DarkStar CBBS V3.0
(C) 1986 DarkStar Systems SoftWare

Designed By: Alan Peters

DarkStar Node 146

NOTICE:

DarkStar Systems Software warrants to the original purchaser of this computer software product that the recording medium on which the software is recorded will be free from defects in workmanship and material for a period of 14 days from the date of purchase. The defective disk(s) will be replaced at no charge, provided that the returned media has not been subjected to misuse, wear, or physical damage. Following this period, any defective media will be replaced at a cost of $8.00 (US Funds) for either disk, or $15.00 for both, and should be returned to the address below. You must also send the proof of purchase, a statement describing the defect, and your return address. Payment must be made in money order, or certified cheque. Local Toronto area residents should write, stating the defect, and should leave their phone number, as we have special replacement procedures for local residents.

This warranty does not apply to the software programs themselves. You have purchased this product "AS IS".

DarkStar Systems Software
113 Valleywoods Road, Unit 95,
Don Mills, Ontario, Canada,
M3A-2R8.

Preliminary:

1. Each disk has been duplicated on both sides to ensure reliability. Side 2 is unprotected, and because of the need to write to the disk often, you should make at least one backup of side 2, and use the backup as the medium to which you will write to. Some of the utility programs will ask for insertion of the "Master Disk". This is side 2. The "Files Disk" will always be the disk you create with the CREATE.DSK program.
2. No Fast Loading cartridges will be allowed to work with the BBS. Efforts have been made to thwart the use of one, as fast loaders and RELative files do not work well together. Please disable or remove the fast loader prior to boot-up.
3. BBS programs tend to wreak a great deal of abuse on the disk medium during prolonged use. 24 hour BBS systems tend to average about 600 disk writes per day, and it cannot be guaranteed that the files disk will last 5 years without being replaced. We recommend that you make backups of the files disk every week.
4. This program will NOT be able to load without 1 1541 disk drive connected to your system. If you have 3 IEEE drives, and you want to run it on this type of system, for example, you will still need a 1541 to boot up from. The device number of the 1541 is not important, nor does the device number have to be that of any of the BBS SYSTEM drives.
5. This version is final, and no mention will be made of the previous DarkStar BBS programs, released prior to this one.
6. A reasonable knowledge of BASIC is assumed, as it will be needed to alter 7 of the 8 CREATE programs. This involves only variable re-assignments, and no programming.
7. If you do not read this manual, you will likely have great difficulty running the BBS. Please read ALL of the manual to avoid frustration and confusion.
8. All CREATE programs will run on side 2 unless stated. Do not remove side 2 from the drive unless otherwise indicated.
9. Authorship of this BBS program need not be stated in any BBS system credits or system files you make; Likewise, you may not take credit for the program, nor may you modify any programs (Except modem files) without prior written permission of D.S.S.
10. This BBS system was designed to give the serial drive user a BBS programe that can compete with more powerful IEEE drive-based programs. Although the BBS can run with 1 1541 drive only, you would be restricted to a very minimum system due to lack of disk space. Two 1541s or any larger system is recommended for best results. Although never designed to take advantage of the IEEE drives, this program has great potential with IEEE drives, all without the need of overlays, or chaining.

In order to save time and space, this manual has been made in a very summarized format, and may tend to be vague in places. But we believe that everything you need to know will be included here. If you feel unsure about anything, or would like to know how to get full potential from this BBS, then send a SASE to us, and we will try to help you out.

The first thing that tends to come to mind about BBS programs and terminal programs is whether it works with your modem or not. Well, if you have a 1650 compatible or a 1670 compatible, then relax. If not, then it will take a little more time and effort to make it work with your modem type. This BBS uses modem files, very small files that contain 5 machine language routines needed so that your modem and the BBS will work together. Procedures for creating a modem file are given at the of this manual, but without some familiarity with machine language, you will have to try the alternate route. This involves you making a list of all peeks and pokes, or AT commands, etc, necessary to do the following routines: ANSWER, HANGUP, go ON-HOOK, and OFF-HOOK. HANGUP is not the same as ON-HOOK in some cases, as what we require in HANGUP is a way to hang up the phone AND stop any incoming calls during the BBS reset period. In addition, there is a fifth routine, which we have reserved for any special routine your modem may require (Ie., modem initialization). If you can supply the info in BASIC or pseudo-code, that will be ok. Send it to the address listed on the first page. We will try to make up a modem file, and send it back in printout. This printout will be a BASIC data file listing that you will type in and RUN, so that a proper modem file may be created.

Custom character sets are used in this BBS. If you have used them with DarkTerm, then you will find that the BBS uses the same character sets. Character sets consist of a FULL character bank, that is both an upper-lower case set, and an upper case only-graphics set. This character bank must have a load address of 57344 ($E000), and must have the upper case-graphics set at $E000 and the upper-lower case set at $E800. All sets on side 2 were made with ULTRAFONT+, merged with MICROMON at $7000 and $7800, and then the load address was changed from $7000 to $E000.

Before going into the process of loading and running the BBS, it is necessary to explain the function of ALL the files on SIDE 2. A printout of the disk directory has been placed in the manual for reference, as all explanations in the manual will use the actual file names as the subject headers.

Note that on side 1, DarkTerm 4.0 has been placed. This is a PUBLIC DOMAIN terminal program, and may be given away, but not sold. The documentation for DarkTerm has been placed on side 2 in a SEQ file, formatted to 80 columns. You will need a printer if you want a proper doc file on paper. Because DarkTerm needs to write to disk in order to change the setup file, you MUST copy all the DarkTerm files to another disk before you attempt to use that program, or you may damage side 1, irreversibly.

(CREATE.CNF). This BASIC program will allow you to change the contents of the CONFIGURATION file, the file that sets up most of the options by which the BBS runs. Because it is such a long BASIC program (356 lines), all line references to the actual variables to alter will be given here. Nevertheless, the entire CREATE.CNF file is filled with numerous REM statements. You should save this file as another file name, or replace the one on the side 2 backup you are using. Then RUN the program.

All line ranges will be given, then an explanation of their purpose.

1230-1300. Physical directory device numbers.
1340-1410. Relative directory device numbers.
1450-1520. Physical directory drive numbers.
1560-1630. Relative directory drive numbers.

The file section uses from 0 to 8 directories for program storage. Physical directories use the actual disk directory as the method of file transfer, using the C,D,U,MD, and MU commands. All access is done straight from the directory. Relative directories are accessed through the RELATIVE file "+PR". This method allows you to place passwords on files on disk, as well as give other information on a program on disk (See EDIT.PROGRAMS). All access is done from the disk directory like the physical method, but via the relative file "+PR". Thus, you may screen and divide the physical directories into 1 or more relative directories. Example: You may only have device 9 for program storage, yet you may make all 8 directories point to device 9 drive 0, and then divide the disk directory. When using any of the file section commands, the directory number runs from 0 to 7 (Ie., LIST0, MU6, D7). Drive numbers can be 0 or 1. Device numbers can be from 8 upwards. You can have all the physical directories point to the same drive, but you should only use as many physical directories as needed for each drive for program storage. NEVER let the device and drive numbers be non-existant, since some commands allow access to all 8 system drives, even if you do not use them. SYSTEM DRIVE always refers to the disk drive with the device and drive numbers of the 8 drives set here.

1730. The FILES DISK drive and device number must be 8:0 at all times. The system will crash if you try to change it to any other drive.

1780. Bulletin drive setup.
1830. Message drive setup.

Drive and device numbers are separate for bulletins and messages, in case you want to create a massive system, and keep the files disk at a minimum. For example, our color BBS system at (416)-445-6788 is one of largest SERIAL-ONLY systems in the Toronto area, running on 2 1571 drives in 128 mode, and 5 1541 drives, all at the same time. Our setup looks like this:

Drive 1 (8:0) : Files disk, with all SYSTEM FILES active, and a full voting section of 99 topics on-line at once.
Drive 2 (9:0) : Message Storage, 144 Messages on the 1571.
Drive 3 (10:0) : Program Storage.
Drive 4 (11:0) : Program Storage.
Drive 5 (12:0) : Program Storage.
Drive 6 (13:0) : Program Storage.
Drive 7 (14:0) : Bulletin Storage, 144 Bulletins on the 1571 in 128 mode (1328 Blocks).

It is possible to have up to 19 drives running, 8 separate for physical dirs, 8 for relative dirs, 1 for bulletins, 1 for messages, and the files disk. While the BBS can run on 1 1541 drive, it will surely be a very tight fit.

Understand that the SYSTEM DRIVES are independant of the bulletin, message, and files disk drives.

1900-1910. The number of physical directories can be from 0 to 7 as said before. You should only set as many physical dirs as you have drives for storage. Relative directories can be from 0 to 7 as well. Note that if you set physical directories to 0, then the commands MD,MU,C,D, and U in the file section will be disabled. Likewise, if you set relative directories to 0, then the LIST,DOW, DEL, and UPL commands will be disabled, AND you may scratch the "+PR" file on the files disk, as it will not be accessed. If both directory limits are 0, then the entire file section will be disabled.

1970-2160. Sound parameters. Rather than keep the same BELL TONE (CTRL-G), keyclick, and CHAT mode pager, you can "slightly" alter the sound output. The keyclick and bell tone both have the same setting, 0-2 to set the waveform, and 0-255 to set the frequency. The higher the frequency, the higher the pitch. Unless you know how to change the ADSR envelope, you can skip the rest of the sound parameters. GATE DELAYS are time delays between each sound in the CHAT pager. The reason it's called gate delay has to do with ADSR, so skip it if you are not sure. Gate delays are in 1/60 second units, and the sum of both gate delays is the duration of each note sounded in the CHAT pager in the BBS. The CHAT pager will sound out the same note 10 times, unless you want the note repeated. The SL variable is the number of times youu want the note repeated for each of the 10 calls the CHAT pager makes. It is set to 1, but if you set it higher, remember that you double the time for each loop in the CHAT pager. The SL and 61/62 variables determine just how long the CHAT pager will try to page the SYSOP until a SYSOP-Not-Here message is displayed. The 2 frequency settings, 2 bytes for voice #1, and 2 for voice #3, are used to create ring modulation for chiming effects. To determine how these variables should be set, load and run the EDIT.CHAT program.

2260-2330. System file abort status. There are 8 system files, SEQ files that the users may see at various points in the BBS. These files appear on disk as "+SY.0" to "+SY.7". Setting the abort status here will make the file unabortable (Ie., you can "S" to stop the listing, "C" to continue, but not "A" to abort). This applies to NORMAL users only. If the user is level "s", or is the SYSOP (you), or you are on the system via keyboard (with a user on-line or not), you can abort the file regardless. The 8 system files are as follows:

+SY.0: New User File. If the BBS is in public mode, this is the new user file, inviting the user to join the system. If the BBS is running validation mode, then this file can be used to tell users how to apply for a validation. If the BBS is running guest mode, then it should explain the system to guest users, and if the BBS is running private mode, it should explain why they are denied access.
+SY.1: Logon File. When a caller gets on-line, this file will be diplayed as soon as he hits return. If the BBS is running in shutdown mode, this file should explain why the BBS is down, because after being displayed, the BBS will disconnect the user. Use this file for opening credits, titles, etc.
+SY.2: Daily Bulletin File. This file is displayed after a guest/new/validated user has logged onto the system.
+SY.3: Log Off File. After a user enters "G" to logoff, this file is displayed.
+SY.4-7: Section Introduction Files. Each of the 4 general BBS sections can have an introductory file. The files, SY4-7, are for the Message, File, Bulletin, And User sections respectively.

Use either "Y" for abort-enabled, or "N" for abort-disabled, for each of the 8 files.

2380-2390. System File Active Status. For each of the 8 files above, any or all of the files may be disabled on the BBS. If disabled, the file will never be accessed on the BBS. If the file is active, but not on the files disk, then it will be searched for, and when found to be missing, will skip (ASCII) or flash the cursor (Color). Use "Y" for file enabled, or "N" for file disabled.

2630. Bulletin Operation Mode. The bulletin section can exist in 2 formats. Relative mode is the standard mode, using the relative file "+bu" to handle all names and descriptions. Primate mode is the way many boards use to handle bulletin sections, by making all bulletins simple SEQ data files. The SEQ file "+bt" is used as an index file to hold descriptions of bulletin text files, bypassing the relative file. For more info on the bulletin section, see EDIT.BULLETINS. Set BM to 0 for relative mode, or 1 for text mode.

2700. Access Level Mode. On this system, access levels play only a minor role in user access. As far as access to various commands in various sections, user access bits in the user record determine a user's status on the BBS. Access levels apply to 2 areas only. The bulletin section uses access levels, IF the bulletin section is operating in relative mode, to screen bulletins for certain users. If a bulletin is posted at a level higher than that of a particular user, then that user will NOT be able to read the bulletin, even though he may list the title and description for it. The second area where access levels are used is the message section. The messages will be screened in the same manner as the bulletins are. If a message is at a higher level than a user's access level, it will be skipped over when a read is attempted. No notice of a restricted access level will be given, unlike the bulletin section. The BBS will handle access levels whether they are enabled or not. If access levels are not used, all messages and bulletins posted will be done at level 1 automatically. Also, messages and bulletins written previously with a level >1 will be readable by any level user, even level 1. The access levels in these messages/bulletins will not be reduced to 1. In this way, you can toggle the access levels on or off at any time without physically changing messages or bulletins. Set AC to 0 for access levels, 1 for no access levels.

