Maximus-CBCS Version 1.02 Operations Manual

Created August 4, 1990.

Documentation produced by Erik Van Riper, Dave Cushing,
Jesse David Hollington, Hubert Lai and Scott Dudley.

TABLE OF CONTENTS

LICENCE

(note Canadian spelling!)

Non-commercial distribution and/or use is permitted under the following terms:

1) You may copy and distribute verbatim copies of the Maximus-CBCS source, documentation, and executable code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy a valid copyright notice "Copyright 1990 by Scott J. Dudley"; keep intact the notices on all files that refer to this Licence Agreement and to the absence of any warranty; PROVIDE UNMODIFIED COPIES OF THE DOCUMENTATION AS PROVIDED WITH THE PROGRAM; and give any other recipients of the Maximus-CBCS program a copy of this Licence Agreement along with the program. You may charge a distribution fee for the physical act of transferring a copy, but no more than is necessary to recover your actual costs incurred in the transfer. Under no circumstances is Maximus-CBCS to be distributed in such a way as to be construed as "value added" in a sales transaction, such as, but not limited to, software bundled with a modem or CD-ROM software collections, without the prior written consent of the author.

2) You may modify your copy or copies of Maximus-CBCS or any portion of it, and copy and distribute such modifications under the terms of Paragraph 1 above, provided that you also do the following:

   a) cause the modified files to carry prominent notices stating that you changed the files and the date of any change;

   b) cause the executable code of such modified version to clearly identify itself as such in the course of its normal operation;

   c) if the modified version is not a "port", but operates in the same hardware and/or software environment as the original distribution, make the original version equally available, clearly identifying same as the original, unmodified version;

   d) cause the whole of any work that you distribute or publish, that in whole or in part contains or is a derivative of Maximus-CBCS or any part thereof, to be licensed at no charge to all third parties on terms identical to those contained in this Licence Agreement (except that you may choose to grant more extensive warranty protection to some or all third parties, at your option); and:

e) send the complete source code modifications to Scott Dudley at the addresses listed below, for the purpose of evaluation for inclusion in future releases of Maximus-CBCS. Should your source code be included in Maximus-CBCS, Scott Dudley retains all rights for redistribution of the code as part of Maximus-CBCS and all derivative works, with appropriate credit given to the author of the modification;

f) You may charge a distribution fee for the physical act of transferring a copy, but no more than is necessary to recover your actual costs incurred in the transfer, and you may at your option offer warranty protection in exchange for a fee;

g) when distributing modified versions of Maximus-CBCS, you must not change the name of the program or the official version number, except to append an identifier which indicates that modifications have been made. For ports to other operating systems, the following convention must be followed:

Maximus-CBCS v<v>.<os>.R<r>

...where <v> is the official Maximus version number, <os> is the name of the operating system which the port runs under, and <r> (optional) is the OS-specific revision number. For example, the second OS/2 revision of Maximus 1.02 must have a version string in this format: `Maximus-CBCS v1.02.OS/2.R2'

Similarly, modifications to Maximus which are designed to run under MS-DOS must also follow a naming convention. The version string must read:

Maximus-CBCS v<v>.<i>.<r>

...where <v> is the official Maximus version number, <i> is three initials (indicating your first, middle and last names), and <r> (optional) is the revision number of your modifications.

For example, a version of Maximus 1.02 modified by Joe T. SysOp must have a version string in this format: `Maximus-CBCS v1.02.jts.1'

3) Mere aggregation of another unrelated program with this program and documentation (or derivative works) on a volume of a storage or distribution medium does not bring the other program under the scope of these terms.

4) You may copy and distribute Maximus-CBCS and its associated documentation (or a portion or derivative of it, under Paragraph 2) in object code or executable form under the terms of Paragraphs 1 and 2 above provided that you also do one of the following:

a) accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Paragraphs 1 and 2 above; or,

b) accompany it with a written offer, valid for at least three years, to give any third party free (except for a nominal shipping charge) a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Paragraphs 1 and 2 above; or,

c) accompany it with the information you received as to where the corresponding source code may be obtained. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form alone.)

For an executable file, complete source code means all the source code for all modules it contains; but, as a special exception, it need not include source code for modules which are standard libraries that accompany the operating system on which the executable file runs.

5) You may not copy, sublicense, distribute or transfer Maximus-CBCS and its associated documentation except as expressly provided under this Licence Agreement. Any attempt otherwise to copy, sublicense, distribute or transfer Maximus-CBCS is void and your rights to use the program under this Licence agreement shall be automatically terminated.

However, parties who have received computer software programs from you with this Licence Agreement will not have their licences terminated so long as such parties remain in full compliance, and notify Scott Dudley of their intention to comply with this Agreement.

6) If you wish to incorporate parts of Maximus-CBCS into a program which is NOT completely free for ALL users, one with "strings attached", then please contact Scott Dudley at one of the addresses listed below. We have not yet worked out a simple rule that can be stated

here, but we will usually permit this. We will be guided by the two goals of preserving the free status of all derivatives of our free software (as it pertains to non-commercial use, as provided by this Agreement) and of promoting the sharing and reuse of software.

7) For those who wish to incorporate parts of this source into a program which is FUNCTIONALLY SIMILAR to Maximus-CBCS, all of the regulations in paragraph 6 also apply. You cannot use this code without prior written permission form Scott Dudley, if you are designing a program which is functionally-similar to Maximus-CBCS.

8) For the purposes of this document, "COMMERCIAL USE" is defined as operation of this software by (or for) any corporation, government, company, foundation, church, or also any other organization or individual, for the purpose of making a profit. The definition of "making a profit" includes systems which run more than three modem lines using the Maximus-CBCS software, systems which advertise products for the system's sponsor or system operator, and also those whose system access fee is above and beyond the cost required to maintain the system. Those wishing to use Maximus-CBCS in a commercial environment (as defined above) must first obtain a commercial licence. Although commercial licences are usually negotiable, a one-node licence is presently going for $110 in Canadian currency (CAD$1 ˜= US$0.85), and includes the Maximus software on disk, and one free non-bugfix upgrade. If you're unsure as to whether or not you require a commercial licence, or if you're interested in a site licence, or even if you're simply looking for more information, please contact the author at one of the addresses listed below. (If you require information only, NetMail or Email messages are the preferred form of correspondence.)

NO WARRANTY

BECAUSE MAXIMUS-CBCS IS LICENSED FREE OF CHARGE, WE PROVIDE ABSOLUTELY NO WARRANTY. EXCEPT WHEN OTHERWISE STATED IN WRITING, SCOTT DUDLEY AND/OR OTHER PARTIES PROVIDE MAXIMUS-CBCS "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF MAXIMUS-CBCS, AND THE ACCURACY OF ITS ASSOCIATED DOCUMENTATION, IS WITH YOU. SHOULD MAXIMUS-CBCS OR ITS ASSOCIATED DOCUMENTATION PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT WILL SCOTT DUDLEY BE RESPONSIBLE IN ANY WAY FOR THE BEHAVIOR OF MODIFIED VERSIONS OF MAXIMUS-CBCS. IN NO EVENT WILL SCOTT DUDLEY AND/OR ANY OTHER PARTY WHO MAY MODIFY AND REDISTRIBUTE MAXIMUS-CBCS AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY LOST PROFITS, LOST MONIES, OR OTHER SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS) MAXIMUS-CBCS, EVEN IF SCOTT DUDLEY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

You can contact the author at one of the address listed below:

Scott Dudley	FidoNet	1:249/106
777 Downing St.	IMEXnet	89:483/202
Kingston, Ont.	Internet	f106.n249.z1.fidonet.org
Canada - K7M 5N3	BBS:	(613)389-8315, 14.4K/HST

Please feel free to contact the author at any time to share your comments about this software and/or licensing policies.

Our thanks to Richard Stallman at the Free Software Foundation, Inc. and BBS Co. for most of the wording of this licence.

INTRODUCTION

Notes from the Author

Maximus-CBCS v1.02 is classified as a `minor' update from version 1.00. However, there are over two dozen pages in the `New Features' document, and a large number of bugs have been identified and fixed. The next release will be a "major" upgrade, and will probably include some earth-shattering changes. Although these are by no means a "sure thing" for the next release, we've scheduled the following additions to both the DOS and OS/2 versions: support for multiple message base types (ie. you will be optionally able to get rid of *.MSG and opt for database-type message areas), support for more than one language for the system prompts/log file, and more. By all means, this isn't a complete list, and is just an indication of where we're planning to go in the future. There will probably be a lot of small changes, but the above three are pretty well on the top of our "To Do" list.

Also, one thing which may be of interest to potential Sysops is that the Maximus source code is finally available NOW, to the general public. Maximus consists of approximately 40,000 lines of C code (ANSI compiler required), and under 500 lines of assembler code. To compile the source code, you'll need either Microsoft or Turbo C, and a MASM or TASM-compatible assembler. The source code will compile under both DOS and OS/2, although you'll need MSC to compile the OS/2 version. As well, plans have already been made to port Maximus to other platforms. If you are interested in porting Maximus to the operating system which you use, please contact the author at one of the addresses listed below.

If you don't know where to get the source code, you can usually get it from participating Software Distribution System (SDS) nodes. Failing that, you can download or file-request it from the author's BBS, using the magic filename of "MAXSRC". Likewise, the latest version of Maximus-CBCS itself can usually be found on the SDS, or can be downloaded or file-requested from the author, as a last resort, using the magic filename of "MAX". If you don't have an assembler and therefore can't assemble the *.ASM files included with Maximus, you can request the assembled .OBJ versions of the .ASM files by requesting the magic filename "MAXOBJ" from the author.

Also, if you are having trouble installing Maximus, have a look in the troubleshooting section of this manual. If you still can't get Maximus to work correctly, then either post a message in the MUFFIN echomail conference, or send netmail to the Maximus Help node, which is FidoNet 1:1/119.

Final note: Maximus-CBCS is another fine CANADIAN program. You know - us Canadians are the guys from up North who all look like "Eskimos", use the word "eh" as if it were punctuation, and drink cold beer twenty-four hours a day. More importantly, that also means we know how to spell correctly! Some of you Americans may think that words like "colour" and "favourite" look odd, but that is actually the correct spelling for the rest of the English-speaking world. <grin>

Scott Dudley	FidoNet	1:249/106
777 Downing St.	IMEXnet	89:483/202
Kingston, Ont.	Internet	f106.n249.z1.fidonet.org
Canada - K7M 5N3	BBS:	(613)389-8315, 14.4K/HST

Credits

Although the majority of Maximus was created primarily by myself, there are several people who should receive acknowledgement for their contributions, and several products mentioned in this text whose authors need to be credited.

First and foremost, my thanks go to Wynn Wagner III. Without Wynn, there would be no Opus, without which there would be no Maximus, at least not in the form it is today. Although we aren't using any of Wynn's code, he did provide the program after which Maximus is modeled. Wynn also wrote a large number of utilities and established the WaZOO standard, without which a lot of the current mailers and BBS programs wouldn't exist.

Secondly, my thanks go to Bob Hartman and Vince Perriello, who are the principal authors of the BinkleyTerm front-end mailer, and who contributed all of the file-transfer protocols for Maximus.

As well, I should thank Peter Fitzsimmons for all of the help he has contributed. Peter is the head of the Maximus-OS/2 development team, but also has made many suggestions and fixes to the MS-DOS version. As well, he wrote a number of utilities to go along with the DOS version of Maximus, including MaxRen and the decompression routines in INSTALL.EXE.

I would also be remiss if I failed to thank all of my alpha and beta testers. Although the list is too long to include here, my sincere gratitude goes out to all of them, not only for their help in tracking down the many bugs which were visible in earlier Maximus releases, but also for their helpful suggestions and comments. Thanks also go to the rest of the documentation team, including Erik Van Riper, Dave Cushing, Jesse Hollington, and the editor, Hubert Lai. These are the folks who get the credit (or blame!) for this document.

Thanks go to the following people who have contributed to the Maximus source, by donating algorithms and/or code. (My sincere apologies if someone has been left out.)

Peter Fitzsimmons
Vince Perriello and Bob Hartman
Andrew Farmer
Scott Friedman
Ray Duncan

Thomas Plum
Alan Hughes
Bob Trower

Thanks also go to Bill Cassidy and James Hollingsworth, who gave me their permission to use some of their PCBoard menu and graphics screens in the PCBoard emulator.

Finally, I wish to acknowledge several products and trademarks which have been mentioned in this document. Use of these names or trademarks neither constitutes an endorsement nor suggests any affiliation with the specified products. The names or trademarks referred to are used for reference purposes only.

ARC and SEAlink:	Thom Henderson
BinkleyTerm:	Bob Hartman & Vince Perriello
BNUcom:	David Nugent, Unique Computing
Courier HST:	U.S. Robotics Inc.
DESQview:	Quarterdeck Office Systems
DoubleDOS:	SoftLogic Solutions Inc.
FEND:	David Luong
Fido & FidoNet:	Tom Jennings, Fido Software
FrontDoor & FDterm:	Joaquim Homrighausen
FView:	Doug Boone
IBM-PC, PC-DOS:	International Business Machines
LHarc and LZH:	Haruyasu Yoshizaki
Minimus:	John Cuccia
MS-DOS:	Microsoft Inc.
Opus-CBCS:	Wynn Wagner III
OpusComm & ConfMail:	Bob Hartman, Spark Software
PCBoard:	Clark Software Development
QuickBBS:	Ultimate Software Technologies
QM, Qmail & AreaFix:	Greg Dawson
RBBS:	Capital PC Users Group
TBBS:	eSoft, Inc.
Telix:	Colin Sampaleanu, Exis Inc.
TheDraw:	Ian Davis, TheSoft Services
TinyTerm, OECC:	George A. Stanislav
Turbo C:	Borland International
V-Series and Hayes:	Hayes Microcomputer Products
VT-100:	Digital Equipment Corporation
X00:	Ray Gwinn, RENEX Co.
Xmodem:	Ward Christiensen
ZIP:	Phil Katz, PKWare
Zmodem and MobyTurbo:	Chuck Forsberg

SYSTEM REQUIREMENTS

Although it is possible to run Maximus on a system with less than the following equipment, the following should be considered the realistic minimum with which you can get by:

- An IBM (or compatible) personal computer running MS-DOS or PC-DOS, with at least 256K of available RAM, with at least 132K free.
- MS-DOS or PC-DOS version 2.0 or greater, 3.2 or above preferred.
- A Hayes-compatible modem. It is possible to use Maximus with a modem which is not Hayes-compatible. However, doing so will make your life unnecessarily complicated.
- A hard disk. Capacity of 30 Mb or greater is preferable.
- A FOSSIL communications driver, revision level 5 or higher.

OVERVIEW

How Maximus is Different From Opus

Although Maximus was first conceived as a drop-in replacement for Opus 1.03, during development it blossomed into something far beyond that. If you are wondering what features Maximus has that Opus 1.03 (and Opus 1.10) do not, then this is the section of the manual for which you have been looking. All of the differences between Maximus and Opus could not be fully described here. Thus, what follows is an overview of Maximus' significant special functions.

In addition to being 99% compatible with the existing Opus 1.03 program, Maximus also offers the following features:

* MaxEd (Maximus Editor) is a new and improved full-screen editor which is faster, and more versatile than OpED. MaxEd offers all of the features of OpED and also supports full paragraph reformatting (ever tried to insert a word in the middle of a long paragraph?) and bidirectional movement in the quote window.

* BORED, Maximus' line-oriented editor offers users the ability to quote the message to which they are replying. As well, with either editor one can read in a text file from disk and make changes to it before saving.

* Maximus sports a new internal user editor which supports all normal user editor commands, and also allows editing of the Maximus-specific fields in the user file.

* Maximus features a speedy internal mailchecker which will display the user's message and permit him/her to reply using either of the two editors. This mail checker is better and faster than anything which is available for Opus and has the added advantage of being directly supported, rather than kludged in.

* Most of Maximus' message-area commands (List, Scan, Next, etc.) function at almost double the speed of the same in Opus. If you utilize the optional SCANBLD utility, Scan will run up to 10,000% faster in large EchoMail areas.

*   Maximus supports an internal `Check for new files' command. Unlike Opus, there is no need to kludge in additional programs such as NEWFILES.

*   Maximus uses a totally redefinable menu structure (unlike Opus 1.03's fixed menus or Opus 1.10's semi-turgid menus). Sysops have the option of displaying a custom *.BBS file (a la QuickBBS), instead of the default menu that Maximus normally generates. Sysops can pair message and file areas in a `SIG' configuration, put all the menu options on one large menu like PCBoard, or go with the standard Opus/Fido menu design. Sysops can add or delete as many menus as they want, and can include up to 256 options on one individual menu. Advanced users will find Maximus' "Linking Menu Options" feature valuable, since it allows one to have more than one command executed with one keystroke, and also to automatically "poke" keys into the typeahead buffer, thereby allowing almost an unlimited number of actions to be performed..

*   In addition to all of the Opus embedded commands, Maximus supports a number of new embedded .BBS commands which allow the sysop to create *.BBS files with much more ease and power. These new commands can be processed by MECCA, the `Maximus Embedded Command Compiler (Advanced)'. MECCA is not only 100% compatible with OECC, but is also over 35% faster. As an added bonus, you no longer need to type a lot of those silly square brackets.

*   Maximus supports up to 1,296 message areas. Each area `number' is actually alphanumeric, so for example, a sysop can name his/her areas `AA', `AB', `A1', 'CZ', etc., in addition to the normal numeric labels.

*   Maximus keeps track of lastread pointers for users in all areas, no matter how many areas there are. Maximus does not have the same limit of 10 areas that Opus 1.03 has. This is long-overdue, since few users restrict themselves to 10 message areas, and losing track of one's lastread pointer is a major irritation.

*   Maximus supports both the version 6 nodelist and the older version 5 nodelist. You can use either or both to maintain compatibility with such programs as BinkleyTerm, Opus 1.03, and Opus 1.10.

*   The Hurl command in Maximus works across physical drives.

*   Maximus can directly display the contents of several types of archive, including ZIP, ARC, PAK, and LZH.

The need for kludging in an external program such as Doug Boone's fine OVIEW is obviated.

*   Maximus supports users with more than two names, while still allowing the old `First Last;Y;Password' sequence to work properly. Thus, users with names like Geoffrey Vaughn Gladdy can log on as Geoffrey Vaughn Gladdy, rather than being forced to truncate their name to Geoffrey Gladdy.

*   Maximus sports a new security system which warns the user if a hacker has been trying to guess their password.

*   Maximus can create an ECHOTOSS.LOG file for messages entered by users. Thus, your mail processor does not have to scan all the areas on your system just to export one message.

*   You can completely exit from Maximus via a chain command or errorlevel exit, run an external program, and re-enter Maximus exactly where you left off. This is completely invisible to the user. Finally, sysops can use memory-hungry programs as doors, or run external programs even on systems with memory constraints.

*   Maximus supports a lightning-fast local video mode, which is aware of DoubleDOS, DESQview, and TopView.

*   Some internal support has been implemented for multi-line systems, and an interface for an external node-to-node chat program has been created.

*   Maximus can optionally store a user's phone number in the user file. As well, if you are running a system which allows handles, the system can be configured to ask the user for his real name and store that information in the user file.

*   Maximus supports translation characters for running external commands, which makes child's play of running programs such as TradeWars.

*   Maximus can generate QuickBBS/RBBS DORINFO*.DEF files, WWIV CHAIN.TXT files, or ANY other ASCII-based door interface file, through the use of the above-mentioned translation characters.

