. If you were using such a modem, you would set this to FIXED. With normal modems, however, you should set this field to NORM. Note that if you use FIXED you will, again, have to use a command in your initialization string to indicate this to your modem.

Field 5: Field five is the default link rate used between your computer and the modem. Normally this should be set to the highest speed supported by the modem. If you are using a high speed modem, you will probably want to set this to 19200 (the maximum baud rate currently supported by OS/2), set field 4 to

fixed, and include the appropriate options in your initialization string.

Fields 6, 7, and 8: These fields control the contents of the data byte. Field 6 is data bits, 7 is the parity, and 8 is the number of stop bits. In most cases, these fields should be 8, N, and 1.

Field 9: This field is used to toggle the line between a fixed link over a null modem cable and a modem. Enter MODEM here if the line is attached to a modem, and TERM if it's attached to a terminal or null modem cable.

Field 10 and 11: These fields are not currently in use. They will, in a future release, allow an "ON" and "OFF" time to be specified for each port.

Field 12: This field holds the initialization string for your modem. The ones provided in the skeleton PORTS.BBS file should be sufficient for most installations if you are using Hayes compatible modems. If not, consult your modem manual for the appropriate options. The following table lists the modem settings required for Omega Point/2 to function properly. The table describes the setting and gives the command used with standard Hayes and Hayes compatible modems.

Description	Hayes AT standard
Numeric response codes	V0
Command mode local echo	E1
Modem speaker off (optional)	M0
Result codes displayed	Q0
Suppress auto-answer	S0=0
Normal carrier detect	&C1
Disconnect on DTR toggle	&D2

The options listed above are generally used by most modems by default and are used by the default PORTS.BBS file. If, however, you are having problems with your modems, consult your modem's manual and change the initialization string as appropriate. Note also that some of these options may not be setable through the

initialization string on all modems. In such cases, the modem should have dip switches that perform the same functions.

Field 13: This field is the string sent to the modem to answer the phone. If your modem uses the Hayes "AT" command set, this string will be "ATA".

Field 14 through 25: These fields represent the numeric values returned by your modem when certain events occur. Your modem should be configured to return numeric rather then text result codes. The first value is the number returned when the modem signals "OK". The next is the value returned when the modem signals "RING". The chart below illustrates which codes should be in each field:

Field	Response	Typical Value
14	OK	0
15	RING	2
16	CONNECT 300	1
17	CONNECT 300 Error Correct	14
18	CONNECT 1200	5
19	CONNECT 1200 Error Correct	15
20	CONNECT 2400	10
21	CONNECT 2400 Error Correct	16
22	CONNECT 4800	18
23	CONNECT 4800 Error Correct	19
24	CONNECT 9600	13
25	CONNECT 9600 Error Correct	17

Notice that there are two connect responses for each baud rate. The first indicates that a standard connection is made, while the second indicates that the connection was made with some type of error correction. The two most common error correcting protocols in use today are MNP Level 5, and V.42bis. Not all modems support these protocols, so the associated fields are useless with these modems. Also, in the chart above, the "Typical Value" fields are taken from the US Robotics HST/V.32 dual mode. The "Error Connect" fields may be specific to this modem. All other fields are fairly standard across Hayes compatible modems. Your modem manual should have a chart similar to the one above. If you are having trouble connecting with the default settings in the PORTS.BBS file, consult the modem manual and verify you have the correct result code settings.

Chapter 4

The System Console

When you first start Omega Point/2, you are presented with the main status screen. This display shows several pieces of important information about your system.

4.1 Presentation of Information

The largest section of the display is the line status window. It shows various pieces of information about each of the available ports. Status information for up to 24 ports is shown. The information is divided into three sections; line number, username, and line status. Depending on the version you purchased, all displayed ports may not be active. The port number column for active ports is in the color cyan, while those for inactive ports are in white. If your version supports more then 24 incoming lines, pressing TAB will display the same information for the top 24 ports. Pressing TAB again will return the display to the lower 24 ports.

The "Username" field will display the name of the user logged onto the corresponding line on the system. The "Line Status" field will display the current state of the line. For example, if the modem is ready and waiting for a call, the text "Waiting..." will be displayed. If the port specified in the PORTS.BBS file is not active or in use by another program, the text "Error Open Port" will be displayed. If the modem could not be properly initialized, the text "Error Init Modem" will be shown. This may happen if there is no modem attached to the specified port or if the modem is not working properly.

Below the Line Status display is the Audit Window. This window is a running log of events on the system. It displays a status line when a user logs off the system. The window displays a diagnostic message if an error occurs. It does not, however, show detailed information about user actions or events. For that, log on to the system and run the MCF command SYSOPUSELOG (linked to the "1" key on the main menu by default).

To the right of the Audit window is the Help Window. This window gives quick help about the most often used console commands. One such command is F1, which brings up a larger help window showing

all of the available console commands. These commands are discussed in the following section.

4.2 Available Commands

This section describes each of the commands available to the SysOp from the console. Some functions require the operator to select a port to act on. For example, ALT-W (the "Watch" command) requires the operator to select which port he or she wishes to "watch". When a command of this type is selected, a red and white arrow will appear on either side of the User Status line. Use the arrow keys to select the correct line or user, and press ENTER.

ALT-A: Sets the SysOp Available state. If the SysOp is "Not Available", he or she cannot be paged to chat by a normal user. Other remote SysOp's may always request a chat. Notice that the "paging beep" is a different sound if the paging user is a SysOp. This can be one three settings, "Available", "Not Available", or "On Timer". If the SysOp Available state is "On Timer", the SysOp may be paged if the system time is between the SysOp In and SysOp Out times described in the following chapter.

ALT-U: Use the system. This allows the SysOp to log onto the system from the console. This is essentially the same as logging on remotely, except that the local user can not do file transfers that require a protocol (i.e., Xmodem), they can attach files to messages directly from their disk, etc.

Note that users with SysOp level access, may use one of several options when logging onto the system. These options are appended to the end of the user's login name following a semicolon. The following options are available:

Option	Effect
;i	Causes the user to be "invisible". The users name will not be displayed in any user listings.
;n	Using this option will prevent the system from entering the user's name in the recent caller log.
;s	Set's the "SysOp Flag" for the user. This causes the "!" character to be printed to the right of the user's name in the "users on-line" display. It has no effect other then the cosmetic. It is intended to highlight SysOp level users for regular users.

Example:

```
Login: sysop;i
```

Would cause the user "sysop" to be logged in in invisible mode.

ALT-W: Watch a user. When this command is invoked, the operator selects the port to "watch". If there is a user logged onto that port, all characters sent to the remote user will also appear on the console screen. In addition, all characters entered on the console will be inserted into the users key buffer. Pressing ALT-W will return the system to console display and end the "watch". "Watching" a user is also referred to as "spying" or "emulating".

ALT-Q: This command terminates the BBS. The system returns to the OS/2 command prompt from which it was invoked.

ALT-K: Kill a user. When this command is invoked, the operator chooses the desired port. The user logged onto that port (if there is one) will be logged off the system. Any external programs or sessions he or she has invoked will be terminated. This is useful for removing troublesome users from the system.

ALT-L: After invoking this command and selecting the desired on-line user, the operator will be prompted for a new user level for that user. This command is useful for "validating" a new user while he or she is still logged in. The access level of that user is changed, and all rights, privileges, and time restrictions are applied immediately.

ALT-H: This command takes the modem "off-hook". In essence, this makes the line perpetually busy.

ALT-M: Modem reset. This command forces a modem reset. This command is useful, for example, if the modem for a particular port was not turned on when you started the system and you wish to manually initialize it. Or, you may want to simply test the modem to be sure it's communicating correctly with the BBS.

ALT-T: This command allows the operator to enter a message, which will be sent to the selected user. The message will be preceded by the text: "From SysOp: ".

ALT-C: Force chat with a user. This will establish a one on one dialogue with the selected user and the operator. Notice that if the console is "watching" a user, there is no need to select a user to chat with, the current user is assumed. There are some

places where this command can not be invoked, for example while the remote user is doing a file transfer.

F1: Displays the Help Screen.

F2: Displays user information for the user being "watched". Alternatively, the operator may select the user to display information for.

F7: Freeze a port. This command is useful for "shutting down" a particular port. You may wish, for example, to execute a communications program that utilizes a port taken by your BBS.

Rather then shutting down your system to use the communications program, you may simply "freeze" the port.

F8: Undoes the affects of the "freeze" command. This command restarts and resets the selected port.

Chapter 5

Basic Setup and Configuration

In the Guided Tour chapter, the SETUP command was discussed briefly. This chapter will discuss the options available in the Setup menu in greater detail. The options in the Setup menu are included inside the BBS program, rather then in an external program, to allow easier access for remote sysops. To enter the Setup menu, enter SETUP from the main menu of the BBS. You must be logged on as the sysop (or as a user with sysop level (99) access). There are four input screens worth of data. You may travel from one screen to the next by entering 'N', or go back to the previous screen by entering 'P'.

5.1 General Information

This input screen includes several options for setting basic operating parameters.

1. First, the BBS name. The BBS name can be included as a

variable inside any of the MCF files (covered later). This allows you to change the name of the BBS without changing all of your menu or MCF files.

2. Next is the SysOp name. You should set this to your name. It too can be printed from within MCF files.

3. Option number three is for your BBS' main phone number.

4. Option number four is a directory name. It points to a directory used to hold various personal files. In other words, files that pertain to specific users. For example, "Profiles" are stored here. Also, a "mailbox" file for each user is kept in this directory. When the program is first installed, the directory ETC is created below the RUNBBS directory for this purpose.

5. Next is the directory name where the message system files are kept. These files can grow to be fairly large, so you may wish to put them on another disk drive. When the program is installed a directory called MESSAGES is created off of the RUNBBS directory, and messages are stored there. To move them to another location, create the new directory, copy the files from MESSAGES to the new

directory, and use this option to change the directory name.

6. The last directory name is for user files. Each user has a user record file that takes up 2048 bytes. On installation, a directory called USERS is created off of the RUNBBS directory.

7. Next is the SysOp "IN" time. This is the time, in 24 hour military time, that the Sysop will be available for chatting. Notice that this is only in effect if the sysop available flag is set to "on timer".

8. Following that is the sysop "OUT" time. It uses the same format as the previous field, except that it shows the time the SysOp is no longer available to be paged.

9. The next option, is a flag to indicate that console should or should not "spy" on any user that logs in. For example, if this option is set to "yes", the system will automatically go into "watch mode" when someone logs on. Note that it will not stop watching one user to watch another user just logging on.

10. Next is the new user access level. This is the access level, 0 to 99, assigned to new users. In the later discussion of MCF files, you'll see how permission is granted or denied to run any particular command based on this value.

11. Another form of access code are access keys. Any command may be run based on an access level, an access key, or a combination thereof. Keys can be used to give someone permission to a particular command that his access level would not normally permit him to use, for example.

12. Download to upload ratio sets the number of files a user is allowed to download before he is required to upload one.

13. In Omega Point/2, a user may be given time based on either a day or month basis. For example, you can set up your system to give users x number of minutes per day or x number of minutes per month. This field allows you to specify which will be used.

14. When a user enters a forum, the program scans the forum to find out how many "new" messages there are. The system can either search for new message based on the date the user was last in the forum, or by the last message the user read in the forum. This option allows you to set which of these two options are used for your system. The default is to search for new messages based on message number.

5.2 New User Information

When a new user logs onto the system, he or she is normally asked to input several pieces of information. For example, the system requests that the users real first name and last name are entered. You may wish to suppress some of these questions. This input screen allows you to do so. You'll notice that after each item is either a "yes" or a "no". Simply enter the number of the item you wish to toggle. Items with a "yes" after them will be requested of new users logging into your BBS for the first time.

5.3 Alloted Time

The following screen allows you, the SysOp, to set the number of minutes per billing period (day or month) the user is permitted

to be online. Notice that this parameter may be set for each user level. This way, more time may be granted for higher user levels.

5.4 Time per Call

In addition to the time allotted for a billing period, a maximum time per call may be set for each user. This is done on input screen 4. When the system is initially installed, all values on this screen are set to 0, which means that the time per call is equal to the maximum time allowed per billing period. It may be desirable, however, to limit the amount of time a user can be logged in on any particular call. Change the values as appropriate for your operation here.

Chapter 6

Menu Configuration Files

The backbone of Omega Point/2, and the feature that distinguishes it from other similar products, is it's Menu Configuration Files (MCFs). Other communications packages allow you to assign commands to certain functions or customize the appearance of menu prompts. These features are also found in Omega Point/2.

Beyond that, however, commands can be used to actually start other programs, or even execute specific functions in a DLL. This is vaguely similar to the "doors" concept found in several DOS based programs. Unlike doors, however, programs executed from MCF files appear to be just another part of the whole system - so long as these programs are written using the input and output routines found in the BBS API developers kit (which accompanies this software). When a BBS API application is executed, the user in that application can still receive pages, the SysOp can still "watch" that user, etc. In addition, such applications have access to a wealth of data about the rest of the system. You may not have ever considered writing your own expansion modules, but since this interface is well documented and readily available, there will be applications created by others that you will be able to take advantage of.

In addition to the functions related to command processing, the MCF files also allow you to set such things as ENTER and EXIT scripts (a set of commands that are executed when the user enters or leaves a menu), the appearance of the prompt (or prompts) associated with a menu, and the "name" of the menu (what appears in the "location" column when a user executes the "who" command).