2830. Maximum # Of Messages (1-199).
2840. Maximum # Of Bulletins, Relative mode ONLY (1-199).
2850. Maximum # Of Programs, Relative Directories (1-699).

If you use primate text bulletin mode, you are only limited by the number of file entries allowed per disk directory. The same applies to the physical directory storage mode. When setting the maximum limits here, they are equivalent to the number of records in the files. The relative files for messages, bulletins, and programs are "+me", "+bu", and "+pr", respectively. It is IMPORTANT to not change these limits without changing the relative files themselves. If you create a files disk, and later on you want to set the maximum for any of the files higher, it will be necessary to expand or recreate the relative file in question. If you recreate the file, all records will be erased. If you expand the file, you should use the program "FIX RELFILES". If you want to set the maximum limit of any file lower, you do not have to change anything. There is no way to remove the extra records from the REL files. The extra will be ignored. The safest way, of course, is to just make a new files disk, or make sure the limits you set at the start are going to remain static. The limits given above are absolute, and may not be set above or below those limits. If you attempt to do so, the system will most surely crash. If you intend to use text mode bulletin storage, you can scratch the "+bu" relative file, as it will no longer be accessed. Similairly, if you set the number of relative directories to 0, you may scratch the "+pr" reltive file, as it too will no longer be accessed by the ISFS system.

2930-2950. Default screen colors. This is for the ASCII version of the BBS only. After the BBS has loaded in all system files, and you place the files disk in drive 8:0, the screen colors will be set to these values. If you are using the CBBS, the colors will default to the first set of screen colors in the color table.

New user status is active when the BBS is operating in PUBLIC or GUEST mode. Certain new user settings will be ignored in GUEST mode, regardless of what values they are set to. The GUEST user will NOT have access to the file section, cannot send messages or mail to anyone except the SYSOP, is not recognized in the system as a user, and thus does not have any status (ST command), and cannot post bulletins. If the BBS is running in public mode, the new user can do whatever you allow him by setting the values here. These values have no bearing in VALIDATION, PRIVATE, or SHUTDOWN modes.

3030. New User Level. Levels range from 1 to 9. Level "s" (lower case) is sysop level, and if given, allows the user access to the editor section, among other things (See EDIT.USERS).

3050. New User Time Status. To save memory, access for users on the BBS is given through individual "bits" in the user record. 8 bits make a byte. A bit has either an on (1) state, or an off (0) state, like a light bulb, which is either on or off. This suits BBS access quite nicely, as access is either given, or not given, to an area on the BBS. All access is given on the binary level, represented in true binary form, or in sequential form. True binary means that bits are arranged with the following values;

Bit 0 : Value =1
Bit 1 : Value =2
Bit 2 : Value =4
Bit 3 : Value =8
Bit 4 : Value =16
Bit 5 : Value =32
Bit 6 : Value =64
Bit 7 : Value =128

These 8 bits form a byte. Visually, a bit string on the BBS is either a series of 8 zeroes and ones, or a series of 8 "Y"s and "N"s. Y and N are for Yes and No, as pertaining to whether the user has access to the function that the bit is designated for. When you see a bit string like: "01001110", bit 0 is the rightmost bit (value 1) and bit 7 is the leftmost bit (value 128). On this BBS, binary strings use the "1" value to indicate NO ACCESS, and a "0" to indicate access IS given. We'll use the time status string, which is only 2 bits long, to show how this works.

The time status is set to "01". This means bit (the rightmost) is "1". Reading to the left, bit 1 is "0". Bit 0 is used for access to unlimited time. Since it is set to "1", the new user does not have access to this function. Bit 1 is set to "0". This is the daily time limit bit, and since it's value is "0", it means the new user has a daily time limit. Daily time limit means that the user can only log onto the system for the set time limit he has, and can only do it once for every 24 hour period. If you set this bit to "1", the new user would not have daily time limit, and since bit 0 is "1", the user would not have unlimited time either. In this case, the user has NORMAL time. Normal time allows the user to log on and use as much time as his limit allows, at which point he will be logged off. The user may then log back on, even if it is the same day. If bit 0 is set to "0", then the user can stay online for as long as the BBS operates. Note that the unlimited time bit takes priority over the daily time limit bit, so that a bit string of "00" would still give the user unlimited time; the daily time limit would be ignored.

3090. New User Time Limit. Time limits are given in minutes, and can range from 1 to 59 minutes. Remember to add a "0" to the start of the limit if set to less then 10 (ie., "05"). If unlimited time is given to a user, this limit is ignored.

3120. New User System Access. This is an 8 bit binary string for general system access on the BBS. Bits run from right to left (0-7).

Bit 0. Restricted File Access. The restricted file is like a SYSTEM file ("+sy,#"), with a file name of "+re", except that all users may not be able to view the file. This file will show up immediately following the daily bulletin after logon.

Bit 1. Multi-File Uploading. This is for access to the MU command for uploading multiple files to a physical directory. It is recommended that you only allow trustworthy users access to this command, unless you keep backups of system disks. The reason being is that multi-uploads do not check current files on disk. While efforts have been made to ensure misuse of the multi-file protocol is not possible, the MU command will save and replace any existing files on disk that have the same names as the files to be uploaded.

Bit 2. Multi-File Downloading. Multiple file downloading is guaranteed to not cause user misuse problems, and MD will only allow as many files to be downloaded as the user's time limit allows. This may however extend beyond the user's time limit, as downloads are not broken in the middle of a transfer if the user's time limit expires.

Bit 3. Physical Directory Commands Access. The 5 physical directory commands, MU, MD, U, D, and C, can be locked out from the user. This was primarily implemented for those who run all public file access through the relative directories. For 1 drive users, this is generally the idea, since the system files would be accessible by physical directory access, unless access could be locked out.

Bit 4. File Section Access. This will lock out the user entirely from accessing the file section.

Bit 5. Bulletin Posting. If the bulletin section is in relative mode, then the "P" (Post) command can be used by any user with access here.

Bit 6. File Section Downloading. If the user has access to the file section (bit 5), you may still deny download access using the DOW command. In this way, a user could browse the file section without downloading.

Bit 7. "+" File Access. All system files used with the BBS use filenames that all start with the "+" character. Thus, all system files can be protected from being downloaded, or seen. "+" files are also used to protect files from being accessed by a user, using the MD or D commands. Any file on any drive can be hidden, from a user without "+" file access, by making the first character a "+". Attempting to download will result in a file not found error, and attempting to multi-download will cause "+" files to be ignored when the directory is placed in memory prior to transfer. The user may conversely upload a file that has a "+" as the first character to hide it from other users. The user need not have "+" file access to do this. But if the user does have "+" file access, all files may be accessed by the user. Once again, MU will not check for "+" files on uploading, so an MU to the files disk in drive 8:0 could conceivably wipe out important system files.

Bits 1,2,3,4,6, and 7 shown above all deal with the file section. There is a priority here, which will be explained. Bit 4 takes highest priority, regardless of all the others. Bit 6 is for the DOW command only. Bit 3 allows further access to the file section by giving access to MU, MD, U, D, and C. If access is given to this, then access to the DOW command becomes irrelavent. Also, bits 1 & 2 for MU and MD become active. The "+" file access applies to any user who has physical access to the directories. This also applies to the relative directories, but in a slightly different way. If you put a "+" file in a relative directory, a user without "+" file access can see the file when it's listed, but may not download that file with the DOW command. In the default system access setting above, access is denied to everything except the file section (bit 4).

3250. Physical Directories Access.
3290. Relative Directories Access.

These accesses are sequential bits, running from left to right for directories 0 to 7. "0" means access to that directory is given, and "1" will deny access to that directory. In the defaults for the new user, the relative directory access shows that access is given to directories 0-3, but not to 4-7. Physical directory access is all "1"s, so no access is given to the physical directories. These 2 access strings take priority over all the file section commands (MD,MU,C,D, and U for physical dirs, and LIST,DOW,UPL, and DEL for relative dirs).

3550. BBS Operation Mode. There are 5 modes of operation. The last one, 5, is not really an operating mode. SHUTDOWN mode is intended to be nothing more than an answering service for your modem. The caller will logon to the point of having the opening logon file ("+sy.1") displayed, then be logged off. The BBS will not allow ANYONE access to the system from remote or from the keyboard when in this mode. You can only disable it by changing the value here in the CONFIGURATION file.

Mode 1 is PUBLIC mode. All callers not found in the user list will be treated as new users, and will be given the the access you set previously in the new user access stuff. The user will be assigned an ID# automatically, such that the ID# will be the first one available, searching from 1 to 255. If the user list is full (255 users), then the caller will be notified of this, and logged off for lack of user space. A new user will not be required to complete a validation, so you may not be aware of a new user logging on without checking the user list.

Mode 2 is PRIVATE mode. Any caller not found in the user list will be immediately logged off the BBS after the non user file ("+sy.0") is displayed.

Mode 3 is VALIDATION mode. This is the same as PRIVATE mode, except that the caller is given the opportunity to fill out an application for validation. After completing (or not completing) the application, the caller will be logged off. Five lines of input are required for the application. These five lines can be up to 25 characters in length, and ALL 5 lines must be input, or the application will not be saved. Because of the ability to assign text prompts to anything you want, the five default input requirements, Real Name, Alias, Password, Calling From, and Phone #, can be changed to anything you want. This application will be added to the validation file ("+va"), and can be read with the "V" command in the editor section.

Mode 4 is GUEST mode. In guest mode, the new user has no ID# assigned to him. In the status line, the guest user ID# is 0, the sysop's. This makes it easy for you to identify if the user is a guest user or not. The options for the GUEST user were explained above. Guest users cannot send messages because they have no ID#, which is required on this BBS. If the guest user does send a message with the S command, it will be done with the LINE editor, and will be written to the end of the validation file ("+va"). The guest user can in no way write to the system message base. In the user log ("+ul"), the guest user will have the ID# replaced with "GST" to indicate that a guest user has logged onto the system.

3640. Bracket Selection. Certain places in the BBS enlose numbers in brackets (ie. (3452) in the message section around the message reference number). The square brackets are the default, and will match the square brackets used in the text file (BBS.TXT) in various prompts. If you change the brackets to something like round brackets (parentheses), then you should use CREATE.TXT to change the text file to reflect the new bracket definitions.

3740. Message Categories. The message section can utilize up to 10 message categories, numbered from 0 to 9, each which is a text string from 1 to 20 characters in length. The CREATE.CAT program is used to make the text strings for each category. If you set CX to 0, then categories will be disabled, and this will be reflected in the lack of the category description in the message header, and the disabling of the CAT, RCAT, and SCAT commands in the BBS. Remember to set CX to one value less than the number of categories you want, unless you want none, in which case CX is 0. As you can see, there is no provision for 1 category. You must have at least 2 (ie., 2 categories is CX=1, 10 categories is CX=9). You need only change the text strings for the number of categories you use, with CREATE.CAT.

3850. Sectioned Help Files. The help file for the BBS can exist in one file, or multiple files, dividing the help up into smaller, easier to understand sections. It is recommended that 1 drive users use just 1 help file to save directory space. NOTE that you only have 144 files per directory on the 1541, and 224 on an 8050, SFD, etc., so, you should be aware of the sum total of all system files, including the number of messages that can be written, bulletins, and programs. If you use one help file, it will be called "+he.0". On side 2 of the BBS, we have supplied a sectioned version of the help file, named "+he.0" thru to "+he.4". Use the EDIT.TEXT program to merge and modify the help files as you see fit. These are default, and are by no means the ones you can use, especially if you change the command definitions with CREATE.COM. If you do split up the help file, the help file displayed depends on what section you enter the "?" help command from.

"+he.0" : Displayed from the main command prompt (currently in no section).
"+he.1" : Displayed from the message section command prompt.
"+he.2" : Displayed from the file section command prompt.
"+he.3" : Displayed from the bulletin section command prompt.
"+he.4" : Displayed from the user section command prompt.

When you type "?" in the editor section, you will be given only a command word summary of the 16 available editor commands on the BBS. See the section in the manual on the editor section for more help on editor command definitions.

3930. Display Directory Header. When using the "C" (catalog) command in the file section, or the "$" (directory) command in the editor section, you can have the directory header (disk name and id) displayed prior to the files in the directory by setting DD to 1. It is left as an option because the directories have their own descriptive headers in the text file (BBS.TXT), which are displayed prior to this directory header, if enabled here.

4000-4030. List Control (XON/XOFF). For listings of SEQ system files, help menus, user listings, logs, etc., the user may want to stop the output at times to read the data. Many ASCII systems use true XON/XOFF to control listings: CTRL-S to stop, CTRL-Q to continue, and CTRL-C to abort. This BBS uses simple S, C, and A control. If you want list control like DarkTerm, you can use the space bar for stop, the space bar to continue, and the STOP key (CTRL-C) to abort. The fourth key, "Q", is used in the message section when reading messages. If the continuous mode characters (+ & -) are added to the message section commands, messages will be read continuously. "A" will only abort to the next message, so "Q" is used to abort to the message command prompt.