*   Maximus can keep a separate log file for uploads, which makes it easy to keep track of who sent what.

*   Maximus supports a series of statements for the lazy sysop. You can now still have an original MSGAREA/FILEAREA.BBS-type display, but it can be generated on the fly by Maximus, without having to mess with any *.BBS files. This is great for sysops who hate having to update zillions of files every time they decide to add or drop message or file areas.

*   Maximus does not need all the SYSTEM*.BBS, DIR.BBS and ORIGIN.* files which are traditionally used in Opus-based systems, and stores all of the information for all areas in only two compact files. (However, the SILT compiler can generate all of the above files if you need them to run Opus-compatible programs.)

*   Maximus offers Internal support for the maintenance of upload/download ratios.

*   Maximus features a local hot-key which allow the sysop to shell to DOS from anywhere in the system.

*   The message and file areas need not be tied to one another, so it is possible to set independent access levels for each.

*   Maximus now supports a full GREP-like Inquire command.

*   Maximus supports a `Lock and Key' system which lets you selectively give users access to certain areas, independent of their privilege level.

*   Maximus has extended support for barricaded areas. Unlike Opus 1.03 (in which only areas in the range 50 to 99 could be barricaded), any area can be barricaded in Maximus. Maximus also has extended the barricade file syntax to allow sysops to create `SIGOps' or users who have extended privileges in some areas only. Sysops may require some users to enter a password to obtain these extended privileges while allowing other users to obtain the extended privileges automatically upon entering the area.

*   Maximus can screen out certain new users based on their login name. The sysop can create a text file with a list of names (or partial names) which he/she does NOT want on his/her system, and Maximus will not allow users with those names to log on.

*   Opus 1.03 has problems handling a large nodelist, such as that used by the vast majority of sysops today. In contrast, Maximus can handle a nodelist of any size. If a sysop can fit it on his/her hard drive, Maximus can use it.

*   Maximus optionally supports a full-screen hot-key mode. Thus, users need not press the <Enter> key after each command. However, the traditional Opus-style menus are supported for those users who prefer them.

*   Maximus supports custom origin lines. If a sysop desires, he/she can have Maximus use a difference origin line for messages in each area.

*   Maximus supports AVATAR graphics. Users who take advantage of this can achieve a significant improvement in apparent transmission speed. As well, Maximus will automatic translate IBM `Extended ASCII' characters to normal ASCII for those users who are not using IBM-compatible computers..

*   Maximus uses a program called `SILT' as a replacement for the old control-file compiler. Not only will SILT parse the system control file, but it also compiles area and menu definitions, which are now also contained in an easy-to-manipulate ASCII file.

*   Maximus supports new local keyboard commands, including those to add or subtract five minutes from the user's time, toggle the Yell command, knock a user off-line with simulated line noise, and more.

*   Maximus supports the use of CD-ROMs in file areas.

*   Maximus can optionally reward users for time spent uploading and time spent chatting with the sysop.

*   With all these improvements and new features, Maximus still uses only 146K of RAM. That is about the same amount of memory required by Opus 1.10 if you plan on using a nodelist.

Overview of Maximus

This section of the documentation is designed to give you, the SysOp, a general overview of what Maximus can do. This isn't complete, by any means, but it should give you at least a general idea of how the system is laid out.

Logging On

When Maximus starts up with a caller on-line, it will display the Maximus name and version, followed by the SysOp-defined log-on screen, which is stored in \MAX\MISC\LOGO.BBS. This screen should be kept fairly short, and should NOT use any ANSI graphics, nor high-bit IBM characters.

After the logo screen has been displayed, Maximus will then prompt the user to enter his/her name. Unlike other BBS programs, Maximus allows a user to use names with more than two names, so even a name such as "Jesse David Hollington" will be accepted. (On the other hand, Maximus will reject callers with a one-word name, unless you have the `Alias System' keyword uncommented in MAX.CTL.)

If the name which was entered exists in the user file, then Maximus will proceed as normal. On the other hand, if the user does NOT exist, then Maximus will display NOTFOUND.BBS, in the \MAX\MISC subdirectory, and confirm that the user really wanted to enter the name they typed. In addition, Maximus will disable key stacking, just in case the user entered a name in the format of `Joe SysOp Y Password', and misspelled the name `Joe SysOp'.

If the user's name wasn't found in the user file, and the user confirmed that s/he wanted to register as a new user, Maximus will then go through the new-user configuration routine. First, Maximus will display APPLIC.BBS, which should normally give the caller some information about your system, and possibly present an application form. (If a caller hangs up before APPLIC.BBS has completed displaying, then the user's configuration will NOT be saved, and no entry will be made in USER.BBS.) When the application has finished displaying, Maximus will then prompt the user to enter his/her city, real name (if applicable), phone number, etc. Finally, Maximus will then display NEWUSER2.BBS, which can be yet another screen directed toward educating new users.

On the other hand, if the user's name was found in the user file, then Maximus will prompt the user to enter his/her password. The user will get five tries to enter the correct password; if all five tries are bad, or if the user pressed <enter> five times, then Maximus will hang up and exit back to your mailer. If the user enters the correct password, Maximus will proceed with the log-on sequence, and display WELCOME.BBS to the user.

Note: if a user's account has a BLANK password, then Maximus will treat that user as a `guest account'. This means that Maximus will ask a user who is using this account for his/her configuration settings at every log-on, and Maximus will also skip the password prompt. This enables the SysOp to create an account specifically for new users (while using a `Logon Level Preregistered' statement), such that users can look around the system, but the user file won't get cluttered with users who choose not to register. (The registration would presumably be done through an on-line questionnaire.)

Maximus also supports a concept known as a `custom welcome screen'. It is possible to define a special welcome screen for each individual user, displayed BEFORE the main WELCOME.BBS. By placing a file called `#.BBS' in the Maximus system directory, where `#' is a number, then Maximus will display that .BBS file to the caller with the specified user number. (You can find out the user number for a particular user by looking in the log file, or by doing a find command in the user editor, and writing down the user number displayed.) For example, if you wanted to show a custom welcome to user #5, then you would create a file called `\MAX\5.BBS'.

The Main Menu

Although Maximus' menus are completely redefinable, this section attempts to explain the commands which would normally be found on the main menu. This is also where the commands appear in the default configuration.

Message Section

> This enters the message section. From here all the message entering and reading features are available. See the section on the message areas for more detail.

File Section

> This takes the user to the Files Section. From here a user can exchange files by uploading or downloading, or simply see what files are available. See the section on the file areas for more detail.

Change Setup

> This takes the user to the Change Setup menu. From here, a user can modify their user profile. They can set their screen length, change their graphics mode, password, toggle the full-screen editor, and more. See the section on the Change Setup menu for more details.

Goodbye

> This option logs the user off the system and hangs up the phone. This is almost identical to what happens if a user simply drops carrier. Maximus will not `hang' if a user drops carrier, but will recycle as if they logged off using this command. This command simply asks the user to confirm that they want to disconnect, asks them if they want to leave a message to the sysop, and then hangs up on them. Comments to the sysop are saved in message area 0. If message area 0 does not exist, the user will not be asked if they want to leave you a comment.

Statistics

> This option displays the user's statistics, including the time the user has been online for the current call, the time online for the day, amount uploaded, amount downloaded, NetMail credit, and so on.

Yell

This command allows the user to attempt to contact the Sysop. You can set when you want users to be able to page you with this command, and you can toggle the local noise on or off with the "!" command from the local keyboard.

Userlist

This command simply displays a list of all the users who are currently in the user file. You can set the maximum and minimum privilege levels to display in the list through options in the control file. Maximus will default to not displaying anybody with a privilege of Sysop or higher, or Twit.

Version

This displays the version number and a few other statistics about the current revision of Maximus and the system that it's running under.

User Editor

This invokes the Maximus Internal User Editor. This command should be made available only to Sysops, for obvious reasons. See the section on the user editor for more details.

In addition, you can define other options in the main menu to call sub-menus, display text files, or run external programs. See the control-file reference section for more details on defining menus.

The Message Section

There are three basic types of message areas within Maximus. These are Local, NetMail, and EchoMail.

Local messages are messages entered by a user on your BBS. They can be either public or private, and remain on your BBS. Other users can only read these messages by logging onto your BBS and being in the same area that the message was entered in. Generally the majority of your message areas will be local.

NetMail messages are messages that are sent from one BBS to another through a network that you are connected to. They are generally private messages.

Unless you are a host or hub, most NetMail messages you encounter will either be entered on your system, or will be sent to your system from another. Maximus is fully compatible with the FidoNet mail standard for these messages. A user entering a NetMail message will be prompted to enter additional information to tell your mail processor where to send the message. Maximus is capable of reading a FidoNet compatible Version 5 or Version 6 nodelist file in order to get its address and cost information. Generally your users will need to have credit in their user accounts in order to send NetMail. Under most circumstances you should only have one of these areas.

EchoMail messages are messages that are shared between several bulletin board systems in a wide-area conference setup. An EchoMail message will be sent through your network to all other systems participating in the conference. Generally EchoMail messages are public only, and private messages in EchoMail areas tend to upset people. To a Maximus user, this area will function identically to a local message area. The difference is that any messages entered in these areas will have special EchoMail processing information added to them, such as origin lines. In addition, when a user enters a message in an EchoMail area, Maximus will append that area's tag name to a file to be used by your EchoMail processor.

Maximus also supports a variety of message area toggles (or "attributes"), each of which can be set independently, on an area-by-area basis. Although a complete list can be found in the AREAS.CTL reference in the Maximus Technical Reference Manual, some of the more common attributes are:

Private Only

All messages entered in these areas will be marked private, and can only be read by the user who sent the message, the user it is addressed to, and the sysop.

Public Only

All messages entered in these areas will be public and can be read by any user. This flag is recommended for EchoMail areas.

Read-Only

Messages in these areas can be read by users, but only users with a privilege level of assistant sysop or higher can enter messages.

Anonymous OK

In an area with this attribute set, users can enter messages under a pseudonym if they do not wish to use their real name. Maximus will embed the user's real name within the message in such a way that only the Sysop should be able to see it. The embedding of the user's real name can be disabled in the Maximus control file for those who don't want to use it. Please refer to the AREAS.CTL reference (in the Maximus Technical Reference Manual) for more details.

Maximus is also capable of assigning passwords to message and file areas, and re-assigning privileges within the area for certain passwords. In addition, you can assign user names to the barricade file, so only certain users can enter an area. See the section on extended barricades for more details.

In addition, the private and public message-area attributes can be defined individually by privilege level, rather than globally for all callers. (In other words, you can allow the SysOp to enter anonymous messages, while still forcing normal users to use their real names.) See the control-file reference for more information on how to assign these attributes to areas.

The Message Menu

The following attempts to explain the options that would normally be used in a message area menu, and how they are displayed in the default configuration.

Area Change

This command allows the user to change to another message area. The user will be prompted to enter the message area they want to go to, or to enter a "?" for a list of the areas that are available to them. The user can also enter the "<" or ">" characters to go to the previous or next area in the list, respectively. If entering a barricaded area where a password is required, he/she will be prompted for the password before they are allowed to enter the area.

Next

This command will display the next message in the current area. To keep reading messages in this direction, the user can press the ENTER key at the next prompt. The ENTER key will repeat the last N)ext or P)revious command.

Previous

This command will display the previous message in the current area. To keep reading messages in this direction, the user can press the ENTER key at the next prompt. The ENTER key will repeat the last N)ext or P)revious command.

Enter a Message

This command will allow the user to enter a message. After the user selects this command, Maximus will prompt them for some information is needs to know to send the message, such as who the message is to, the subject of the message, and whether the message is public or private, and any other information allowed by the configuration of the current area.

If the area does not allow public messages, or does not allow private messages, the user will not be able to select whether they want the message to be public or private.

If the area allows anonymous messages, the user will be able to change who the message is from as well. If the message area is a NetMail area, the user will be prompted for a network address to send the message to. When entering the address, the user can either enter the address, or use the following keys to get listings:

#: This will display a list of all the nodes in the current net.

/: This will display a list of all the nets in the nodelist.

It should be noted that Maximus will default to using your net, if you only enter a node number.

After entering this information, the user will be placed in the message editor to enter the message. For more details, refer to the section on message editors.

Reply to Message

This command allows the user to send a response to the author of the current message. The reply command is similar to the enter command, except that some of the message fields will be filled in (the name of the author of the message to which you are replying will automatically be inserted in the To: field). Also, once in the editor, the user will be able to QUOTE the message they are replying to. See the section on the message editor for details. "RK" [R)eply/K)ill] will kill the original message after the reply is finished.

Read Non-Stop

This command will allow a user to read all of the messages in the current area, starting with the current message, without pausing between each message. This is useful if users want to capture the messages to a disk file for later perusal.

Read Original

This command will allow the user to display the original message to which the message they are reading is a reply. Messages that are replies to another will have a "*** This is a reply to #xx" tag at the bottom of the message.

Read Reply

This command will allow the user to display any messages that are replies to the message they are reading. Messages that have replies to them will have a "*** See also #xx" tag at the bottom of them.

List

This command will show a list of all the messages in the current area (starting with the specified message) in a brief format, so a user can quickly select the message to read, without having to see the full text of each message. "LV" [L)ist Verbose] will display the subject of each message, in addition to the header information. "L*" [L)ist New] will list only messages after a user's lastread pointer for an area. "L?" (L)ist ?)help] will provide help on the L)ist command.

Scan

This command will scan all of the message areas a user has access to, and display any messages addressed to the user. There are several modifiers to the S)can command. These are as follows:

"SV" [S)can Verbose] will display the subject line of each message scanned, in addition to the normal header information.

"S*" [S)can New] will only display messages which a user hasn't received.

"SR*" [S)can Read New] will display the body of each message scanned, and give the user a chance to either reply to the message, see the message again, kill the message, or terminate the scan. This command activate the Maximus internal mail checker. The mail check can also be programmed into a separate option on the message menu, or used as an embedded command. Also, if you only want the Scan command to display NEW messages, in addition to displaying the body of each message, then you should use "SR*" command instead. See the control-file reference for details.

"S?" [S)can ?)help] will provide help on the S)can command.

Inquire

This command will allow a user to scan all of the messages in the current area for a pattern of text in the `To:', `From:', or `Subject:' fields. Entering a "?" at the inquire prompt will provide additional help on the I)nquire command. The following "wildcard" characters can be entered as part of the string to search for:

.: matches any character; i.e. "Th.t" finds "This" and "That"

*: finds zero or more occurrences of the previous character. For example, "a*b" would find "b", "ab", "aab", etc.

+: finds ONE or more occurrences of the previous character. For example, "a+b" would find "ab", "aab", "aaab", etc.

To find a LITERAL question mark, asterisk, period, or plus sign, precede it with a backslash. For example, "\?", "*", "\.", "\+". In addition, Maximus supports the character-set matching capability of UNIX's GREP utility.

Goodbye

This is identical to the G)oodbye command at the main menu. It will log off the user.

Main Menu

This will return the user to the main menu.

Kill a Message

This command allows a user to delete a message in the current area. Unless a user has Sysop privileges, he/she will only be able to kill messages which are TO or FROM them.

Upload a Message

This command allows a user to directly upload a text file as a message to the current area. This is identical to the E)nter message command, except instead of invoking the editor, Maximus will start an Xmodem upload. The user may then upload a pre-typed ASCII text file which will be stored as a message.

Forward

This command allows a user to make a copy of a message in the current area, and send it to someone else. The user enters the message number to forward, and the name of the person to forward it to. The user can also forward the message directly into another area by typing the area number when prompted. The F)orward command supports two special modifiers, as follows:

"FK" [F)oward K)ill] will delete the original message after forwarding it.

"FB" [F)orward Bombrun] will allow a user to specify a filename containing a list of users to forward the message to. In order for a user to use this command, their privilege level must be equal to the privilege level required to create a message from a file as defined in the Maximus control file (see the control-file reference for more information). In addition, the user cannot specify a filename of "CON" or "COM1" unless they are either local or have Sysop privileges. The format of the bombing run file is as follows:

```
<username> <dest_net/dest_node> [-x]
```

-x can be one of the following switches:

```
-h:  Message should be marked as HOLD FOR PICKUP
-c:  Message should be marked as CRASH
-n:  Message should be marked as NORMAL (default)
```

The bombing file can contain any number of lines.

The <dest_net/dest_node> and [-x] fields are only used for NetMail messages, and should be omitted for local bombing runs.

In the username field, spaces in a user's name must be represented by underscores.

For example:

```
Sysop                         225/337
Scott_Dudley                  249/106     -c
Hubert_Lai                    249/102     -h
Vince_Perriello               141/191     -n
Jesse_David_Hollington        225/1       -c
```

This would send carbon copies of a message to the five people names, sending the messages to Sysop and Vince Perriello as NORMAL messages, and the messages to 249/106 and 225/1 as CRASH messages.

Hurl

This command is used to move messages from one area to another. It will ask the user which message to hurl and which area to hurl it to.

In addition, you can define other options in the message menu to call auxiliary menus, display text files, or run external programs. See the control-file reference for more details on defining menus.

Message Entry

For ANSI and AVATAR callers, Maximus supports a sophisticated message entry screen. After selecting one of the options which begins the message entry process (such as E)nter, R)eply, or U)pload), Maximus will then display a "template" for the user to fill in. The template indicates whether or not the message is private or public, the name of the recipient, the recipient's matrix address (if in a netmail area), the subject of the message, and optionally, an "alias" or alternate name for the sender.

The user can move back and forth through the various items if they have an ANSI/VT-100 or IBM-compatible terminal emulator, and if not, users can also use the WordStar-like Control-E and Control-X keys to move up and down the fields (respectively). Control-Y will delete the contents of the current field, and pressing <escape> TWICE will abort the message. Assuming that all of the fields in the template have been filled, when the user presses <enter> on the last field, or uses the cursor to move off the bottom of the template, then Maximus will invoke the message editor.

Message Editors

Maximus supports two types of message editors:, MaxEd, the full screen editor, and BORED, the line-oriented editor. Maximus also supports an external editor for local use, but that is covered in the control file documentation.

MaxEd

MaxEd is the Maximus full screen editor. This can only be used by users who are capable of receiving ANSI or AVATAR graphics, and have a screen width of 80 columns and a length of at least 23 rows. The full screen editor has a number of advantages over the line editor, the most obvious being that it is far easier to use. MaxEd is more like a word processor than a BBS editor; you can use the cursor keys to move around, insert and delete text in the middle of paragraphs, and so on.

MaxEd uses a mixture of WordStar, generic VT-100 and IBM-PC commands, a list of which can be obtained by typing ^n (Control-N) from within the editor.

Also, if the message you are editing is a reply to another, then you can quote text from the original message, and place it inside your own, which greatly increases readability. You can look at four lines at a time through a "quote window", and optionally copy those lines into the message you are writing. You can also page through the original message in either direction, forward or backwards.

MaxEd also has a special menu, which is accessible via either ^kH (Control-K and then 'H'). or the <F10> key The options available on the MaxEd menu are similar to those found on the BORED menu, and includes the following:

Continue

> This will return to MaxEd and allow the user to continue entering the message.

To

> This allows the user to change the addressee of the message.

Subject

This allows the user to change the subject of the message.

From

This will allow the user to change who the message is from. The privilege level of this command should be set rather high, as this command can be used from any area, whether it's anonymous or not.