An Omega Point/2 system may have up to nine MCF systems. Each system is contained in it's own editable ASCII text file. You may use the OS/2 system editor or any other standard ASCII text editor to modify these files. The nine MCF files are called:

MAINMNU.MCF	Main menu. Executed at login.
MAILMNU.MCF	E-Mail System.
MESSMENU.MCF	Message System.
FILEMNU.MCF	File (uploads/downloads) System.
NEWSMNU.MCF	News and Information
GAMEMNU.MCF	Games and Entertainment.
AUX1MNU.MCF	[defined by the SysOp]
AUX2MNU.MCF	[defined by the SysOp]
AUX3MNU.MCF	[defined by the SysOp]

The AUX1, 2, and 3 MCF files are not provided with the software, but may simply be created and linked from any command on any other MCF system.

Each MCF file is read and compiled by the system when it is started. Therefor, any changes made while the system is running will not take effect until after it is restarted.

The MCF files use the same convention as the PORTS.BBS file with respect to comment lines; any line that begins with with a '#' character is ignored, as are any characters on the line following a '#' character. In addition, the command links are composed of multiple fields delimited by commas.

A MCF file is comprised of two basic components, the "Header Options" include the prompt description(s) and ENTER/EXIT macros. The second component is the body of the "Command Links". The rest of this chapter will discuss the syntax and options associated with both parts. The following chapter will describe each of the internal functions which may be linked to a command in an MCF file.

6.1 MCF Header Options

There are three basic header options; ENTER or EXIT macros, PROMPTs, and NAME. The ENTER and EXIT commands are identical in syntax. Likewise, in the MCF files that use more then one prompt, the PROMPT commands (PROMPT, PROMPT2, and PROMPT3) are identical. Each command uses the same basic format, simply the name of the variable to set (i.e., ENTER, EXIT, PROMPT, etc), followed by an equals sign, followed by a string of characters wrapped in quotation marks. Consider the default EXIT command found in the MAINMNU.MCF file:

```
EXIT="@10"
```

This sets the EXIT macro for this MCF file to "@10", which in this case happens to display a "saying of the day". The effect here is that when the user exits the menu, the command "@10" is pushed onto his "command stack", and executed by the MCF file just as if the user had typed it in himself.

6.1.1 The ENTER and EXIT Commands

As described above, to create an ENTER or EXIT macro, enter the variable name, followed by an equals sign, followed by the macro (enclosed by quotes) on a line in the appropriate MCF file. The command string enclosed in quotes will be executed when the user enters that particular MCF system (or exits in the case of the EXIT variable). For example, if you wanted users logged in to see

a list of other users logging in at that time, you could use the following command:

```
ENTER="WHO"
```

Note that the command "WHO" would have to be linked to the internal function UsersOn.

Commands in ENTER and EXIT messages may be chained. This same rule applies to command entered by the user, as well. Chained commands are delimited by the semicolon (';') or the space character. If spaces are required in your command, you may use the "_" to indicate this; they will be mapped to spaces after command chaining takes place, but before the command is processed. For example, if you wanted the user to see a list of other users logged on, and then immediately go to the Message section upon logging in, you could use the command:

```
ENTER="WHO;M"
```

The EXIT command uses the same syntax.

6.1.2 The PROMPT Command

When a program pauses to allow the user to input some piece of data or command, it displays a prompt. This prompt should indicate to the user what type of information is required and permitted. Omega Point/2 gives you the ability to customize the way a prompt is displayed for a user. In an Omega Point/2 MCF file, there is always at least one PROMPT command. Some special MCF files have more then one, such as MAILMNU.MCF and MESSMNU.MCF.

The syntax for the prompt command is essentially the same as that of the ENTER and EXIT commands discussed above. The appearance of the prompt is described in text surrounded by quotation marks. However, unlike ENTER and EXIT commands, prompt strings may contain variables or display commands. In addition, the prompt may contain any text that you wish to display. Consider the following example:

```
PROMPT="$n$c(2)$b : $t(0)$nSelect [$c(7)1,2,3$c(2)]: "
```

In this example, it is possible to pick out two distinct types of data; the text the user will see, and various commands or special operators, which are preceded by the '$' character. The operators available to you are detailed below, for now follow along with the above example.

First, the $n indicates that a new line should be started. Next, the $c(2) tells the system to change the color to green. The next command, $b, tells the system to print the variable "board name" (the name of your system. You used SETUP to enter this). Next is

the colon, which is printed as you see it by the system. The $t(0) tells the system to print the current time in format 0, which happens to be: DD-MMM-YY HH:MMpm. After the time is printed, the $n prints another new line. The system prints the text following the $n up to the $c(7), at which time the color is changed to white. the text "1,2,3" is printed, the color is again changed back to green, and the remainder of the text is displayed. As you can see, the prompt commands are both flexible and easy to use. The following section describes each of the special operators that may be used within a prompt string.

6.2 PROMPT String Variables

$i This variable prints the total number of messages in the current message forum. It should only be used in the MESSMNU.MCF.

$a This variable prints the number of messages AFTER the current message in the current message forum. Like $i, it should only be used in the MESSMNU.MCF file.

$1 This variable prints the message number of the first message in the current message forum.

$2 This variable prints the message number of the last message in the current message forum.

$t This command formats and prints the current time in one of two formats. The format is specified by enclosing it's code inside of parentheses following the $t. Time code 0 prints the time and date in the format: DD-MMM-YY HH:MMpm; and code 1 prints the time only in the format HH:MMpm. For an example, see the section above.

$m Displays the name of the current message forum.

$f Displays the name of the current file database.

$n Begins a new line. Similar to pressing ENTER.

$r Prints the minutes remaining before the user runs out of time for this call and is forced to hang up.

$c Changes the color of the prompt. Similar in nature to the $t command, which requires that code enclosed in parentheses follow the command, the color operator needs to be told what color you wish to use. The following table lists the valid options:

Code	Color
0	Turns all color off
1	Bright Red
2	Bright Green

3	Bright Yellow
4	Bright Blue
5	Bright Magenta
6	Bright Cyan
7	Bright White
8	Red
9	Green
10	Yellow
11	Blue
12	Magenta
13	Cyan
14	White

Note that if the user does not have ANSI selected for his terminal type, no colors will be displayed and the $c commands will be ignored.

$u Displays the users name, or his handle.

$p Displays the port number on which the user is currently logged in.

$w Displays the date and time the user logged onto the system.

$l Clears the screen.

$b Prints the board name.

6.2.1 The NAME Command

The final Header Command is NAME. This simply sets the text that will be displayed in the UsersOn listing; when the user enters "WHO", for example. If you wished to change the text displayed for the "News and Information" section, for example, you could add the following line to NEWSMNU.MCF:

```
NAME="No News is Good News!"
```

6.3 MCF Command Links

With Omega Point/2 you can "link" almost any key or set of keys to a particular action. The action can be one of four things; a) an internal function that is part of the BBS system, b) an external program, c) an external session (similar to an external program, but is used primarily for programs not written using the BBS API and/or requiring direct access to the keyboard or screen) and d) a particular function from a DLL (Dynamic Link Library). As delivered, Omega Point/2 has most commands linked to internal functions.

In a typical MCF file, the Header Commands are followed by a set of Command Links. Each line in the MCF file (excluding comment lines) represents one link, or one user command that will create some response from the system. Each such link is composed of several fields, each field being delimited by a comma. The number of fields and their meaning varies slightly depending on what type of action is to be performed (function, program, session, or DLL). In all four cases, the first four fields of a link are the same. The significance of the this first set of fields is discussed below.

6.3.1 Common Fields

Field 1: Is a quote enclosed field representing the command as it will be typed by the user. For example, "WHO" in this field would mean that any time the user entered "WHO", this command (whatever it is) should be executed. This field is not case sensitive; the user may either enter "WHO" or "who".

Field 2: This field indicates what type of link this is going to be (function, program, session, or DLL). The following table summarizes the valid options for this field:

Field Entry	Meaning
FUN	Execute requested internal function
PRO	Execute requested BBS API program
SES	Execute non-BBS API program
DLL	Run a function from a DLL

Field 3: Describes the access level and/or access keys that are required to execute this command. Most often, this is simply a number between 0 and 99. 0 indicating that any user may execute the command, and 99 indicating that only the SysOp may do so. Optionally, you may also tell the system that only users that have the required level AND a particular access key may run this

command. Too, you may say that users that have the required access level OR a particular key may run the command. The correct syntax is as follows:

```
10 and 2
```

indicates that an access level of at least 10 AND key number 2 is required to execute this command.

```
99 or 4
```

states that the user needs to have either an access level of 99 (SysOp level) or key number 4 in order to run this command.

Field 4: This field is quote enclosed and contains one or more special flags. The valid characters inside this field are:

- M Re-displays the current menu after the requested command has been executed.
- T Stops the communications port read thread for this user. This option is only of use if you're executing a non-BBS API program (usually with the SES command in field 2).
- O Displays the string "One moment please..." so as to reassure the user while an external program is loading.

Note that these special options are most often used for the execution of external modules.

The remaining fields vary depending on the type of operation to be performed. The next four sections will cover the remaining record layout for each of the possible operation types.

6.3.2 Functions

Field 5: Simply contains the name of the function to be executed. The list of available functions and their behavior will be discussed in the following chapter. Note that in addition to "normal" functions, you may also link to another MCF file here. For example, if you wanted to jump to the message system MCF, you could put MessMenu in this field. This is typically done from the Main Menu MCF.

Field 6: This field is optional, and need not be included for all cases. It can be used, however, to respond to a prompt that will be presented after the specified function is run. To do this,

simply add the string that should be pushed onto the command stack and enclose it in quotes in this field.

6.3.3 Programs

Field 5: This field holds the name of the EXE file that is to be executed. Note that for the PRO option to work correctly, the EXE should be a BBS API application, in other words it should have been compiled using the I/O routines discussed in the developers kit documentation.

Field 6: Holds any command line arguments that may be needed by the invoked program. This field is quote enclosed.

6.3.4 Sessions

Field 5: Similar to the PRO option, this field holds the name of the EXE file that is to be executed. The difference between this option and the PRO option, is that a program invoked this way is placed in it's own session. Therefor, the program will have it's own virtual screen and keyboard. This is best used for programs that have not been compiled with the BBS API. Such programs would probably write directly to the screen and/or "steal" the

keyboard, therefor in order to maintain system integrity, they would have to be run in their own session.

Field 6: Identical to field 6 in the PRO section, this field simply holds any command line arguments to be sent to the invoked program. It is quote enclosed.

6.3.5 Dynamic Link Libraries

Field 5: The DLL option provides a great deal of flexibility. It combines the best attributes of internal functions with those of external programs. For example, when a DLL function is loaded, it becomes part of the in-memory code, so no loading need take place the next time it's loaded. On the other hand, you can specify that the function be removed from memory when not in use, thus achieving a good balance between execution speed and memory consumption. This field may hold one of three options:

- NEEDED: The function is loaded when it's called, and removed from memory as soon as it has completed execution. This is the most memory efficient option.

- ALWAYS: After this function is called for the first time,

it is kept in memory until the BBS is terminated. This option provides the best execution performance.

- USE: Once the function is called for the first time, it is kept in memory until there are 0 users logged on the system, at which time, it is removed from memory. This is a good compromise, since the function need not be loaded as long as there is a chance it will be used, but as soon as the system is cleared out, the memory can be freed.

Field 6: Holds the name of the DLL module that the function is to be loaded from. This module will have the extension DLL and must be in one of the directories specified in the LIBPATH environment string (set in CONFIG.SYS).

Field 7: Contains the name of the function within the DLL module that is to be run.

Chapter 7

MCF Functions

As discussed in the previous chapter, Menu Configuration Files contain command links which invoke either an internal function, a program, a session or a DLL function. This chapter will describe each of the internal functions which may be called from an MCF file. There are over 100 such functions in the current version of Omega Point/2.

7.1 MCF Control Functions

These functions are generally used inside a MCF. These commands are used to alter the current state of the menu.

QUIT
Calling this function tells the program to leave the current MCF system and return to the one it was called from. If the current MCF happens to be MAINMNU, the user will be logged off.

SHOW
This tells the system to display the current menu. You'll notice that in the delivered system it's used for the '?' or 'H' commands in most menus.

FORCESHOW
Normally the SHOW command will only print the next menu if there are no commands waiting. Use this command to show the menu even if the user has commands waiting in his queue. For example, when the user enters the MAILMNU, FORCESHOW is called to display the menu. This is because the ENTER string also contains a command to display the number of email messages waiting for a user. If SHOW was used, the menu system would see the mail display command and therefor not display the menu.

PUSH
This command can be used to push a command onto the users command stack. Notice that here, as elsewhere in the system, commands can be chained.

PAUSE
This command prints the following prompt:

```
    Press [ENTER] to continue:
```


and waits for the user to press enter.

WAIT
This command is similar to pause, except that it prints the prompt:

```
    ..More [Y,n]..
```

and waits for the user to press a key.

COLOR
Can be called to change the current color. The table of color

codes can be found in the preceding chapter in the section on the PROMPT command.

CLEARSCR
Clears the users screen. The method used to do this is dependent upon the user's terminal type, and therefor may not produce the desired affect for all users.

SHOWTIME
Displays the time in the requested format. The table of valid format codes can be found in the preceding chapter in the section on the PROMPT command.