"S" : Stop A Listing.
"C" : Continue After Listing Stopped.
"A" : Abort A Listing At ANY Time.
"Q" : Quit Continuous Read Mode.

4110. Number Of Voting Topics. The BBS has an optional voting section. You can disable the VOTE command, and thus disable the voting section, by setting the number of topics to 0. From 1 to 99 vote topics may be selected. These vote topics will be accessed thru the "+vo" relative file for tabulating results. See CREATE.VOT for more info.

4240. User Editor Lock. This applies only to the CBBS, which has 2 editors, a line editor, and a color editor. By overwhelming demand, this late addition was made to prevent users from accessing the line editor, and to use the color editor only, for the purposes of "encouraged color creativity". If set to 1, the user does not have the option of what editor to use, and must use the color editor only. The level "s" user can however still use both editors regardless of this value, and the guest user must still use the line editor when writing a message to the sysop.

4310. Enable Restricted File. The "+re" restricted file can be disabled from use on the BBS by setting DF to 0.

4390. Display Voting Results. After a user casts a ballot on a vote topic, or after attempting to cast a ballot after having done so previously, he may be able to see a tabulated result of the selected vote topic if DV is set to 1. The vote operation is explained under the VOTE command. If DV is set to 0, the user will not be allowed to see the tabulated results. This way, the voting section can handle secret balloting.

4470. Upload Restrictions. For each of the 8 physical AND relative directories, you can set whether a user can upload files to that directory. The string should be set with "0" for uploads allowed, and "1" for uploads restricted, with the "0"s and "1"s for each directory from 0-7 running from left to right in the binary string. Level "s" users may still upload to any directory regardless of this option.

This is the end of the CREATE.CNF program. You should save this to a backup of side 2, replacing the old file with this new CREATE.CNF. Then RUN this program on your backup of side 2. You are now ready to go to CREATE.DSK.

(CREATE.DSK). Once you have the configuration file changed, you will need to create a FILES disk. This is the disk that will run off drive 8:0, and contain all system files. Depending on the message and bulletin drive settings in CREATE.CNF, you may or may not have the message and bulletin text data on drive 8:0. The voting section text files will also be placed on this disk.

Because this is a BASIC program, it will take some time to create a files disk, depending on the maximum sizes of the relative files. The program will format the disk, so use a blank disk, or one with nothing of value on it. Just LOAD and RUN this program. It will prompt you for which disk to insert at which time. Once the files disk is created, it will have the following files on it:

"+sf" (PRG) : System File (See FIX +SF FILE for it's use).
"+us" (REL) : User Record File (See EDIT.USERS for details).
"+pr" (REL) : Program Storage File (See EDIT.PROGRAMS for details).
"+me" (REL) : Message Base File (See section under MESSAGE SECTION for details).
"+bu" (REL) : Bulletin Title File (See EDIT.BULLETINS for details).
"+ul" (SEQ) : Daily User Log (Accessed with LOG command).
"+va" (SEQ) : Validation File.

At this point, you should do the following:

-If you have any vote topics selected, use the CREATE.VOT program to install the vote result file.
-Add the other system files, which may include:
-help menus (+he.#)
-SYSTEM files (+sy.#)
-the restricted file
-the vote description file, if vote topics used ("+vt.00")
-any vote topic files ("+vt.##")
-the "+bt" bulletin description file IF bulletin text (primate) mode is used
-scratch "+bu" if bulletin text mode is used
-scratch "+pr" if you are not using any relative directories

Some of the above items have yet to be explained, so you can hold off the first time around, until you are familiar with the system.

(CREATE.VOT). The voting section consists of 2 parts. The first is the "+vo" vote result file, which is a RELative file that is created with this program. The other part is the SEQ description files, which exist on the files disk as "+vt.##", where ## is the topic number (ie., topic 1 would be "+vt.01", topic 32 would be "+vt.32"). It is important to always add a leading zero if the topic is less than 10 (1-9). The descriptions for ALL vote topics, which can range from 1 to 99, should be described in the file "+vt.00" (Topic 0 is not recognized, thus the reason why the file name is used). The "+vo" file itself consists of 2 records of 129 bytes per record for each topic. This will store a vote outcome for each user, including the sysop. As you can see, 99 vote topics requires a large amount of relative file storage, slightly over 100 blocks, but necessary, since this file would hold 25344 outcomes. Outcomes for balloting is explained in more detail under the VOTE command. What is important now is to set up the "+vo" file. LOAD and RUN this program on side 2. Follow the on-screen prompts.

When done, you will have the relative file set to that number of topics which you selected. If you change the number of topics in CREATE.CNF after this point, you MUST run CREATE.VOT again. Every time you run this program, all vote outcomes for all topics will be cleared. Please consult the VOTE command for how the actual voting procedures take place. Note that using REL files for tabulating results gives very FAST balloting.

Note that the remaining 6 create files should all be RUN on side 2 after the programs have been modified.

(CREATE.CAT). If you are using message categories, then load this program, and list it. There will be 10 data statements, each a string of 20 characters. Categories run from 0 to 9, and in this case, top data statement to bottom statement. You need only change as many of the text strings as you have categories. You can use any characters you want in the string, upper/lower case, etc., and, if you are running the CBBS, you can use the graphics characters, or even color characters (ie, a sequence like F1 (blue) F3 (red) F5 (yellow) will change the screen colors just by listing the message category!). The important thing to be aware of is that the string MUST be 20 characters long. If you have less than 20 characters in your category, then pad the remaining ones with blanks; they will be stripped from the category at bootup time. You can change category descriptions at ANY time.

(CREATE.COL). If you decide to run the CBBS (Color BBS), you will need a color table. There are 33 sets of screen colors, used to change the screen colors when entering any section, or when accessing some (not all) of the commands. The commands that do have screen color changes are the ones that involves large amounts of text display. Although you can toggle the color changing on the BBS with the COL command, you may want to do it here, if this feature of the CBBS bothers you. To disable the screen colors, set ALL 33 color combinations to the same values (ie, same 33 border, background, and text colors). That way, the screen colors will always change to the same values. Some color terms/BBSs do not offer screen color changing, and base their operation on black backgrounds. You can emulate a more simple color BBS by setting all background colors to black, but the creative user with DarkTerm can still get away with writing very colorful messages in the message section. As mentioned, the COL command toggles the BBS between color mode, and mono-color mode. In mono-color mode, the screen colors are always set to the FIRST set of screen colors in CREATE.COL. These are also the same colors used at boot-up time, and at system reset time (between calls). The mono-color mode colors can be changed with the CCOL command, but only during the duration of a user logon. The 33 default color combinations are set for good readability, but you may want to change them. There is a very small program on side 2 called COLOR TEST which will allow you to test the screen colors.

(CREATE.COM). System commands for each of the 4 sections may be defined to any 1-4 character words you like. The 16 commands in the editor section have been built into the system to conserve memory. There are 40 general commands in this file, excluding the various subcommands in the message section, which are explained later. Each command in this file is documented with a REM statement, to summarize it's function. You may skip over modification until you are more familiar with the system. If you find that one command or another is un-necessary to the type of system you want to run (ie. password changing on an assigned password system), then you can disable the command by using non-input characters in the command string. These type of characters can be color characters, control characters, or anything that isn't printable. Graphics symbols can be used in the CBBS for command words, if you want to get fancy, but they may not be used in the ASCII BBS. For each command, there is a 4 character string, and a one byte number. This number is the length of the command. The command string MUST be 4 characters in length. The extra space in a shorter command will be ignored when parsed (ie., the "M" command to enter the message section is padded with 3 blanks, but is defined to length 1). Make sure you set the length to match the string, or the command will not be recognized (NOTE: you may also disable a command by setting it's length to 0, or a number greater than 4). Note also that the commands are divided into sections, except for the 4 universal commands, denoted as transversing, because they are valid in all 4 sections. The message, file, bulletin, and user section commands may be the SAME, yet no conflict will occur. For example, you can use "R" as a command in the message section, AND also use "R" as a command in the user section. Because the sections are different, the commands will not interfere with each other. However, the 4 universal commands, VOTE, G, BAUD, and CHAT, will interfere with any command in the 4 sections, so a command "G" in the user section will conflict with the "G" (logoff) universal command. When all commands have been changed, you should replace this file with the default on the backup of side 2. Then you MUST set the file type variable in line 1200. This variable, CL, determines whether a CBBS command (C.COMMANDS), or an ASCII command (A.COMMANDS) file will be created. Set CL=0 for ASCII, and CL=1 for color. Then RUN this program on side 2 to create the file. A last note of warning: DO NOT use numbers (0-9) in any command. The parser makes use of numerical references (ie. BAUD450) that will conflict in the input string. Never use numbers in a command definition.

(CREATE.TXT). All text you see on the BBS exists in the form of strings, each with a defined, yet variable length. Each of these strings is referred to in this manual as a "prompt", since the function of the string is to prompt you to do something. There are 132 such prompts on the CBBS and ABBS. These are numbered from 0-131, not 1-132. If you have a printer hooked up, you should do the following. On side 2, LOAD the file "text.pal". Then, from immediate mode, type open4,4,7:cmd4:list (RETURN). Then, when printing is done, type print#4:close4 (RETURN). This will give you a visual listing of all the prompts, as their default values. This listing may appear strange, but don't worry. It is a source listing done with the commercial assembler, PAL. If you have PAL, and you understand assembly language and/or machine language, you can do a much better job in modifying the text file than you can by using the CREATE.TXT program. This source is provided so that you may modify it, and so that you can see where each prompt occurs on the BBS, to aid in your modification of the text file. NOTE that prompts #124 to #128 are used for the CBBS only, and prompts #61 to #64 are used for the ASCII (ABBS) only. The last prompt in the text file is used as an end-of-file prompt, to tell the BBS where the end of the last prompt is found. This is set to a value of "EOF" in the default file, and is not accessed on the BBS. You should just use the "T" command in CREATE.TXT to transfer it off to the new file. The N133 in line 5620 of TEXT.PAL is the EOF marker for the file itself. If you modify the source code, you must always make the last line 3 null bytes for end of file. Before, discussing the method of changing the source code, we'll discuss CREATE.TXT, LOAD and RUN this program. You will be asked for a file name. Hit RETURN to load in the default file, BBS.TXT. You have a very small command set with this program. The FREE counter at the upper left corner of the screen indicates free memory space for the text file. This indicator counts down from 8704, which means you have 8704 bytes, or about 35 blocks available for your text file as an upper limit.

It is important to NEVER let this counter reach 0 before you have modified all prompts, or you will have to restart. This program works by swapping one prompt at a time from the old file, and then places it, modified, or un-modified, into a temporary buffer in memory. Because the length of each prompt can change with each redefinition, the prompts are accessed in sequence, from 0-132. You may not change what has been accessed previously, since the lengths have already been encoded into the buffer. This means that you MUST create the file entirely, without partial saving and continuation another day. If you do want to stop, and finish another day, the only way to do this is to transfer all remaining prompts to the new file, then resume at the most recently defined prompt. As indicated when you run the program, you may do 4 things at any point along redefining the text file. Transferring means take the old prompt, as is, and place it untouched into the new file. Restarting means to begin again at prompt 0, scrapping everything done so far, and trying again. This is what you must do if you run out of free text space. Abort will quit the program, and put you back into BASIC. Changing a prompt will clear the screen, and then you will be required to enter text for the current prompt. You will see a cursor at the top of the screen.

Enter any printable characters you want. You may use graphics symbols if you run the CBBS, but not if you are running in ASCII. You may hit RETURN to start another line of text, and keep going until the prompt is done. When viewing a prompt in it's old format, it will appear at 600 baud. You may pause this prompt display with SPACE, resume with SPACE, or abort it's display with the STOP key. Another thing you will see is the appearance of 2 left arrow characters. The first one is to mark the start of the prompt, the other to mark it's end. Why the left arrows? Well, some prompts, like #11-17, the message header prompts, do not have returns (chr$(13)) in them. These prompts should not use a return character, or you will double space the output. The left arrows will make it easy to identify which prompts end in a return character, and which ones do not. After you enter new text, you must enter a CTRL-C to end your input. STOP and CTRL-C are the same value, but hitting stop here will cause nothing to be printed, since STOP will stop the output of the new prompt before it can be displayed. So always use CTRL-C to end input. You may also use CTRL-G as a prompt character. This is the bell tone, and will sound everytime you press it. If you make a mistake during input, you may backspace, but may only do so to the end of the current line. It is not possible to delete upwards, so you will have to abort the input if you need to do that.

Once you hit CTRL-C, you will see the new prompt in review, and be asked to save it to the buffer. If you do not want to save it, enter NO, and you will be able to start over on that same prompt. When all 132 prompts are done, you will have to save the file. To prevent erasure of the original file, "BBS.TXT" will not be allowed as a file name for saving the text file. Use any name you want other than that one.