Handling

This is another command for which the privilege level should be set high. It will allow the user to change the flags on the message. Flags such as Private or Public can be changed, in addition to NetMail-type flags such as Crash, Hold or File Attach.

Read File

This will allow the user to enter a path to a file on your hard disk and read it in as a message. The privilege level of this command should be set fairly high.

BORED

BORED is the Maximus line-editor. BORED can be used by anybody, regardless of whether they have graphics or not. Most users who have graphics will most likely prefer to use MaxEd.

BORED allows the user to enter a message, one line at a time. When the user is finished entering the message, they are presented with the editor menu. The commands available on the editor menu are as follows:

Save

This will save the message the user has just entered.

Abort

This aborts the message without saving it.

List

Lists the message, preceding each line of the message with it's line number.

Edit

This command edits a line in the message, to correct any mistakes. Firstly, the user must select the line number that they wish to edit, then enter the text that they wish to replace, followed by the text you wish to replace it with. To insert at the beginning of the line, just press <Enter> for the "Text to replace" prompt.

Insert

This command will insert a blank line in the message preceding a specified line number.

Delete

This command will delete a specified line of the message.

Continue

Allows the user to continue entering their message, by appending to the end.

Quote

This command allows the user to quote text from the message they are replying to. The user must enter the starting and ending numbers of the lines that they wish to quote.

To

This allows the user to change the addressee of the message.

Subject

This allows the user to change the subject of the message.

From

This will allow the user to change who the message is from. The privilege level of this command should be set rather high, as this command can be used from any area, whether it is anonymous or not.

Handling

This is another command for which the privilege level should be set high. It will allow the user to change the flags on the message. Flags such as Private or Public can be changed, in addition to NetMail-type flags such as Crash, Hold or File Attach.

Read File From Disk

This will allow the user to enter a path to a file on your hard disk and read it in as a message. The privilege level of this command should be set fairly high.

The File Menu

The following attempts to describe the options that would normally be used in a file area menu, and how they are displayed in the default configuration.

Area Change

This command allows the user to change to another file area. The user will be prompted to enter the file area they want to go to, or to enter a "?" for a list of the areas that are available to them. The user can also enter the "<" or ">" characters to go to the previous or next area in the list, respectively. If entering a barricaded area where a password is required, they will be prompted for the password before they are allowed to enter the area.

Locate

This command allows a user to search all of the file areas for a particular filename or description. The text that the user enters will be matched anywhere in the filename or description, so wildcards are not required. There are a couple of modifiers to the L)ocate command.

"L*" [L)ocate New] will search all of the file areas for any files that have been uploaded since the user was last on the system.

"L?" [L)ocate ?)Help] will display help information on the L)ocate command.

File Titles

This command will display a list of files in the current area, along with their descriptions. New files will be flagged with a flashing asterisk (*). An an argument to this command can be specified in the same manner as for the L)ocate command, however F)ile Titles will only search the CURRENT file area.

Type

This command will allow a user to display the contents of any ASCII text file. The file is checked to make sure that it is an ASCII file, and the user is informed if it is not.

Goodbye

This is identical to the G)oodbye command at the main menu. It will log the user off.

Main Menu

This will return the user to the main menu.

Download

This will allow a user to download a file from the system. Maximus will ask the user to select a protocol and the name of the file to be downloaded. Once the user specifies the filename, Maximus will begin attempting to send the file to the user. Maximus has several built-in protocols, including Zmodem, SEAlink, Xmodem-1K (sometimes called Ymodem), and Xmodem. In addition, Maximus has ten "slots" into which Opus-compatible external protocols, such as Ymodem-G, ASCII and Kermit, can be inserted.

Upload

This is the reverse of the download command, and allows a user to send files to your system. Maximus will ask the user which protocol they are using to upload, and in some cases the name of the file they are uploading (if the user selects a batch protocol, such as Zmodem or SEAlink, the filename is transmitted automatically in the transfer, so Maximus won't bother prompting the user). The protocols are identical to those used for the Download command.

Statistics

This option displays the user's statistics, including the time the user has been online for the current call, the time online for the day, amount uploaded, amount downloaded, NetMail credit (if any), and so on. It will also display the CPS of the last file transfer.

Contents

The C)ontents command will allow a user to look into a compressed file and see what files are contained inside. The C)ontents can view any .ZIP, .ARC, .PAK or .LZH file. For other compression methods you are on your own.

Raw Directory

This will display a listing of ALL the files in the current file directory, not just the files listed in the files listing.

Override Path

This will allow the user to supply a path to a different directory than the one specified in the AREAS.CTL file for the current file area. This command should, for obvious reasons, be accessible only to users with high privilege levels. All changes made with this command are temporary, and the area's path will revert back to normal once you leave the area.

Hurl

This command will allow a user to move a file from one area to another. It will ask the user the name of the file to move, and the number of the file area to hurl it to. This command should be set to a rather high privilege level.

Kill File

This command will allow the user to delete a file from a file area. They will be asked for the name of the file to kill, and will then be asked to confirm that they want to delete it. If they answer "n" to the "Delete?" prompt, they will be given the option of leaving the file but simply removing the entry from the file listing. For obvious reasons, this command should be set to a high privilege level.

In addition, you can define other options in the file menu to call auxiliary menus, display text files, or run external programs. See the SILT Documentation for more details on defining menus.

The Change Setup Menu

From this section, a user may change as many of their settings as you choose to allow them to. Upon entering the change menu, the user's profile will be displayed. The menu options are as follows:

City

Allows the user to change his/her city.

Phone Number

Allows the user to change his/her phone number.

Real Name

Designed for an alias system. Allows user to change his/her real name.

Password

Allows the user to change their password. The user will be prompted to enter his/her old password, then enter his/her new password twice. If the user gets his/her old password wrong five times, he/she will be ignominiously disconnected. If the new passwords don't match, the password will not be changed. Users should be encouraged to change their passwords every so often to prevent other people from finding them out.

Help Level

Allows the user to change his/her help level. There are four different help levels available in Maximus:

NOVICE:	Full Menus.
REGULAR:	Abbreviated Menus.
EXPERT:	No Menus
HOTFLASH:	Full-screen, hotkey interface.

Nulls

Allows the user to change the delay after each transmitted line. Most users won't need to use this unless they are using a really ancient terminal.

Width

Allows the user to change the width of his/her screen. The screen width is used to determine where Maximus should word-wrap, and how wide the menus should be, among other things.

Length

Allows the user to change the length of his/her screen. The screen length is used to determine when the "More?" prompts appear.

Tabs

Allows the user to toggle the translation of tabs. Normally tabs are sent unaltered, which speeds up the display time marginally. If this option is off, tabs will be translated to spaces before being sent.

More

This allows the user to toggle the `More [Y,n,=]?' prompts on and off.

Video Mode

This allows the user to change Video modes. Currently, only TTY (plain), ANSI and AVATAR video modes are supported. More video modes will be added in future releases of Maximus-CBCS for compatibility with other systems.

Full-Screen Editor

This command allows users to toggle the use of the MaxEd full-screen editor.

Screen Clear

This allows the user to toggle the sending of screen clearing codes in case his/her terminal cannot handle the TTY clearscreen (ASCII character 12) or the ANSI `CLS' command.

IBM Graphics

This allows the user to toggle whether or not Maximus will send IBM extended ASCII characters. The IBM (and compatibles) have a special `extended' 8-bit character set, which allows things such as box-drawing and block graphics, while running in text mode. Most non-IBM systems do not support these extended ASCII characters. For those users that have this option turned off, Maximus will translate IBM extended ASCII characters into standard ASCII characters in the range of 32 (decimal) to 128.

Hotkeys

This option will allow the user to toggle his/her hotkeys setting, independently of the user's help level.

Quit

This will return the user to the main menu.

Due to the dynamic menu structure of Maximus, it is possible to configure the change menu to include any commands. The above are only the commands that are most often used in the change menu on most systems.

INSTALLATION INSTRUCTIONS

This section of the manual describes how to install Maximus from scratch, or converting to Maximus from a non-Opus BBS. If you are converting from Opus to Maximus, you should skip this section, and instead read the section entitled `Converting From Opus'.

Step 1: Where do I put these Files?

This is perhaps the easiest part of installing Maximus. Maximus is distributed in three separate parts: MAX102-1.LZH, MAX102-2.LZH and MAX102-3.LZH. You'll need ALL THREE FILES to install Maximus, since each file contains crucial system information.

The first step in the installation is to unarchive all three files into an empty subdirectory, using LHarc. (Obviously, since you're reading this file, you already know how to operate LHarc!)

Most of the files in the distribution kit are packed using a proprietary FIZ compression method, so you'll see a few files with a .FIZ extension, some documentation files, and the install program itself. The only way to unpack the .FIZ files is through the installation program, so that's what you should run next.

When run, the installation program will display some copyright information, and then proceed with the installation. It will ask you where to place the contents of each .FIZ file; unless you know what you're doing, or want to use a different directory structure, it is recommended that you use the defaults. To select the default path, simply press <enter> at each prompt. (The installation program will create directories which do not exist.)

After extracting each .FIZ file, the installation program will then ask whether or not you want to configure Maximus. If you type `Y' for YES, then the installation program will take you through the idiot-proof configuration procedure. This portion of the program is fairly straightforward; simply answer the questions which the installation program asks you. (Note: if you aren't a member of FidoNet, or don't yet have your system set up for network activity, then please answer NO to the `Are you a member of FidoNet' question. The answers you give here aren't permanent, and can be changed in the future.)

If you've installed Maximus using the idiot-proof configuration, then you can skip on to section 2, `Configuring Your Modem'. The rest of this section deals only with manual configuration.

On the other hand, if you answered NO to the `Configure Maximus' prompt, then you'll have to configure Maximus manually. Although this is covered in detail later in the installation section, you should create the following directories at this stage:

```
     \MAX\TEMP
     \MAX\MSG
     \MAX\FILE
```

All of these directories are needed to set up a basic Maximus system. (These should have been created automatically, if you used the idiot-proof configuration program.)

\MAX\TEMP is a special directory, and one that you should NEVER use. Maximus uses this directory at various times for internal purposes. Any files that you leave in this directory should be considered `fair game', since Maximus can and probably will play with it.

\MAX\MSG and \MAX\FILE are the starting points for your message and file areas. They will be discussed in greater detail later. For now, just create the directories and then leave them alone.

That's it for the first step of the installation. Wasn't that easy?

Step 2: Configuring your Modem

The modem is your gateway to the rest of the world. It can also be your biggest headache. Since this documentation obviously cannot cover all aspects of configuring and installing your modem, you should refer to the manual that came with your modem if you are experiencing difficulties.

To begin with, a Hayes-compatible modem is strongly recommended for use with Maximus-CBCS. Although you may be able to use Maximus with a modem which is not Hayes-compatible, this would not be an ideal situation.

If you do have a Hayes-compatible modem, but you are having problems getting it to work, chances are that it has to do with DCD, or Data Carrier Detect. DCD is a signal which your modem sends to your computer when it is connected to another modem. However, when most modems are shipped from the factory, they have DCD forced on all the time by default. Unfortunately, this is exactly the opposite of what most bulletin board programs require, Maximus included. Without having DCD set correctly, Maximus will not be able to tell when a user has hung up, and your mailer software may have problems handling incoming phone calls. There are two ways to change the way your modem handles DCD, depending on the type of modem you have.

If you own a 1200 bps modem, then chances are that your modem's DCD handling is controlled through DIP switches. These are those miniscule switches on the bottom or rear of your modem. (You may have to remove one or more panels on your modem to access these switches.) On most 1200-baud modems, DIP switch #6 toggles whether or not DCD is forced, and should usually be set in the UP position. (However, it is a good idea to check your modem manual, just in case.) Make sure that this switch is set so that DCD reflects the modem's actual state, instead of being forced on. Also make sure that your modem will send back verbal result codes (as opposed to numbers). The DIP switch which controls these result codes varies by modem manufacturer, so you will need to consult your modem's manual to determine which one to use.

If you own a 2400 bps (or greater) modem, then you will be spared the trouble of fiddling with DIP switches. Since almost all of these modems use non-volatile RAM (or NVRAM) instead of DIP switches, you can modify the modem's settings directly, using your favourite communications package. Once you load up your favourite communications package, type in the command `AT' and press <enter>, to make sure that your modem is on-line. If all is well, your modem should emit a response of `OK'. After you have established that your modem is okay, type in the command `AT&C1&S1&D2&W' and press <enter>. This will set your modem up so that in the future DCD will always reflect the modem's actual state, rather than always being forced on.

One other thing with which you may have problems is your modem cable. If your cable does not have the right number of signals wired through, then your modem may behave as though the DCD state is set incorrectly, regardless of DIP switch or NVRAM settings. One way you can verify your cable configuration is by borrowing the modem cable from a fellow SysOp who already has Maximus-CBCS running correctly. If you determine that your cable is at fault, you can go to your local computer store or service center, and ask them for a cable that has all seven RS-232 signals wired through properly.

Step 3: Installing a FOSSIL

Unlike most communications programs, Maximus does not handle its serial communication routines internally. Like BinkleyTerm and Opus, Maximus requires a FOSSIL. `FOSSIL' is an acronym which stands for `Fido/Opus/SEAdog Standard Interface Layer'. It is a program which will handle all of Maximus' low-level serial communication needs. This simplifies the use of Maximus on computer systems which do not exhibit 100% compatibility with standard serial hardware. However, it also means that you will need to find a FOSSIL package before you can begin using Maximus. Some of the more commonly-used FOSSIL drivers are X00, OpusComm, and BNU. If you can't find any of these programs on a BBS near you, then one can be usually be obtained from a local Software Distribution System ("SDS") node, or from the author's BBS. (See the `Notes From the Author' section for details on the author's BBS.) When you're searching for a FOSSIL driver, always make sure that the FOSSIL supports at least "FOSSIL Revision 5" or above, since Maximus won't work with lower FOSSIL versions.

There are two principal techniques use to install a FOSSIL. Some FOSSILs, such as OpusComm and BNU, are loaded as TSR (Terminate and Stay Resident) programs in your AUTOEXEC.BAT file. Others, such as X00.SYS, are loaded via your CONFIG.SYS file. Although different FOSSILs have different set-up instructions, it is fairly easy to install a FOSSIL for a basic configuration. Here are the basic installation instructions for the three most popular FOSSILs. These cover only a generic COM1: setup, so you should consult your FOSSIL manual if you are having problems getting your FOSSIL to work in your particular set-up.

OPUSCOMM Installation:

> To install OpusComm on COM1:, simply insert the following command at the beginning of your AUTOEXEC.BAT:
>
> ```
> OPUSCOMM
> ```
>
> Make sure that OPUSCOMM.EXE is on your current PATH, or else this will not work.

BNUCOM Installation:

> To install BNUcom on COM1:, simply insert the following command at the beginning of your AUTOEXEC.BAT:

BNU

Make sure that BNU.COM is on your current PATH, or else this will not work.

X00 Installation:

To install X00 on COM1:, simply insert the following command at the beginning of your CONFIG.SYS:

```
DEVICE=X00.SYS
```

Make sure that X00.SYS is in your C:\ root directory, or else this will not work.

If you have advanced requirements (such as if your modem is on COM2, or if you're running a 9600 bps modem), you should consult the documentation which came with your FOSSIL for more information.

Step 4: Editing Configuration Files

In order to run Maximus on your system, you will need to make several changes to your system set-up, especially in your CONFIG.SYS and AUTOEXEC.BAT files.

In the CONFIG.SYS file, you'll either need to EDIT the following items if they already exist, or ADD them to the end of CONFIG.SYS if they do not. To edit your CONFIG.SYS file, you may use either a text editor or a word processor in `non-document' mode.

The first change you have to make is to the `BUFFERS=' statement. (Again, if you don't already have a BUFFERS= statement, add it to the end of CONFIG.SYS.) If the system you are using is an XT or a PC, make sure that the line reads `BUFFERS=20'. However, if you are using an AT or an 80386, this line should read `BUFFERS=30' for improved performance. The BUFFERS statement controls the amount of space that DOS uses for buffering files that are being read from or written to disk. Setting the BUFFERS command to the wrong number may slow down your system, but Maximus will still function correctly.

The second change is that to the `FILES=' statement. Unlike the BUFFERS= statement, Maximus will not be able to run properly unless you have this statement set to a number GREATER THAN OR EQUAL TO 20. If you are running under a multitasking environment such as DESQview or DoubleDOS, then this number should be at least double that. In other words, if you have no multitasker, use `FILES=20'. However, if you are running a multitasker, you should use at least `FILES=40', or else Maximus will probably exhibit erratic behavior. Again, if you do not already have a `FILES=' statement, simply add it to the end of your CONFIG.SYS.

The third change you need to make is to add the line `DEVICE=ANSI.SYS' to the end of your CONFIG.SYS. If you look on your MS-DOS distribution disk(s), you should find a file called `ANSI.SYS' on one of them. After making the aforementioned change to CONFIG.SYS, copy ANSI.SYS from your DOS disk to the root directory of your hard disk.

NOTE: If you are using IBM-compatible hardware and wish to set Maximus up to use direct screen writes, you do not need to install ANSI.SYS. Please refer to the MAX.CTL documentation for more details, on the `Video IBM' command, since your system may or may not be able to handle this video mode.

Once you have made all of the above changes and saved the files, you should press <Ctrl-Alt-Del> to reboot your computer, so that the changes you made to CONFIG.SYS and AUTOEXEC.BAT will take effect.

Step 5: Setting Up the Control Files

This is perhaps the most complicated step in setting up your BBS. Maximus has three basic configuration files which control the operation of your system: MAX.CTL, AREAS.CTL, and MENUS.CTL. MAX.CTL is the main control file, and it controls almost everything about your system from the log-on screens displayed to the time `reward' given to users who upload. AREAS.CTL controls all of the message and file areas of your BBS, and MENUS.CTL controls all of the menus and options available on your BBS. The control files are all straight ASCII text, so to edit these files, you must either use a text editor, or a word-processor in a "non-document" or ASCII mode.

If you've installed your system using the idiot-proof configuration program, then all of the changes described in this chapter will have already been made for you. However, it is a good idea to read this anyway, since a lot of the information given here will apply to any future changes that you make.

There are a lot of fairly complicated commands in these control files which give you control over the most minute aspects of your BBS. Along with this power to tailor your BBS to your particular needs, there also comes the potential for hopelessly screwing things up. Therefore, if you are a new SysOp, it is advised that you refrain from playing too much with these files until you get your BBS up and running, and have become more familiar with the various options which are available.

The first step in setting up your control files is to edit only those portions which need to be modified, to get Maximus running on your system. Again, it is strongly recommended that you try to resist the temptation to change something other things in the control file. Please leave those extraneous changes for later. For now, you should make the following changes: (Note: these do not apply if you've used the idiot-proof configuration program.)

* MAX.CTL, line 46: Edit the `Name' statement to the name of your BBS. This name will be displayed in various places around the system, and it will also serve as the default origin line for EchoMail messages entered on your system.

* MAX.CTL, line 55: Edit the `SysOp' statement and insert your name, as you wish it to appear to users.

- MAX.CTL, line 98: Edit the `Path System' statement to reflect the drive on which you placed your \MAX directory. Since all of the other paths in MAX.CTL are relative to this one, you only need to specify the full path in one place.