7.2 Miscilanious and Information Display

Functions in this group are typically attached to commands on the main menu but may be called from any menu. These functions either display a certain piece of system data or allow the user to customize his environment.

VERSION
Displays the version data for your copy of Omega Point/2 BBS. This data includes the version number, copyright notice, and your specific license information.

SHOWUSE
As delivered, MAINMNU calls this function in it's ENTER string. The function displays the number of calls the user has made, the number the system has received, and the number the system has received that day.

SHOWSAYING
As delivered, MAINMNU calls this function when the user logs off. It simply displays a short "saying of the day".

SHOWNEWMESS
As delivered, MAINMNU calls this function when a user logs in. It displays the messages waiting for a user and includes email as well as each of the valid message systems.

CHANGEYOURSETTINGS
This function is initially mapped to the 'Y' key on the main menu. It allows the user to change things such as his name, address, etc. Off this menu is also another section used for changing specific user flags. These flags include the BIG BEN chimer and whether or not line numbers display when editing

messages.

RECENTCALLERS
Displays the names of the last 60 users to call the system. Includes the time they logged in and the number of minutes they spent on the system.

USERSON
Lists the users currently logged onto the system. It shows their name, city, stat, age, sex, line, and login time.

USERLIST
Lists all of the users that have logged onto the system.

PAGEUSER
Used to send a page to a user. A page is a message that appears on the receiving user's screen no matter where he or she is on the system (except for places where it would be impossible, such as file transfers).

MORE
Displays the requested file but stops after every screenful with the "..More [n,Y].." prompt. If the user then enters 'n', the output is terminated. Note that no such prompt will be presented if the user has selected a page size of 0 (which indicates that he or she wishes to have continuous output).

SHOWFILE
Displays a file to the screen but does not pause the display when the screen is full.

7.3 MCF Switch Functions

These functions cause the system to begin executing another MCF system. The system always starts processing with MAINMNU. After that, command links in MAINMNU may call other MCF systems.

The following table lists the appropriate function name followed by the menu called as a result:

Function Name	Menu or Subsystem Executed
FILEMENU	Calls the File System Menu
MESSMENU	Calls the Message System Menu
MAILMENU	Calls the E-Mail System Menu
GAMEMENU	Calls the Game System Menu

NEWSMENU	Calls the News and Information Menu
AUX1MENU	Calls the AUX 1 Menu
AUX2MENU	Calls the AUX 2 Menu
AUX3MENU	Calls the AUX 3 Menu
AUX4MENU	Calls the AUX 4 Menu
AUX5MENU	Calls the AUX 5 Menu
MNDOTELECONF	Calls the Teleconferencing System

7.4 File System Functions

This set of functions allows the user to scan the database of downloadable files and initiate file transfers. This set of functions should only be called from within the FILEMNU.MCF. This is because when the user initially enters this MCF, the correct files are opened and the file system is initialized.

FSYSLIST
This function lists the files in the current file section. It's behavior is dependent upon the users current file system preferences as well as the type of database in use (RAW, DATABASE, FILE). The use of the various types of file systems is covered in a later chapter.

FSYSBROWSE
This function is simpler to FSYSLIST except that is displays the long description as well as the short if the file system type is DATABASE for the current section.

FSYSDATELIST
Lists the current files by date, regardless of what the users File System Preferences are.

FSYSSEARCH
This function allows the user to search all files or a DOS wild-card subset of all files. A text string key may also be specified. The program searches for this key string in the short description of each of the matching files.

FSYSAREA
Allows the user to change to another area of the file system.

FSYSVIEWARC
Allows the user to view the contents of an ARC or ZIP file. If no extension is entered, the program defaults to ZIP.

FSYSEDITPREF
Allows the user to modify certain parameters used by other file system functions. The following attributes may be changed:

Files Sorted by Name or by Date
Files listed in Ascending or Descending order
Default file Upload Protocol

Default file Download Protocol
File list format (3 line or 1 line format)

FSYSUPLOAD
Allows user to upload a file to the system. This function uses the default upload protocol if one has been selected.

FSYSDOWNLOAD
Allows the user to download a file from the system. This function uses the default download protocol if one has been selected.

FSYSSYSOPEDIT
This function should have an access level high enough to allow only sysops to use it. It allows the user to perform maintenance function on the current file system. This function and it's relative, MESSBOARDEDIT, will be covered in their own chapter later in the manual.

7.5 Message System Functions

Like the File System Functions, these functions should only be called from the MESSMNU.MCF. This group of functions allow the user to read, write, and scan messages. Your system may have up to 100 message forums in your system.

MESSBROWSE
Function browses the messages in the current message forum. This function lists the full message header for each message found.

MESSWRITE
Allows the user to write a new message in the current forum.

MESSSCAN
Scans the messages in the current forum. This function simply lists a short description for each each message. This is a one line summary that includes the message number, the sender, the receiver, the date, and the subject.

MESSREADMARKED
This function reads the "marked" messages in the current forum. A message is considered "marked" if it is new to the reading user and it was sent to the reading user.

MESSREADINDIV
Prompts the user for a message number to read. If that message number exists, the system makes that message the current message and displays it.

MESSREADREV
The user is prompted for a message number to start reading from. If the user presses ENTER, the last message is selected. The

selected message becomes the "current" message, and the "read state" is set to reverse.

MESSREADFWD
The user is prompted for a message number to start reading from. If the user presses ENTER, the first message is selected. The selected message becomes the "current" message and the "read state" is set to forward.

MESSREADNEXT
This function makes the next message the "current" message and sets the "read state" to forward.

MESSREADPREV
This function makes the previous message the "current" message and sets the "read state" to reverse.

MESSREADNEW
This function makes the first message that is new to the reading user the "current" message. The "read state" is set is set to "forward".

MESSQUICKNEW
Many users prefer to read all the new messages on the system with one simple command. This function allows them to do just that. By default, it is linked to the command "RA" in the MESSMNU. Entering this command tells the system that you wish to scan all new messages. The system changes forums to the first available forum, reads all the new messages in that forum, then continues on to the next forum. Users may configure this function using MESSSCANCONFIG (see below) to only read messages in those forums that they have an interest in.

MESSSCANCONFIG
This function is linked to the "CS" command as delivered. It allows users to decide which forums are scanned or not scanned when using the MESSQUICKNEW function. When this function is executed, users are presented with a four item submenu. There are options for "Add a forum", "Remove a forum", "List forums", and "Quit". By default, all scan flags are set so users executing MESSQUICKNEW scan all forums. The user may use this function to remove forums from his or her list. If this is done, that forum will no be scanned when executing MESSQUICKNEW.

MESSDELETE
If there a current message has been established and the user has permission to delete that message, it is removed. If a current message has not been selected, the user is prompted for a message

number. If the user is permitted to delete that message, it's removed. Any SysOp or any user with their SysOp flag set for the current forum may delete any message.

MESSUPLOADTEXT

Allows the user to upload a prepared file into a message. The file may be uploaded using any of the supported file transfer protocols. It must not have any line longer then 70 characters and may not have more then 60 lines.

MESSAREA
Allows the user to change his current message forum.

MESSREPLY
If there is a "current" message, this function allows the user to enter a reply to it.

MESSPROCNEXT
This function is normally linked to the ENTER key. It processes the next message based on the current "read state". For example, if the "read state" is forward, this function reads the next message. If the state is reverse, this function reads the previous message. Note that there must be a valid "current" message, otherwise and error message is generated.

MESSBOARDEDIT
This function allows the SysOp (or the Forum SysOp) to change various defaults regarding the current forum. This command is covered in greater detail in it's own chapter later in the manual.

7.6 Electronic Mail Functions

Like the File System and Message System functions, these functions should only be called in the MAILMNU.MCF. They permit the user to manipulate the messages in his or her personal mailbox.

MAILREAD
Prompts the user for a message number to start reading at. ENTER defaults to the first message in the mailbox.

MAILBROWSE
Lists the messages in the user's private mailbox.

MAILDELETE
Deletes the current message if one has been selected. If no message has been selected, the user is prompted for a message number to delete.

MAILWRITE
Allows the user to input a new message. The user is prompted for the name of the user to receive the message and the subject.

MAILREPLY
If a message has been read, this function allows the user to reply to that message.

MAILFORWARD
Allows the user to forward the current message to another user. The message remains in the forwarding user's mailbox. This function only works if a message has been previously selected.

MAILSAVEREPLY
This function is very similar to MAILREPLY, except that the message being replied to remains in the user's mailbox.

MAILTRANSFER
Omega Point/2 allows users to attach uploaded files to E-Mail messages. This function is called to allow the receiving user to download an attached file.

MAILREREAD
Allows the user to re-read the currently selected message.

MAILCHECK
This function displays the number of waiting messages in the user's mailbox. As delivered, Omega Point/2 calls this function when the user enters the E-Mail section.

MAILENTERPROC
This function is normally attached to the ENTER key. If there is a currently selected message, this function reads the next one.

7.7 Resume/Profile Functions

This small set of functions allows a user to display or customize his "resume" (also called a "profile"). The profile is a free form description of him or herself entered by the user. These profiles may be viewed by other users on the system.

RESUMEMENU

This function calls a small menu system that allow users either to enter or customize their own profile or view that of another user.

RESUMELOOKUP
This function displays the profile for the requested user.

7.8 SysOp Functions

In the course of running your system, you will find the need to diagnose problems, manage user accounts, and monitor performance. These functions allow you to perform those functions and more.

SETUP
Runs the system setup menu. Allows the SysOp to set various system wide defaults.

CONSTAT
Short for Console Status, this function displays a variety of system statistics and information. The header of this function displays the port the console is currently "watching" (if any), the SysOp available status, the version number of the program, the time the program was started, the number of pages sent since the system was started, the largest available memory block, and the thread id of the main thread for the user calling this function. After the header, one line per port is displayed. The following table indicates the meaning of the displayed data:

Field Header	Description of Data
#	Line number
On	Port on/off in PORTS.BBS
Act	Port active and working
Cnt	BPS rate of active connection
Em	Port "emulating" displayed port
Idle	Seconds since last key stroke
Status	Line status
Function	Last MCF function executed

RUN
As described in previous chapters, the RUN function can be used to manually call any of the MCF functions. It is especially useful for calling some of the SysOp only functions which you may not want to have linked to specific commands. For example, you could enter "RUN CONSTAT" at any MCF prompt to run the CONSTAT function. In addition, a SysOp level user can also run MCF functions from teleconferencing. This is done using the "/r"

command. For example, in teleconferencing, you may simply enter "/R CONSTAT" to run the CONSTAT function.

LISTFUNC
Lists all valid MCF function names.

CONMESSAGE
Prints a message in the AUDIT window of the console.

SYSOPCHAT
Pages the sysop to chat. If the SysOp available status is not set to available, an error message will be displayed.

SYSTEMSTATS
Displays various system statistics.

SYSOPUSELOG
Displays the system audit log, which holds information such as who logged on the system, what they did on the system, any errors that occurred, etc. This information is kept in the file LOG.BBS.

SYSOPDELUSELOG
Deletes the system audit log described above.

SYSOPERRORLOG
Displays the system error log. An entry is made in this log when a modem failed to initialize correctly or if the a critical error occurred. The data is logged in the file ERRLOG.BBS. If any errors are logged here, you should copy the file to a diskette and send it to ExcelSoft.

SYSOPDELERRLOG
Deletes the system errorlog discussed above.

SYSOPKILLUSER
This function logs a user off the system.

SYSOPCHGLEVEL
This function changes the level of a user logged onto the system.

SYSOPMDMRESET
Resets a modem manually. This can be used either to reset a modem that failed to initialize correctly or one that was perhaps added to the system after it was started.

SYSOPMDMOFFHOOK
Takes the modem "off hook". Note the modem must be Hayes Compatible for this to work.

SYSOPUSERMOD
Allows the SysOp to modify the personal information and statistics of a user.

SYSOPEDITFILE
Allows the SysOp to edit a standard ASCII text file on the system.

SYSOPPAGEALL
Used to send a "global page" to all users logged on.

SYSOPTIMEDSD
Shuts down the system after a specified period of time. The system will send a "global page" to all users every minute until the shutdown time is reached.

SYSOPRESET
Used to shutdown the system quickly. This function can be used by remote SysOps to reset the system in the event of some hardware error or other problem. If you are using the DOBBS.CMD batch file, it will automatically restart the system.

SYSOPPS
There are two "threads" associated with each user on the system, a main thread and a communications thread. This function lists each of those threads for each port.

SYSOPSUSPENDUSER

This function suspends both threads associated with a given port. Effectively, this "locks" the port.

SYSOPRESUMEUSER
This function undoes the "lock" associated with SYSOPSUSPENDUSER.

SYSOPMESSDUMP
Displays various memory and performance statistics associated with the Message and Mail Systems.

SYSOPKILLPROC
Kills the specified process. The process must be one created by the BBS system. The process ID may be obtained by executing CONSTAT. CONSTAT shows the IDs of any processes and sessions created by the system.

SYSOPKILLSESSION
Kills the specified session. This session must have been created

by the BBS.

SYSOPPORTOFF
This function turns the specified port "off" on the system. The port will no longer answer calls. It will not disconnect any user currently logged onto that port.

SYSOPPORTON
This function turns the specified port "on".

SYSOPSPYON
Has little practical value, but turns the Console "watch" on and sets it to the specified port.