This next part is for those who want to delve into the machine language method of modifying the text file. Although the TEXT.PAL file is written in PAL source code, it still is relatively simple in structure, and would be very easy to convert to other assembler formats. The file is nothing more than a series of .byte and .asc instructions, pseudo-ops for the assembler. BASIC equivalents would be DATA (numeric) and DATA (string). Pretty simple, eh? The originate of this file is $9A00 hex, or 39424 decimal. This file is set to output to file #2, which has been opened in the opening line (.opt o2 - send object code to logical file #2). The n0 to n133 labels are for reference only, and are not needed for assembly purposes. Comments would do just as well. A text prompt has a simple structure: starts with a null byte (0), and ends with the start of the next prompt, which is, of course, a null byte (ascii 0). In the case of the last prompt, 133, it is 3 zeroes, to indicate end of file. Now, as for the body of the prompt, ANY character, with value from 1-255, can be used. This allows much more flexibilty in modification than the CREATE.TXT program does, as you may use color characters, control characters, or anything you like. For example, consider the string .asc "(F1)(RED)(F3)(BLACK)(F5)(WHITE)Command:(CTRL-G)". This would print the text prompt "Command:", but would first change the border color to red, the background color to black, and the text color to white, and after printing the text, would sound the bell. This cannot be done with CREATE.TXT. .BYT (.BYTE) has to be used in places where a quote (chr$(34)), or a carriage return (chr$(13)) were used, since characters like those cannot be enclosed in quotation marks. Always remember to use the nulls to start off each new prompt. A BASIC aid like POWER would be excellent for use in changing the text source for PAL format. If you use PAL, and RUN "text.pal" after modification, you will see the copyright notice, a "2" (pass 2), and then the address range for the text file. It should read $9A00-$XXXX. XXXX must be a hex value less than $BC00, or you will have to start over, as you have exceeded the 8704 byte capacity of the text file. NOTE: if you have problems making a text file, we will be happy to help devise "themes" for you system. When you buy our system, we will be more than happy to help you set the system up, circumstances permitting. It takes a bit to explore all the possibilities, as some things require a little more insight than other things.

(CREATE.NAM). This is a very small and simple program. LOAD and LIST this program. It contains a 25 character string. This string is the name of you, the SYSOP, or the handle that you want to use as sysop of the BBS (ID# 0). SYSOP is generally considered a standard on many BBS systems, large and small alike, but on a theme board, you may want something fancy. However, some users may still like to write messages to the SYSOP. Once you change the name from SYSOP here, it opens up the possibility for SYSOP to be used in the user list (+us) file, taken as the identity for a standard user, not the real sysop. You should add a dummy user with the name SYSOP to dicourage this from happening, or else drop the system out of public mode. The string must be 25 characters in length, and if you use a shorter name, pad it with blanks like it was done with SYSOP. Note that it also MUST be UPPER CASE characters to allow recognition in the user table. Only alpha-numerics are allowed in the name. BUT due to the nature of the ID # system this BBS runs in, names are not necessary in the operation of the BBS. You could use a sysop name with color characters, or graphics symbols if you want. But you must remember now that your name will not be allowed as input. Only your ID# will. It's just a feature of the CBBS that can be exploited.

(CREATE.TIM). Although there are multiple levels of time access for each user in the user record, there is nothing that can determine at what time during the day certain users may have access to the system. CREATE.TIM is used to set time restrictions on the BBS for certain 1 hour periods on the BBS. What this does is allow from 1 to 25 users special access to the BBS at any hour of the day, such that no other user may access the BBS at this time. This program contains 2 sets of DATA statements. The first set of 25 is the user ID#s of those users who are permitted access during the set time periods. If you do not want to add 25 users to this list, then leave the rest of the data statements set to 0. The next set of 24 DATA statements set the time periods that the users in the above list may access the BBS without other user intervention. Each data statement uses an hour number, 11,12,1,2,3,4,5,6, 7,8,9, or 10, followed by an AM or PM indicator, "a" for AM, "p" for PM (ie., 12 am (midnight) would be DATA 12,"a"). Make sure the "a" or "p" is lower case. Examine the default setting. For each 1 hour period you want the users in the above list to have restricted access to, you set that here, and you must NOT use any hours that are not in the restricted time period. For example, if you have an assistant sysop who wants to get access to the BBS at 4 am without ANY other user (except the SYSOP ID#0, who is always permitted access) from accessing the system, and the person has ID# 255, then you add 255 to the first data statement, and set ALL time period statements to 4 am. Why set all to 4am? Because if you don't, then something like DATA 3,"P" will also not allow any users to access the system at 3 pm except the SYSOP and ID# 255. If you want to add another user to this list, say ID# 23, then put him in the next data statement in the first set.

We'll give a more complex example here. We have 15 users who want special access to the BBS at the hours of 9 am to 11 am, and 8 pm to 10 pm. The user ID#s are 1,3,5,7,9,11,20,30,40,50,100,150,200,225, and 250. We will place each of these 15 numbers in the first 15 DATA statements of the first data statement set. The rest we leave as 0's. Next we change the time period data statements. The first four would look like:

```
DATA 9,"a"
DATA 10,"a"
DATA 8,"p"
DATA 9,"p"
```

This reserves the time periods for those 15 users, and the SYSOP. What about the remaining 20 data statements? Well, if they are left as is, then those 15 users will be the only ones to access the BBS at 4 am as well. So, to prevent that, all time period data statements not used should be set to any of the previous settings that are used. So, set the other 20 to DATA 9,"a", or DATA 10,"a", or the other 2 proper times, so that the unused time periods become valid time periods. Now, you may not want to use time periods at all, and would rather give access to the BBS to all users at all times. Well, in line 1170, the variable TR determines whether time restrictions will be allowed on the BBS or not. Set TR=0 to disable it, TR=1 to enable it (ignore what the REM's say). RUN this program on side 2 when you are ready. Twenty four time periods are given so that you could run the entire 24 hour day as a restricted time period, although I don't think it would be too useful for general BBS operation.

IF a user logs on during a restricted time period, and is not the SYSOP and is NOT on the list of users who have access, then the BBS will respond with "Restricted Time Period.", and then the user will be logged off the system. This prompt is built into the BBS, and may not be changed. If you are running public or validation mode, and a non user calls in, then the BBS will proceed to hangup on him without any notice whatsoever. Time restricted periods are taken literally with this BBS. If a user is still on when a restricted time period starts, he will not be logged off until his time is up, or unless he uses "G" to logoff. For example, 9 am - 10 am is a restricted time period, and a user logged on at 8:45 am, with a 30 minute time limit. He will be allowed to stay on until 9:15 am, even though 15 minutes elapsed beyond the start of the restricted time period. If your system involves long time limits, then you should use consecutive restricted time periods.

Be aware that the time settings in CREATE.TIM are 1 hour periods. So DATA 3,"p" reserves 3 pm to 4 pm inclusive.

(EDIT.BULLETINS). The bulletin section can operate in one of two ways. The first method is through the use of SEQuential files, no RELative file storage. This was set in CREATE.CNF to primate, or text mode. It is a simple, but relatively fast method to store bulletins. All main bulletin descriptions and titles should be placed in a SEQ description file named "+bt". Before you create this file, you should delete the "+bu" relative file, and overwrite it with this file, as relative files are no longer needed. What this means is that the bulletin section now operates like a SEQuential file reader, nothing more. You will also lose access to the Delete and Post bulletin commands, since simple text storage has no provision for adding or deleting bulletins from the bulletin description data file. Also, since the "+bu" file holds the access levels as well, you will lose the access level option in the bulletin section, using text mode. The only real advantage of this mode of storage is that there is zero search time in finding bulletin names. As an example, searching through the entire "+bu" file, if you have 60 bulletins, will take about 15 seconds to find the 60th bulletin, this being on a 1541 drive. Also, with SEQ storage, you can hold as many bulletins on disk as you want, as long as you are aware of the number of files your system can hold. Relative storage uses the "+bu" relative file to store the name, description, and access level of ALL bulletins, whether they are main bulletins or sub-bulletins. This is done through a unique internal linking system, that will divide bulletins up into smaller sections (ie. like categories in the message section). The only way to explain how both modes work and how they differ is through some examples.

We will attempt to place the following bulletins in the bulletin section:

```
GENERAL : Main bulletin description for the following 3 bulletins:
          GEN1 : General bulletin 1 (a sub-bulletin).
          GEN2 :    "       "    2 (      "      ).
          GEN3 :    "       "    3 (      "      ).
HELP    : Main bulletin for 4 help file sub-bulletins.
          HELP1 : Help sub-bulletin 1.
          HELP2 :  "       "      2.
          HELP3 :  "       "      3.
          HELP4 :  "       "      4.
MODEMS  : Main bulletin text file describing modems.
SECRET  : Secret bulletin, not listed on the main listing, or anywhere else.
```

GENERAL and HELP are 2 main bulletins, that when read, will list the names and descriptions for the sub-bulletins GEN1-GEN3 and HELP1-HELP4 respectively. So, the "+bt" file would appear something like this:

Description	Enter
General BBS Information	GENERAL
Help With The System	HELP
About Modems	MODEMS

This would be the main bulletin, SECRET is hidden of course, and does not appear in the listing. When you try to read GENERAL, for example, you'd get something like:

Description	Enter
The Message Section	GEN1
The File Section	GEN2
The Bulletin Section	GEN3

The same type of thing would be for HELP when you try to read it. If you try to read MODEMS, then you'd get a text file "+b.modems ", a SEQ file about the topic, since it is not a file that lists any sub-bulletins. But "+b.general " and "+b.help " would each be a SEQ text file that would list the descriptions and names for their respective sub-bulletins. Bulletin text files exist in the format "+b.name ", where the "+b." is mandatory for a bulletin text file, and "name" is a 1-8 character name, alpha-numerics ONLY, padded with spaces so that the length of the file MUST ALWAYS be 16 characters. This is important to remember that the file name be 16 characters, and padded with spaces, NOT shifted spaces, or the file will not be found when a read is attempted. The name part (excluding the "+b.") must never exceed 8 alpha-numeric characters either (you may also uses spaces in the bulletin name, ie. "+b.test 1 " is a valid bulletin name). In text mode, this is the way to organize a bulletin section with sub-bulletins. The SECRET bulletin can be read, since it is just another file, despite the fact that it is not listed on any other SEQ description file. You cannot place a password on it, nor can you place an access level on it. If you really want a password on a bulletin, just use the password as the bulletin name, and change the file name often. Easy to do.

The RELative way to organize this same example at first may seem difficult to grasp, but will become easier to understand once you try it a few times. In this mode, the bulletin section assigns each bulletin a reference number. In fact, this number is the actual record number in the "+bu" file. When you attempt to read a bulletin using the reference number, then the search time is zero, like the text mode method, since the BBS will not have to search records; you wil be placed at the exact record by specifying the reference number. To read a bulletin using the reference number, consult the manual under the BULLETIN SECTION. We will show now how to make the above example the RELative way.

The GENERAL and HELP files are description files, and in relative mode, each bulletin has it's own description, whether it is a sub-bulletin or not. Thus, it is not necessary to really make up a description file, is it? So, what we do is assign a main bulletin link numbers. A link number is a value from 0-255 that determines a) whether the bulletin is a main bulletin or a sub-bulletin, b) whether the bulletin is hidden, and c) if the bulletin is a sub-bulletin, what main bulletin this bulletin is linked to.

Now the files GEN1, GEN2, and GEN3 are linked to GENERAL, each being a sub-bulletin. Link #0 is a special link value, indicating that the bulletin is always a main bulletin. Each bulletin actually has 2 link numbers. The first link number is what was just explained; whether it is a main bulletin or a sub-bulletin. The second link number is the sub-bulletin link value. This is a number from 1-255, which sets the link between a main bulletin and the sub-bulletins. A second link number of 0 can ONLY be used by a main bulletin, which has a first link number of 0. This will make the main bulletin a text file. A diagram may prove worthwhile here:

```
GENERAL L1:00 L2:01    HELP L1:00 L2:02          MODEMS L1:00 L2:00    SECRET L1:255 L2:255


GEN1 GEN2 GEN3         HELP1 HELP2 HELP3 HELP4

L1:01 L2:01            L1:02 L2:02
(FOR ALL 3)            (FOR ALL 4)
```

The GENERAL bulletin is assigned a link #1 of 0, indicating a main bulletin. The second link number is 1, indicating that this bulletin will list the descriptions and names of all sub-bulletins with a link number of 1. This is the case with GEN1-3, each which has both link numbers set to 1. If you make a sub-bulletin on disk, BOTH link numbers MUST have the same value. With EDIT.BULLETINS, the syntax is a little different, but more on that later. HELP has link #1 set to 0, another main bulletin. Link #2 is set to 2, which means that this will list all sub-bulletins with link numbers of 2 when HELP is read. HELP1-4 each have both link numbers set to 2. MODEMS does not have the function of listing any sub-bulletins, so, we do not link it to anything. It IS a main bulletin, so we have to assign it a link #1 value of 0. Link #2 must be 0 now, because we are not linking it to any sub-bulletins. So both link numbers are 0. The SECRET bulletin is not to be listed anywhere on the disk, so we must assign it a link value that cannot be accessed by any main bulletin. This link pair is 255. By setting both links to 255, it is not possible to list this bulletin at all. If you were to try to POST a bulletin with link #1 set to 0 (main) and link #2 set to 255, the BBS will not allow it. Link #255 is reserved for non-listing bulletins. Why 254 links? Well, it is the limit, but you will never use all the links, since you can only have 199 bulletins maximum. This means only 199 link numbers can be used. What about text file storage on disk? With text mode, we had to make a text file for each bulletin. GENERAL and HELP would require a SEQ text file each, each listing the names and descriptions for their respective sub-bulletins. With RELative mode, a main bulletin that has a second link number from 1-254 will NOT require a file on disk. When you try to read GENERAL, it will list the descriptions and names of the bulletins it is linked to, from the sub-bulletins' record data.