- MAX.CTL, line 176: If you are using a DOS-compatible multi-tasking environment such as DoubleDOS or DESQview, uncomment the statement that reflects the multitasker you are running. (`Uncommenting' refers to removing the percent sign in front of a line, which tells the control-file compiler that you want it to process that line. Similarly, `commenting out' refers to placing a percent sign in front of the specified line, so that the compiler will NOT process that line.)

- MAX.CTL, line 229: There are a dozen or so `Output' statements in this section of the control file. These control which COMx: port you wish Maximus to use. If you are using COM1:, the default, then you can just leave these alone. Otherwise, uncomment the statement which reflects the COMx: port you wish to use.

- MAX.CTL, line 243: Edit the `Baud Maximum' line to reflect the fastest speed your modem can transmit at. Valid baud rates are 300, 600, 1200, 2400, 4800, 9600, 19200, or 38400.

- MAX.CTL, line 324: Edit the first `Address' statement to reflect your FidoNet address, if you are currently a member of FidoNet. If you do not belong to FidoNet or any other electronic network, then just leave this line alone.

- MAX.CTL, line 349: Edit the `Path NetInfo' statement to point to the directory where your `Version 6' nodelist files are contained, if applicable. This is ONLY necessary if you plan to send `NetMail' (also known as `Matrix Mail') using Maximus. If you do not intend to use that capability, then you should leave this statement alone. Maximus also supports two nodelist versions: you can tell if you are running with a version 6 nodelist if you see a file called `NODELIST.DAT' in your nodelist directory. If you have only a version 5 nodelist, you'll instead find a file called `NODELIST.SYS'. If you have a version 5 nodelist, you should COMMENT OUT the `Nodelist Version 6' statement, and UNCOMMENT the `Nodelist Version 5' statement. If you're using a version 6 nodelist (default), then you can leave the control file alone.

- MAX.CTL, line 1079: Somewhere around this point in the file, you should find a `Save Directories' command, followed by a string of letters. Those letters represent drive letters on your computer; if you have floppy drives using drive letters other than `A:' or `B:', or if you have any other types of removable drives, make sure to remove those drive letters from this statement.

Step 6: Compiling the Control Files

Once you have made all of the required changes to MAX.CTL, the next step is to `compile' your control files. If you make any changes to your control files and forget to compile them, Maximus will not realize that anything has changed, and will still run using your old configuration.

(Note: you don't have to follow any of the instructions here, if you've run the idiot-proof configuration program, since it compiled the control files for you. But again, it's a good idea to read this section, since you'll need to follow these steps the next time you make a change to the control files.)

Compiling your control files is easy; the program that compiles all of your control files is called SILT. Just type in the command `SILT <ctlname>' and press <Enter>, where <ctlname> is the name of your MAIN control file. If you're working with the default configuration, then `SILT MAX' should get you running and compile all three of the control files. (The main control file has an `Include' statement near the end, which tells SILT to read in MENUS.CTL and AREAS.CTL automatically. You cannot compile either AREAS.CTL or MENUS.CTL individually, so you must always give SILT the name of the main control file.) The first time you run SILT, it will probably complain that a few directories didn't exist, and that it is creating them for you. You need not worry, since this is perfectly normal. If you have made any errors in the control file, such as misspelling a keyword, then SILT will warn you about those too. If you have made any errors, restart your text editor, move to the specified line number, fix the error, and then recompile using SILT.

Step 7: Starting Maximus

Once you have compiled your control files, you are finally ready to start Maximus. Although your BBS is still be pretty bare-bones, it will at least be usable.

To start up Maximus for the first time, CHDIR to the \MAX directory, and type in the command: `MAX -c'. The `-c' switch tells Maximus to create a new user file, and won't be normally needed after the first time you run Maximus.

After a bit of disk activity, Maximus should display a copyright banner, and print out a message about the lack of an existing user file. Maximus will then display the prompt: `Your Name Here [Y,n]?'. Type the letter `Y' to answer the prompt, and continue your login.

After doing this, Maximus will display a bit of text which describes your BBS. This text is contained in a file called APPLIC.MEC, in your \MAX\MISC directory. (Files with an extension of .MEC will be discussed in greater detail later in this document.)

Maximus will also prompt you for a few pieces of information, including your city, phone number, and password. Maximus will also prompt you for ANSI screen controls, the MaxEd editor, and IBM-PC characters. Answer `Y' to all three of these prompts.

After Maximus has finished the interrogation, it will display a welcome screen and a bulletin file, and will then finally place you at the main menu. All of these screens are completely redefinable. Such customization will be described later in this manual.

Now that you have Maximus working, you will probably want to look around for a while. Feel free to play around, and explore the different features of your new BBS system. If you would like to be able to send some test NetMail, first try going into the user editor and giving yourself some matrix credit.

When you have finished looking around at your new BBS, type `G' from any menu to log off, and Maximus will exit back to the DOS prompt.

Step 8: Setting Up Batch Files

Since Maximus-CBCS will not answer the telephone by itself, or `talk' with other FidoNet systems, you need a front-end mailer to handle things such as incoming calls. There are about five or six mailers on the freeware/shareware market, and three or four on the commercial market, so you have plenty to choose from. Although setting up your front-end mailer is beyond the scope of this document, you will find several sample batch files for different mailers in one of the appendices.

Although your mailer's documentation will probably not have any specific details for connecting itself to a Maximus system, you can safely follow any instructions that have been provided for hooking your mailer up to Opus, since Maximus uses the same command-line interface. When Maximus starts up with a caller on-line, it expects to find a minimum of one parameter: `-b<speed>', where <speed> is the baud rate of the caller currently on-line. If you can manage to pass this information from your mailer to Maximus, then you are pretty well set.

Once you are able to get Maximus started from your front-end mailer, you are halfway there. When Maximus finishes execution, it needs a way to tell your batch file what to do next. For example, if a user entered an echomail message, you may want to run a utility to process the just-entered message, or you may want to run some sort of logging utility. To accomplish this, Maximus sets a numeric variable in DOS, which is called an `errorlevel'. As mentioned, Maximus has several errorlevels to juggle for various events, such as a user entering an echomail message, a user entering a netmail message, logging off before the user enters a name, etc. You'll note that in several places throughout the control file, you can tell Maximus which errorlevel to use in certain situations. Errorlevels are always numeric, and always have a value from 1 through 255. (However, Maximus reserves errorlevels 1 through 4 to indicate errors, so you shouldn't use these in the control file.) The control file comes pre-set with certain errorlevels, so unless you have a special case, you shouldn't need to modify these.

Once Maximus is set up to use errorlevels, the only other task is to make your batch file detect the errorlevel Maximus used, and react accordingly. Errorlevels can be detected in a batch file quite easily, through the use of the `If Errorlevel <e> <a>' statement. <e> is a number, and should indicate the actual errorlevel to check for, and should also correspond to the errorlevel specified in

MAX.CTL. <a> specifies an action, which is simply a normal batch file command.

However, there is one item which you should pay special attention to: DOS process all errorlevels using a greater-than-or-equal-to operation. In other words, the statement `If ErrorLevel 10 echo Hi!' will display the word `Hi!' if the errorlevel was set to 10, but will ALSO display `Hi!' if the errorlevel was set to 11, 12, or above. For this reason, if you have more than one errorlevel to check for, you should always arrange the group of errorlevel statements in a DESCENDING order. For example, to check for errorlevels 1, 3, 9, 10, 11 and 12, this would be the proper way to place the statements in your batch file:

```
     MAX -p%1 -b%2 (other cmd-line parameters here)

     If ErrorLevel 12   rem Do op `A' here.
     If ErrorLevel 11   rem Do op `B' here.
     If ErrorLevel 10   rem Do op `C' here.
     If ErrorLevel  9   rem Do op `D' here.
     If ErrorLevel  3   rem Do op `E' here.
     If ErrorLevel  1   rem Do op `F' here.
```

The only other thing to remember is that ALL programs modify the DOS errorlevel. If, in the example given above, you were to run a program called (for example) ABCD.EXE for errorlevel 12, ABCD would RESET the DOS errorlevel. Since the batch file is executed one line at a time, the errorlevel statements which follow would use the errorlevel set by the ABCD program, rather than the value set by Maximus. To get around this DOS limitation, you must instead use a GOTO statement for the `action' portion of the errorlevel statement.

The GOTO command allows your batch file to jump to a completely different location within the same batch file, and start executing commands from that point. This is accomplished by using a statement in the form of `GOTO <l>', where <l> is a LABEL. A label is a unique, alphanumeric, ONE-WORD name, which indicates where in the batch file you wish to jump to. Some valid labels could be called `DoScan', `Heres_The_Scoop', and `ScanBLD'. However, a GOTO statement alone is not enough to complete the GOTO operation. You must also place the same label somewhere else within the batch file, which lets DOS know where the GOTO statement should end up. You can do this simply by placing a colon (`:') at the beginning of a line, simply followed by the label name.

For example, the following sample batch file:

```
Line 1:   :BigLoop
Line 2:   echo This will be shown repeatedly
Line 3:   goto BigLoop
```

...would cause the line `This will be shown repeatedly' to continuously display on the screen, until the user hits control-C to abort. (Omit the `Line X:' tags when typing this in.) When DOS starts the batch file, it will process each line in sequence. When DOS reads line 1, it will recognize that `BigLoop' is simply a label definition, and will ignore it. Next, DOS will read line 2, and process the ECHO command, by displaying `This will be shown repeatedly' on the screen. Once DOS encounters line 3, it will realize that it contains a GOTO statement, and will parse the label `BigLoop' out of the command. Having done that, DOS will then scan for the prior `:BigLoop' label, and jump back to line 1, thus continuing the cycle.

However, the GOTO command does have practical applications. The above example could be re-written like this:

```
     MAX -p%1 -b%2 (other cmd-line parameters here)

     If ErrorLevel 12   goto OpA
     If ErrorLevel 11   goto OpB
     If ErrorLevel 10   goto OpC
     If ErrorLevel  9   goto OpD
     If ErrorLevel  3   goto OpE
     If ErrorLevel  1   goto OpF

:OpA
REM * Do operation `A' here.
goto End

:OpB
REM * Do operation `B' here.
goto End

:OpC
REM * Do operation `C' here.
goto End

:OpD
REM * Do operation `D' here.
goto End

:OpE
REM * Do operation `E' here.
goto End

:OpF
REM * Do operation `F' here.
goto End
```

```
:End
```

In this situation, DOS would first compare the errorlevel returned by Maximus to those listed in the `If ErrorLevel' portion of the batch file, and then jump down to the corresponding label. For example, if Maximus exited using errorlevel 10, DOS would jump down to `:OperationC', and process the statements which follow. The REM statement is simply a remark, and is ignored by DOS. (Typically, there would be one or more commands placed in that portion of the back file, instead of the REM command.) After processing the REM command, DOS then reads the next line of the batch file, and processes it. The `goto End' statement is necessary, to make sure that DOS doesn't keep going, and execute the commands for `OperationD' as well. (Recall that DOS just ignores LABEL DEFINITIONS, such as `:OperationD'. Without the extra `goto End', the batch file would just `fall through' to the statements under OperationD, OperationE, etc. The extra goto specifically instructs DOS to jump to the `:End' label, which is located at the end of the batch file. On the other hand, some times you may WANT the batch file to `fall through', since it allows one to have several similar commands to be processed, when using the same errorlevel. Fortunately, cases where intentional fall-throughs are needed are few and far between.)

Arranging the batch file like this allows for more than one command to be executed for a certain errorlevel, and in addition, gets around the above-mentioned problem of other programs changing the errorlevel.

Maximus' errorlevel numbers are entirely configurable, but if you are using the default set-up, then the following errorlevels will be used:

- Errorlevel 255: This means that Maximus terminated with an undefined error condition. Your batch file should return to your front-end mailer.

- Errorlevel 16: If Maximus exits with this errorlevel, it indicates that an internal stack error was generated by the Turbo C run-time. Your batch file should to return to the mailer.

- Errorlevel 12: This indicates that a user entered EchoMail and perhaps also NetMail during this session. In response, your batch file should call whatever scanner/packer program you use, such as QM SCAN PACK, or ConfMail and oMMM. After calling the scanner/packer, your batch file should then restart your mailer.

*   Errorlevel 11: This indicates that a user entered NetMail but no EchoMail during this session. In response, your batch file should call your mail packer, such as QM PACK or oMMM. After calling the packer, your batch file should then restart your mailer.

*   Errorlevel 5: This indicates that a user logged off without entered either EchoMail or NetMail during his/her session. In response, your batch file can execute a program such as the SCANBLD mail database utility. Alternatively, you may have your batch file simply restart your mailer.

*   Errorlevels 4 through 1: These four errorlevels are used to indicate an error condition, and should simply cause your batch file to restart your mailer. (In addition to errors, errorlevel 2 is also used to indicate that the caller has hung up before entering their name, during the log-on sequence.)

A generic batch file for Maximus might look something like this:

```
rem * This is where you call Maximus itself.  Change
rem * the `%1' and `%2' as necessary, to make Maximus
rem * work with your mailer.  (See the appendix on Sample
rem * Batch Files for more information.)

Maximus -b%1 -p%2
if errorlevel 255 goto Error
if errorlevel 16  goto Error
if errorlevel 12  goto EchoMail
if errorlevel 11  goto NetMail
if errorlevel 5   goto Nothing
if errorlevel 4   goto Error
if errorlevel 3   goto Error
if errorlevel 2   goto Error
if errorlevel 1   goto Error

:EchoMail
rem * Invoke scanner here.
rem * This should FALL THROUGH to the next NetMail exit,
rem * since you want to both scan AND pack during this
rem * stage.

:NetMail
rem * Invoke packer here.  This should FALL THROUGH
rem * to the Nothing exit, since you want your
rem * After-Each-Caller utilities to be run after each
rem * caller, regardless of whether or not they enter
rem * any messages  This is why there is no `goto End'
rem * command at the end of the `EchoMail' portion, just
rem * as there is no such command at the end of this
rem * section, either.

:Nothing
rem * Invoke after-each-caller utilities here.
goto End

:Error
rem * Something funky happened, so let's say so.
echo There has been an error!

:End
rem * This should return to your mailer.  If
rem * you're using Binkley's BBSBATCH method, then
rem * this should probably be just an `exit'
rem * statement.  If you're using FrontDoor or
rem * another mailer, this should probably be a `goto'
rem * which jumps to the main portion of your batch
rem * file.
```

Finally, since this section pertains to batch files, it would be a good idea to read the chapter on installing SCANBLD (in the `Maximus Utility Documentation' section), if you intend to use the internal mail-checker. Although the mail-checker will work without SCANBLD (albeit somewhat slowly), it is strongly advised that you install SCANBLD before you allow users to access the mailchecker. You may experience unpredictable and bizarre results without it, depending on your hardware/software combination.

Step 9: About Priv Levels and Locks

Unlike other BBS programs, Maximus does not use numbers to represent a user's access level. Instead, Maximus uses a series of keywords, which are called `Priv Levels'. Listed in descending order, the following privilege levels are currently supported:

Hidden (special, see note below)
Sysop
AsstSysop
Clerk
Extra
Favored
Privil
Worthy
Normal
Limited
Disgrace
Twit

All of these privilege levels, except for `Hidden', can be applied to an ordinary user or menu command. The`Hidden' privilege level is a special case, and if applied to a user, it ensure that the user will NOT be able to log onto your system, since Maximus will disconnect as soon as the user enters his/her password. Similarly, setting a menu option or a message/file area priv of `Hidden' means that nobody can access that option - not even the Sysop. This is useful for hiding commands that you don't want even yourself to be able to access.

The only privilege level which confers special capabilities is that of `Sysop'. For example, users with the privilege level of Sysop can read private messages in any area, regardless of who the actual addressee is. It should be noted that simply having your name listed in the `Sysop' section of the control file does NOT automatically confer Sysop privileges upon you. Your actual user profile, contained in the USER.BBS data file, must have your privilege level set to `Sysop'.

The remaining privilege levels do not confer any special capabilities and can be assigned to any users you wish, regardless of what the name of the privilege level implies. The privilege levels that are required to access menu options and message/file areas are controlled in MENUS.CTL and AREAS.CTL, respectively. These two files will be discussed later in this document.

When first setting up your BBS, try and define a set of rules for using privilege levels. For example: "First-time callers get a privilege level of DISGRACE, validated users get a privilege level of NORMAL, and users who have donated money to the Sysop receive a privilege level of PRIVIL." If you don't lay out a plan for assigning privilege levels when you first start out, you will find it very easy to lose track later in the game of who has access to what.

Privilege levels are not the only way to control user access to various areas or menu options. Maximus' has a lock and key system. Using Maximus' locks, you can give specific users access to certain areas or options, independently of their privilege level. Once an option or area is `locked' with a specific lock number, a user must have the same key number to access that particular option or area. Valid lock/keys are numbered from 1 to 8. To add a lock to a message/file area or a menu option, simply add a forward slash after the privilege level, followed by the lock numbers you wish to use. To illustrate, an area with an access level of `Privil/167' would be accessible to only those users whose privilege level was at least `Privil', and who had keys 1, 6, and 7.

Step 10: Customizing *.BBS Files

Now that Maximus is functioning, you are probably interested in customizing the screens and menus, and getting the BBS ready to handle callers. Your first step will probably be to modify the welcome screens and information files which came with the default Maximus package.

Almost all of the files which are displayed to the user are stored in the \MAX\MISC subdirectory, and this is where you will be doing most of your customization work. Different files are used for different purposes, and each file has its own unique name. For example, the first screen displayed to a new user is stored in a file called APPLIC.BBS, the bulletins are stored in BULLETIN.BBS, etc. You can change these filenames in MAX.CTL if you want, but it will be simpler if you leave the names alone for now. Each file which can be displayed to the user ends with the extension `BBS'. However, you will not be working directly with these files. Just like the Maximus *.CTL files, screens to be displayed to the user must be COMPILED before they can be displayed. Although it is possible to directly enter text into the *.BBS file, it is usually much easier to edit the `source code', which is contained in a file with an extension of `.MEC', which is an acronym for `Maximus Embedded Commands'. However, if you really have a burning urge to insert MECCA codes directly into the *.BBS files themselves, a list of the MECCA codes and their translations can be found in the MECCA command language reference, which is located towards the end of this manual.

The advantage to using MECCA, instead of simply creating files with ANSI graphics, is two-fold: for starters, you can imbed user- or system-specific information into any screen displayed to the user, which gives your BBS a personal touch. Secondly, MECCA allows you to directly enter colour tokens and cursor controls. The advantage to this is that Maximus will STRIP OUT these colour and cursor controls for callers who don't support them, which makes the *.ANS and *.ASC kludges of other BBS programs unnecessary. Only one file is needed for any given menu, and Maximus will convert it on-the-fly for callers with or without ANSI graphics, with or without the ability to display IBM graphics characters, or any combination of the above. This greatly reduces the time required to maintain your BBS, and saves disk space, too.

A *.MEC file is composed of straight text which is displayed to the user, plus some optional `embedded commands'. If you want to see an actual *.MEC file, start up your text editor, and load the file `\MAX\MISC\NEWUSER2.MEC'. As you can see, the file is mainly composed of straight ASCII text, but with a few special commands inserted in, mainly to control colour and to perform special functions such as displaying the user's name.

Generally speaking, anything in a *.MEC file which is NOT inside a pair of square brackets is treated as straight text, and is therefore displayed to the user without altercation. Anything which IS enclosed in square brackets is called a `token', and is interpreted specially by Maximus. Tokens have various functions, which can range from changing the colour of the text, to running an external *.EXE or *.COM program, to invoking the internal mail-checker. Although you can see quite a few of the tokens being used in the *.MEC distribution files, a complete list of these tokens is available in the MECCA documentation, in the Maximus Technical Reference manual. A complete walk-through to creating a MECCA source file is given in that same documentation, so you should at least read the first few pages of that section.