SYSOPSPYOFF
Turns console "watch" off.

SYSOPREMSPY
Used to "spy" on a user while logged onto the BBS. Any keys entered will be pushed into the spied on user's buffer. Pressing the 'key returns the spying user to where he or she left off.

SYSOPSETCALLS
Sets the number of calls on the system. This is provided to allow the sysop to artificially set the number of system calls for whatever reason.

SYSOPDIR
Allows the SysOp logged onto the BBS to list a directory.

Chapter 8

Message and File Forum Management

In a previous chapter, you learned how to use the SETUP command to change several parameters that affect the general behavior of the system. You also learned how to change the command structure of your menus to suite your system's needs. There is only one remaining issue you need to understand to fully customize and tailor your system.

Omega Point/2 supports up to 100 "forums". A forum is a section on the BBS that has either a file library, a message system, or both. Each forum has it's own name and it's own operating parameters.

A user other then the SysOp can be given "moderator rights" to the forum. The moderator of a forum may assign it's name, change the operating parameters, decide which users may use the forum and which specific access flags they will have, manage the library and file sections, etc. Assigning a user as the moderator of a forum can reduce the amount of work the SysOp has to do. Also, many users enjoy being a forum moderator and can bring new experiences and outlooks to your system.

As the SysOp, you may edit individual forums. To do so, go to the message section ('M' at the main menu), change to the forum you wish to edit ('C' from the message menu. Notice that the sysop can change to any forum, even those that are not active), and at the message menu for the forum you wish to modify, enter the command 'EDIT'.

8.1 Basic Options

After entering EDIT at the message menu, you will be presented with a menu of 15 items. The top of the menu will say "Forum Edit" and indicate the forum number you are currently editing. The first 11 options affect the general operation of the forum, while the remaining options are used for maintenance or more specific adjustments. The following list covers the first 11 options:

1. Option number 1 allows you to change the current title of the forum. This is the title that's displayed when the

AreaChange MCF function is used. It can also be printed with the MCF PROMPT command.

2. Number 2 assigns allows you to change the name of the forum moderator. Note that is option is for display purposes only. Therefor, you may put anything you like here, it does not even have to be the name of a user on the system. Putting an actual user name here will not make them the forum moderator or give them any additional rights. Assigning moderator rights is discussed below.

3. Number 3 allows you to assign the level and/or access key needed for "default" forum rights. As we'll discuss below, each forum operation requires that the user have a specific forum flag. The group of all flags is called the user's forum "rights". There are two levels of rights for each user. The two level system allows you to assign low level rights to users that you may not wish to have full access to a forum. For example, perhaps you want new users to only be able to read messages, but not write them. The format for this field is the same as for the access field of MCF files. For example, you can enter just a simply access level (such as "20"), or you can make access dependent upon both an access level and a particular access key ("20 and 3", for example).

4. Option number 4 works the same as option 3. User's that meet the user level and access key requirements of this option, however, are only assigned low-level access rights. Just what these rights are and how they're assigned are covered in more detail below.

5. Option number 5 sets the download directory for the forum. This is the directory where uploads are saved in and where downloads are taken from. Note that in order to insure adequate system security, only the actual SysOp may modify this field. Forum Moderators may not. If the string entered here is not a valid and existing path name, Omega Point/2 will issue an error message when started. Notice that this field will never have any effect as long as the "File System" is not turned "on".

6. Number 6 is a flag that indicates whether or not uploaded files are automatically "verified". The term "verified" simply means that the file data (it's name, size, descriptions, etc) will be viewable by other users. This option is only effective if the "File Directory Mode" (covered below) is set to "Database".

7. Option number 7 is used to toggle the message system "on" and "off". When the message base is "on", users that meet either the low or high level access requirements may user the message system. The functions that may be performed within the message system, however, are affected by which

set of access rights the user has been assigned (high or low).

8. Option number 8 is used to toggle the file library "on" or "off". The file library is where all files that belong to this particular forum are stored. The file system may operate in one of three "modes" (covered below). Like the message system, the actions each user may perform are determined by his access rights, which are assigned based on his access level and access keys.

9. Option number 9 toggles the "private" flag "on" or "off". Private forums are not much different from public forums. If a forum is private, the level needed for default access and low-level access fields are ignored. Instead, the SysOp or moderator must assign the desired access flags to each user specifically.

10. Option number 10 is used to specify whether or not messages may be posted anonymously or not. If this field is "yes", then whenever a user writes a message in this forum, he or she will be asked if the message should be posted anonymously or not. If he or she selects "yes", then readers will not be shown the name of the writer. The SysOp and forum Moderator, however, will still be able to see who posted the message.

11. The last of the basic options allows the SysOp to set the file directory mode. Omega Point/2 supports three modes. The mode affects what information is stored for each file uploaded or otherwise added to the system and how it is displayed for any user operating in the file library for this forum. The first type is "Database". When this mode is selected, any user who uploads a file will be asked to enter a short and long description. This information will be stored in a database. When users scan the file library for the forum, they will be able to view this information or search for certain key words in the data. The second option, "Raw", simply displays the raw OS/2 file directory when a user selections one of the scanning MCF options. This option is particularly useful for "SysOp or staff only" forums used for exchanging semiprivate information. The final type is "File". When a user selects one of the "list" or "browse" functions in a file library with "File" mode, a particular text file is displayed. The text file displayed is either "FILEBROW" or "FILELIST", for either a "list" or "browse" function. The extension ".ANS" is used for users with ANSI terminals, and ".MSG" for users with no terminal type. If the ".ANS" file does not exist, the ".MSG" file is used. The file is created by the SysOp or moderator and placed in the "Download Directory" for this forum. For example, if the Download Director is set to 'E:\FILE1', the user selects "browse" at the FILEMNU MCF, and the user has selected ANSI emulation, the file

'E:\FILE1\FILEBROW.ANS' is displayed. If that file doesn't exist, or the user does not have ANSI emulation selected, the file 'E:\FILE1\FILEBROW.MSG' is displayed.

8.2 Message Management Menu

By selecting option number 12 on the Forum Edit menu, you enter may enter the Message Management and Configuration Menu. This menu has three options that let you tailor the functionality of the message system.

1. The first option allows you to set the maximum number of active messages in for this forum. After the maximum number of messages has been reached, the first active message posted in the forum will be deactivated each time another new message is written. By adjusting this option, you can make your forums self circulating. This lowers the maintenance time for the forum, since old messages will automatically be removed from the system.

2. The second option is the "autopack" toggle. Making this option equal to "yes", tells the system to check the number of messages in the forum when the BBS is started. If the number of messages in the system has exceeded the maximum allowed (option 1), then the first active message will be moved up so that there are a certain number of free "slots". The number of slots the system creates is set with option number 3. Note that the autopacker does not purge deleted message. For that you should use the MESSPACK utility. Generally speaking, the MESSPACK utility should be used over the autopack option.

3. The final option sets the number of free slots the system creates when it performs an autopack. This option is relevant, because the system incurs more overhead when it has to "shift" messages to create a free slot. This number should be approximately equal to the number of messages you would expect to have posted in this forum per day.

8.3 Adjust Access Rights for a User

Selecting option number 13 from the Forum Edit menu brings you into another menu. This menu allows you to edit the access rights for the specified user. After entering the name of the user you

wish to modify, you are presented with a menu that looks something like this:

```
          User          Flag    Forum Def.    Flag
          ------------  ----    ------------  ----
     [ 1] Mess See      Yes     Mess See      Yes
```



```
     [ 2] Mess Read     Yes     Mess Read     Yes
     [ 3] Mess Write    Yes     Mess Write    Yes
     [ 4] Mess Sysop    No      Mess Sysop    No
     [ 5] File See      Yes     File See      Yes
     [ 6] File List     Yes     File List     Yes
     [ 7] File D/L      Yes     File D/L      Yes
     [ 8] File U/L      Yes     File U/L      Yes
```

This table shows a list of 8 flags. This set of flags together are called the user's "rights" for this forum. Each user has a set of flags for each forum. If the forum is Private (see above), each flag can be modified individually. If the forum is not private, then the flags will be assigned to the user for each forum when the user logs in. What flags the user gets is dependent upon the user's access level and access keys. These two things are compared against option numbers 3 and 4 (above) to determine whether the user has "default" access, "low-level" access, or no access at all.

The one exception to this access scheme is the moderator flag. If the 4th flag (Mess Sysop. Referred to as the "Moderator" above) is set for a user, the user's rights will not be changed, no matter what access keys or level he or she has. This way, you can assign a specific moderator and not have to concern yourself about his or her actual access level.

At this menu, you may select any one of the flags to change (1-8), 'D' to assign the default flags to the user, 'Q' to quit, or 'S' to quit and save any changes you have made.

The purpose for each of the 8 flags is as follows:

- Mess See: Allows the user to "see" the Message System. If the user does not have "Mess See" rights, then the forum name will not be shown in the Area Change menu and the user will not be able to enter this area. Basically, having this flag allows the user to enter the message system for this forum (assuming the message system is "on").

- Mess Read: Allows the user to perform any of the MCF functions that read messages in this forum. For example,

ReadForward, ReadReverse, etc.

- Mess Write: Allows the user to write messages in this forum. Note that this includes replying to messages as well.

- Mess Sysop: A user with this flag is called the forum "moderator". The moderator may perform any action in the the forum for which he has moderator rights. The user has no special abilities outside of the forum for which he or she has moderator rights. The moderator may, in addition,

perform any of the Forum Edit functions (with the exception of changing the download directory).

- File See: This flag is similar to the "Mess See" flag discussed above, with the exception that it applies to the file system. A user who does not have this flag set, will not be able to see or attach to this forum.

- File List: This is the equivalent of "Mess Read" for the file system. Any user that does not have this flag will be unable to use any function that displays file lists (such as MessBrowse).

- File D/L: A user with this flag set will be allowed to download files from the file library associated with this forum.

- File U/L: A user with this flag will be allowed to upload files to the file library associated with this forum.

8.4 List Users with 'See' Rights

This function scans the user list and prints the name of any user that has his or her "Mess See" flag set for this forum. This amounts to a list of any user that can attach to this forum.

8.5 Adjust Default Forum Rights

Selecting option number 15 from the Forum Edit Menu allows you to adjust the default flags assigned to any user that matches either the "Level needed for default access" or the "level needed for low-level access". When this option is selected, you are

presented with a list of 16 options. You may toggle each of these options on or off. The first 8 are the flags assigned to a user that meets the requirements for "default" access, while the upper 8 are those assigned to a user who meets only the requirements for "low-level" access.

It's important to remember that if the forum is 'Private', then these flags have no effect. The user's rights remain at what the SysOp or moderator set them at with option number 13. Also, if a user has his "Mess Sysop" flag set, he will not be affected. Likewise, any user with SysOp level access (99) can do anything in any forum so it makes no difference what his or her flags are set to.

8.6 File System Management

In additions to the menu options discussed above, Omega Point/2 also provides tools for managing the file system libraries associated with each forum. You may enter the file system by using the 'F' command from the main menu. From the file system menu, the SysOp for the forum moderator may enter 'EDIT' to get to the file system management menu.

There are six options on this menu, numbered 20 through 25. The function of each of these options is as follows:

1. The first option, number 20, allows the SysOp or forum moderator to list the files that have been uploaded to this library but have not yet been validated. This option is only relevant if the "File Directory Mode" (see above) is set to "Database". Also, if "Auto-Verify" is turned on for this forum, all uploaded files will be automatically set to "validated", and therefor will not appear in this list. The term "validated", as it applies to Omega Point/2, simply means that the file will appear in any of the "list" and "browse" displays invoked by other users.

2. The next option, number 21, is used to toggle the validated flag for a file. When you select this option, you will be asked to enter a file name. The file name must be in the database for this file system. Like the previous option, this one is only valid in forums with a "File Directory Mode" of "Database". After entering a valid file

name, the file's validated flag will be toggled to either on or off.

3. Option number 22 allows the SysOp or moderator to remove a file from the file system database and/or delete the file from the download directory. Simply select this option and enter the name of the file you wish to remove. You will then be asked if you wish to delete the database entry, the actual file, or both.

4. Option number 23 provides the SysOp or moderator with the ability to add database entries for files that have been manually placed in the download directory of the forum. The SysOp might, for example, download files from another BBS and copy them into the download directory for a given forum. After which, he or she would use this option to create entries in the file system database for them. This option is only valid in forums that use the "Database" file mode.

5. Option 24 allows the SysOp or moderator to rename a file in the download directory for this forum.

6. And finally, option 25 provides the ability to display a the directory contents of the download directory.

Chapter 9

Using Teleconferencing

On-line multiuser teleconferencing allows several users to all simultaneously, and in real time, carry on conversations with one another. While most options and commands used by Omega Point/2 are initiated by Menu Configuration Files, functions within the teleconferencing system are not. As delivered, the "T" command is used to enter the teleconferencing system. The following section describes the commands that are available within

teleconferencing:

9.1 Teleconferencing Commands

In teleconferencing, users enter lines, a character at a time. when they press ENTER, the line they entered is sent to the other users in the "room" (the particular area of the teleconference). In addition to this standard conversation, there are several special commands that may be used inside teleconferencing. All commands in the teleconferencing system are preceded by the "/" character. This tells the system that the rest of the line is to be interpreted as a command. The following commands are available:

/Q
Quits the teleconferencing system and returns to the MCF the user entered teleconferencing from.