You will still need to make text files for GEN1-3, HELP1-4, MODEMS, and SECRET, stored in the same file format as described in text mode. To summarize, the bulletins can be posted by linking in the following ways:

1. Main bulletin that is itself a text file with no sub-bulletins: Link #1:0 Link #2:0 File : "+b.name"
2. Main bulletin that lists sub-bulletins when read (X: 1 to 254): Link #1:0 Link #2:X File : none.
3. Sub-bulletin that is linked to main bulletin X (X:1 to 254) : Link #1:X Link #2:X File : "+b.name"
4. Secret Sub-Bulletin, not linkable to any main bulletin X=255 : Link #1:255 Link #2:255 File : "+b.name"
5. Lost Bulletin: X is a link # that no main bulletin uses : Link #1:X Link #2:X File : "+b.name"

A lost bulletin will occur when the sub-bulletin is unable to link to any main bulletin. For example, if the second link number for GENERAL was changed from 1 to 23, then the 3 sub-bulletins, GEN1-3, would no longer be listable. When you switch link numbers around, you will realize the flexibilty of this form of bulletin organization. Say you posted a bulletin GEN4, but mistakenly gave it link numbers of 2. What this will do is link GEN4 to HELP, and when you read HELP, it will show up in the listing along with HELP1-4. How do you get GEN4 into the GENERAL listing? Switch both link numbers to 1. What if we wanted to make GEN4 a main bulletin? Switch both link numbers to 0. What about making GENERAL a text file, and not a linker to the files GEN1-3? Switch the second link number to 0. If you switch both link numbers of GENERAL to 2, then GENERAL will become a sub-bulletin linked to HELP. But now you need to give GENERAL a text file, where one didn't exist before. By moving link numbers with the "B" command in the editor section, you can re-orginize the bulletin section without having to change any text files.

Also, you can assign access levels for any bulletin, main or sub-bulletin. A user with an access level below the level of the bulletin posted will be able to list the bulletin's description and title, but will not be able to read the bulletin, and will be notified of this.

At this point you will be either confused or you will understand to some extent. So, we'll go onto the bulletin editor. The bulletin, program, text editor are all full screen editors, meaning you can change data by simply cursoring around the screen, and typing over any previous data. At any time, you can use the cursor up, down, left, and right keys. The delete key only operates on the largest field of each editor. In the case of the bulletin section, this would be the description field. To delete in another field, use cursor left and SPACE. LOAD EDIT.BULLETINS and RUN this program. It will search side 2 to retrieve the maximum bulletin count. If you removed side 2 after loading, you will be prompted to insert the MASTER DISK. This refers to side 2.

At the top of the screen, you will see the current number of bulletins, the current bulletin the cursor is positioned on, and the maximum number of bulletins that you have set with CREATE.CNF. The F7 key is the function access key. The following functions are supported:

L: Load bulletin file into memory. Note that if you are using text storage, you do not need to use this program at all.
S: Save bulletins back. This will re-write the "+bu" file back to the files disk.
P: Print bulletins.
D: Disk directory.
E: Erase all bulletins from the current bulletin down.
+: Insert a blank line.
-: Delete current bulletin.
X: Exit to BASIC.
CURSOR UP gives 5 times scrolling speed up.
CURSOR DOWN gives 5 times scrolling speed down.

As well as the F7 functions, HOME will place the cursor at the first bulletin, and CLR will place the cursor beyond the last position. F1, F3, and F5 will change the 3 text colors, F2 and F4 will change the border and background colors. Hitting RETURN from any field places you in the next field, or to the next record if the current field was the last one.

The display for bulletins appears like:

Name Description LT-LK

The first field is the bulletin name (alphanumeric). The second field is the bulletin description (25 characters, any type), the L in LT is the access level, the T is the bulletin type (M for main, S for sub bulletin), the LK is the second link number. The program will automatically set the first link number to the second in the case of a sub-bulletin when saving (ie., S122 will set link #1 to 122 when saving).

(EDIT.CHAT). This is the sound test program for setting the parameters for the CHAT mode pager in the BBS. Although it would have been nice to allow full control of the SID chip when creating sounds for the CHAT pager, it was not done, in order to conserve memory. The sounds generated will all be ring modulated, an effect used to produce bell and chime sounds. All information for experimenting with this small program is given on screen, so it will not be presented here. Read back to CREATE.CNF for more information. If you are not familiar with the operation of the SID chip, consult the Commodore owner's manual or reference guide.

(EDIT.COLOR). If you want to create color text files for use on the CBBS, there are many ways to do so. The buffer editor in DarkTerm 4.0 and the color editor here are virtually the same program. Information on how to make color files is given in the DarkTerm doc file on side 2, but it will be given here as well.

There are 7 options to the editor: Edit, Load, Save, Append to disk, Append from disk, Review, and Quit. Selecting edit places you in the edit buffer. Note that anytime you select edit, the buffer WILL be cleared. This is NOT the case with the color editor in DarkTerm. The number of free bytes is set at the upper left hand corner. There are 45056 bytes available for your color file. This is 178 disk blocks, which should be sufficient. While in the edit buffer, anything you type will be placed in the buffer. The CBBS and DarkTerm communicate using a special color mode, sometimes called keyboard mode. Keyboard mode allows you to use anything on the keyboard as input. This means all color keys, edit keys, and graphics symbols. The color mode we use has the following characteristics:

-full non-destructive cursor control.
-23 line display, with cursor row and column position values placed on the status lines of both term and BBS.
-Insert and delete do not lock into quote mode, and will operate only on the current line range.
-CONTROL (CTRL) 6 is the bell tone.
-Reverse Video, Home, CLR, and the color keys (C= 1-8, CTRL 1-8) all operate as in BASIC immediate mode.
-The RETURN key strips all characters from the current line, after the point of RETURN.
-The F1, F3, and F5 keys will set up for screen color change. Changing the screen colors involves 2 keypresses. The first is the function key, the next is one of the 16 color keys. F1 will change the border color, F3 will change the background color, and F5 will change the text color. For example, F1, then (RED) will set the border color to red.
-CTRL-I enables upper case/graphics mode. CTRL-H restores the screen back to normal lower case mode.

While in edit mode, CTRL-D will remove the current character from the buffer. Delete's are buffered, so CTRL-D must be used instead. CTRL-D will remove any characters, even color characters, from the edit buffer. But to show how it works, we'll use a line of text as an example:

"this is a tesT" : The cursor is at the "T" in tesT. Entering a control d will remove the T from the buffer. But if you have a case like: "this isS a test", and the cursor is at the S in isS, CTRL-D will remove the "S" AND will shift the entire buffer, after that, one position over. If you are working on a color screen, and you try to use CTRL-D, you may put the color screen out of synch by doing this, as deleting a color key will move the text over one position, as well as remove the color character.

To exit edit mode, enter a CTRL-A. Use CTRL-C or STOP when using the color editor on the CBBS. At this point you should review the edit buffer. When selected, your text data will scroll by at 300 baud. By entering the "S" key during a review, you can speed up or slow down the rate of text display. Enter STOP to abort the listing, or SPACE to stop/start it. If you enter CTRL-X will the text is being displayed, you will be placed back into edit mode. This is like placing the cursor at that point in the buffer. You may resume typing now, and whatever you type will write over the current buffer contents, AT that position in the edit buffer. Use CTRL-A (STOP in CBBS) to exit again. While in edit mode, you may also use CTRL-X to exit. BUT in this case, the buffer will be truncated from the point of exit onwards. So, if you re-entered 256 bytes into a buffer that held 600 bytes of data, and hit CTRL-X, the remaining 344 bytes are removed.

Append will either take a file from disk and add it to the buffer, or take the buffer and add it to a file, depending on which one you select. If you review the text to the end (ie. don't hit STOP at all), you will be placed back into edit mode, so that you may resume where you left off. Enter CTRL-A (STOP in CBBS) to exit if you are done editing.

(EDIT.PROGRAMS). This program works in the EXACT same way as the EDIT.BULLETINS program, with a few exceptions. First, the arrangement of the fields on-screen are different. The arrangement appears like this:

```
File Name        T-SZ D ACC SID PASSWORD
---------------- ---- - --- --- --------
```

The first field is the file name, as it appears on disk. Text will be in upper-case, but the file itself on the disk MUST be in lower case. This applies to ALL programs you place in the file section for user access. Upper case file names, file names with illegal characters, and a file name with any blanks following it are not allowed (ie. "modem 1650" as opposed to "modem 1650 ", in which the latter has a blank after the last character, making it an invalid file name). If you use physical directory storage, and bypass the "+pr" file, then you need not use this program at all. All access will then be done right off the disk itself. As such, you are no longer required to remove padding from the end of any file name (ie. "modem 1650 " is ok now), but you still are required to use lower case only. File names with non-printing characters like color keys (ie., koala painter pics) can not be accessed off a disk physically using the D or U commands. However, there is no restriction on file names when using the MD or MU commands. In fact, anything that can be stored on a disk directory in file format can be sent via multi-file transfer (Note: if some non-printing character does appear in a file name, the "C" command may distort the appearance of the disk directory. This is not important. It may also alter the appearance of the file transfer screen on your side; Also not important. No harm will be done).

The next byte, T in T-SZ, is the file type, in this case being a P (program), S (sequential), or U (user). Note that Punter protocol has no provision to single file transfer user type files. In the case of user files, they will be sent over as SEQ files, if the D or DOW commands are used. The multi-file transfer commands will always get the file type right.

SZ indicates file size, in blocks. With relative storage, you need not make the block count match that of the actual program on disk. Since this BBS was oriented towards serial drives, the block size can't exceed 999 for any file (nevertheless, a 1300 block file can still be downloaded or uploaded to the BBS; the high digit will simply not be tallied in the directory listing).

D is the directory number. As mentioned previously, relative directories range from 0 to 7 (not 1-8). You may alter directory numbers at any time for any program. You will end up hiding programs by placing them in directories that don't exist (ie. you place it in dir 7 when you have 3 relative directories).

ACC is system accesses. This is a download counter of the program. Everytime a program is downloaded, the BBS will update this access counter. It ranges from 0-999, and you may place false counter values in here, if you want.

SID is the sender ID#. Use any number from 0 to 255. This ID# is important for one reason. The DEL command for deleting programs allows deletion if the user is at level "s" (SID ignored), OR if the SID matches the ID# of the user who accessed the DEL command. Since DEL erases the program and record, you should never assign SIDs to any random user. After a UPL is performed, and the user has uploaded the program, his ID# will be set as the SID, giving him access to delete the uploaded file at any time.

PASSWORD is a 1-8 character password. This password will place download restriction on that program, unless the user sent the program, is at level "s", OR knows this password. The user who uploads with UPL can also mark a file with this security code, in order to protect the program from any normal user. If you do NOT want to place a password on a program, you must enter the LEFT ARROW key at any position in the PASSWORD field. This will place 8 consecutive lower case "a" characters in the field. All passwords exist in upper case. Thus the reasoning behind the lower case "a"s to signify no password.

Another interesting feature of relative directories is that they too can be referenced by the record number. When you use the LIST command to list a relative directory, a reference number will be shown, in much the same way as in the bulletin section. This number is the record number, and here too has it been provided to allow zero search time when accessing files via the DOW command.

Efforts have been made to parse all input in the fields for this and the EDIT.BULLETINS program properly, but it is your obligation to ensure no fields are left blank. In the numeric fields (SZ, ACC, SID), hitting RETURN alone will automatically place zeroes in the field. Also, when entering numbers, the parser will justify all input with leading "0"s to ensure the field has been properly filled.

IMPORTANT! One function was not named, accessed with the F7 key in both EDIT.BULLETINS and this program. Enter F7 then "O" to sort (order) the bulletins/programs alphabetically. This quicksort routine is very fast and efficient, and will make your bulletin section and file section look neater, if the records are ordered (unless you use chronological ordering, or some other sorted order).

(EDIT.TEXT). When it comes to the world of text editing, the best text editor is undeniably a word processor. When it comes to bulletin boards, and text files on a BBS which has only a 40 column display, then it's not as necessary to have quite so formal a text display. This program is a very simple, full screen, 40 column text editor. It will only edit SEQ text files, and is an EXACT duplicate of the EDITLINK file in DarkTerm 4. Although help for this program is provided in the DarkTerm doc file, it will be repeated here.

The capacity of this text editor is 1125 lines, which is quite good (we are dealing with a small program of few features). This works out to about 45000 bytes of text buffer area. As said before, this program is designed for 40 columns, the size for creating system files on the BBS. If you are going to run the CBBS, you can put color in your files with EDIT.COLOR, since color messages cannot be created with this program.

Editing Keys: Cursor Keys, CLR moves cursor to bottom of text, HOME to the top of the text data. Insert and Delete will only work on the current line the cursor is on (you cannot shift paragraphs with the program). The RETURN key will strip all characters on the current line after the point of return.