Once you have finished creating or modifying a *.MEC file, you must then compile it using MECCA, the Maximus Embedded Command Compiler (Advanced). Compiling a file with MECCA is easy. Simply type in the command `MECCA <filename>', where <filename> is the name of the *.MEC file you wish to compile. For example, to compile the file called `APPLIC.MEC' into the file called `APPLIC.BBS', type in `MECCA APPLIC'. MECCA will then compile the specified file, and warn you if you made any errors.

It's also a good idea to test your creations, before allowing your users to see whatever dreadful mistakes you may have made. Chances are that your compiled *.BBS files will occasionally have problems, and users might get stuck in an endless loop (or something equally embarrassing), if you've been especially careless with some tokens. The Oracle utility will allow you to view *.BBS files off-line, without needing to start up Maximus to view them. Please see the section on Oracle for more information.

Step 11: Customizing Msg/File Areas

The next step in customizing your bulletin board system is to set up your own message and file areas. Although the Maximus distribution kit came pre-configured with three message and three file areas, you will probably want to expand beyond this, particularly if you are a member of FidoNet and carry a number of EchoMail conferences.

The first thing you should know is that all message and file areas are defined in a file called AREAS.CTL. You can have up to a total of 1,296 areas on your system. However, it is usually a good idea to start with a small number of areas, and then create new ones only as the need arises.

Each area definition in AREAS.CTL is demarcated by a starting \`Area <area_no>' line, and an ending \`End Area' line. Everything between those lines pertains to that specific area only, and has no effect on other areas in the control file. <area_no> specifies the area \`number' that is being defined. Note that this is a bit of a misnomer. The area \`number' doesn't necessarily have to be a number. It is actually a label which can be up to two characters long. These characters may be a number from 0-9, or a letter from A-Z. (There is no difference between uppercase and lowercase numbers.) Therefore, \`1', \`2', \`A', \`XY', and \`A1' are all valid area \`numbers'.

Inside each area definition are a sequence of keywords which define what the area looks like. Although there are other advanced options available, you really only need a dozen or so basic keywords to define a message or file area. If you wish to see a complete list of the options which can be used in defining a message or file area, please see the section entitled "Control File Reference", in the Maximus Technical Reference Documentation.

Please note that each area definition does NOT necessarily need to have both a message AND a file area associated with it. If you do not want a specific message area to be accessible as a file area, simply omit the defining statements for file areas, and vice versa.)

THIS KEYWORD IS REQUIRED FOR ALL AREAS:

Access <priv>

> This statement is REQUIRED for all areas, no matter what type of area it is. This statement defines the access level, that a user must possess to access this area. You can define separate access levels for the

message and file areas if you like. Information on how to do this is contained in the AREAS.CTL reference, in the Maximus Technical Reference Manual.

THESE ARE ONLY REQUIRED FOR MESSAGE AREAS:

MsgInfo <desc>

This statement tells Maximus what to describe this message area as to the user.

Local <path>
EchoMail <path>
Matrix <path>

Depending on what type of message area this is, you will use ONE of the above three statements to tell Maximus the type of messages that are in this area and in what subdirectory the messages are located. The area type can be LOCAL (Messages entered in this area stay on your system alone.), ECHOMAIL (Messages entered in this type of area are broadcast to other systems who are participating in the EchoMail conference.), or MATRIX (Messages entered in this area are sent directly to the destination node, sometimes referred to as `NetMail'.). <path> specifies the directory in which the messages are located. Note: Each message area must have its own separate subdirectory! Yes, that is correct. You must have a separate subdirectory for each message area, just as you must have a separate subdirectory for each file area. SILT will create this directory for you if it does not already exist.

Private Only
Public and Private
Public Only
Read-Only

Again, you can use only one of these four statements in a given message area. These commands tell Maximus which type of messages to allow users to enter in this area. The first three statements perform as expected. Users may enter only private messages, they may be given the option of entering either private messages or public messages, or they may enter only public messages. For most users, a read-only message area means that they cannot enter any messages. Instead, they can only read existing messages. This is useful for those sysops who want to set up a message area in which he/she can post bulletins for users to read. Obviously, there must be some way for messages to be entered, or users would have nothing to read. Users with a with a privilege level of at least AsstSysop are permitted to enter messages in read-only areas. Users below AsstSysop will receive the message, `This is a read-only area', and Maximus will not allow them to enter a message.

THIS IS ONLY REQUIRED FOR ECHOMAIL MESSAGE AREAS:

MsgName <tag>

This keyword tells Maximus what the `echo tag' of the current message area is. This tag should be the same as the one which you have defined for this area in the control file for your EchoMail utility, usually called AREAS.BBS.

THESE ARE ONLY REQUIRED FOR FILE AREAS:

FileInfo <desc>

This statement tells Maximus what to describe this file area as to the user. You can add some descriptive comments if you like, so long as all of the text will fit on one line.

Download <path>

This statement defines the subdirectory in which the files for this file area are contained. In other words, this is where users will be able to download files from.

Upload <path>

This statement defines the subdirectory in which uploads for this file area will be placed. You have two options for defining an upload path:

*   Set it to the same subdirectory as the download path. This means that users should be in the correct area when they upload files, since the file will be available for download from the specified area as soon as the user finishes uploading.

*   Set ALL upload paths in ALL file areas to point to the DOWNLOAD path for area 0, which is normally accessible by only the sysop. This is the most secure option, since it allows the sysop to check files which are uploaded before they are put on-line. Only after the sysop has checked the file out and Hurled it to the appropriate area is the file available for download by users. This also means that users can upload a file of any type anywhere and not have to worry about getting

it in the correct area, since there is only one area for uploads.

There is one area name in the control file which has a special significance, and that is area zero, or `Area 0'. When logging off, a remote caller will be asked if they want to leave a comment to a Sysop, and if they do leave a comment, area 0 is where the comment will be placed. If you do not wish to let users leave a comment to the Sysop when logging off, simply delete the definition for area 0. TIP: For ease in replying to users' messages, define the path for Area 0 to be the same path as you use for your local private message area. By doing so, you can reply to messages in that area, and have your users will still be able to read them even though you have not Hurled them into an area which is accessible to them.

One last reminder: Don't forget to recompile your control files with SILT after you make changes to them!

Step 12: Maintaining File Areas

Although message areas are easy to create (by simply adding the appropriate area definition to AREAS.CTL), file areas require a bit more maintenance. Not only do you still need to create the area definition, but you also need to create a listing of files which are available for download in each file area.

If you have no files to go into a particular area, then you don't have to do anything. Maximus will create a file catalog as needed, when a user uploads a file.

However, if you already have some files you'd like to place in a certain file area, the task is a bit more involved. For starters, CHDIR to the DOWNLOAD directory for the file area, as you specified it in AREAS.CTL. When you get there, just copy all of the files you wish to have in that file area into the current directory.

Once you've done that, to initially create a FILES.BBS catalog, simply type in the following command at the DOS prompt to create a file catalog.

```
     FOR %F IN (*.*) DO ECHO %F >> FILES.BBS
```

You should see a bit of disk activity as commands are executed for you, and then the DOS prompt should re-appear. You have now created the basic file catalog, but you are not yet done. Start up your text editor, and load in FILES.BBS, the file you just created. Inside this file, you should see a list of the files in the directory. If you wish to add a comment to a particular file, you can just add a space after the filename, and insert your comment there. (You can use up to forty-eight characters if you wish to keep the comment on one line. If the comment is any longer, then Maximus will automatically wordwrap it onto the next line. You can make the comment as long as you want, up to 255 characters in length.) Here are the contents of a sample FILES.BBS for a hypothetical file area:

```
TEST.ZIP     This is a text file
ABCDEFGH.TXT This is another interesting text file
ACKTHPPT.ZIP A digitized Bloom County comic strip
```

If you want to add files to your catalog after performing the initial `FOR %F' command, you can simply use your text editor to place the filename on a new line of FILES.BBS, followed by the description. Similarly, to delete a file from the catalog, just delete the line containing the file entry you wish to get rid of. You will also need to delete the actual file from the subdirectory.

When using the default `File Date Automatic' setting in the control file, Maximus will automatically place the file size and date beside the filename, in addition to adding a bit of colour to the catalog.

Finally, you may add comments to the FILES.BBS listing which are NOT specifically related to one file. If the FIRST character on a line is alphanumeric, then Maximus will treat the line as a file entry. Otherwise, Maximus will treat the whole line as a comment and display it to the user, with all of the usual *.BBS file colours and commands still usable. Also, if the first character on a line is a hyphen, then Maximus will set the colour to WHITE and display that line as a comment.

Step 13: Customizing Menus

Maximus' menus are completely redefinable and therefore they can sometimes get fairly complex and difficult to manipulate. If you are just starting out as a new sysop, it is not advisable to mess around with the menus a lot, apart from changing a few basic features. If you are new, one of the few things which are safe to play around with is the access level for each command. This is usually located in the second or third column of the line for each menu option. The other field which is safe to change is the `Command as it appears to user' field. This is the name of the command as it will appear on the menu to the user. Remember that the first character in the command is what activates that option, so make sure that no more than one of your commands uses a given letter as its first character.

Once you have become accustomed to the Maximus menu system, you can, customize your menus by moving some commands to different places, or adding or deleting options. However, if you are just starting out, it is best that you leave the menus alone for a while.

The last thing which is safe to play around with is the `MenuFile' option. By using this command, you tell Maximus that instead of generating a `canned' menu from the options you have specified, it should display the customized *.BBS file specified after the `MenuFile' keyword. This will allow you to greatly customize your BBS, and make it look completely different from any other BBS, whether or not the other BBSs are running the same software. Refer to the section entitled `Using Custom Menus' for more information on this topic.

Step 14: Miscellaneous Information

You have now completed the installation of Maximus-CBCS. Although you are now officially finished, there are a few things you should think about and keep in mind:

* Renumbering Utility: If you take part in any large EchoMail conferences, then a utility of this nature is crucial. Maximus comes bundled with a message-renumbering utility called RENUM which can be used to renumber and trim your message areas based on a variety of specifications. Documentation for the renumbering programs is provided separately in RENUM.DOC. If you prefer to use the original version of Bob Hartman's RENUM program, then make sure to specify the area by using the subdirectory rather than the area's number. The reason for this is that the original version of Bob Hartman's RENUM does NOT support Maximus' new AREA.DAT file, and can therefore not keep the users' lastread pointers up-to-date. It is strongly recommended that you use the Maximus-specific version of RENUM that is supplied with the distribution package. It is a drop-in replacement for Bob Hartman's RENUM and does support all of Maximus' new features. Also see MAXREN.DOC.

  Again: both of the renumbering utilities perform similar functions, so you only need to use ONE of the above-mentioned utilities. The only functional difference, besides what was mentioned above, is that MaxRen will not delete messages. (If you want to use MaxRen, you'll need a separate utility to do the deletion, such as ConfMail's MAINT feature.) Otherwise, they both function identically.

* If you are having trouble installing Maximus, chances are that you have not followed these instructions to the letter, or that you have forgotten to use a FULLY-QUALIFIED PATH in at least one place. Since Maximus repeatedly changes its current directory as it executes, you can NOT make any assumptions about what the current directory will be at any given time.

HOW TO UPGRADE FROM OPUS 1.xx TO MAXIMUS

If you are already running Opus 1.03, installing Maximus should be a snap. Converting from Opus 1.1x may take a little more work, but will be almost as easy. At this stage in the game, Maximus and Opus 1.03 are virtually "plug-compatible," which means that you can still use most of your old Opus 1.03 utilities. Maximus will also use most of your existing Opus 1.03 configuration files, either directly or by converting them. Maximus does not directly support any Opus 1.1x data files; however, they can all be converted to a Maximus-compatible format. In addition, Maximus can also directly display any Opus *.BBS files, and includes an OECC and OACOMP-compatible compiler, which allows you to compile old *.OEC files, in addition to creating your own, which can also utilize some of the new Maximus extensions to the OECL language.

Note, however, that Maximus does not have any built-in mail processing capabilities, and will not even answer the phone by itself. A front-end mailer, such as BinkleyTerm, is required for Maximus to function properly.

First, before doing anything else, make a backup of your old BBS files. If, for some reason, you can't get Maximus working right away on your system, you'll want something to fall back on.

To start off the installation, check the archive that you received Maximus in. Maximus was originally distributed in a .LZH-type archive, called MAX_100.LZH, although it may have been re-packed to use a different type. The first step is to extract all of the files contained within the archive, whichever type it may be, into a blank directory. When everything is extracted, you should end up with a file called README.1ST, the documentation files, CONVERT.DOC (the file you're reading now) and two other, more detailed manuals, a program called INSTALL.EXE, and possibly several other files.

For now, we'll focus mainly on the INSTALL program. The purpose of INSTALL is to extract the Maximus system files, contained in files with an extension of .FIZ, to the appropriate directory. To start off, you should run the install program, and press ENTER at each prompt, to select the default path. (Answer `N' to the `Configure Maximus now?' prompt.) You can move the files into another directory (such as your previous \OPUS directory) later, but it is best to first place Maximus in it's own directory, since some of Opus' filenames are identical to Maximus'. If you just press ENTER at each prompt, then INSTALL will drop the appropriate files into the correct directory.

The first thing you should do with the control files is to move them into the directory where you keep your SYSTEM*.BBS or SYSTEM*.DAT files. In most cases, this will be your main \OPUS subdirectory. At any rate, it is the directory specified by the "Path System" statement in your Opus control file. In addition, if you are involved in sending or receiving EchoMail, and if you're running Opus 1.03, you should make sure a ConfMail-style AREAS.BBS or an Opus ECHO.CTL file exists in this same directory. If you keep this file in a different directory, temporarily copy it into the same directory as your SYSTEM*.BBS files. (These files are not required for the Opus 1.10 conversion)

Once you have done this, you should take a plain ASCII text editor (you can use EDLIN, which is on your DOS disk, if you don't have anything else), or a word-processor in NON-DOCUMENT mode, and edit the MAX.CTL file to suit your system configuration. The MAX.CTL file is pretty self-explanatory, and you should be able to use your Opus control file as a reference. Anything extra contained in MAX.CTL, which you can't find in your Opus control file, you should ignore for now. You can add frills later, once you get the basic system working. (A verbose list of all of Maximus' control-file keywords, and in-depth explanations of those keywords, can be found in the Maximus Technical Reference Manual.)

After editing MAX.CTL to suit your tastes, double-check to make sure that the Opus-style SYSTEM*.BBS or SYSTEM*.DAT files, and for the Opus 1.03 conversion, also make sure that either AREAS.BBS or ECHO.CTL are in the current directory. Once you have done that, you should run the conversion program, called PIPER. PIPER takes the area information in either your SYSTEM*.BBS files (for Opus 1.03), or your SYSTEM*.DAT files (for Opus 1.10), and AREAS.BBS/ECHO.CTL files, and creates AREAS.CTL, which is the file Maximus needs for its area information.

*   ADVANCED USERS: PIPER takes the path names and flags (i.e, Read-Only, EchoMail, etc.) for each area from the SYSTEM files, and for Opus 1.03, the area names from DIR.BBS files in each subdirectory, and the EchoMail tag name information from the AREAS.BBS or ECHO.CTL file. (Opus 1.10 includes this information in its SYSTEM*.DAT files, so the other files are not required.) You may also want to edit the AREAS.CTL file later to use certain advanced area control features, such as locks and extended barricades.

At this point, you should compile your control files. Simply type "SILT MAX" at the DOS prompt. SILT will inform you of any errors that it encounters in the control files. If errors are detected, correct them and repeat this step.

* ADVANCED USERS: There is one other Maximus control file which you may want to edit. MENUS.CTL is the file that contains the Maximus menu definitions. The MENUS.CTL provided with the distribution archive should give you a menu setup much like that of Opus. If you wish to customize your menus, however, this can be done by editing the MENUS.CTL file with any text editor. See the comments inside MENUS.CTL, and the control-file documentation for more details.

If all goes well, SILT should create the following files:

AREA.DAT	Message/File Area defn's
AREA.IDX	Message/File Area index
MAIN.MNU	The main menu
MESSAGE.MNU	The message menu
FILE.MNU	The file menu
CHANGE.MNU	The change menu
EDIT.MNU	The edit menu
MAX.PRM	The main, compiled .PRM file
VER14.PRM	An Opus 1.03-style .PRM file
VER17.PRM	An Opus 1.10-style .PRM file

The next thing you should do is find your USER.BBS file, and run the provided CVTUSR utility on it. If you're converting from Opus 1.03, all you need to do is to change to the directory where USER.BBS is kept, and type the command `CVTUSR -o103' at the DOS prompt. If you're converting from Opus 1.10, simply type `CVTUSR -o110' instead.

* ADVANCED USERS: CVTUSR initializes some fields in the USER.BBS file which Maximus uses. These fields were previously unused by Opus 1.03. (The Opus 1.10 and Maximus user files are not directly compatible, so CVTUSR will write out a new USER.BBS, and leave your old Opus 1.10 USER.DAT alone.) However, the Opus 1.03 and Maximus user files are virtually 100% compatible, in that you can still use programs which access the Opus 1.03 user file. However, if you use an external user manager, there are a couple of things that you must remember. First of all, you will not be able to access the extended Maximus features, such as the extra privilege levels, the user keys, and so on, while using a non-Maximus-aware user editor. In addition, there is one thing you should never do to the user file with an external manager, and that is to add a new user. Doing so will cause Maximus to cross-link that user's lastread pointers with another user, which is not desirable. (Should you accidentally add a user with a non-Maximus-compatible program, you can re-run CVTUSR with the `-m' parameter, to fix the cross-linked

pointers.) Also, but only to a lesser extent, you should not delete users through a non-Maximus-aware user editor. Although the consequences aren't as severe when you delete a user, it will cause a one-time mix-up of lastread pointers, for NEW users.

Maximus supports the direct transmission of raw AVATAR graphics to the user. In order for this to work, all *.BBS files that were previously ANSI must be converted to AVATAR. To accomplish this, there is a utility included in the distribution archive, called ANSI2BBS. This program will convert ANSI files to AVATAR specs. You should use this utility to convert your ANSI files to AVATAR. If in doubt as to which is which, simply run it on all your *.GBS files. Note: ANSI2BBS has some problems in dealing with "super-animated" ANSI files, particularly those which use a lot of animation. If the converted *.BBS file doesn't look quite right when displayed, then you can do one of two things: 1) Try de-animating the file. There are some ShareWare ANSI editors around (such as TheSoft's TheDraw), which will let you read in an ANSI file and undo any animation. Then try running the non-animated ANSI file through ANSI2BBS again. 2) If step #1 fails, then you have no choice but to simply use the ANSI file, without converting it to AVATAR. Although the file will look icky on the local end, if you're using the `Video IBM' direct-writing video mode, it should look okay remotely

*   ADVANCED USERS: If you used the Opus Embedded Command Compiler (OECC) with Opus, your *.BBS files will already be in AVATAR format. If you used utilities such as TheDraw to create your screens, they are most likely ANSI.

Included in the distribution archive are two separate utilities for renumbering message bases: One is a patched version of Bob Hartman's RENUM program, which should be used for DOS versions 3.3 and below. The other is Peter Fitzsimmons' MaxRen, which can be used for all DOS versions, including DOS 4.0. These utilities are identical to their Opus counterparts, except they correctly update Maximus' lastread pointers.