/X
Does the same thing as /Q

/H
Display the teleconferencing help menu.

/?
Does the same thing as /?

/U
Displays a list of the other users in the room. In addition, this shows who is "moderating" the room and if the room has any special attributes.

/W

I used to display the users currently online. Works the same as the MCF function, USERSON.

/P [user] [message]
Sends a private message to a user. If the user is in the room, you name followed by "PRIVATE" will precede the message. If the user is not in the room, a page message will be sent to him or her. For example:

```
/p john Hi, how was the movie?
```

/A
Toggles the room in and out of "Anonymous" mode. When the room is anonymous, instead of the user's name preceding each message, the text "[??]" will be used instead.

/M [conference name]
Moderate the current conference. A user may become the moderator of any conference which does not already have a moderate, except room 1. Other users may enter a moderated room only at the moderator's invitation (except a SysOp level user; They may enter any room). Example:

```
/m This is my room!
```

/I [user]
Invite a user. This command allows the moderator of a room to "invite" a user to his or her room. The invited user will receive a page message indicating an invitation has been extended. The user may then enter the room at will. In addition, the moderator may use "all" for a username. Doing this will make the room available to anyone.

/EJECT [user]
This command can be used by the moderator of the room to forcefully eject a user from the room. It may also be used by SysOps, who need not be the moderator.

/J [room]
Join another teleconferencing room. Note that the user will not be allowed in a moderated room unless he or she has received an invitation. Example:

```
/j 2
```

/S
Scans the teleconferencing rooms. This command shows which users are in what rooms.

/Y [user] [message]

This command sends a page message to the indicated user. It is similar to the /P command, except that it always sends a page, even if the receiver is in the current teleconferencing room.

/C [user]
May be used to request chat with another user. When in chat, each

key typed by each user is displayed for both users. This form of conversation is more interactive then "standard" teleconferencing and is normally used for more involved discussions. When a user requests chat, the receiving user receives a page message telling him or her that they have been asked to chat. To enter chat, the receiving user simply enters "/C".

/E
Toggle message echo. By default, whenever a user types a message in teleconferencing, it is echoed back to the user. This may be suppressed by enter "/E". This "echo state" is saved from login to login.

/L [user]
Look-up a user's resume. This command allows a user to view the resume, also called the profile, of another user.

/FORGET [user]
Occasionally, one user may find another annoying or bothersome. The resolution for such situations is the forget command. Using this command will squelch out all pages and private messages from the forgotten user.

/REMEMBER [user]
This command undoes the effects of the forget command.

9.2 SysOp Only Commands

There is one command that only be run by a user with SysOp level access. The "/r [function]" command lets a SysOp run MCF functions from teleconferencing. For example, entering:

```
/r constat
```

would execute the console status function, listing the status of each incoming line in the system. Likewise, entering:

```
/r sysopkilluser
```

would allow the SysOp to force a user off the system.

9.3 Chat Commands

In a previous chapter, the 'chat' function was discussed. There are several commands that may be used within the 'chat' section. These are as follows:

/Q
Quit chat and return to teleconferencing.

/X
Same as /Q.

/H
Display the list of commands available from 'chat'.

/?
Same as /H.

/P [user] [message]
Virtually identical to the page command in teleconferencing, this command sends a page message to the requested user.

/W
Same as the /W command in teleconferencing, this command lists the users currently online.

Chapter 10

Display Menu Files

Depending on the scope of your system, you may have users call your system using a variety of different computers and screen types. The current version of Omega Point/2 support two types of displays, either "ANSI" or "none". By default, Omega Point/2 asks the user to select a terminal type when he or she first signs up. After that, the user may change terminal types by using the ChangeYourSettings MCF function (assigned to the 'Y' command on the main menu, by default).

The term "ANSI" refers to the terminal emulation standard supported by nearly all DOS and OS/2 compatible machines. When the user selects "ANSI", the program will send the correct ANSI standard codes to do things such as change colors, clear the screen, position the cursor, etc. In order for these codes to be interpreted properly by the user's system, he or she must have "ANSI emulation" turned on. How this is done is dependent upon the type of software in use.

The term "none" implies that the user does not have ANSI terminal emulation, and therefor can not display color and can not process any of the other ANSI sequences (such as cursor positioning). The only terminal control Omega Point/2 attempts to do with users who have "none" for their terminal type is clear the screen. Omega Point/2 does this by sending the ASCII character 12 (form feed). This may not work on all terminals or with all communications software.

In the section on Menu Configuration Files, you learned how to use the PROMPT command to set the current color for text displayed within an MCF. In addition to the text displayed by the MCF file itself, there are also two Menu Display Files associated with each menu system. One menu is for ANSI terminals and the other is for terminal type "none". The following table lists the file names for each Menu Display File:

MCF File	ANSI Menu	Normal Menu
MAINMNU	MAINMNU.ANS	MAINMNU.MSG
MAILMNU	MAILMNU.ANS	MAILMNU.MSG
MESSMNU	MESSMNU.ANS	MESSMNU.MSG
FILEMNU	FILEMNU.ANS	FILEMNU.MSG
NEWSMNU	NEWSMNU.ANS	NEWSMNU.MSG
GAMEMNU	GAMEMNU.ANS	GAMEMNU.MSG

AUX1MNU	AUX1MNU.ANS	AUX1MNU.MSG
AUX2MNU	AUX2MNU.ANS	AUX2MNU.MSG
AUX3MNU	AUX3MNU.ANS	AUX3MNU.MSG
Tele init	TELEINI.ANS	TELEINI.MSG
Tele help	TELEMNU.ANS	TELEMNU.MSG

Whenever a user enters a new menu system, the appropriate menu file will be sent and displayed on the user's screen. If the user has selected ANSI emulation, the ANSI menu will be sent. If the user does not have ANSI selected, the normal menu will be sent. If the user has selected ANSI emulation, but the ANSI menu does not exist in the current directory, the normal menu will be sent (assuming, of course, that it exists). No menu will be displayed when changing menu systems if the user has selected "expert" mode.

The teleconference system menus represent a special case. The first teleconference menu, TELEINI, is displayed the first time the user enters the teleconferencing system. The other, TELEMNU, is displayed whenever the user users the teleconference help command (/h or /?).

As you can see from the table above, the text that is displayed on the user's terminal is contained in a simple OS/2 file. These files may be edited using any standard ASCII editor (including the OS/2 system editor). Additionally, you may find it helpful when editing the ANSI files to use a tool designed for that purpose. ExcelSoft used a shareware DOS product called TheDraw to create the default ANSI menus.

Creating varied menus is a great way to make your system stand out from others.

10.1 Other Files

In addition to the menu files described above, there are several files that are displayed at various times. Like the menus discussed above, the following files may also be edited with any standard ASCII text editor. These files are as follows:

INIT.MSG — This file is the first thing displayed when a user first logs onto the system.

LOGIN.MSG — This file is displayed after the user enters his or her username and password.

NEWUSER.MSG This file is displayed when a new user logs onto the system and asks to register.

Chapter 11

External Utilities

The Omega Point/2 package includes several external utilities. This set of programs allows you to better manage your system and the data it produces. For example, you should use the MESSPACK utility frequently to keep the amount of disk spaced used by your system to a minimum. The USERLIST utility can help you keep tabs on who is using your system. It creates a listing of all the users on your system in either a "human readable" format, or in a format that can be read in by your favorite spreadsheet or database program.

11.1 BBSSEND - Send Requests to the System from other Sessions

Usage: BBSSEND [-h] [-i=n] [-p=n] {-f -d -s[=n]}

BBSSEND can be used from other OS/2 full screen or window sessions to affect or alter the operation of the BBS. It uses an OS/2 Named Pipe to send a message to the running Omega Point/2 BBS. This program is best used from CMD files or called from other programs. It can be used to perform certain actions, such as shutting down the BBS, on a timed basis. This utility should only be executed while the BBS is running.

Options:

-h Displays the help screen for this program.

-i=n This parameter is optional. It is used to specify that the message should be sent to another

instance of Omega Point/2. If not specified, BBSSEND defaults to 0, or the first instance of the system executed.

-p=n This parameter is need if either the -f or -d options are used. It allows the caller to specify which port the -d or -f option is to affect.

One, and only one, of the following options must be specified when executing BBSSEND:

-f "Freezes" the port specified with the -p option.

The port will be unable to answer calls. The device driver for the port will be closed so it may be used by another program.

-d "Defrosts" a port previously frozen with the -f command. This function reopens the port so that it may once again answer calls and otherwise be used by the BBS.

-s[=n] This command shuts down the BBS. If 'n' is specified, the BBS will be shut down in 'n' minutes, warning all users online each minute until the shutdown occurs. If 'n' is not specified, 0 is assumed and the BBS will be shutdown within 30 seconds of issuing the command.

11.2 MESSPACK - Message Packing Utility

Usage: MESSPACK [-h] [-a=n] [-f=n]

As people use your BBS, messages will be added and deleted frequently. Since messages are stored in a set of two files for each forum, the system can not free the disk space used by deleted messages without rewriting the message files. That is what MESSPACK does, rewrites the message files so that no disk space is taken up for deleted messages. It can also be used to purge messages that have been in existence past a specified period of time. In order to insure that your system is running at it's best, it is imperative that you use this utility on a regular basis, perhaps once a week.

Options:

-h Displays the help screen for this program.

-a=n Used to purge messages older then 'n' days in addition to those that have already been deleted.

-f=n By default, MESSPACK scans each of the active message forums, purging messages as it does. You may, however, wish to purge the messages for only a specific forum. For example, it is a good idea to pack E-Mail messages more frequently, since they are typically read by the receiver and then deleted. Use the -f option to specify a particular forum to purge. Use '0' for 'n' to purge E-Mail, or the number of the forum, 1-99.

11.3 USAGE - Export Per/Call Statistics

Usage: USAGE [-h] -m=mon [-l=name]

The USAGE utility is used to export a one line summary of each call received by the system for a particular month. A separate binary log file is stored by the system for each month. The files are stored in the RUNBBS directory and use the following naming convention: AUDITxxx.BBS, where 'xxx' is a three character month abbreviation (JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, or DEC). This program reads the specified log file and creates an ASCII text file that can be read in by most spreadsheet and database programs. The text fields are enclosed in quotes while the numeric fields are not. A total of 12 fields are output for each line. These fields are summarized in the following table:

Field	Description	Type
1	Users name	Alpha
2	Port logged in on	Numeric
3	User's sex	Alpha (M or F)
4	User's age	Numeric
5	E-Mail written	Numeric
6	Public messages written	Numeric
7	Files downloaded	Numeric
8	Files uploaded	Numeric
9	Time logged in	Alpha Numeric

10	Time logged out	Alpha Numeric
11	Minutes online	Numeric

Notice that the purpose of this utility is not to provide human readable reports, but rather to give you the tool to produce your own specific reports using a separate tool (database or spreadsheet) more appropriate for the task.

Options:

-h Displays the help screen for this program.

-m=mon The program requires that you specify the 3 letter abbreviation of the month for which you wish to generate a report. The acceptable abbreviations are listed above.

-l=name By default, the program outputs the report to a file called USAGERPT.LOG. If you would like the output to be sent to another file, specify the desired filename in 'name'.

11.4 USERLIST - Generate a Listing of All Users

Usage: USERLIST [-h] [-r=n] [-l=name]

The USERLIST program outputs a list of all the users that have signed up on your system. The output can be in one of three forms, two of which are human readable and one which can be read into your favorite spreadsheet or database. The two human readable reports are in columnar format. The contents and order of these columns are detailed in the following table:

Default Report:

Column Header	Description
Num	User sequence number
Handle	The user's name on the system
Name	The user's real name, first and last
Age	The user's age in years
Sex	The user's sex, 'M' or 'F'
Lev	The user's access level

Last On	Date last logged on the system

Report Number 1:

Column Header	Description
Num	User sequence number
Handle	The user's name on the system
MLI	Minutes logged in this month
MOL	Times logged on this month
LOG	Total number of logins
MIN	Total number of minutes online
AVG	Average minutes per call
TUL	Total files uploaded
TDL	Total files downloaded
Bytes UL	Total number of bytes uploaded
Bytes DL	Total number of bytes downloaded
Mess	Total public messages written

The next report is written in spreadsheet importable form. Fields are delimited by commas, and text fields are enclosed by quotes while numeric fields are not.

Report Number 2:

Field	Description	Type
1	User's handle	Alpha
2	Password	Alpha
3	Age	Numeric
4	Sex	Alpha (M or F)
5	First name	Alpha
6	Last name	Alpha
7	Computer type	Alpha
8	Address	Alpha
9	City	Alpha
10	State	Alpha
11	Zip code	Alpha
12	Phone number	Alpha
13	Time allowed per day	Alpha
14	Time allowed per month	Alpha
15	Deleted flag	Alpha (Y or N)
16	Access level	Numeric
17	Access keys	Alpha
18	Expert mode	Alpha (Y or N)

19	Terminal type	Numeric
20	Screen size	Numeric
21	Last login	Alpha (date)
22	Total time online	Numeric
23	Total logins	Numeric
24	Total uploads	Numeric
25	Total downloads	Numeric
26	Total size uploaded	Numeric
27	Total size downloaded	Numeric
28	Total message sent	Numeric
29	Account active since	Alpha (date)

Options:

-h Displays the help screen for this program.