The functions supported with the F7 function select key are:

-L: Load File. (SEQ files only).
-S: Save Buffer. (Saved as SEQ file).
-M: Merge File. This will add the file to the end of the current text data in the buffer.
-A: Append Buffer. This will add the buffer contents to the end of any SEQ file on disk.
-P: Print Buffer. This is not a fancy printer option. This is provided for draft quality output only.
-+: Insert A Line.
--: Delete A Line.
-$: Send disk command.
-#: Set disk device number to 8 or 9.
-E: Erase all text from cursor position down.
-D: Disk Directory.
-C: Toggle CAPS mode.
-X: Exit to BASIC.
-Cursor Up and Down will give 5 times scrolling speed through the text data.

(EDIT.USERS). This user editor offers more than the BBS user editing commands in the editor section do, although you can still get away without ever using this program. But it will be faster for mass editing if this program is used. You should be careful to always save user data back if you ADD, DELETE, or MODIFY any user record. By re-writing, you will save all changes made. A simple exit will not save those changes. If you are only going to get a summary, or a listing of the users on printout, you need not save the records back to the "+us" file.

The user record is organized as follows:

-Access Level. Can be a number from 1 to 9, or "s". Access levels were explained before, with the exception of level "s". Level "s" users are designated as special access users, and this applies even if you do not run the BBS in access level mode. A level "s" user has the following capabilities that other users do not have:

-access to the editor section, with full access to 14 of the 16 editing commands ($ and K excluded).
-access to "+" files.
-access to download any file with a password on it
-access to read any messages, public or private, and the ability to delete, or forward them to anyone.
-unlimited time
-no time outs for sitting around doing nothing. A normal user will be thrown off the BBS if he does not input anything after 2 minutes. A level "s" user can stay on forever.
-ability to abort any of the "+sy.X" files, even if abort status is removed.
-ability to add, delete, or modify any user, including himself.
-access to use the D command in the bulletin section to delete any bulletin. Also can delete any program in relative file storage using the DEL command.
-unlimited access to the entire file section, and the files disk.
-as such, a level "s" user has much power on the BBS, and you should only give this type of access to a trusted user or assistant sysop.

-User Name. The user name must be 5 to 25 alphabetical characters (with optional spaces as separators), and will always exist in upper case format. In addition to A-Z, and SPACE, the "." may also be used in the user name. You may use any characters you want really, but the name identifier routine only recognizes these characters. Using illegal characters means no names can be used as inputs; however, ID numbers are still ok, even if the name is invalid. So, if you want creative names, you may end up restricting yourself to an ID# input system only.

-Password. This can be from 1 to 8 characters, upper case (the EDIT.USERS program always converts the case, so you can still use lower case as input with this program), any characters allowed.

-Last On. This is the date of last access on the BBS. This value can not be changed with the BBS or this program, unless you use a disk doctor program to do so. When adding users on the BBS, this date is set to the current date on the BBS. When adding users here, this date is set to "00/00/00-00:00a".

-Time Limit. Ranges from 1 to 59 minutes. If you try 0 or a higher value, the BBS will most likely crash. If you feel the need for longer time limits, then grant the user unlimited time, or give him normal time limit, so that he may re-log onto the BBS as many times in a row as desired.

-Time Used. This will always be set to 0 for any user, unless that user has a daily time limit. This is the running time counter for the user's time on the BBS. It will count from 0 to his time limit, and reset to zero at logoff/logon if the user has normal or unlimited time. The daily time limit will accumulate for each user logon, and be stored here, to be retrieved at next logon. Do not set this limit beyond the user's time limit.

-Time Status. This is the 2 bit binary time status string outline under CREATE.CNF. Remember, bit 1 is daily time limit, and bit 0 is unlimted time. With EDIT.USERS, you will not use "0"s and "1"s, but will instead use "Y" for Yes ("0"), or "N" for No ("1"). Thus, the time status would have the following outcomes:

```
YN: Daily Time Limit (bit 1) : Yes - Unlimited Time (bit 0) : No  - Result : Daily Time.
NY:   "    "     "      "    : No  -     "        "     "   : Yes - Result : Unlimited Time.
NN:   "    "     "      "    : No  -     "        "     "   : No  - Result : Normal Time.
YY:   "    "     "      "    : Yes -     "        "     "   : Yes - Result : Unlimited Time (Takes precedence).
```

-Relative Directory Access. Instead of "0" and "1", use "y" and "n". For example, "ynynynyn" gives the user access to directories 0,2,4,6, and no access to directories 1,3,5, and 7.

-Physical Directory Access. Identical to relative directory access, but applies to the physical directory commands.

-System Access. Use "y" and "n" instead of "0" and "1". The 8 bits are as follows:

Bit 0 - View "+re" file.
Bit 1 - Multi-File upload (MU).
Bit 2 - Multi-File download (MD).
Bit 3 - Access to physical directory commands (MD,MU,C,D,U).
Bit 4 - Access to the file section (F).
Bit 5 - Posting bulletins.
Bit 6 - File section downloads (DOW).
Bit 7 - "+" file access.

All these bits were explained under CREATE.CNF. Remember that bits run right to left, not left to right. For example, "ynnynynn" would, reading from LEFT TO RIGHT, summarize as:

-Can access "+" files.
-Cannot use the DOW command.
-Cannot post bulletins.
-Can access the file section.
-Cannot access physical directory commands.
-Can use multi-downloading.
-Cannot multi-file upload.
-Cannot view "+re" restricted file.

-High Reference Number Of Last Message Read. The message section uses reference numbers to allow greater flexibility in reading and writing messages. As a user reads newer messages, the newest message read will also have the highest reference number. This number will be saved every time the user logs off, or hangs up, or is forced to log off. You can change this number, but since it is always updated automatically in the message section, it's not really necessary to change.

-Total Uploads. This is a 16 bit (values from 0-65535) counter that keeps track of all uploads the user makes on the BBS.

-Total Downloads. Like the upload counter, 16 bit.

-Messages To User. When one user writes a message to another user, this counter goes up for every message addressed to him. This is a 16 bit counter as well. If a message is forwarded to the user, it will not be counted in this total.

-Messages By User. 16 bit counter of all messages the user has written.

-Total Logons. 16 bit counter of all logons the user has made to the system since the first day he was added to the user list.

Given this information, you should now be able to use the commands in EDIT.USERS.

1. List Users. Give a list of users, the same as using the U command in the user section. Use SPACE to stop/start the listing, and STOP to abort the listing.

2. Add A User. Enter the ID# of the name of the user to add. Enter the information as outlined above. Failure to enter any of the information will abort the current addition. ADD, DELETE, and SUMMARY commands wrap around, so that you can use the command easier for multiple entries. Use RETURN to abort the command.

3. Delete A User. Removes a user from the user list.

4. Modify A User. Modify is mandatory, as far as selection goes. When you select the user ID# or name of the user you want to modify, you must enter either new data for any of the fields of the record, or you should enter RETURN by itself. By doing this, you will leave the current field unchanged. Thus, if you select the wrong user, enter return alone for all the fields, and the user record will remain as is. Note that the hi ref# for messages does not have to be set to a precise value. Setting it to zero for new users in the list is the proper way to do it. The lower the reference number, the more NEW messages there will be to read.

5. User Summary. This gives a complete on screen summary of any user, much better than the ST command on the BBS does.

6. Print Users. Since there is a lot of information to print out, it is necessary to make the output as brief and to the point as possible. When you select print users, set the printer paper to top of page. Each user record uses 2 lines of output. Use STOP to abort a user printout. The user printout will appear like:

```
001-JOHN DOE                -PASSWORD-23/09/86-12:34a L:8
59/00:YN:YNYNYNYN:YNYNYNYN:YNYNYNYN:02376:00012:00011:00002:00003:00008:
```

Line 1 gives the user ID#, name, password, date last on, and user level.

Line 2 gives the user time limit/time used, time status, relative and physical directory access, system access, high message reference number, total uploads, downloads, messages to user, from user, and logons.

7. Swap Users. This will swap the ID# of one user with another user (ie. swap user at #43 with user at #67).

8. Transfer Users. This will move a user from one ID# to another ID#.

9. Write User Records. If you use the add, delete, or modify commands, you MUST use this when finished, or your changes will not be saved.

10. Exit. This will exit to BASIC without saving any changes.

Do not sort users! The ID#s should never be changed once assigned, without taking notice of that user's contribution to the system. For example, swapping user #34 with user #45 will give all messages that are in #45's control to #34, and vice versa. This also applies to upload recognition in the relative directories, among other things. This BBS is name based, but it is ALSO ID# based, and one can't function without the other. Just as names shouldn't change, neither should ID#s, at least not without good reason (ie. deleting a bad user).

(CHARSET0-5). On side 2 are 6 character set files. These DarkTerm compatible files are 2 character sets in one file, the first set being the upper case and graphics symbols set, the second being the upper and lower case character set. These files have a load address of $E000 (hex) or 57344 (decimal), and were created using Computa's UltraFont + character editor. You may define your own character set files, but must be sure that you merge an upper case-graphics set with an upper-lower case set, then change the load address to $E000. If you have micromon 64, or another monitor with similar commands, then you would make a character set file as follows:

1. Design an upper case / graphics set and save this as "file 1".
2. Design a lower case / upper case set and save this as "file 2".
3. Load in the monitor. Micromon usually loads at 49152. Type NEW then SYS49152.
4. Type L 7000 "0:file 1" 08. This will load "file 1" at $7000.
5. Type L 7800 "0:file 2" 08. This will load "file 2" at $7800.
6. Type S 7000 8000 "0:charset" 08. This will save both character sets with the file name "charset".
7. Load up a program that changes a file load address, or a disk doctor, and change the load address from $7000 to $E000.

This will create a complete character set file. The 6 character set files on side 2 may be selected when the BBS asks for "Character Set:". Enter the name of one of these files, or use your own custom file. You may use any type of redefined graphics you want in a character set. It does not have to be old graphics symbols, letters, or numbers.

(COPY.DISK). This is a fast disk backup utility. It was obtained from a public domain bulletin board system in Toronto, and seems to get the job done. Without notice of copyright, and since it was found on a P.D. BBS, we assume this program is in the public domain, and if this not the case, then the fault lies with the SYSOP of that bulletin board, and not us. Since we did not write this program it's use is left up to you to understand.

(COPY.FILE). This program will ONLY work with 2 drives. This is a universal 2 drive file copier. It will copy files from any type of drive to any type of drive, as long as the disk for each drive exists in it's own specific format. This is not a fast file copier, despite the fact that it is written in machine language. Yet it will copy any size file. The copier will show each block being copied on screen as it goes. There is no need to swap disks. Note that this program will save and replace any file on the destination drive that matches that on the source drive to be copied. The 6 options at the top of the screen are:
(Y)es (N)o (A)ll The Rest (D)one Selecting (Q)uit Or Abort and (X) Exit To BASIC. Do not set the source and destination drive and device numbers equal. One drive users will have to use some other program for file copying. There is no error-checking used by this copier. The program will likely crash in the event of a read or write error. The entire process of copying is driven through the KERNAL I/O routines CHKIN, CHKOUT, etc., using a simple OPEN/READ/WRITE/CLOSE method, so there should be no drive incompatibility problems. Use (C) to initiate the copy process, and then select all files to be copied using the above options.

(COPY.REL1). This is our relative file copier, used to back up the 5 relative files on the files disk. A normal file copier will not copy relative files, even if an attempt to do so seems to work. The side-sectors will not be copied with a standard file copier. This copier was designed to work best with the storage format of the BBS relative files, and may not work as well with other relative files. Still, this program should be able to copy any relative file with a size of less then 200 blocks. Just specify a file name, and the rest is automatic. You do not have to scratch a relative file off the destination disk if it has the same name as the file you are going to copy. It will be left as is, and the records from the new file will be written to it.

(COPY.REL2). This program is better known as TRANSCRIBE 64, a very good machine language relative file copier published in the TRANSACTOR computer magazine. This program is sure to work with all relative files, and there is no limit on file size. It will however run about 2 to 3 times slower than our relative file copier. The choice is yours. This program was not written by us, and credit deserves to be given to the TRANSACTOR for publishing and Richard Evers for writing this excellent program (TRANSCRIBE 64 appeared in TRANSACTOR Volume 7 Issue 2 Page 42 Sept. '86).

(FIX RELFILES). If you change the limits on any of the relative files, such as messages, bulletins, or programs, then you will have to either make a new files disk, or expand the file. You may also want to recreate one of the files, like the user list (+us). This program will expand or create the relative file in question. The voting file, "+vo", does not use this program. Use CREATE.VOT to fix that file. Set the variable FL in line 1110 to what file you want to change (see the REMs, 1130-1160). Follow the REMs in the program, as it gives supplemental information on this program. Lines 1350-1360 set the start and end records to be cleared in the new file. If you want to start at record 1, then scratch the file, then run this program. For example, you had the message base set to 80 message maximum, and now you want 50 maximum. The only way to make the file "+me" smaller is to scratch the old file, then set S to 1 and E to 50. This will make a new message base with a smaller "+me" file. By not scratching the file, you would still be left with 30 unused records. As an expansion example, we want to take our bulletin file, and expand the limit from 25 bulletins to 100 bulletins maximum. In this case, we set S to 26, and E to 100. This will expand the relative file by adding blanks records, 26 to 100. The first 25 records will not change at all. If there is any data stored in the first 25 bulletin records, it will remain unchanged. You may use this program to modify any or all of the relative files. If you are not sure of this program, you should simply re-create another files disk with CREATE.DSK.