*   ADVANCED USERS: The patched RENUM handles the new format of lastread pointers that Maximus uses. Maximus keeps two different sets of lastread pointers, which allow for an unlimited number of pointers for the users. Maximus maintains the LASTREAD file for the SysOp in each message area, which is identical to the one used by Opus 1.03, msged, and several other utilities. This file keeps the lastread pointers for the user who resides in record zero of the user file. This is usually the SysOp. Maximus also maintains a LASTREAD.BBS file in each message area, which contains

> the lastread pointers for all the other users. As a result, users can have in each and every message area they access, instead of only the ten message areas which Opus 1.03 will keep track of. This is why a patched version of RENUM is necessary. If you use the normal version of RENUM, the user's lastread pointers will not be updated correctly when you renumber areas. Also, when using Bob Hartman's RENUM, make sure to specify the area's name by PATH, rather than area number, or else the lastread pointers will not get updated correctly. It is, however, okay to use area numbers with Peter Fitzsimmons' MaxRen.

Maximus uses an embedded command language similar to that of Opus and OECC, only more advanced. The program MECCA is 100% backwards-compatible with OECC, and is approximately 30% faster to boot. If you used OECC under Opus, then simply replace OECC.EXE with MECCA.EXE, and rename *.OEC to *.MEC. You can still use OECC with Maximus if you really want to, since the basic control codes used by the two are the same, but then you won't be able to access any of Maximus' new MECCA keywords/tokens.

Also, Maximus uses an external utility known as SCANBLD to create data files in each message directory, which speed up the time required to do a scan or mail-check of all message areas. This utility should be run on all local and EchoMail areas after messages are entered. SCANBLD is not required. However without it, the scanning and mail-checking features will function much more slowly. Please see the section on SCANBLD for more installation details.

Maximus should run properly in a basic configuration with only slight changes to your Opus batch files. However, since Maximus is unlike Opus in that it cannot run as a standalone BBS/mail system, certain modifications may need to be made. Maximus has no built-in mail capabilities and will not answer the phone by itself. A front-end mailer, such as BinkleyTerm or FrontDoor, is required to run Maximus.

Maximus is called from the command line or from a batch file in a format almost identical to that of Opus. Although a complete list of command-line parameters can be found later in one of the appendices, the basic format for executing Maximus is as follows:

```
     MAX [<prm>] [-p<port>] [-b<baud>] [-t<time>] [-k]
```

<prm> is the name of the compiled *.PRM file for Maximus to use. If none is specified, then MAX.PRM is used by default. <port> is optional, and tells Maximus which COM port to use, if not the one specified in the control file. <baud> specifies the baud rate of the on-line caller. This

argument must ALWAYS be specified if you wish to run Maximus in remote mode. <time> specifies the maximum amount of time, in minutes, that a user is allowed to remain on-line for. Unlike Opus 1.03, the <time> parameter actually works, so you can now include it in your batch file. `-k' specifies that you wish to load Maximus in keyboard (aka LOCAL) mode. If the <baud> parameter is not specified, then Maximus will load in local mode anyway, so the `-k' is actually redundant.

Examples:

```
     C> MAX
```

This would run Maximus in local mode, defaulting to MAX.PRM in the current directory for the name of the parameter file.

```
     C> MAX C:\Config\MAX -p1 -b2400 -t30
```

This would run Maximus using MAX.PRM in the C:\Config directory, with a caller online on COM1 at 2400 baud, with 30 minutes until the caller is forced off-line.

```
     C> MAX -p2 -b1200
```

This would run Maximus with a caller online on COM2 at 1200 baud, and would default to looking for MAX.PRM in the current directory.

```
     C> MAX BBS -p1 -b300
```

This would run Maximus with a caller online on COM1 at 300 baud, and would look for BBS.PRM in the current directory.

To set up Maximus in your batch file, all you need to do is to replace the Opus command line with the Maximus command line. For example, with BinkleyTerm's SPAWNBBS.BAT:

```
     Remove  `OPUS -p%2 -b%1'
     Insert  `MAX -p%2 -b%1 -t%3'
```

This essentially completes the Maximus installation. A couple of other quick notes follow:

a) For the sake of being concise, Maximus will only display the one WELCOME.BBS file, between the password prompt and the main menu. If you wish to have your BULLETIN.BBS file displayed after the WELCOME.BBS file, as Opus does, then you will have to insert the command `[onexit]D:\Path\Bulletin' at the top of your WELCOME.MEC file (substituting in the correct name and path for your bulletin file), and then compile it using MECCA. By placing another such statement at the top of your BULLETIN.BBS file, you can display an unlimited number of files between the password prompt and the main menu. (If you only have a WELCOME.BBS file, or if you didn't use MECCA or OECC to create your welcome screen, instead insert the text `^oFd:\path\bulletin' on the first line of the welcome screen. This is the compiled equivalent of the above-mentioned `[onexit]' token. Please note that `^o' represents a Control-O, while the `F' is a normal uppercase letter.)

b) SILT is capable of generating an Opus 1.03 or 1.10 *.PRM file, in case you need it for running some Opus-specific external programs. SILT is also capable of generating the Opus 1.03-style SYSTEM*.BBS files that may be required for an external utility, such as a file-area manager. However, if you don't need an Opus-compatible parameter file, or your SYSTEM*.BBS files for anything, you may delete them. Please see the section on SILT operation, for details on how to create SYSTEM*.BBS files.

c) At this point, if you plan to still call your main BBS directory `\OPUS', so you don't have to modify all of your other batch files, you can now start to move all of the Maximus files into the appropriate subdirectory. You can probably directly copy everything from \MAX to \OPUS, since these are mainly executable files, which do not use the same filenames as Opus. You can also probably delete everything in \OPUS\HLP too, since the Maximus help files, in \MAX\HLP, are more descriptive, and easier to modify. The only tricky part is in moving the files in \MAX\MISC into \OPUS\MISC. Since you probably already have a number of customized *.BBS files, you will only want to move the static files, or those which you haven't changed. Most of the following files should be also copied into your \OPUS\MISC directory, since they have been updated to reflect several new features of Maximus:

```
BADLOGON.*
NOTFOUND.*
WHY*.*
NOMAIL.*
CANTENTR.*
NOSPACE.*
FFORMAT.*
SHELL*.*
ACTIVE_2.*
NUM_M.*
QUOTES_M.*
TIMEWARN.*
```

d) Make sure that the FOSSIL you are using supports at least "FOSSIL Revision 5" or above. Maximus needs some of the extended commands which were introduced with revision 5, and will not run correctly with a lower FOSSIL version.

e) Maximus starts displaying the LOGO.BBS file on the SAME LINE as the "Maximus-CBCS vX.YY" banner, so it is now possible to insert extra information on that line. However, to have your LOGO.BBS file to appear as it did when using Opus, you'll have to insert two carriage returns at the beginning of your LOGO.BBS.

f) Caveat SysOp: Make sure that you have specified FULL paths and FULL filenames everywhere you can. Unlike Opus, Maximus doesn't stay in the same directory throughout execution, for speed reasons. This means that Maximus probably won't be in the right directory to call a *.BBS file specified with only a relative path name. SILT and Maximus will compensate for this, by trying to expand "relative" paths to their full equivalent path, if possible. This cannot be done in all circumstances, so double-check all your paths and filenames if you're having problems.

This completes the formal Maximus installation. Now that you are done, and after you have tested the system to make sure that it works, you may want to move some of the Maximus files that don't have to be in your main directory to other areas. It is usually a good idea to keep your main directory as uncluttered as possible, by creating subdirectories to store seldom-used files.

You should now take the time to at least skim through the rest of the documentation to get a feel for Maximus. Again, Maximus and Opus are very close, but Maximus contains a number of advanced features that Opus does not have, all of which are covered in the documentation.

MAXIMUS UTILITY DOCUMENTATION

ACCEM Operation Guide

ACCEM does the reverse of what the MECCA utility does: it takes a compiled .BBS file, and converts it back to a human-readable .MEC file. This can be useful if you've lost the source for one of your .BBS display files, or if you're trying to change a compiled .BBS file which someone else has given you.

Once you've run ACCEM over a .BBS file, you can freely edit the resulting .MEC file, and recompile it as you wish. ACCEM can convert a .BBS file back to the complete .MEC file. The .MEC file created with ACCEM should be identical to the original .MEC file, with one small exception: since label names aren't stored in the .BBS file, MECCA will simply convert these into numeric labels in the form of `/L0', `/L1', `/L2', etc. However, the reverse-engineered .MEC file will still compile correctly, and after compiling, the output from the new .MEC file should be identical to the original .BBS file.

The format for ACCEM is:

```
     ACCEM <infile> [outfile] [-s]
```

<infile> is the name of the .BBS file to convert. If no extension is given, then ACCEM will automatically use an extension of .BBS.

[outfile] is the name of the .MEC file to write to. If no extension is given, or even if [outfile] is omitted, then ACCEM will default to the <infile> filename, but using an extension of .MEC.

[-s] tells ACCEM to split lines which are over 100 characters. Using this will make ACCEM place an empty brace at the end of each line, thereby limiting the length of lines in the .MEC file, but without affecting the .BBS output. This is useful if you're decompiling a .BBS file with some very long lines, and if your editor can only display a limited number of columns.

For example, to convert TEST.BBS to TEST.MEC, all of the commands following would work equally well:

```
     ACCEM TEST
        or
     ACCEM TEST.BBS
        or
     ACCEM TEST.BBS TEST
```

If one wanted to split lines over 100 characters in length, the following would work, too:

```
     ACCEM TEST -s
     ACCEM TEST.BBS -s
     ACCEM TEST.BBS TEST -s
```

ANSI2BBS and ANSI2MEC

ANSI2BBS and ANSI2MEC are two programs which will process a file containing ANSI graphics, and convert it into a file displayable by Maximus. ANSI2BBS will convert a file containing ANSI graphics directly into a .BBS file, which can be immediately displayed by Maximus. On the other hand, ANSI2MEC will convert a file with ANSI graphics into a file containing MECCA commands, as opposed to the compiled AVATAR sequences which are generated when ANSI2BBS is run. ANSI2BBS is useful for a one-time translation, when you're sure that the output will look right. ANSI2MEC is useful if you wish to display a file containing ANSI graphics, but also want to add some special effects, such as customizing the screen with MECCA tokens, or adding menus. After running ANSI2MEC and making any changes, the .MEC file must then be compiled into a .BBS file through MECCA, the Maximus Embedded Command Compiler. Please refer to the MECCA command language reference manual for more details on the operation of MECCA.

The format for ANSI2BBS (and ANSI2MEC) is as follows:

```
     ANSI2BBS <infile> [outfile]
        or
     ANSI2MEC <infile> [outfile]
```

<infile> is the name of the input file which contains ANSI graphics. If no filename extension is specified, then ".ANS" will be used by default.

[outfile] is the name of the file to place ANSI2BBS/ANSI2MEC's output in. If no output filename is specified, then ANSI2BBS will add a `.BBS' extension to the input filename, and send the output to that file. ANSI2MEC will do similar, except it will use an extension of `.MEC' instead.

Although ANSI2BBS and ANSI2MEC will try do the best job they can when converting an ANSI file, due to some ambiguities in the ANSI cursor-movement syntax, it is not always possible to correctly convert all ANSI graphics files. ANSI2BBS and ANSI2MEC will particularly have problems with some `highly-animated' screens, including some of TheDraw's alternate scanning modes, such as `Diagonal', `Gate', `Squiggle', etc. Maximus can handle almost all straight-through ANSI files, so unless you're using one of those scanning modes, then you shouldn't have any problems.

However, once you have converted an ANSI screen, it is a good idea to put it in a place where Maximus can access it, and test it in local mode, or with the Oracle utility. If the file didn't convert correctly and has some semi-bizarre formatting glitches, then you have two choices:

* If the file is animated, load the file using TheDraw, an excellent ANSI screen editor by Ian Davis, and turn off the animation by pressing Alt-J and then `N' to convert the drawing to normal mode. Then try ANSI2BBS again, and hope it works.

* Convert the file using ANSI2MEC, and try to edit the MECCA tokens to fix the problem, and then compile the .MEC file using MECCA.

* Leave the file as-is, and send straight ANSI codes to the caller. Although it won't be viewable by AVATAR or TTY callers, and it will look icky if you have the `Video IBM' statement enabled, it should work okay for remote ANSI callers, if you enclose the graphics inside `[colour]' and `[endcolour] tokens.

CVTUSR and User File Conversions

CVTUSR is a utility which will allow you to convert foreign user files into a format Maximus can handle, from several other popular BBS programs. In addition, CVTUSR is also capable of generating an Opus 1.10-style USER.DAT file, FROM Maximus' own USER.BBS. Not only can you use CVTUSR to switch from another BBS program to Maximus, but you can also use CVTUSR to create an Opus 1.10-style user file, which means that you can run some of the newer Opus programs, as well.

The command-line format for CVTUSR is:

```
     CVTUSR [-m] [-o103] [-o110] [-q] [-x110]
```

`-m' tells CVTUSR to reset the lastread pointers in a Maximus-style USER.BBS. This option is normally only used to fix cross-linked lastread pointers, and should only be used when needed, as this may reset the lastread pointers of some or all users.

`-o103' tells CVTUSR that you're converting an Opus 1.03-style USER.BBS to a Maximus-style USER.BBS. Be warned that this conversion procedure will overwrite your old Opus 1.03 user file, so you should make a copy if you want to save it. All fields in the Opus 1.03 user file are retained. However, please note that the Maximus user file is still backwards-compatible with Opus 1.03. Almost all of your 1.03 utilities will work, as will 1.03 user editors and such. When "converting" an Opus 1.03 user file, CVTUSR is simply initializing several fields which Opus never used, to allow for several of Maximus' added features.

`-o110' tells CVTUSR that you're converting an Opus 1.10-style USER.DAT to a Maximus-style USER.BBS. This procedure will convert almost all of the Opus 1.10 fields, with the exception of the expiry dates, personal welcome screens, and any utility-specific fields which may be stored in the user file. Your old USER.DAT is NOT overwritten by this procedure, so you don't need to make a copy of it.

`-q' tells CVTUSR to convert a QuickBBS-style USERS.BBS to a Maximus-style USER.BBS. This conversion function isn't as complete as some of the others; it will leave out things such as ANSI graphics and "More [Y,n]?" prompting. However, the next time the user logs on, Maximus will ask for all the information which couldn't be converted, so the loss is minimal.

`-x110' tells CVTUSR to convert the Maximus-style USER.BBS to an Opus 1.10-style USER.DAT. CVTUSR will translate all of Maximus' fields into the equivalent Opus 1.10 fields, and will also attempt to store some Maximus-specific information in some "spare" fields. This means that it MAY be possible to convert the Maximus USER.BBS to the Opus USER.DAT, run a program which modifies the Opus version of the user file, and convert it back. Although this theoretically should work without problems, it isn't advised, and doing so may cause some fields to be lost in the Maximus portion of the user file.

EDITCALL Documentation

EditCall is a small utility which was written to dummy up the `number of callers' count contained in BBSTATxx.BBS. This program is useful if you've recently changed from another BBS package, and want to set the caller count to reflect the actual number of callers to your system.

The command-line format for EditCall is:

```
     EDITCALL <task_num> [num_calls]
```

<task_num> should indicate the task number whose caller counter you wish to set. If you're running only one line, then use 0 for <task_num>.

[num_calls] should indicate the new number-of-calls variable you wish to set for the specified task. If you don't specify this parameter, then EditCall will simply display the number of callers for the specified task number, without changing anything.

MECCA Operation Guide

MECCA is a companion utility which will compile *.MEC input files into binary *.BBS files, which can then be displayed by Maximus. MECCA is 100% backwards-compatible with Opus' OECC utility, and can compile any preexisting *.OEC files without problems.

The operation of MECCA itself is fairly simple. The command-line format is:

```
MECCA <infile> [outfile] [-t]
```

<infile> is the name of the input file, and if no extension is specified, an extension of `.MEC' will be used by default. <infile> ca include wildcards, so entering `MECCA *.MEC' is perfectly valid.

[outfile] is the name of the compiled output file. This parameter is optional, and if not specified, it defaults to <infile>, using an extension of `.BBS'.

In other words, typing `MECCA BULLETIN' would cause MECCA to try to compile the file `BULLETIN.MEC' into a file called `BULLETIN.BBS'.

[-t] tells Maximus to compare the date stamps of the input and output files, and to skip the current file if the output filename is newer than the input filename. This is useful for recompiling an entire directory of .MEC files, if you can't remember what has changed. Simply type `MECCA *.MEC -t', and MECCA will automatically scan all of the files in the current directory, and recompile those which have changed.

Documentation on the internal format of a *.MEC file itself, and the keywords used therein, is contained in the MECCA Command Language Reference, in the Maximus Technical Reference Manual.

ORACLE Operation Guide

ORACLE is an off-line .BBS file viewer. Unlike other BBS programs with embedded command languages, Maximus allows you to view compiled .BBS files without logging on, while still having the screens displayed exactly as they would be through Maximus itself.

The command-line format for ORACLE is:

```
     ORACLE <bbsfile> [[-x]...]
```

<bbsfile> is the name of the compiled .BBS file you wish to view. If no extension is supplied, then .BBS will be used by default.

`-x' can be any of the following command-line parameters:

-hX Sets the current help level to `X', where `X' is the first letter of a valid help level. Valid options are `-hN' (Normal), `-hR' (Regular), `-hE' (Expert), and `-hH' (Hotflash).

-i Disables high-bit IBM characters. With this option enabled, ORACLE will automatically translate IBM Extended ASCII to the ASCII equivalent.

-kX Sets the user's keys to X, where X is simply a listing of keys to assign to the user. For example, using `-k1237' would give keys #1, 2, 3 and 7 to the user.

-mX Sets the local video mode to X, where X is one of `D' (DOS), `F' (FAST), or `I' (IBM). By default, ORACLE will use the video mode defined in the control file. However, if you wish to use ORACLE from remote, it may be necessary to use the DOS video mode, since output from the IBM and FAST video modes normally cannot be redirected to a COM port.

-pX This tells ORACLE to read the Maximus .PRM information from the file `X'. If no PRM file is specified, then ORACLE will default to using MAX.PRM, in the current directory. THIS PARAMETER IS REQUIRED!

-q This option enables the `quick' hotkey mode.

-slX This option sets the virtual screen length to `X' rows. This doesn't change your physical screen length; however, it does determine when the `More [Y,n,=]?' prompts are displayed. This option defaults to 24 lines.

-swX This option sets the virtual screen width to X columns. This doesn't change your physical screen width; however, it does change the screen width, and controls when virtual screen wraps will occur.

-t The -t parameter forces Oracle into TTY video mode. This will disable all ANSI and AVATAR graphics commands, and display the file just as it would be shown to a TTY caller.

-vX This sets the user's privilege level to `X', where `X' is the first letter of a valid priv level. For example, `-va' would set the user's priv level to AsstSysOp, while `-vl' would set the user's priv level to Limited.

In addition to the ability to specify the above parameters on the command-line, you can also semi-permanently set these options through an environment variable. Instead of typing all of the parameters on the command-line, you can simply place the same options into the ORACLE environment variable.