-r=n If this option is not specified, the program prints the default report. Otherwise, 'n' may be either 1 or 2 for the optional report to print.

-l=name By default, USERLIST writes it's report to the file USERLIST.LOG. If you wish to send the output to another file instead, specify the desired filename in 'name'.

11.5 USERPACK - Packs the User Database

Usage: USERPACK [-h]

USERPACK falls into the same category as MESSPACK. It should be run on a regular basis to insure that your system is functioning at it's peek. This program simply purges any user records that have been deleted by the sysop.

Options:

-h Displays the help screen for this program.

Chapter 12

Programming Guide

12.1 Introduction

The Omega Point/2 system includes a set of tools designed to help you create add-on programs or modules for your BBS. With the tools you can either create stand alone programs or DLL modules that are loaded on demand, or every time the system is started. Add on programs can be attached to virtually any menu command on any menu and run virtually seamlessly with the main system. This functionality is accomplished by taking full advantage of OS/2's powerful features and far surpasses the capabilities of other systems.

12.2 System Requirements

To create add-on modules, you need only this toolkit and the Microsoft 'C' Compiler. Version 6.0 is recommended, since it offers far better OS/2 support then did version 5.1. It may be possible to use other compilers, but none have been tested.

12.3 Setup

The following include files can be found on the distribution diskette:

BBSEXPAN.H	Main include module for expansion programs. Simply include this at the top of your code. This file will include the other files as needed.
OS2BBS.H	Contains all of the BBS typedefs and #defines that will be needed by your programs.
BBSAPI.H	Contains the prototypes for the BBS Expansion functions, which are loaded at runtime from the DLL file; BBSAPI.DLL.
OS2VER.H	Contains version #defines and a #define for the maximum number of ports your system is configured for.

In addition, the file BBSEXPAN.C contains several routines that you will need. The most important is UseChildAppInit(). You'll need to call this at the top of your code in order to properly initialize your application. One final file, BBSAPI.LIB, is an export library for the BBSAPI.DLL library. This should be linked in with your application.

12.4 Building a Simple Application

This section will describe how to make the famous "Hello World" program into a BBS expansion application. Enter the following code:

==========

```
//
//  hello.c
//  The hello world program as a BBS expansion //
#define INCL_DOS
#include <os2.h>    // this includes OS/2 stuff, comes with MSC

#include <stdio.h>
#include <stdlib.h>
#include "bbsexpan.h"    // BBS expansion include file

main(int argc, char *argv[])  // Start here!
{
   short rc;

   rc = UseChildAppInit(argv[1],argv[0],&anchor,
              &unhand,&semhand,&usernum,&instance);
                        // this is a standard call to
                        // initialize all applications
   if( rc != 0 )        // check for initialization error
      return(1);

   COLOR(BRWHITE);      // Make the color bright white
   SerWritef(unhand,"Hello World!");
                        // Print our string..
   exit(1);
}
```

And that's all there is to it. Simply include the BBSEXPAN.H header, and insert the call to UserChildAppInit(). Beyond that, all that need be done is be sure all I/O is done using the functions detailed below, and included in the BBSAPI library. Now let's create the makefile for this program:

```
#
#  Makefile for hello.exe
```

```
#
OPT=-MT -c -Zp1 -Gs -Os -WX -DEXPAN  # standard compiler command
LNK=/NOD /NOI

HELLO.EXE: hello.obj bbsexpan.obj
   link $(LNK)
hello+bbsexpan,hello,hello,llibcmt+os2+bbsapi,hello;

HELLO.OBJ: hello.c                   # your program file
   cl $(OPT) hello.c

BBSEXPAN.OBJ: bbsexpan.c bbsexpan.h  # expansion routines
   cl $(OPT) bbsexpan.c
```

One final file, you'll need a linker definition file. Definition files are very simple for BBS expansion programs, and the one for our program should look like this:

```
;------------------------------------------------------- ;
HELLO.DEF  module definition file
;-------------------------------------------------------

NAME        HELLO    WINDOWCOMPAT

DESCRIPTION 'Omega Point/2. Copyright (c) 1990 ExcelSoft
Software' PROTMODE
STACKSIZE   6144
```

Now simply enter the command "nmake hello.mak". When you're finished, you should have a program called HELLO.EXE. This is a finished, working, BBS expansion module! To attach it to a BBS menu item, insert the following line in your MAINMNU.MCF:

```
"HELLO",PRO,0,"",HELLO.EXE
```

Load the BBS, log in, go to the main menu, and type "HELLO". You should see "Hello World" on the screen.

12.5 Components of the BBSAPI

The BBSAPI is a set of functions that can be called by programs or DLLs that are external the main BBS. By using these functions, a consistent and manageable interface is established. With this interface, all interaction with the user, both from his point of view and that of the system, is consistant. For example, if a user in the teleconferencing section sends a page to user 2 running an external module, user 2 will receive the page. No special action or code needs to be implemented by the programmer of the module to take advantage of this. Likewise, if the sysop

is "spying" on a user, and that user runs an external module, the "spying" continues just as if the user was still in the main module. With few exceptions, all interaction between external programs and the main system is handled seamlessly by the BBS API.

The code for the API functions is distributed in DLL (Dynamic Link Library) format in the file BBSAPI.DLL. This library handles all of the input and output between users and the BBS. Any input or output your expansion programs do should always be through calls to this API. While it is possible to write programs with redirected STDIN and STDOUT or that run in separate sessions, they are inferior to those built upon the BBS API.

There are simple conventions that should be followed in order to insure that your application runs in harmony with the main BBS. The following text will describe these while providing an overview of the library.

12.6 Programming Considerations

As we demonstrated with HELLO.EXE, it is a straight forward process to create and run expansion modules. Two simple rules apply in setting up your application, first include the header file BBSEXPAN.H at the top of every 'C' module that will be calling any BBS API function. Next, at the very start of your main() function, initialize your application with a call to UseChildAppInit(). This function call sets up several shared memory segments and semaphores used to communicate between the main BBS and your module.

When UseChildAppInit() is called, five key variables are initialized:

anchor	A base pointed to the dynamic user information segment
unhand	A pointer to the entry in the user information segment corresponds to the user executing your program. This value is passed to most BBS API functions.
usernum	This is the user number that is executing your process.
instance	This is the instance of the main BBS program. It

	is possible for several copies of the BBS to be running, and this value indicates from which one your program was started.
semhand	A system semaphore handle used to signal the termination of the expansion module.

Of these variables, 'unhand' and 'anchor' are the most important. If you read the BBSEXPAN.H file, you'll notice they are both of type PORT_REC. This data type is defined in OS2BBS.H. A great deal of information is stored in this structure. For example, you'll notice one element of PORT_REC is 'p_user', of type USER_REC. This will contain all of the user information of the user calling your module. USER_REC is also typedef'd in OS2BBS.H. If you wanted to print the name of the person calling your program, you could do this:

```
SerWrite(unhand->p_user->u_handle,unhand);
```

The function SerWrite() takes two arguments, a pointer to the string to print (unhand->p_user->u_handle) and a pointer to the information segment belonging to the user who you wish to write the information to (unhand). If you look at the bottom of OS2BBS.H, you'll notice that there are several #defines to help simplify your code. One such #define is 'PU'. Using PU, we could shorten our example above to:

```
SerWrite(PU->u_handle,unhand);
```

Again, most BBS API functions take a pointer to PORT_REC as their last argument. This tells the BBS API functions where to send the output or read the input. By calling UseChildAppInit(), the variable 'unhand' of type PORT_REC is set to the information segment for the user calling your application.

This manual will not attempt to describe the application of each data type specified in OS2BBS.H. This include file is well commented, and the contents of most variables should be fairly self explanatory. Most applications will not need variables other then the user specific ones contained in the USER_REC structure. There should be little reason to modify other variables, other then through the BBS API functions.

12.6.1 Input Loops

In an online application such as a BBS program or an expansion module for a BBS program, it is very important to have the ability to detect error conditions and back out of your program

gracefully. For example, while it might be acceptable to wait forever for a user to enter a keystroke on a local console, waiting "forever" for a key to be sent over a serial port could be disastrous. The user could easily disconnect, and leave the program waiting forever. Fortunately, the BBS API provides the tools to avoid this run-on situation.

All input operations should be done in a loop. The function SerGetCmd() reads a command of a specified maximum size from the specified user. This function can, however, return an error code for a number of reasons. First, the user could disconnect; second, the user might take to long to input the text and the

function will have to return so the application can take action if needed. There are other possible conditions, such as the user receiving a page, but the consequences are the same. If the function returns an error code, you need to be sure it's okay to continue waiting for data, or if you should "error out" and return yourself. You should check things such as the time the user has left on the system, if he's still connected, and if he's been "kicked off" by the sysop or some other action. In OS2BBS.H is a #define called 'LoggedIn()' that does all this for you. The following sniplette of code is taken from TTT2.C, a 3-d tic-tac-toe demonstration program:

```
while( LoggedIn() ) {      // check user up here
   SerColorWrite("\n\rDo you want the rules?"
                 ,BRYELLOW,unhand);
                           // write a query
   COLOR(BRCYAN);          // change to a good input color
   if( SerGetCmd(buf,2,unhand) >= 0 )
      break;
}
if( !LoggedIn() )
   return();
....
Process Input
....
```

Most input functions return the number of characters read, or a negative number to indicate some sort of error. If the user hits only ENTER, 0 is returned. Therefore, in the example above, the loop waits until the user enters data or hits return and then breaks out. After every few seconds without any user input, SerGetCmd() will return an error, at which time the the macro 'LoggedIn()' is run to be sure the user is still online. Your program should exit if this condition is false.

Chapter 13

Programming Reference

13.1 Introduction

This section will describe the functions that make up the BBS Application Programming Interface. There are several groups of functions, each will covered seperatly.

13.2 "Ser" Functions

Name:

```
SerCheckPause(PORT_REC *unhand);
```

Description:

Checks to see if the user has requested a pause in the current output stream.

Remarks:

This function checks for a key in the ports input ring buffer. If one exists, it tests the character. If the character is a space, an 's' or an 'S', the function returns a 1. If the character in the buffer is a 'P' or a 'p', SerCheckPause() calls SerPause(), which prints a message similar to: "..More[Y,n].." and waits for a key stroke. If 'n' is entered or the function times out, SerCheckPause() returns a 1.

Return Value:

Returns a 1 to indicate that the user has requested that the current stream of output be halted, else it returns a 0.

Name:

SerClearQueue(PORT_REC *unhand);

Description:

This low level function executes an OS/2 system call to clear the device driver input queue for the port.

Remarks:

This is a low level function, it simply request's that the Operating System maintained device driver queue be cleared. It does not clear characters that may already be waiting in the BBS input queue. This is used primarily before and after a binary protocol file transfer.

Return Value:

This function always returns a 1

Name:

SerColorWrite(char *str, int color, PORT_REC *unhand)

Description:

Prints a string to the user in the requested color

Remarks:

Prints 'str' to the user in the color 'color'. Valid values for 'color' are specified in OS2BBS.H and include WHITE, BRWHITE, GREEN, BRGREEN, CYAN, BRCYAN, etc.

Return Value:

This function always returns 1 unless the line counting is on (variable "lcounton" in PORT_REC) and the maximum number of lines for the screen has been reached ("lcount" in

PORT_REC stores the number of lines), in this case the function SerPause() is called and 0 is returned if the user indicates he wants no more output.

Example:

```
SerColorWrite("Hello World!\n",BRGREEN,unhand);
```

Writes "Hello World" followed by a line feed in Green.

```
SerColorWrite("Background colors!\n",BRWHITE | BACKGREEN,
              unhand);
```

Notice that background colors can be selected by performing a bitwise 'or' on the background and foreground value.

Name:
SerComClose(USHORT handle);

Description:

This is a low level function that closes the communications port handle, 'handle'.

Remarks:

It is unlikely that any external program will every have a need to call this function and is used internally by the BBS API.

Return Value:

Returns the value returned by the OS/2 system call, DosClose()

Name:
SerComOpen(short *err, char *name);

Description:

This function opens the device specified in 'name'.

Remarks:

It is unlikely that any external program will every have a need to call this function and is used internally by the BBS API.

Return Value:

This function returns the handle opened as returned by DosOpen(), or 0, in which case *err will contain the OS/2 error code.

Name:
SerConnected(PORT_REC *unhand);

Description:

Tests to see if there is a user connected to the specified port.

Remarks:

This function simply tests the status of the Carrier Detect signal on the serial line. It always returns 1 if the port is port 0 (the console).

Return Value:

Returns 1 if there is a user connected or the console is specified, 0 if there is no user connected.

Name:
SerGetCmd(char *str, int max, PORT_REC *unhand);

Description:

Reads a command string from the user.

Remarks:

This function capitalizes the input string. This function also parses the input string into multiple commands if either a space or a ';' delimiter is found in the string. If you would like string capitalized as the user entered it, call SerGetiCmd(). If you would like an unparsed string, call SerGetStr();

Return Value:

Returns the length of the string entered or a negative value on some error. An "error" can occur because the user dropped carrier, received a page while the system was waiting for input, or the system timed out. Appropriate action should be taken by the application (i.e., test for carrier, loop and call the function again, etc).