(FIX +SF FILE). The system file on the files disk, named "+sf", is a small program file that keeps track of 4 16 bit counters. These counters are the number of callers, hi message reference number, total uploads, and total downloads. The 16 bit range gives values from 0 to 65535. The hi message reference number is the reference number of the last message written into the message base. If you re-create a files disk, the 4 counters in the +sf file must be preserved. You can copy the old +sf file from the old disk to the new files disk, or you can note the values of the 4 counters on the system before you create a new files disk. If you do the latter, then place the 4 values into this program in lines 1170, 1210, 1250, and 1290. Then RUN this program on the files disk, not side 2.

(COLOR TEST). This is a very simple program that sets the border colors, and prints their values on screen. It's a simple way to set up screen color combinations when creating a color table with CREATE.COL.

(QUICK CYCLE). If you want to do message recycling off line, use this program. The message base will be recycled in memory, a faster way of recycling, but not necessarily a more effecient way. This program is completely automatic. Just RUN it on the files disk. Note that the message base is not infallible. The scratch command sometimes fails to work sometimes, and a message text file may not be scratched even if the relative record itself is deleted. You will need to periodically examine the files disk directory for "extra" message text files, and delete them.

(TEXT.PAL). This is the source code for BBS.TXT. See CREATE.TXT for more information about this file.

(M1650.PAL). This is the PAL source code for "modem.1650". See the technical docs at the end of this manual for details.

(M1670.PAL). This is the "modem.1670" PAL source file. Consult the technical notes at the end of this manual for more information.

(DARKTERM DOCS). This is the documentation file for DarkTerm 4.0. It is pre-formatted to 80 column pages. Use any standard printing utility to print out this file.

(+HE.0 - +HE.4). These are the example help menus for the files disk. You can load and examine them with any text editor.

DARKSTAR BBS V3.0.

Version 3.0 is an all machine language bulletin board system. Side 1 contains only one file that concerns the BBS. This file is the first one on the disk, "(c) 1986 d.s.s.". The files after this in the directory are files for DarkTerm 4.0. You should remove the 15 DarkTerm files from side 1, since you will be unable to write to side 1. File copy DarkTerm to another disk before you use it. The documentation for DarkTerm 4 is on side 2.

This BBS will ONLY load from a 1541 disk drive, or one that is 100% compatible. Side one has been copy protected, and duplicated on both sides. It may occur that you might want to run a dual drive with device 8, like an 8250 using drives 8:0 and 8:1. In this case, you cannot load the BBS from device 8. The program will load independant of what device number you use. Even if you do not intend to use the 1541 on the system as a system drive, you will still need to have it connected to the computer to load the program. If you need to change the device number through the software method, do it like this:

```
open1,od,15:print#1,"m-w"chr$(119)chr$(0)chr$(2)chr$(32+nd)chr$(64+nd):close1
```

OD is the old device number, ND the new device number. You may load the program after this from this drive. In cases where you will be using many serial drives (our system uses 2 1571s and 5 1541s), you will have to change all device numbers to what you need them to be, unless they are hardwired to the set values. You may freely mix serial and IEEE drives together without harm to the system. There is no reason why an 8250, SFD, and 3 1541 drives can't be used at once. A 90X0 series hard drive will work with the BBS, or any other hard drive IF the linkup to the 64 does not use any RAM from $0000-$F000, and the interface emulates a serial drive exactly (ie. the serial status register at $90 (144) must operate like a serial drive). A note to 24 hour BBS operators: if you are using an IEEE system, you can skip this. If you are using 1541s around the clock, you should keep cooling fans on them at all times, or else periodically take the system down a few hours a week (ie. turn the drives off). Some 1541 drives, especially the older models, will tend to heat up quite a bit. On this or any other large scale BBS system, disk access is extremely intensive. You may experience "lockups" on the serial bus if your drive overheats. A lockup will be noticed when the drive light is active, the clock keeps ticking, yet the system fails to respond. This is a serial bus lockup caused by a 1541 overheating. Note that this is proportional to the number of 1541s you hook up to the system. Cooling fans, or periodic shutdown will save you a lot of frustration. The program is not at fault. If you run a BBS only for part of a day, you will not experience this problem unless your 1541 is very susceptible to overheating.

Note that a 1571 drive allows you to use double sided mode to access 1328 blocks on a disk. Refer to your 1571 user manual for how to set up the drive for double sided mode. The BBS will be able to access a disk run in this mode, giving you double capacity. The BBS will NOT load from a 1571 in 1541 mode. You may use the BBS with a 128 in 64 mode, however.

(LOADING). Make sure all device numbers for all drives are set. Place side 1 in the 1541 drive you want to load the BBS from. Any device number will do.

Type: load"?$",device,1. This will load the program, and give the title screen. The BBS exists in 2 formats. Color mode and ASCII mode are compatible with each other in every aspect, except that color files will usually not work on an ASCII system. ASCII files will work on the color system as is, so there is no need to change anything when switching from an ASCII system to a color system, other than make a color command file with CREATE.COM. As said before, the CBBS uses FULL color and graphics, both character sets simultaneously, and allows screen colors to be changed via the modem. Many color BBS and terminal programs do not offer color to this extent. Nevertheless, you can tone down the color on the system, so that any color terminal program will operate with it. You will need a DarkTerm color mode compatible terminal program to take full advantage of the CBBS, however.

Select "C" to load the CBBS, or "A" to load the ASCII BBS.

You will be prompted to insert side 2 after the BBS has loaded. Insert the working (backup) copy of side 2 that all the changes to the CREATE files were made on. The BBS will load in all necessary setup files, and then prompt you for the following:

MODEM FILE. Version 3.0 uses custom modem files that are designed to work with a specific modem type. The current version is only provided with 1670 and 1650 modem files. See the TECHNICAL NOTES at the end of this manual for information on how to make/obtain modem files for different modem types. You may enter RETURN by itself to load in the default modem file, "modem.1650", the 1650 compatible file. Otherwise, enter the file name of the modem file you want to use. For example, for the 1670 you would enter "modem.1670".

CHARACTER SET. Enter the file name of the character set you want to use on the BBS. The files "charset 0" to "charset 5" are provided on side 2. You may also use your own file here, if you have created a proper character set file. If you do not want a custom character set, just enter RETURN alone.

SYSOP PASSWORD. The 255 users have their own passwords in their respective user records in the "+us" user list file. You will have your own password, which you will have to enter here each time you boot the system up. Enter any 1-8 character password, and use this at logon time, or for remote SYSOP access.

SYSTEM PASSWORD. The BBS may use an optional system password. This password must be entered by all users (SYSOP too) at the FIRST return prompt. A normal logon will display "###" as soon as the caller connects. Then the first return prompt will appear, seen in the default text file as "Hit RETURN...". At this point, the user MUST enter the system password if one is used. If there is no system password, then the user just hits RETURN like the prompt says. Failure to enter the system password will result in the caller being immediately logged off the system. Enter RETURN alone at this prompt to not use a system password, or else enter any 1-8 character word for your system password.

TEXT FILE. This is the text prompt file, the one you may have modified with CREATE.TXT (see CREATE.TXT for details). If you enter RETURN alone, the BBS will load in the default, "BBS.TXT". Otherwise, enter the name for your custom text file.

DAY. Enter the current day, a number from 1 to 31.

MONTH. Enter the current month, a number from 1 to 12.

YEAR. Enter the current year, using only the last 2 digits (ie. for 1986, enter 86).

TIME (HHMMSS). Enter the time, in hours/minutes/seconds format. This is a 24 hour military time input. For example:

Midnight : 000000 - Noon : 120000 - 3 Pm : 150000 - 4:15:23 Am : 041523.

NOTE: you must always enter side 2 when prompted to do so. Failure to do so will result in the BBS crashing. No file not found detection is provided in the boot. Also, never enter an invalid day or month. If you do so, the automatic calendar in the BBS will cause something unpredictable to occur when the day turns over at midnight.

At this point, be QUICK about getting the files disk in drive 8:0, and hitting return. The calendar, and thus the clock will not start until you hit RETURN after the files disk is in the drive. If you do not want to be rushed, set the time about 20 seconds ahead of the current time.

Be sure that the files disk is in drive 8:0 when you hit RETURN. Failing to do so will require a complete re-boot of the system, since the BBS will be unable to find any of the system files, thus failing to setup the IFS system properly.

The screen will then shift to the BBS display screen, with the 2 status lines appearing at the bottom of the screen. The ISFS system will begin intialization at this point. Just wait for this process to finish, and the Waiting For Call prompt will appear. The BBS will be ready for your use at this time. You should have all system disks in all drives at this time (Note: the ISFS system is explained in the TECHNICAL NOTES at the end of this manual).

Before going any further, the calendar display format will be explained. Words are not used for each month of the year. Numbers are used to display the date, in the format DD/MM/YY-HH:MMx.

DD: Day - MM: Month - YY: Year (last 2 digits) - HH: Hour - MM: Minute - x: "a" for AM or "p" for PM.

The local modes will be mentioned first, since they deal with the status lines at the bottom of the screen. The actual system commands will be explained later.

LOCAL MODES. There are 2 "local modes" on the BBS. These are referred to as local modes since you have the opportunity to control the system from the keyboard (locally). The first is while you are waiting for caller, the second is active while the BBS is on-line with a user (local or remote), provided it is enabled. While any user is on-line, you will be able to enter anything from the keyboard, as would the user on the other end. While not in the local mode, you can do anything that the user can remotely do, at the same time, while that user is watching. The status lines at the bottom appear as follows:

```
 1  2                      3    4
=======================================
000-SYSOP                    -S-YYYYYYYY  : Status Line 1
12:34p 01-03 CFMO YN0059YYYYYYYYYNNNNNNNN  : Status Line 2
=======================================
  5     6   7  8 9 A  B      C
```

Each item is described as follows:

1: ID# of the user currently on-line.
2: Name of user currently on-line.
3: Access level of user currently on-line (1-9 or s).
4: System Accesses, reading from LEFT to RIGHT are:
   1:"+" file access 2:access to DOW 3:posting bulletins 4:file section 5:physical dirrectory commands
   6:multi-downloading 7:multi-uploading 8:access to view "+re" restricted file
5: System clock, 12 hour format.
6: Cursor row and column position.
7: System status flags:
   C: CHAT command active. Inverted means CHAT is closed and SYSOP will not be paged when CHAT is issued.
   F: File section active. When inverted, the entire file section is closed to the users (F disabled).
   M: Modem suppress. Modem suppression is a way to cut off input and output from and to the modem. By suppressing modem I/O, you can access the BBS locally at FULL speed, much faster than having to send each character over the modem. When enabled, the user at the other end (if one is on-line) will not be able to see anything come his way, and any input he enters will be ignored until modem I/O is re-enabled. Carrier detect will not function while modem suppress is enabled.
   O: Carrier Detect Enable. When inverted, the BBS will not reset the system if carrier is lost. Under normal circumstances, the BBS would hang up and reset for the next caller if a carrier was dropped. This will not happen if carrier detect is disabled. When logging on the BBS locally via F5 or F7, you will notice that the "O" is always inverted. This is because a local logon is usually done without a user at the other end.
8: Time Status. First Y/N is daily time limit, second value is unlimited time.
9: Connect Time (minutes).
A: Time limit of user currently on-line.
B: Relative directory access. 8 Y/N values, one for each of the 8 possible relative directories.
C: Physical directory access. Same as relative directories, as far as values go.

Note that there are quick reference sheets in this manual for a short summary of the status lines and local mode commands.

LOCAL MODE 1 - WAITING FOR A CALL.

Functions:

F1: Change System Passwords. You can change either the SYSOP or the SYSTEM password. If you do not want to change either of the passwords, hit RETURN alone at the one you don't want to change, and it will retain it's current value. If there is no system password up, then you will see 8 consecutive lower case "a" characters for the system password. This is it's value when there is no system password. A system password will appear as 8 upper case characters (ie. "aaaaaaaa" is not a system password, but "AAAAAAAA" can be a system password). If you want to remove a system password, enter the LEFT ARROW key, and the password will be removed.

F2: Modem Defined Key. This is the key that calls the user defined modem routine in the modem file. See the TECHNICAL NOTES at the end of this manual for how to program this key.

F5: 1200 Baud Local Logon. This will perform a local logon at 1200 baud. Local logon works just like remote logon, without (or with) a user at the other end.

F7: 300 Baud Local Logon. Like 1200 baud, but only 1/4 the speed. In both cases, hit RETURN to enter the system after you press the function key.

F8: Exit to BASIC. Resets the C64, and puts you back to BASIC immediate mode.