For example, issuing the following sequence of commands:

```
     SET ORACLE=-pD:\Max\Max2.Prm -vS -q
     ORACLE D:\Max\Misc\Bulletin
```

...is identical to entering all of this at once:

```
     ORACLE D:\Max\Misc\Bulletin -pD:\Max\Max2.Prm -vS -q
```

However, although the first example looks like more typing, you can easily place the SET command into your AUTOEXEC.BAT, and only have to type `ORACLE <filename>' for each future file you wish to display, while still using all of the defaults set in your AUTOEXEC.BAT.

SCANBLD and Mail Checking

SCANBLD is a utility which will greatly enhance the performance and speed of Maximus' S)can command, and also the speed of the internal mailchecker. At a glance, this performance increase can be up to 10,000% in large EchoMail areas.

SCANBLD works by creating a database of all of the messages in each area of your system. Although SCANBLD has to be run after certain events, including after running a message renumbering utility, after processing EchoMail, etc., the speed increase while checking for mail should more than justify this minor disadvantage. Although the mailchecker CAN be used without ever running SCANBLD, it will be relatively sluggish, and may take several minutes to scan the message areas on a large system.

The command-line format for SCANBLD is as follows:

```
SCANBLD <user_bbs> <area_dat>   [[All │ Local  │ Matrix  │
     Echo │ @<afile> │ <areaname> │ !<areaname> │ /x]...]
```

Only the <user_bbs> and <area_dat> parameters are required.

<user_bbs> specifies the location and name of your USER.BBS file. <area_dat> specifies the location and name of your AREA.DAT file.

After the above two mandatory parameters, any of the following commands can appear, in any order:

All: Specifies that SCANBLD is to scan ALL areas, regardless of what type of message area it is. This is the default, and all areas will be scanned if no parameters are specified.

Local: Specifies that SCANBLD should scan LOCAL areas.

Matrix: Specifies that SCANBLD should scan MATRIX areas.

EchoMail: Specifies that SCANBLD should scan ECHOMAIL areas.

@<afile>: Specifies that SCANBLD should read the named file, which should contain a list of message area tags. SCANBLD will compare those tags to those specified for the `MsgName' keyword in each area, and process those which match. This file can be Maximus' own

ECHOTOSS.LOG, or it can be the import data files produced by ConfMail or QM.

<areaname>: Specifies a specific area number for SCANBLD to process.

!<areaname>: Specifies that this area is to be excluded from a normal scan, and is not to be processed. This is useful if you have two separate area numbers pointing to the same physical message path, or if you want to exclude certain areas from one of the above EchoMail/Matrix/Local scans.

/C: Forces SCANBLD to do a full compile of each area processed. By default, SCANBLD will normally try to update the mail database in the areas processed without rebuilding the entire area. You should ALWAYS use this option after renumbering messages, or else the message database will become out of sync with the actual messages.

/ND: Informs SCANBLD that you do NOT want the @<afile> specification to be deleted after processing. This is useful if you have other utilities which need the specified file, even after SCANBLD has finished with it.

/Q: This switch forces SCANBLD into quiet mode. Instead of displaying each area's statistics, SCANBLD will instead display a single hash sign (`#') for each area processed.

The options specified on SCANBLD's command-line are cumulative, so entering the following:

```
scanbld user.bbs areas.dat echo matrix 45 !22 @et.log
```

...would cause SCANBLD to process all EchoMail areas, in addition to all NetMail areas, plus area number 45, plus the areas listed in the ECHOTOSS.LOG-format ET.LOG, with the exception of area number 22.

It is recommended that you use SCANBLD as follows, to assure that the mail database receives all necessary maintenance, and remains synchronized with the actual messages:

AFTER A USER ENTERS ECHOMAIL (usually errorlevel 12):

```
scanbld user.bbs area.dat local matrix @echotoss.log
```

AFTER A USER ENTERS MATRIX MAIL (usually errorlevel 11):

```
scanbld user.bbs area.dat local matrix
```

AFTER A USER ENTERS LOCAL MAIL (usually errorlevel 5):

```
     scanbld user.bbs area.dat local
```

AFTER IMPORTING ECHOMAIL:

```
     scanbld user.bbs area.dat local matrix @echotoss.log
```

AFTER RUNNING ANY MESSAGE-RENUMBERING UTILITY:

```
     scanbld user.bbs area.dat all /c
```

Finally, after using an external message editor, you must SCANBLD all of the areas which you entered messages in. If your editor can produce an ECHOTOSS.LOG-like file, then you should run SCANBLD after your editor, using the command shown for `If a user enters EchoMail'. On the other hand, if your external editor does not produce an ECHOTOSS.LOG (or similar) file, then you must scan all areas, using the following command:

```
     SCANBLD user.bbs area.dat ALL
```

IF THESE INSTRUCTIONS ARE NOT FOLLOWED TO THE LETTER, then SCANBLD may miss messages which would otherwise be flagged as new mail.

SILT Operation Guide

SILT is a utility similar to Opus 1.03's OPUS_CTL, and Opus 1.10's SALT. It takes the raw ASCII control files which you have created, and turns them into something Maximus can use. SILT can also optionally parse only part of your control file, and it can also generate several structures for Opus compatibility. (Such as the Opus version 14 & 17 *.PRM files, and SYSTEM*.BBS.)

Starting SILT is fairly easy; the command syntax of SILT is:

```
SILT <ctl_file> [-a] [-m] [-o]  [-p]  [-s103]  [-s110]  [-u]
     [-x]
```

<ctl_file> specifies the name of the control file you want SILT to process, and is the only required argument. If only the name of the control file is given with no other arguments, then SILT will process everything EXCEPT the SYSTEM*.BBS files. Otherwise, SILT will only process the parts of the control file which are given on the command-line. When specifying the control file, do not include the .PRM extension.

`-a' tells SILT to compile only AREAS.CTL.

`-m' tells SILT to generate the *.MNU files from MENUS.CTL.

`-o' will cause SILT to sort message/file area numbers in alphanumeric order, which will make the `A?' areas display look slightly more ordered, and will make the `A>' and 'A<' (Area Next and Area Previous) commands work in proper sequence. Normally, SILT will write the areas to the data file in the order it sees them, unless this switch is specified. Not specifying `-O' is useful for changing the order in which Maximus changes between areas, and for changing the order of the `A?' area display. Normally, all of the areas in your control file will be in numerical order anyway, so this shouldn't present a problem. However, if you have used message- and file-area definitions in separate blocks in the control file, then using the `-O' option may be necessary to have the area lists display correctly.

`-p' will instruct SILT to generate MAX.PRM, and if you requested them, the Opus version 14 and 16 *.PRM files.

`-s103' tells SILT to create SYSTEM*.BBS files for Opus 1.03 compatibility, in addition to compiling only AREAS.CTL.

`-s110' tells SILT to create SYSTEM*.DAT files for Opus 1.10 compatibility, in addition to compiling only AREAS.CTL.

`-u' tells SILT to run in "unattended mode". Normally, SILT will prompt the user for input in certain situations, including when a specified directory doesn't exist. (In such a case, SILT would ask the user whether or not s/he want to create the directory.) Using `-U' will tell SILT not to stop to ask for directions, and to create any nonexistent directories.

`-x' will cause SILT to compile everything, INCLUDING the SYSTEM*.BBS and SYSTEM*.DAT files.

RUNNING EXTERNAL PROGRAMS

Although Maximus itself offers a large selection of internal features, chances are that you'll want to execute programs OUTSIDE of Maximus while a user is on-line. Maximus can run almost all types of external programs, from customized file-transfer protocols, to `door' programs written for another BBS program.

Maximus can execute as many external programs as you wish, from either a menu option, or from a MECCA embedded command file. Since these two pieces compromise the whole of the Maximus software, it means that you can run any external program anywhere, and at any time.

Execution Methods

In general, Maximus supports four different methods of running external programs. You can determine what type of exit you need by looking at the list below, and comparing the methods' advantages and disadvantages to the requirements of the program you wish to run.

DOS: This is the so-called `normal' exit type. Maximus will load a second copy of COMMAND.COM, and then run the external program.

* This is the only way to run a batch file as an external program.

* Since COMMAND.COM has to be loaded, your external program will have about 160K less space to work in. (150K/Maximus + 10K/COMMAND.COM = 160K)

* If the program is located on the DOS PATH, no explicit path needs to be given.

* You can execute internal DOS commands using this method, such as DIR, TYPE, CHDIR, etc.

RUN: This exit is identical to `DOS', except that COMMAND.COM is NOT loaded. This means:

* Your program will have a bit more memory to run in, since a second COMMAND.COM isn't in memory.

* You cannot run a batch file with this command.

* This method will run faster than the DOS method, because COMMAND.COM doesn't have to be loaded.

CHAIN: This command is just like `RUN', except the external program will be loaded ON TOP of Maximus. In other words, the external program will overlay Maximus in RAM. NOTE: this execution format may not work with all hardware and software combinations.

* Since the program is loaded on top of Maximus, it will be able to use all available system memory.

* This is a one-way command, and control will not be passed back to Maximus when the program terminates. It is the responsibility of the external program to reload Maximus with the appropriate parameters once the program has been executed. (See below about restarting Maximus after a CHAIN command.)

ERRORLEVEL: This command tells Maximus to exit completely from memory, and exit to the calling batch file or program.

* This command is slow, since the transient potion of COMMAND.COM must be reloaded.

* The only interface Maximus has with the external program is an errorlevel. (However, this isn't totally true. See below about Errorlevel Batch Files.) Also, see below about restarting Maximus after an errorlevel exit.

ErrorLevel Batch Files

When exiting via an errorlevel exit, Maximus uses a concept similar to BinkleyTerm's `BBSBATCH' command ,which allows Maximus to pass command-line parameters to an external program. To create an errorlevel batch file, instead of specifying only an errorlevel as the command to execute, add a single SPACE character (or an underscore, if you're running the external program through a menu option), and then the name of the command you wish to run.

ie. `Xtern_Erlvl 65' -> `Xtern_Erlvl 65_Myprg_Arg1_Arg2'.

When Maximus encounters an argument after the errorlevel, it will write a file called ERRORLVL.BAT in the Maximus startup directory, containing the argument specified after the errorlevel. (If you have a task number defined in MAX.CTL, then Max will write a file called `ERRORLxx.BAT' instead, where `xx' is the task number, in hexadecimal. However, aside from the filename change, the use of ERRORLVL.BAT is identical to that in a single-node environment.) In the case of the above example with MYPRG.EXE, the ERRORLVL.BAT file would contain:

```
     Myprg Arg1 Arg2
```

Once the errorlevel batch file has been written, then Maximus will exit with the specified errorlevel. You can then trap this errorlevel in your batch file, and use a `CALL ERRORLVL.BAT' command to execute the command. (If using DOS 3.2 or under, replace the `CALL' with `COMMAND /C'.) After executing the external program, you can then restart Maximus by the method described in the next section.

Restarting After Chain/Errorlevel

Unlike Opus, after executing an external program via the CHAIN or ERRORLEVEL exit methods, Maximus can restart itself exactly where it left off, and appear as if Maximus had remained in memory for the entire time that the external program was running.

This feature is made possible through the `-r' command-line parameter. When Maximus is invoked using `-r', it will read a file called RESTAR*.BBS from the root directory. This file was written to disk just before Maximus executed the chain/errorlevel command. This file contains all of the information that Maximus needs to start up again, so Maximus will simply pick up right where it left off, whether Maximus was displaying a menu or in the middle of a *.BBS file.

Also make sure to specify the *.PRM file name on the command-line, if you are not using MAX.PRM. In addition, if you are using a NON-ZERO task number, then you MUST accompany the `-r' option with the `-nXX' (set task number) option.

WARNING! Never attempt to use an `[xtern_erlvl]' token before a new caller has reached the NEWUSER2 file. Maximus cannot perform a restart until it knows who the user is, and that means that the user must have entered their name, password, graphics selection, etc.

This is an example batch file which utilizes errorlevel batch files, and also the restart option:

```
REM * These first "%1 %2 %3" parameters will be passed to
REM * the batch file by your mailer.  However, they
REM * aren't really important when dealing with errorlevel
REM * batch files, so we'll just assume that they are
REM * correct.  Also, make sure that the `:DoErlvl' label
REM * comes AFTER the main `Max -b%1 ...' command.

Max -b%1 -p%2 -t%3 -n2

:DoErlvl
if errorlevel 65 goto Outside
REM * [..more errorlevels here..]
if errorlevel  1 goto end
goto end

:Outside
REM * Replace the `Call' with a `Command /C', if using DOS
REM * 3.2 or below!  Also, make sure that the number after
REM * the `-n' parameter specifies the Maximus task number
REM * to use, if not the one specified in the control
REM * file.
REM *
REM * Finally, if you're using a non-zero task number, keep
REM * in mind that the filename Maximus writes will be
REM * `ERRORLxx.BAT', where `xx' is the hexadecimal task
REM * number.

call C:\Max\Errorlvl.Bat
Max -r -n2
goto DoErlvl

:End
```

After you've created a batch file such as this, using errorlevel exits becomes just as easy as any of the other exit types. In MECCA, instead of using something in this format:

```
     [xtern_run]D:\Path\Progname.Exe Arg1 Arg2
```

...you could easily replace it with something like this:

```
     [xtern_erlvl]65 D:\Path\Progname.Exe Arg1 Arg2
```

As you can see, once you have added the errorlevel code to your batch files, adding new options requires only a minimal amount of work.

External Program Translation Chars

When passing a command-line to an external program (and also when parsing some special MECCA tokens), Maximus can include information about the user and sysop, by using special format characters. A format character is lead in by the `%' (percent) character, and followed by one more CASE-SENSITIVE character. Maximus will interpret the character following the percent sign, and replace it with the variable which that character represents.

Maximus currently supports the following external program translation characters:

Ch	Translation
%A -	The user's FIRST name, in upper-case.
%b -	The user's baud rate. If the user is a local caller, then this will translate to `0'.
%B -	The user's LAST name, in upper-case. (If the user has no last name, then this will translate into `NLN', `No Last Name'.)
%c -	The user's city.
%C -	The response to the last `[menu]' MECCA token.
%d -	The area number of the current message area
%D -	The area number of the current file area
%e -	The user's password
%E -	The user's screen length, in rows
%f -	The user's first name, in mixed case.
%F -	Path to the current file area.
%g -	User's graphics mode -- `0' for TTY, `1' for ANSI, and `2' for AVATAR.
%G -	User's Daily DL limit, in kilobytes
%h -	The user's phone number.
%H -	Number of kilobytes downloaded today
%i -	Total downloads
%I -	Total uploads
%j -	Minutes on-line, this call
%k -	The current node's task number. (`0' for no task number.)
%l -	The user's last name, in mixed case. If the user has no last name, then this will translate into `NLN'.
%L -	If the user is REMOTE, this will translate into the string `-pX -bY', where X is the port number (1=COM1, 2=COM2, etc) and `y' is the baud rate. If the user is LOCAL, this will translate into a simple `-k'.
%M -	Path to the current message area.
%n -	User's full name, in mixed case.

%N - The name of your BBS, as defined in MAX.CTL.
%p - The current port number (0=COM1, 1=COM2, etc).
%P - The current port number (1=COM1, 2=COM2, etc).
%q - Path to the current msg area (NO trailing backslash)
%Q - Path to the current file area (NO trailing backslash)
%r - The user's real name, if applicable.
%s - The SysOp's last name, in mixed case. If the SysOp has no last name, then this will translate into `NLN'.
%S - The SysOp's first name, in mixed case.
%t - The amount of time the user has left, in minutes.
%T - The amount of time the user has left, in seconds.
%u - The user's user number.
%U - Simply translates to an underscore.
%v - Path to the current upload area (with trailing backslash)
%V - Path to the current upload area (NO trailing backslash)
%Z - Translates to the user's full name, in caps.

In addition to the above translation characters, there is also another set of almost-identical characters, which can be used when giving Maximus the name of a file to display. However, the first character in sequence should be a "+", rather than the usual "%". The second character WILL be treated as shown above, and translated normally. For example, to display a file called D:\<#>.BBS, where <#> is the user's number, you'd use the following command in MENUS.CTL:

```
   Display_File   D:\+u.BBS   Twit  "Display It!"
```

Please keep in mind that the usage of the "+" is only required when specifying a filename to display - the percent sign can be used in all other cases.

Finally, there is one additional shortcut for *.MNU menu names. If you wish to substitute the current task number in a filename, then substitute the "*" character where you wish the task number to appear, and Maximus will translate it automatically. For example, the following line...

```
     First Menu  MAIN*
```

...would cause task 0 to display a menu called MAIN00.MNU when first executed, task 1 to display MAIN01.MNU, etc. (Keep in mind that the task number is in hexadecimal, and therefore the menu displayed for task 12 would be MAIN0C.MNU.)

Running Opus Message/File Utilities

Opus 1.03 and 1.10 have a rather curious interface for external programs designed to run from the message or file menus. These programs were declared through Opus' `External XXXX_Management' control-file options (or "_OUTSIDE" menu options for Opus 1.10), and this section pertains only to installing that specific type of utility.

Utilities designed to run from the Opus message/file menus expected to find two things:

1) A <progname>.CTL file, which contains some information about the current message or file area, and some general information about the system itself.

2) Three special command-line arguments (seven for Opus 1.10), which tells the external program the baud rate of the caller, and several other pieces of information.

To run a utility of this type from a menu option (It is not possible to do so from a *.BBS file), you must take the following steps:

* Make sure that the option has a type of `Xtern_Run'. Otherwise, it may not run correctly.

* Add the `Ctl' modifier before the `Xtern_Run', so that Maximus will generate a control file for your program.

* For the program's command-line, type in the path to the executable itself, followed by the name of the program, but with a *.CTL extension. (This should be done for both Opus 1.03 and Opus 1.10 external commands.) For Opus 1.03, then add an underscore, and add `%L' to the end of the command. On the other hand, if the program is supposed to be run with Opus 1.10, then add an underscore to the end of the command-line, followed by the following characters:

```
%L_-t%k_-m%d_-f%D_-r%t
```

In other words, you should end up with a line in your MENUS.CTL which looks something like this:

(For Opus 1.03)

```
Ctl Xtern_Run Prog.Com_C:\Max\Prog.Ctl_%L Disgrace  "Program
     #1"
```

(For Opus 1.10)

```
Ctl  Xtern_Run  Prog.Com_C:\Max\Prog.Ctl_%L_-t%k_-m%d_-f%D_-
     r%t Disgrace "Program #1"
```

If you follow these directions exactly, then you should have no trouble running Opus-specific message or file utilities from a menu option, for either Opus 1.03 or Opus 1.10.

Running `Other' Door Programs

A `door' is just a fancy name for an external program which can be run and interacted with by an on-line user. Most door programs contain modem routines, so they can keep track of a user's time limit, make sure that the user doesn't drop carrier while inside the door, etc.

However, running a door program presents a special problem. There are several conflicting standards for `door interfaces', which are what controls the way the BBS program passes information to the door. Most modern door interfaces can pass out the user's name, whether or not the user supports ANSI graphics, the name of the SysOp, etc.

Maximus includes built-in support for the Opus 1.03 LASTUSER.BBS standard, as well as the capability to DIRECTLY write ANY text-based door interface file. The distribution version of Maximus comes with MECCA scripts which allow you to create door interface files for the following formats: DORINFO1.DEF (QuickBBS and RBBS), CHAIN.TXT (WWIV), CALLINFO.BBS (WildCat!) and DOOR.SYS. In addition, you can write your own MECCA scripts, which allow you to generate a door interface file for almost any other system type.