Example:

```
SerGetCmd(str,10,unhand);
```

User enters "this;is;a;test", str would contain "this", the next call to SerGetCmd() would fill str with "is", etc. If there are no delimiters in the text, the full string is returned. Notice that function will return immediately, without without waiting for input, if there is already input buffered.

Name:

```
SerGetColorField(char *cmd, int max, short color,
                         PORT_REC *unhand);
```

Description:

Retrieves a user input string from a field painted in 'color'.

Remarks:

If there is no input buffered, this function paints a field of size 'max' in a color 'color' (see SerColorWrite() for more information on colors). All characters input will be in the specified color combination. This function is useful for inputting fixed length fields where using a background color other then the standard would produce a useful effect.

Return Value:

Returns the length of the input string, or a negative value on error.

Name:

```
SerGetField(char *cmd, int maxlen, PORT_REC *unhand);
```

Description:

Gets a field of size 'maxlen', stopping input at 'maxlen' characters.

Remarks:

This function is very similar to SerGetCmd() and related functions. The difference being that if no input is buffered, and the function has to take input directly from the user, it will halt input beyond 'maxlen' characters. Other functions allow for longer input and only return a string of the length specified. This is to allow for the stringing of commands.

Return Value:

Returns the length of the entered string or a negative on error.

Name:
SerGetModemResp(int timeout, PORT_REC *unhand);

Description:

Waits for a response from the modem.

Remarks:

This function is used internally by the BBS API.

Return Value:

Returns the value of the modem return code, or a negative on error.

Name:
SerGetStr(char *str, int maxlen, PORT_REC *unhand);

Description:

Get an unmodified, unparsed string from the user.

Remarks:

This function is very similar to SerGetCmd() except that it does not convert the string to upper-case, and it does not parse on delimiters. The enter string is returned as entered by the user.

Return Value:

Returns the length of the string or a negative on error.

Name:
SerGetiCmd(char *str, int maxlen, PORT_REC *unhand);

Description:

Reads a parsed string from the user, but does not convert it to upper-case.

Remarks:

This function is exactly like SerGetCmd(), except that it does not covert the returned string to upper-case.

Return Value:

Returns the length of the string or a negative on error.

Name:
SerInputReady(PORT_REC *unhand);

Description:

Checks to see if characters are waiting in the BBS ring buffer.

Remarks:

This function checks the BBS ring buffer to see if characters have been input and are waiting to be processes.

Return Value:

Returns 1 if there are characters waiting, 0 otherwise.

Name:
SerLoadCmdBuf(char *str, short code, PORT_REC *unhand);

Description:

Loads a value into the command stack buffer for a user.

Remarks:

This function can be used to alter the string of input waiting to be processed by a user. It is called primarily by the MCF handler, which uses it to insert ENTER and EXIT commands into a users command buffer. The value 'code' can be either PUSH (pushes 'str' onto the command stack ahead of everything else), REPLACE (replaces what's there with 'str'), or APPEND (places 'str' after the commands already in the buffer).

Return Value:

Returns the value of the CMDWAIT flag. This flag indicates that there are or are not commands waiting to be processed.

Name:

```
SerMultiColPrompt(char *str, int shl, int esl, int hcol,
                  int rcol, PORT_REC *unhand);
```

Description:

Prints a prompt string in two colors.

Remarks:

Omega Point/2 BBS offers users the ability to chain commands together, separated by either a space or a semicolon. This feature presents certain problems to the input and output system. For example, if the user enters the input that will be required at the next three prompts, it look rather silly to display those prompts only to continue on without waiting for input. To solve this problem, the BBS API offers to distinct forms of output, the first set, which consists of functions like SerWrite() and SerColorWrite(), displays the requested character string whether or not there are command waiting. The second group, which is used for prompting for input and consists of functions such as SerPrompt() and SerMultiColPrompt() only display the output if there are no commands in the command buffer. If there were commands in the user's command stack that needed processing, SerMultiColPrompt() would display nothing. This feature allows application programs to handle command chaining transparently.

This function prints a string in two colors, a highlight color and a normal color. 'shl' signals the start of the highlight color and 'esl' signals the end. The color to be used for highlighted text is 'hcol' and the regular color is 'rcol'. This function, of course, only prints color information to users that have requested a terminal type that accepts color codes.

Return Value:

This function always returns 1.

Example:

```
SerMultiColPrompt("Enter [O]ne or [T]wo:",
                      '[',']',BRWHITE,BRGREEN,unhand);
```

This example would print the 'O' and the 'T' in bright white, and everything else in bright green. The call would generate no output if there were commands waiting in the command buffer. Notice that this call differs from SerMultiColWrite() in that it will not print if there are command waiting, and that the characters used to signal

highlight on/off are printed, where as with SerMultiColWrite(), they are not.

Name:

```
SerMultiColWrite(char *str, int shl, int ehl,
                  int hcol, int rcol, PORT_REC *unhand);
```

Description:

Writes a string to the user in two colors.

Remarks:

This function is very similar to SerMultiColPrompt(). (see that functions remarks for a discussion on the difference between 'write' and 'prompt' output functions) There are two differences between this function and SerMultiColPrompt(). First, this function does not check to see if there are commands in the user's command buffer, it displays the output regardless. Second, this function does not print the 'shl' or 'ehl' characters.

Return Value:

This function returns 0 if output has been stopped by the user, otherwise it returns 1.

Name:

```
SerPause(PORT_REC *unhand);
```

Description:

Displays a "..More[Y,n].." string and waits for the user to hit a key.

Remarks:

This function is typically used only within the BBS API, it may be helpful to some applications, however. It prints a the "more" prompt and waits for the user to hit a key. If the user hits 'n', it indicates he wishes to halt output and the function returns 0. Any other input causes the program to return 1.

Return Value:

Returns 0 if the user enters 'n' for "no more", otherwise it returns 1.

Name:

SerPeekBuf(PORT_REC *unhand);

Description:

Checks a user's input buffer for waiting, unprocessed, characters.

Remarks:

Checks to see if there are characters waiting in the input ring buffer for a user. This should not be confused with the waiting command stack buffer, which stores commands already parsed by the program. This function simply checks to see if there are any unprocessed characters waiting in the buffer.

Return Value:

This function returns 0 if no characters are waiting, otherwise it returns the ASCII value of the next available character.

Name:

SerPrompt(char *str, int color, PORT_REC *unhand);

Description:

Displays a prompt for input if no commands are waiting in the command buffer.

Remarks:

See the description for SerMultiColPrompt() for a discussion on the difference between 'write' functions and 'prompt' functions.

This function prints the character string 'str' in the color 'color' if there are no commands waiting in the user's command buffer.

Return Value:

Returns the value of the variable unhand->cmdwait which is true if there are commands waiting in the command buffer and false otherwise.

Name:

SerReadc(PORT_REC *unhand);

Description:

This function returns the value of the next waiting in the user's input ring buffer.

Remarks:

This function reads the next character from the input ring buffer. If no characters are waiting in the buffer, this function will wait either for the user to input something, or for some error condition to occur. For example, a timeout condition or other abort signal. If such and error occurs, this function will return 0.

Note that it almost all cases, it is best to not call this function directly, but rather to call one of the string input functions, such as SerGetCmd() or SerGetStr().

Return Value:

Returns the ASCII value of the next character read, or 0 on

timeout or abort signal errors.

Name:

SerReadln(char *str, int len, int tmout, int echo,
PORT_REC *unhand);

Description:

Medium level function used to read a line of input from the user.

Remarks:

This function is called by SerGetCmd(), SerGetStr() and other related functions. Most applications should call those functions rather then calling this directly.

SerReadln() waits for and reads input from the user and places it into 'str'. If available input reaches 'len', input will be concluded and the function will return. If

input takes longer then 'tmout' seconds, the function will return an error. If 'echo' is true, each character will be echoed back to the user as he types them.

Return Value:

This function returns the total number of characters read, unless an error occurs in which case a negative value is returned.

Name:

SerSetDtr(PORT_REC *unhand);

Description:

Clears the DTR line on the serial port for approximatly 2 seconds.

Remarks:

This clears the DTR signal on the user's serial port to the modem. This essentially disconnects the user.

Return Value:

This function returns 0 unless an OS/2 device driver error occurs, in which case it returns the value of that error code.

Name:

SerSetXon(int xon, unsigned short rto, PORT_REC *unhand)

Description:

Sets the XON/XOFF flow control flag and also sets the serial port read timeout.

Remarks:

This function can be used to turn XON/XOFF flow control on or off. If 'xon' if TRUE, flow control is turned on,

otherwise it's turned off. This function is also used to set the read timeout. This is the number milliseconds a read request waits before returning an error code. Normally, this should be set to DEF_TIMEOUT (defined in OS2BBS.H).

Return Value:

This function always returns 1.

Name:

SerSimulateXon(PORT_REC *unhand);

Description:

Forces restart of output that has been halted by XON/XOFF flow control.

Remarks:

This function forces the resumption of output that has been stopped by XON/XOFF flow control.

Return Value:

This function always returns 1.

Name:

SerWrite(char *str, PORT_REC *unhand);

Description:

Outputs 'str' to the user.

Remarks:

This function displays the string, 'str', to the user. Unlike the 'prompt' style of commands, this function does not check to see if there are command waiting in the command buffer.

Return Value:

This function returns 0 if the maximum number of lines per page has been reached for this user and the user responded with a 'n' at the "more" prompt. Otherwise it returns 1.

Name:

```
UseAddPage(char *pgtxt, short pgusr, short type,
                  PORT_REC *unhand, PORT_REC *anchor);
```

Description:

Used to send one of several types of pages to a user online.

Remarks:

This function is used internally to send pages to users for such thing as login/logout messages, private pages, and notification when an E-Mail or regular message is sent a user while he or she is logged in.

'pgtxt' is the text that is sent to the receiving user. 'pgusr' is the port number on which the receiving user is logged in. 'type' is a code indicating the type of page to be sent. These codes are #define'd in OS2BBS.H but the most common one used from external programs is PAGE_REGULAR. 'unhand' is the PORT_REC handle of the calling user, 'anchor' is the PORT_REC anchor pointer. Both of these values are received in the call to UseChildAppInit();

Return Value:

This function returns either 0 if the page was sent, or one of the PERR values defined in OS2BBS.H.

Name:

UseDisplayPage(PORT_REC *unhand);

Description:

Called primarily from within the BBS API, this function displays any page messages waiting for the user.

Remarks:

Omega Point/2 BBS software supports the concept of user pages. These pages can take several forms; a message from one user to another, a sysop message to all users, or a login/off notification. This function displays any pages that may be waiting for any user.

Return Value:

This function returns 1 if there pages waiting and they were printed. If no pages were waiting, it returns 0.

Name:

UseEnterSysopChat(PORT_REC *unhand);

Description:

Called internally by BBS API to place a user in chat with the sysop.

Remarks:

This function is called internally to place a user in chat with the console user.

Return Value:

This function always returns 1.

Name:

```
UseTm(PORT_REC *unhand);
```

Description:

Checks to see if the user still has time allowed online.

Remarks:

When a user logs in, the system determines how much time he is permitted online. If this time has been exceeded, this function will return 0, otherwise it returns 1.

Return Value:

Returns 1 for time left, 0 for no time left.

13.4 "Trm" Functions

Name:
TrmClrScr(PORT_REC *unhand);

Description:

Sends a clear screen command to the users terminal and resets the line count counter for the user.

Remarks:

This functions sends the appropriate clear screen command to the user's terminal. It also sets the line counter to 0. This counter is the variable 'unhand->lcount'.

Return Value:

This function always returns 1.

Name:
TrmColor(int color, PORT_REC *unhand);

Description:

Sets the color of the user's terminal if the selected terminal type supports color.

Remarks:

This function sends the appropriate terminal control commands to the users terminal to change colors. The color to set the terminal to is 'color'. More on color codes can be found in the description for the function SerColorWrite(). Note also that the #define COLOR can be used to simplify the calling of this function. That definition, along with the applicable color codes, can be found in OS2BBS.H.

Return Value:

This function returns 0 if the user's terminal type does not support color, otherwise it returns 1.

Name:

TrmPosn(short x, short y, PORT_REC *unhand);

Description:

Sets the cursor position on the user's terminal if his terminal supports cursor movement.

Remarks:

This function sends the appropriate terminal control commands to the user's terminal to reposition the cursor. The cursor is positioned on column 'x' and row 'y'.

Return Value:

This function returns 1 if the user's terminal supports cursor control, otherwise 0 is returned

13.5 Line Editor Functions

13.5.1 Introduction

Omega Point/2 BBS has it's own integrated line editor. This editor is invoked in order to enter messages, e-mail, profile entries, file system entries, etc. In addition to these internal uses, the editor may be invoked by expansion applications. The include file OS2LED.H contains the structures and prototypes for these functions.

All of these functions take some action on a passed variable of type LNTYPE. This type is defined in OS2LED.H. When a message or other entry is created using these functions, one structure of LNTYPE is dynamically allocated using malloc(). Each structure becomes an element in a linked list of LNTYPE structures. It is important to free the memory used by these linked lists after you have finished with them. There is a function to do this for you.

13.5.2 Reference

Name:

```
FileToLntype(LNTYPE **lnret, char *fname, int maxcol,
             int maxline, int escok, int *err, int *lines);
```

Description:

Converts a standard ASCII text file to a LNTYPE linked list.

Remarks:

This function can be used to convert a file to a LNTYPE which can then be edited or modified using any of the other editor functions.

The first argument passed to this function is a pointer to a pointer of type LNTYPE. No memory should be preallocated for this address, since this function will allocate one LNTYPE element for each line read in. You must remember to free the list after you've finished with it, however, by using FreeLntype().

The variable 'fname' is the name of the file you wish to

read in.

'maxcol' should be set to the maximum number of columns that the function is allowed to read in. This value may not be greater then MAXCOL, defined in OS2LED.H.

'maxline' should be set to the maximum number of lines the function is permitted to read in.

'escok' should be true if ESCAPE (ASCII 27) characters should be permitted. You should set this to true if you wish to read in files that have ANSI codes.

'*err' will contain any error codes that were generated by the function.

'*lines' will contain the number of lines actually read in by the function.

Return Value:

The return code of this function does not contain anything useful. However, the value '*err' will contain -1 if the file was not found, or a file error occured, -2 if invalid characters were found in the file (i.e. non printable control characters), -3 if the maximum number of columns in a line is exceeded, and -4 if the maximum number of lines is exceeded. In addition, 'lnret' will either be a pointer to a linked list of type LNTYPE, or NULL.

Name:

FreeLntype(LNTYPE *);

Description:

Called to free a linked list of LNTYPE elements.

Remarks:

This function may be called to automatically free a linked list of LNTYPE elements. Several functions, such as FileToLntype() and LineEdit() allocate memory for these elements and it is important to free the list with this function.

Return Value:

This function always returns 1.

Name:
FreeTxtBuff(char *txtbuf)

Description:

Used to free text buffers created with LntypeToTxt() calls.

Remarks:

Although buffers created with LntypeToTxt() are simply storage areas created with malloc(), it is best to free these buffers with this function call for portability and future expansion.

Return Value:

This function always returns 1.

Name:
LineEdit(LNTYPE **lnret, int maxline, PORT_REC *unhand);

Description:

Standard line editor function. Accepts user input to create a linked list of line elements.

Remarks:

This function allocates one LNTYPE element for each line entered by the user. These lines, together, form a linked

list. This list should be freed with a call to FreeLntype(). 'lnret' contains a pointer to the first LNTYPE structure, or NULL if some error has occured or the user entered no text.

Return Value:

The value 0 is returned by this function unless some error has occured, in which case a positive integer is returned. The invoker of this function knows that there is valid text to process if the value '*lnret' is not equal to NULL.

Name:

```
LnEdit(LNTYPE **lret, LNTYPE *fline, int maxline,
              int endcmd, int *totln, PORT_REC *unhand);
```

Description:

Function used to invoke the edit mode of the Omega Point/2 line editor.

Remarks:

This function is useful for editing text that has been converted with FileToLntype() or previously created with LineEdit().

The variable 'lret' is used to return the new LNTYPE linked list created as a result of editing actions taken by the user (add, delete, etc). 'fline' is a pointer to the LNTYPE structure to be edited. This value will no longer be valid after this function returns, instead the edited contents will be transfered to the newly allocated 'lret'.

'maxline' is the maximum number of lines this entry is allowed to hold.

'endcmd' is used by the function LineEdit() to pass a command from the input section to the editing section. This can be any of the valid editing commands, such as 'S' to save, or 'A' to abort. Normally, this should be set to NO_COMMAND (defined in OS2LED.H).

'*totln' contains the number of lines in the finished, edited, product.

Return Value:

The caller of this function should check '*lnret' for a value of NULL. Such a value indicates that the user has aborted input or an error has occured. Any other value requires the text to be processed.

Name:

```
LntypeToTxt(int *lines, int *bytes, LNTYPE *lin,
                  char **text);
```

Description:

Converts a LNTYPE linked list of lines into a text buffer.

Remarks:

This function converts a set of linked list LNTYPE elements into a single text buffer. The output is placed in the '**text' buffer, which is allocated using malloc() by this function.

'*lines' contains the number of lines converted by this function.

'*bytes' contains the total number of bytes copied into 'text' by the function.

'lin', upon entry, contains a pointer to the first element in the LNTYPE linked list.

'text', upon entry, contains a pointer to a pointer of type char. The buffer is allocated and it's address copied into this variable upon exit.

Return Value:

This function returns the number of bytes converted. The only possible error condition is if a value NULL is passed in 'lin'. In this case, 0 is returned, and *text will be set to NULL.

Chapter 14

Advanced Topics

14.1 Using the Line Editor

The line editor can be called to allow a user to input text, and create a dynamically allocated buffer to store the input text. The following code demonstrates how this is done:

```
GetText()
{
   LNTYPE *mln;
   // linked list array for input text

   short x;
   // Holds the number of lines input after LntypeToTxt();

   short bcount;
   // Holds the number of bytes input after LntypeToTxt();

   char *textbuff;
   // buffer will be allocated by LntypeTxt();


   LineEdit(&mln, 60, unhand);
   // allow up to 60 lines to be input, the editor will
   // allocate memory as needed for 'mln'.

   // mln will be NULL if the user aborted or an error occured
   if( mln ) {
      LntypeToTxt(&x, &bcount, mln, &textbuff);
      // converts the linked list 'mln' to a text buffer

      FreeLntype(mln);
      // free the 'mln' linked list buffer

      //  At this point, 'textbuff' points to a buffer of
      //  'bcount' size. With this buffer and the byte
      //  count, the data can be written to disk or output.

      FreeTxtBuff(textbuff);
      // free the text buffer created
   }
   return(0);
}
```

14.2 DLLs with Omega Point/2

In the last few chapters, we have discussed the outlines and procedures used to create programs that can be attached to any command in any Menu Configuration File. This alone provides functionality far superior to that of any similar program. But Omega Point/2 goes even one step further; it is also possible to use the BBS API to create DLL modules which contain functions that may be executed from an MCF menu command.

By using the OS/2 Dynamic Link Library function, your expansion modules can achieve better performance then program modules while making more efficient use of memory. If you are not familiar with the concept of DLLs, you may wish to consult the "OS/2 Programmer's Reference" or "Advanced OS/2 Programming" by Ray Duncan before reading further.

Under Omega Point/2, a DLL function can be 'attached' to any command in any Menu Config File. When you make the entry in the MCF file, you specify the DLL name, the function name within that DLL, and a special flag that tells Omega Point/2 how to handle your request. This flag can be one of three values:

"NEED" - Tells Omega Point/2 to only load the function when and while it's in use. As soon as the DLL is no longer in use, it is discarded from memory. This is the most efficient in terms of memory, since no unused libraries are in memory. There is a slight performance penalty, however, since the module must be loaded each time it's called.

"ALWAYS" - This option specifies that once the module is first loaded, it should always be kept in memory. This is the most efficient in terms of performance, since the module need only be loaded once. It will, however, take up memory even when not in use.

"USED" - This type of function, like the others, is loaded the first time a user requests the function. With this type of DLL, however, the library is kept in memory until there are no longer users online. With this type of DLL, you may keep a library in memory as long as there is a chance it will be needed, only dumping it when the board is empty. This is a good compromise between performance and memory concerns.

There is an example DLL on the distribution disks along with the makefile, definition file, and source code.

The MCF entry for DLLTEST should look like this:

"DLLTEST",DLL,99,"",NEEDED,DLLTEST,TestProc

Reading from left to right, this entry says: the command DLLTEST is a DLL expansion module, it requires an access level of 99 to

execute, there are no special options, the module is loaded every time it's needed, the name of the module is DLLTEST (do not give the .DLL extension) and the procedure I wish to call within that module is TestProc.

Remember to copy your DLL into one of the directories specified in the OS/2 LIBPATH. This is set in the CONFIG.SYS file. The program will be unable to find it if it's not in a directory specified in that path.

All procedues within a DLL that are called from an MCF file must have 3 arguments, the first is a pointer to type SYSCONFIG. This structure is detailed in OS2BBS.H and contains most system configuration data. The second two are pointers to type PORT_REC and are exactly the same as 'unhand' and 'anchor' described in previous chaptors. See the DLLTEST.C file for details.

Appendix A

Expansion Program Examples

As you read in previous chapters, your Omega Point/2 package includes tools that give you the ability to create add-on modules for your BBS. Several example programs are included with your system. Each of the examples includes the needed 'C' files, include files, make files, module-definition files and data files. If you have Microsoft 'C', version 6.0, you may simply compile the programs by entering NMAKE <name>. If you have a different compiler, you may need to make minor changes to the make files or program files. The executable versions of each program is included, so you need not re-compile them in order to use them. The following programs are available:

A.1 ADVENT.EXE

This is the classic Adventure game! To compile it, simply enter: NMAKE ADV. This is a fairly large program that shows how easily almost any standard text based program can be ported to take

advantage of the BBS API system.

A.2 CZAR.EXE

This program is used to correctly execute CZARWARS, which is a game not included with this system. CZARWARS is written by Ray Yeargin and Troy Carroll. It is a shareware game and it available on CompuServe and many BBS systems. You may also write to the author directly at:

Troy Carroll
Rt 16, Box 9022
Tellahassee, FL 32310-9710

A.3 PREDI.EXE

This is a program used to redirect standard output and, optionally, standard input of another program. Using this program, you may spawn other programs that simply use standard in and standard out. For example, the PSTAT command in the

MAINMNU.MCF calls this PREDI, which then calls the OS/2 program PSTAT. Use the command line option "-i" to tell PREDI that you wish to also redirect standard input. Note that input redirection is not perfect with this program. Input is not echoed back until you press ENTER.

A.4 TTT2.EXE

This is a 3-D Tic-Tac-Toe game. This is a simple program that demonstrates how to use the the BBS API functions effectively.

Appendix B

Add-on Product Catalog

As Omega Point/2 continues to grow and develop, ExcelSoft and other developers will create new add-on modules for the system. This section will be updated to include the current list of such programs, who wrote them, where to find them, and how much they cost. If you develop your own add-on module and would like it listed here, please write or call. Even if you don't want your product listed here, we'd like to hear about what you're doing

with the system!

B.1 Add-On Modules

Name: Byte Information Exchange News Reader
Author: ExcelSoft
How to Obtain: Contact ExcelSoft Software
Price: $20
Available: Now
DLL/Program/Session: Program
Function:

BIX, the Byte Information eXchange, is a commercial information service operated by Mcgraw Hill Publications (publishers of BYTE magazine). On BIX, they offer a special service to BBS operators called BBX (BBS Exchange). Subscribers to this service can download computer industry news briefs daily. This set of programs parses, indexes, and allows users to view these news briefs. The BBX service must be subscribed to separately, and as of this writing the cost was approximately $200/year.

Name: Classified Ads
Author: Nick Steel
How to Obtain: Contact ExcelSoft
Price: (undetermined)
Available: End of 1990
DLL/Program/Session: Program
Function:

This system allows users to post classified ads. Your "classifieds section" can include as many as 25 different categories. Posted ads can be anonymous and when another user replies to the ad, the reply is sent via the Omega Point/2 E-Mail

system. The classifieds section is superior to regular messages because ads are regularly recycled and old ads removed. This means the system isn't cluttered with old, no longer applicable, messages.

Appendix C

Future Directions

Most software developers are very hesitant to make any statements about the anticipated future shape of their products. This is for good reason, even vague probing statements are often interpreted as firm product announcements. Well, since this is our first commercial product, we are entitled to make a mistake.

ExcelSoft plans on developing the concept of remote networking. We plan to implement at least one of the many networked message and mail systems, such as FIDO Net's Echo-Mail. In addition, we plan on developing networked teleconferencing and other related systems.

Future versions of Omega Point/2 will be more suitable for LAN use. OS/2's advanced named-pipes can be used to great advantage. For example, on a multiple machine network, one machine could be set aside simply as a game server. Programs running on the server would communicate to other systems on the net via named pipes. This way, the server programs would not affect the performance of the regular system. This same concept can also be expanded to include message servers and database servers.

We also hope to use LANs to make instances of Omega Point/2 running on different nodes all appear as one to the users. So, for example, you would be able to teleconference and send messages to any user on any node on the LAN.

When OS/2 2.0 is officially released, Omega Point/2 will be ported to take advantage of it's 32 bit environment. We anticipate a significant increase in performance.

Appendix D

Development Tools

A seemingly vast array of hardware, software, books, and magazines were used in the development process. The following list mentions those that helped out. Trademarked names are owned by their perspective companies. The following tools were used in the development of the Omega Point/2 Bulletin Board System:

Software:

IBM OS/2, version 1.0
IBM OS/2, version 1.1
IBM OS/2, version 1.2
Microsoft C, version 5.1
Microsoft C, version 6.0
Microsoft Macro Assembler, version 5.1
Microsoft Presentation Manager Toolkit, version 1.1
The Hamilton C Shell for OS/2, by Hamilton Laboratories
BRIEF, by Underware
Sprint, by Borland International
Railroad Tycoon, by MicroProse (for distraction)

Hardware:

IBM PS/2 model 60
Club American Hawk II 486
US Robotics HST/V.32

Thanks and Acknowledgments:

The development of this system has been both challenging and rewarding. As with any major undertaking, many people influenced it's creation. Therefor, I would like to thank the following people. Christine R. Lindmark for pointing me in the right direction; Gerry Ceres for opening the door of computers; my grandfather, A.A. Boehm, the eternal gentleman; my baby sister, Amanda, the teenage mutant heartbreaker; and all of the hundreds of unnamed people, on-line and off, that contributed to Omega Point/2's development.

Contents