Maximus can achieve this through the use of the `[write]' MECCA token. Although the `[open]' and `[post]' commands were originally implemented in Opus to allow on-line questionnaires, they serve a dual purpose under Maximus. The `[write]' token will simply write a line of text to the previously-opened file, while making translations to the string, as described in the `External Program Translation Characters' section, above.

For example, the only requirement to make Maximus write a QuickBBS or RBBS-compatible DORINFO1.DEF file is to copy the following MECCA script into a file called DORINFO.MEC, and compile it. (Note: if you're using the standard distribution package, then you can find this file, including the compiled .BBS version, in the \MAX\MISC subdirectory.)

When copying this into a file, be sure to line up all of the text against the left margin. Also make sure to change the [delete] and [open] tokens to reflect the path where you want the DORINFO1.DEF interface file to be placed.)

```
[delete]C:\Max\Dorinfo1.Def
[open]C:\Max\Dorinfo1.Def
[write]%N[ comment           Write the BBS name      ]
[write]%S[ comment           Write the SysOp's f.name]
```

```
[write]%s[ comment           Write the SysOp's l.name]
[islocal write]COM0[ comment   Write the COM port    ]
[isremote write]COM%P[comment (local is always COM0) ]
[write]%b BAUD,N,8,1[comment Write the baud rate     ]
[write] 0[ comment           Say we're not networked ]
[write]%A[ comment           Write the first name    ]
[write]%B[ comment           Write the last name     ]
[write]%c[ comment           Write the city          ]
[write]%g[ comment           Write the graphics      ]
[sequal  write]100[ comment  Write the security level]
[sxclude write]50[ comment             Ditto         ]
[write]%t[ comment           Write the time remaining]
[write]1[  comment           Say we're using a FOSSIL]
[quit      comment           And we're done!         ]
```

You can create similar files for other door interface types, by simply creating another MECCA file containing the appropriate commands. (A list of the external program translation characters has been provided in the prior section; however, you can use the above DORINFO.MEC file as a guide to designing your own door interface files.)

There are three ways to have DORINFO1.DEF (or any of the above-mentioned files) created when running an external program:

TO CREATE DORINFO1.DEF FROM A .MEC FILE:

To have the appropriate door file created, simply include the following line, whenever you wish to have DORINFO1.DEF written:

```
[link]C:\Max\Misc\Dorinfo
```

As mentioned above, the distribution version of Maximus also comes with MECCA scripts to generate several other types of door interfaces. The format for using these is similar to the interface described above:

```
[link]C:\Max\Misc\WWIV         - To create CHAIN.TXT
[link]C:\Max\Misc\CallInfo     - To create CALLINFO.BBS
[link]C:\Max\Misc\DoorSys      - To create DOOR.SYS
```

TO CREATE DORINFO1.DEF FROM A MENU OPTION:

Similarly, you can achieve the same results through a menu option, by simply linking the appropriate .BBS file to the menu option. (For more information, please see `Linking Menu Options' in the Maximus Technical Reference Manual.)

For example, to create a DORINFO1.DEF file for running a program called `C:\Max\Prg.Exe', you'd use something similar to the following in MENUS.CTL:

```
NoDsp Display_File  C:\Max\Misc\Dorinfo  Twit "Run Prg.Exe"
      Xtern_Run     C:\Max\Prg.Exe       Twit "R"
```

Again, the same concept can also be applied to the other MECCA-created door scripts, by simply substituting the name of the script into the Display_File command.

TO HAVE DORINFO1.DEF CREATED AUTOMATICALLY:

If you wish to have DORINFO1.DEF written every time Maximus exits for an external program, for whatever reason, you can simply edit the `Uses Leaving' statement in MAX.CTL, such that it reads like this:

```
Uses Leaving    C:\Max\Misc\Dorinfo
```

This will instruct Maximus to create DORINFO1.DEF whenever Maximus runs an external program, without needing to be specifically instructed to.

What follows is a demonstration of how to install a non-Maximus door in a menu file, assuming that you have NOT implemented the above `Uses Leaving' statement in MAX.CTL.

In MENUS.CTL, you should add the following to the menu which you wish the door to appear on.

```
      Display_File  Misc\DorInfo  Disgrace  "Play BoreDoor"
NoDsp Xtern_Dos     BD\Bore.Bat   Disgrace  "P"
```

The `Display_File' command tells Maximus to write the DORINFO1.DEF file, which will always be written to the C:\MAX directory (unless you've changed the .MEC file).

The `Xtern_Run' command tells Maximus to run the batch file called BD\BORE.BAT, which you'll need to create later. (The `NoDsp' in front tells Maximus not to show `P' on the menu a second time, since you only want the first `Play BoreDoor' to be visible. See the section on Linking Menu Options (in the Maximus Technical Reference Manual) for more details.)

When a user selects `P' from the menu, Maximus will execute the above options in order. That means that DORINFO1.DEF will first be written, followed by the execution of BORE.BAT.

Although the contents of the batch file are highly specific to the door program you'll be running, in general, you should use a format similar to this, in BORE.BAT:

```
echo off

REM * Change to the right directory
cd \Max\BD

REM * Copy the DORINFO1.DEF file from the
REM * main Maximus directory into the
REM * current directory, which is probably
REM * where the door program will look for
REM * it.
copy \Max\Dorinfo1.Def

REM * This is the door program itself.  The
REM * command-line parameters will be
REM * specific to the door you're running, so
REM * you should consult your door's installation
REM * instructions for more details.
BoreDoor

REM * Now change back to the Maximus
REM * directory.
cd \Max

REM * And exit back to Max!
exit
```

On-Line User Modification

Some door programs may be written specifically for Maximus, and may need to directly change part of a user's profile (such as the user's remaining time, ANSI/AVATAR preference, phone number, etc.), even while the user is on-line. Maximus supports this feature through a series of special keywords and characters, which cause it to re-read the LASTUSxx.BBS file after returning from an external program.

If you're running the external program through an option in MENUS.CTL, then the fastest way to enable on-line modification is to place the `ReRead' modifier in front of the usual `Xtern_xxx' option. In other words, instead of invoking the program like this:

```
        Xtern_Run   D:\Path\Prog.Exe    Disgrace "Program"
```

...you'd place the following line in MENUS.CTL, which would enable on-line modification:

```
ReRead  Xtern_Run   D:\Path\Prog.Exe    Disgrace "Program"
```

Similarly, you can perform the same operation when using the [xtern_xxx] MECCA tokens, by using an `@' as the first character in the program name. For example, instead of using this:

```
     [xtern_run]D:\Path\Prog.Exe
```

...you would use this, instead:

```
     [xtern_run]@D:\Path\Prog.Exe
```

However, keep in mind that most programs don't need this feature. For security reasons, you shouldn't use this feature, unless the external program's documentation states that on-line modification will be performed.

MULTI-LINE OPERATION GUIDE

In addition to general multi-line support, Maximus 1.02 supports an integrated paging and inter-node chat facility, which makes it ideal for multi-line systems. In addition, Maximus uses NetBIOS-compatible file opening calls (using the SH_DENYNONE attribute), which makes Maximus even more suited for network applications.

This section is merely a guide to running Maximus in a multi-line environment. Undoubtedly, there will be some problems which aren't covered by this section, and there will be some questions left unanswered. However, this section will hopefully answer most of the basic questions, and at least give you a head start on installing a multi-node version of Maximus.

Installation

Installation of a network version of Maximus is fairly identical to a normal installation. Simply run the INSTALL program, and answer all of the questions it asks.

However, there are several important things to consider:

*   You will need a SEPARATE batch file for EACH copy of Maximus you wish to run. Although you can reduce duplication by moving common parts of the batch file (such as SCANBLD commands) into a separate batch file, and then using `CALL' or `COMMAND /C' to execute the secondary batch file, you'll still need a separate batch file for each node you wish to run.

     Fortunately, you only need one copy of the Maximus .PRM file: you can use the `-nXX' and `-lX' command-line parameters to adjust the task number and log filenames at runtime. However, you DO need to specify a separate log file for each task. Naming the logs files as Line01.Log for task 1, Line02.Log for task 2, etc., is normally a good idea. (If you don't wish to have a log file for a certain node, then simply use the `-l' command-line parameter, without specifying a filename.)

     Even if you use the same .PRM file for all tasks, you can still display node-specific files to the user, through the use of the `*' token, which will translate into a two-digit, zero-padded, HEXADECIMAL node number of the current node. For example, if you specified the file `D:\Max\Misc\Welcom*' as the welcome file in

MAX.CTL, Maximus would display WELCOM01.BBS for task 1, WELCOME02.BBS for task 2, etc.

- When setting up your batch files, you should make sure that ALL copies of Maximus are started from the same directory. This will allow you to share some files between nodes, in addition to providing a cleaner directory structure.

- Some NetBIOS kernels (such as some Novell versions) have a bug in the DOS findfirst call (int 21h, func 4eh), which cause the `Check If Directory Exists' routine in Maximus to fail, if the specified directory contains no files. The solution for this is to let the INSTALL and SILT programs create all directories by themselves. If you need to create a directory on your own, you should place an EMPTY FILE in the directory, so that Maximus can `see' that the directory exists.

- Normally, even if you're part of FidoNet, you'll only want to run a mailer program on one line, unless you have special needs. Fortunately, there are several smaller programs which will only answer the phone and pass control to Maximus, without all the memory and set-up overhead of a full FidoNet mailer. One of these programs is called `FEND'; another is named `Minimus'. If you can't find these on a BBS near you, you should be able to find one at the Maximus help node, at FidoNet address 1:1/119.

- When installing a FOSSIL, you may need to do a bit of searching. FOSSILs are fairly finicky, and may not get along with all NetBIOS drivers and versions. If you're having mysterious communications problems, then try switching to a different FOSSIL. There are at least three different types for the IBM PC, so you should have no problem finding one which works with your hardware.

- In your AUTOEXEC.BAT, you may wish to include commands to delete ACTIVE*.BBS and UTASK*.* from the main Maximus directory, and IPC*.BBS from the Inter-Process Communications directory. These are temporary files created by Maximus during execution, and shouldn't be left lying around. If you need to restart the network while Maximus is running, then these files won't get deleted, and may cause future problems. To fix this, you should include the above-mentioned delete commands in your AUTOEXEC.BAT, to make sure that you start with a clean slate whenever you reboot. (In the case of a network, the delete commands should be put in the server's AUTOEXEC.BAT. If you're running DESQview, or some other multitasker on a single node, then you can also place those statements in AUTOEXEC.BAT.)

*   Maximus requires that the operating system support file and record locking, if you wish to use either of the multi-node chat or paging facilities. In DOS versions 4.0 and above, this is built in, so there are no special requirements. However, versions of DOS below 4.0 don't automatically support record locking, and you'll need to load SHARE.EXE to be able to use most of the multi-line features. Although it is possible to use Maximus in a multi-node environment without loading SHARE (through the `No Share.Exe' option in MAX.CTL), this is strongly discouraged, and no guarantees are made if you don't load SHARE.

*   Make sure that all copies of Maximus you're running have a NON-ZERO task number. If the task number is set to zero, Maximus will assume that you're running in a single-node environment, and won't bother to check the inter-process communications area. In fact, not many of the multi-line features will work, unless you use a non-zero task number.

Multi-Node Chat Operation

The main way in which Maximus takes advantage of multiple lines is through the integrated multi-node chat and paging facility. These features are much like those found in the commercial PCBoard and TBBS programs, and are just as flexible. Users can toggle whether or not they can be paged by others, they can display a list of who is on-line, and they can actually enter into a real-time conversation with other on-line callers.

The first step in enabling the multi-node chat is to enable the `Path IPC' statement in MAX.CTL. (Make sure to follow the instructions in the `Path IPC' description about installing SHARE.EXE and creating a RAMdisk!)

The second step is to edit MENUS.CTL, and uncomment the Display_Menu option which calls the CHAT menu. Although you can use a custom MenuFile for the chat section, it is best to leave this for later, and use the built-in `MenuHeader Chat' for now. You can worry about tweaking the cosmetics once everything is running smoothly.

Having changed MENUS.CTL, the only remaining step is to recompile the control files. But before allowing users to call the system, you should first test it yourself, by logging onto two nodes locally. (You'll have to use two different user names, since Maximus will only let one user hog one node at a time.)

Before testing the chat mode itself, enter the Chat Section, and look at the menu display. The table should show the node number which you're logged on to (including your name, and the `(you)' designation), in addition to the same information about the second node. (If there is no display, then check to make sure that you've implemented the `Path IPC' keyword, and that it points to a valid drive and directory. Another possibility is that you've forgotten to load SHARE.EXE. Also see the section above, on buggy NetBIOSes and empty directories.)

If the menu display seems to be in order, the next step it to try toggling your chat availability a few times. After your status has been toggled, the table should indicate whether or not you're available for chat, in the `Status' portion of the table. You can also check that the other node was informed of the change, by simply entering the Chat Section on the second node, and looking at the table on that system.

Finally, after you've confirmed that everything else is working, you can enter the multi-node chat itself. To initiate a chat, select the P)age option. Then enter the number of the other node you've logged onto, and wait for the chat request to register. (This should take no longer than about 15 seconds.) You should then see a `You are being paged by Joe SysOp (node XX)' message on the other node. To answer the chat, simply select the A)nswer Page option, and enter the node number of the user which sent the chat request. This should place you inside chat mode: the user who initiated the page should see a `User Name joins the conversation' message, to indicate that the other user answered the chat request. The user who answered the page won't immediately see anything; However, to find out who is participating in the conversation, you can simply type a `/w' command at the beginning of a line, and Maximus will display the list of callers in the same chat.

Once inside the chat, users can send a message to each other, by simply typing in the text they wish to send. Maximus will automatically word-wrap at the end of lines, and will transmit the text a line at a time. It's best to try typing a few times from each node, if possible, to make sure that the chat function is working properly.

Once you're finished testing, you can use the `/q' command on each node, to exit chat mode. (When a node exits chat, the other nodes participating in the same chat should see a `User Name leaves the conversation' message.)

In addition to the private chat facility (which is what you just tested), Maximus also supports a group chat, or a `virtual CB' chat. The CB chat is useful when you have three or more nodes, and want to have more than two callers in one conference. Maximus supports 255 concurrent `channels', which means that there can be up to 255 separate group conversations going on at the same time. However, the CB chat has no paging ability; it's up to the callers to look at the status screen in the Chat Section, and see which channel everyone else is using.

Aside from the differences in invoking the CB chat, once you get inside the chat mode itself, Maximus will behave just as it does inside the private chat, even using the same commands. For more information on using Maximus' multi-line chat, please see the chat help file, included in the Maximus distribution package. (Assuming a standard system, the help file is accessible using the `?' command from the chat menu, or through the `/?' command, while inside chat mode.)

USING CUSTOM MENUS

Maximus allows you to create custom menus with relative ease: simply insert a `MenuFile' command in the appropriate section of MENUS.CTL, and you're done. However, there are several tips and tricks you may find useful when designing custom menus, especially when using fancy ANSI or AVATAR graphics.

* When using a menu which contains a `[cls]' MECCA token, you'll notice that output from some of the internal commands (such as Version or Statistics) disappears, since the [cls] command in the menu erases it, before it can be seen by the user. The solution for this is to link a `Press_Enter' menu option after the appropriate command, which will cause Maximus to wait until the user presses <enter>, before re-displaying the menu. (For more details, see the section entitled `Linking Menu Options' in the Maximus Technical Reference Manual.) For example, to make Maximus to pause after displaying the user's statistics, you might use something like this:

```
            Statistics       Disgrace    "Statistics"
      NoDsp Press_Enter      Disgrace    "S"
```

* If you're using a custom MenuFile statement in the message or file areas, and you choose to disable the MenuHeader statement, then you must modify your MenuFile to compensate for this. Due to the flexible way that Maximus handles menus, you need to inform the menu handler when a particular menu represents a message/file area, so it can read certain pieces of information from AREA.DAT. Since the MenuHeader statement usually informs Maximus of this, disabling it will make Maximus think that the menu just represents a normal area. The solution for this is to place either the `[message]' and `[file]' MECCA tokens at the top of the custom MenuFile, depending on the type of area (message or file) you want the menu to represent. These tokens must be used before any of the message-specific tokens (such as `[msg_cname]') are used. The [message] or [file] token only needs to be used when a message area is first entered - this means that you can place the [message] or [file] token in the custom HeaderFile as well, although it will work equally well in the MenuFile.

* When designing a custom menu with an input prompt at the bottom, you may have some trouble getting the cursor to stop at the appropriate place. Most text

editors automatically insert a carriage return after the last line of the file, and since Maximus reads the entire file, this will cause the cursor to skip down to the next line after the entire file is displayed. There are two solutions to this: the first is to use a text editor that DOESN'T insert a carriage return at the end of the file. The other solution, if you're using a .MEC file to create the menu, is to insert a `[quit]' token where you want the cursor to stop. As soon as Maximus encounters this token, it will stop displaying the file, without displaying an extra carriage return. On the other hand, if you're creating the menufile manually, or if you're using an ANSI editor such as TheDraw to create it, and then running the ANSI file through ANSI2BBS, you can insert the compiled equivalent directly into the text, which is `^oQ'. (Control-O and then a capital letter `Q'.). This has the same affect as the [quit] token, and will cause Maximus to behave in the desired fashion.

MISCELLANEOUS INFORMATION

This chapter is for miscellaneous information which didn't fit anywhere else in this documentation.

Filename Specifications

Wherever you specify a filename, make sure to specify a FULL path, including drive specifier and leading backslash. For speed reasons, Maximus changes the current directory as it executes. This mean that you can never assume anything about the current directory. SILT will try to compensate for this, but cannot do so in all circumstances.

Hard-Coded Filenames

Maximus uses several few hard-coded filenames, whose names are not changeable:

BADUSER.BBS: If a file named BADUSER.BBS resides in the main Maximus directory, Maximus can use it as a screen when a new user logs on, to keep out users with unwanted names. This file is a simple ASCII text file, containing a list of names not to be allowed on the BBS, one to a line. Each name listed in the file will be matched to either the first, last, or the entire name of the user. If Maximus finds a match, then it will try to display a file called BAD_USER.BBS in your miscellaneous directory, and then hang up.

YELL.BBS: When yell is turned off and a user tries to yell for the sysop, Maximus will look for this file in the Maximus root directory. If it exists, it will display the file to the user. If it does not exist, Maximus will display the standard `Yell is turned off' message.

NOTIN.BBS: When a user yells and the sysop does not respond, Maximus will look for this file in the Maximus root directory. If it exists, then this file will be displayed. If it does not exist, Maximus will display the standard `Sorry, there's no answer'. (Compare to YELL.BBS.)

RAWDIR.BBS: If a file of this name exists in a file area directory, it will be displayed to the user when s/he attempts a R)aw directory command. It will be displayed AFTER the command is selected from the menu, but BEFORE the `Enter mask:' prompt.

READONLY.BBS: If this file exists in a read-only message area, and a user tries to enter a message, then this file will be displayed, as opposed to the built-in, "canned" message which Maximus normally displays.

HLP\CHAT.BBS: This is the help file which will be displayed inside the multi-node chat.

MISC\CHG_SENT.BBS: This is the help file displayed when a user tries to edit a message which has already been sent, packed, or scanned as EchoMail.

MISC\CHG_NO.BBS: This is the help file displayed when a user tries to edit a message which was written by someone else.

INDEX