Note that the 1670 modem may "act up" during a local logon. The 1670 modem does not have a power switch, and since you cannot plug and unplug the modem to and from the user port while the computer is on, you cannot shut off the 1670 from "responding" to AT commands. With the CBBS any lower case "at" characters and any upper case "AT" type characters could cause the modem to behave unpredictably. The AT command protocol in smartmodems works when the modem is usually without a carrier, as is the case with the 1670. In local logons, carriers are absent, so the AT commands are active. This means, that on the CBBS, a message read by you, during a local maintenance period, that has a phrase like "phone this BBS at 234-5678" would not be good. The 1670 would read the "at" part as the attention command, and if a "d" was somehow found after the at, then it would be like "atd234-5678", and the BBS would dial the 1670. The word "attack" would be interpreted as "ata" and would put the 1670 into answer mode. Things like this can cause problems for the 1670, since you do not have a power switch to "turn off" the AT commands. For this reason, you should be careful of what might happen while on line locally with a 1670. The F2 key used in CHAT mode and while waiting for a call will toggle modem responses. Since the BBS operates in full duplex, you might wonder what would happen if the word "at" was read in a message, with the modem responding "OK". In full duplex, the BBS echoes all input to the modem as well as the screen. The "OK" response will cause the BBS to go into an endless loop due to a duplex conflict. A good way to demonstrate this is to load up DarkTerm, enter "ATE1", go into full duplex, then type "AT". The program will go into an endless loop of printing RETURNS. During a reset after a caller logs off the BBS, there will be a 2 second interval when the 1670 sets up for answering the next call. During this 2 second period, a call may come in, and may be answered by the 1670. If this happens, the caller will not be able to access the BBS, and will be forced to log off. The only way to remedy this problem is to write another 1670 modem file that might correct this problem.

NOTE: C= is the Commodore Logo Key
CTRL is the control key
SH is the shift key

C= C : Toggle the CHAT command (toggles the "C" in the status line).
C= F : Toggle file section open and closed (toggles the "F" in the status line).
C= M : Toggle modem suppression (toggles the "M" in the status line).
C= O : Toggle carrier detect (toggles the "O" in the status line).
C= U : Toggle upper case mode (all letters from A-Z will be upper case).
C= K : Toggle keyclick.
C= 1 : Change border color.
C= 2 : Change background color.
C= 3 : Change text color.

LOCAL MODE 2 - SYSTEM ON-LINE.

This is the primary mode for accessing control of the BBS even while another process is going on. In this mode, a user could be reading a file while you access functions. All functions are invisible to the user, locally or remotely, and will not disturb the process of operation.

To engage this local mode, enter SHIFT and RUN/STOP at the same time. The top status line will change to reversed video. When in reverse video, the local mode is active. When active, you will not be able to enter any input on the BBS locally, other than any of the local mode functions. However, the remote user can continue to access the BBS, doing whatever command is available, while you perform the local mode function(s). To disengage local mode, enter SHIFT and RUN/STOP again, and the top status line will return to normal video.

Functions:

C= C,F,M,O,U,K,1,2, and 3 are the same functions as those in local mode 1.

The number keys, 1 to 8, will toggle the user's system accesses, the first set of Yes/No values in the top status line. Refer to the previous page for the system access values. For example, pressing the 1 key will toggle the first Y/N in the status line. This will toggle the user's access to "+" files to Y for yes, or N for no.

The shifted number keys, SHIFT 1 to 8, will toggle the user's access to the relative directories, which is the first set of 8 Y/N values in the second status line. For example, SH 1 will toggle the user's access to relative directory 0.

The CTRL number keys, 1 to 8, will toggle the user's physical directory access, the second set of Y/N values in the second status line. For example, CTRL-8 will toggle the user's access to physical directory 7.

C= 4 : Move user's connect time up 1 minute.
C= 5 : Move user's connect time down 1 minute.
C= 6 : Move user's time limit up 1 minute.
C= 7 : Move user's time limit down 1 minute.
C= T : Reset user's connect time to 0 minutes.

Do not exceed the time limit bounds! Do not let time limit go less than 1 or greater than 59. The same applies to the connect time. The results of exceeding the bounds will be erratic and unpredictable.

C= S : SYSOP Emulate. When you enter this key, the state of the current on-line user will shift to the SYSOP, and will replace the current user without him noticing. This command is provided to allow you to take control of the system while a "normal" user is on-line. The ID# will switch to 0, and full access will be assigned. The name of the on-line user will remain the same however, to indicate that sysop emulate was used. If a guest user is on line (system mode 4), and you want to automatically validate him, then it will be necessary to use the SYSOP emulate function, use the A command in the editor section to add the guest user to the user list, then autolog the user with the U command in the editor section.

NOTE that the functions accessed with C= 6, C= 7, 1-8, SH 1-8, and CTRL 1-8 are not active when a guest user is on-line (system mode 4 only).

C= D : Delete User. This will remove the current user from the user list file (+us), and log him off the system immediately. This is a very quick way to get rid of bad users on the system. This command will not remove the SYSOP.

C= X : This function will force the current user on-line to log off the system.

C= L : This will cycle the access level of the current user on-line. Values will run from 1 to 9, then to "s", then back to 1. Level "s" will allow editor access.

C= @ : This toggles the daily time limit status of the user; the change will be seen in the status line.

C= * : This toggles the unlimited time status of the user; the change will be seen in the status line.

C= + : Place Modem Off-hook. This puts the modem on-line, so that a voice communication can be switched over to a modem linkup by throwing the data switch on 1650 type modems. This also has the effect of stopping incoming calls. Thus you can prevent a call from coming in during a local maintenance period by entering F5 or F7 to logon, then SHIFT RUN/STOP and C= + to place the modem off-hook. Note that for the default 1670 modem file, this function will send an "ATA" to enable a carrier detect for the other user to use for modem linkup.

C= - : Place Modem On-hook. This takes the modem off-line, in effect disconnecting any user who may have been on-line. A quick hangup routine. The 1670 modem file will hang up and send an "ATZ" to reset the modem.

When you change a user's access to the system with the functions above, none of these changes are made permanent. When the user logs off, all access changes will be ignored, and the user will retain his old system access he had prior to those temporary changes. To make all changes permanent, you must enter CONTROL-W at any point of input on the BBS. You must also be OUT of local mode to do this. For example, you changed a user's access level from 5 to 8. If the user logs off, the access level is still set to 5. If you enter CTRL-W while out of local mode (status line normal video), then the level 8 will be written to the user record. A point of input is anyplace where the BBS is sitting idle while on-line with a user. This can even be while the BBS is in the message text editor.

F7 will engage CHAT mode. When you want to talk to a user, you enter F7 while OUT of local mode, and at any point of input (see above). The user will be notified that you request a CHAT, and you and the user will be placed into CHAT mode. While in CHAT mode, you may also set color and duplex so that you can use the CHAT mode as a crude terminal mode (see CHAT command).

LOGGING ON/OFF THE SYSTEM.

As soon as carrier is detected, or you logon via F5/F7, three asterisks ("***") will appear on the screen. This is to indicate connection to the system, AND to set 2 parameters for the ASCII BBS. For the CBBS, you just hit RETURN at this prompt. With the ABBS, the bell tone (toggled with BT) and formfeed (toggled with FF in the user section) status can be set at this point, depending on the value of the next keypress. If the user enters an ASCII value of less than 27 at the "***", then the bell tone and formfeeds will be enabled. Any value over 27 will disable these 2 options. A value of 27 includes keys like RETURN, DELETE, and all keys from CTRL-A to CTRL-Z. Thus, hitting return here leaves the above 2 options enabled. If a key like any alphanumeric character (ascii value > 27) is pressed, the above 2 options are disabled.

At any point on the BBS (even during text file displays), CONTROL-L will toggle linefeeds on and off in the ASCII BBS, and CONTROL-F will toggle the cursor flashing in the CBBS. You will notice cursor flash after text file displays and certain prompts on the CBBS.

After the "***", the "Hit RETURN..." prompt will appear. You will either enter the SYSTEM PASSWORD at this point, or hit RETURN.

The "+sy.1" system file, or the logon bulletin, will be displayed.

The user will now be prompted with the date, and be given the prompt to log onto the system. The user will enter his name or ID#. If the user is not found in the log, he will be asked if he is a new user. If the user responds NO, then he will be able to try to logon again until a proper logon is achieved. If the user responds YES, then he will be designated as a non-user, and follow the actions of whatever mode of operation the BBS is running under.

Mode 1. Public Mode: The non-user is assigned the first available ID# in the user list. If none is found, he will be notified and logged off the system. The non-user will see the non-user file (+sy.0), the contents of which should explain the system to this "new" user. After entering a new password, the user will be assigned the default access settings set with CREATE.CNF, and follow the normal logon procedures for validated users.

Mode 2. Private Mode. The non-user file will be displayed, if active, and the non-user will be logged off the BBS.

Mode 3. Validation Mode. The non-user file will be displayed, giving information on how to apply for access to the system, after which the user will complete an application, or fail to do so. In either case, the user will be logged off the BBS.

Mode 4. Guest Mode. The non-user file will be displayed, giving information on what a guest user has access to and what rules apply (all this non-user file stuff is, of course, optional). The non-user will be assigned guest user access, and be allowed to log onto the system as a guest user (see CREATE.CNF for the particulars of a guest user).

Mode 5. Shutdown Mode. No matter who logged onto the system, the caller will be logged off the BBS. This will occur even BEFORE the user can enter his name or ID#. The logoff will occur after the opening bulletin is displayed, which should contain information as to why the system is down. Of course, if you de-activate the opening bulletin, the BBS will only answer calls, then hang up on the user. This does nothing more than keep the phone line tied up and out of BBS service.

When the validated user is found in the user list, he will be prompted for his password. If the SYSOP logs on, he will be asked for his password one time only. If the input was incorrect, the caller is logged off. The validated user in the user list has 4 tries to get the correct password. Note that the BBS considers a name less then 5 characters long invalid. Other characters mentioned under EDIT.USERS are invalid as well. Failure to enter a valid name after 4 tries will also log the caller off the BBS. With a correct password entered, the user's date of last logon is updated to the current date, and is logged on the system, UNLESS the user has a daily time limit. If the date of last logon matches the current date, AND the time used is equal to his time limit, then the user will be notified that he has used his daily time limit, and cannot log onto the system. A daily user's time-used will be set to 0 when the current date does not match that of his last logon.

At this point, a user will either be logged on the system, or logged off.

Logging off the BBS can be accomplished in several ways:

1. The user does not pass the logon procedures outlined above.
2. The user's time limit has been reached.
3. The user did not input anything for 2 minutes, and the idle time routine logged the user off for lack of input (this does not happen for the level "s" user).
4. The user was deleted locally with the C= D function.
5. The user was forcibly thrown off the BBS locally with the C= X function.
6. The user logged off the system with the "G" command.
7. The user was disconnected or hung up on the BBS, and carrier was lost, with carrier detect being enabled. (Note that carrier detect is always enabled when a call is answered by modem).
8. The local mode function C= - was used to put the phone on-hook, thereby disconnecting the user.

Logoff procedures for the BBS can take no time at all, or can take a few seconds, depending on what the BBS has to do. The normal full system reset at logoff is:

-the user's system status is updated to his record in the user file.
-the user is added to the user log.
-the +SF system file is updated with new counter values.

At logoff, the modem will stay off-hook for the 1650 modem file, to prevent incoming calls. The 1670 modem file will send an "AT S7=0 D", which has the effect of stopping incoming calls. There may be other modem files on side 1 or 2 of the BBS disks. These will have been late additions to the system, added after this manual was written. All modem files will start with "modem." to let you know of their addition; supplemental information will be given separately for additional modem files.

When the user is logged onto the system, the screen clears, and the following appears:

1. The daily bulletin (+sy.2) is displayed.
2. The optional restricted file "+re" may be displayed, and will be seen only by those users with access to see it.
3. The number of caller is displayed.
4. The number of users is displayed.
5. The number of messages is displayed.
6. The number of programs is displayed ONLY if the file section has relative directories active.
7. The number of bulletins is displayed ONLY if the bulletin section is operating in relative mode.
8. The number of messages written to the user on-line is displayed ONLY if the user is NOT a guest user.

The user is then placed at the main command prompt. From here, the user may enter any section by entering one of the 5 section entry commands. He may also enter any of the universal commands here. A user may NOT enter one section from another. For example, the user may not enter "M" to enter the message section while he is in the user section. Entering RETURN from any of the 5 sections will place the user at this main command prompt again.

This BBS is divided into 5 sections, each which is independant of the others. By sectioning off the BBS, it allows one to make better use of defining custom command words with CREATE.COM. It also allows access to the system to be set for each user at a better level of efficiency. Only the 4 universal commands, G, VOTE, BAUD, and CHAT, plus CTRL-Z, can be used from any section, as well as from the main command prompt.

SYSTEM COMMANDS.

Section Entry Commands. There are 5 section entry commands. The default values for all system commands will be used in this manual. You may change their values at any time. The default section entry commands are:

M: Enter The Message Section.
B: Enter The Bulletin Section.
U: Enter The User Section.
E: Enter The Editor Section. Entry is granted only to level "s" users.
F: Enter The File Section. This section will not be available if:
-the user does not have access to it.
-the file section is disabled locally with C= F.
-the number of physical and relative directories sum up to 0.
-the user is a guest user.
-the user does not have access to physical directory commands on a physical dierctory-only system.

If a user enters any command that is not defined in the command table, then the user will be given a repeat of the current command prompt. No prompt like "Enter ? For Help" will be given.
The "?" command is universal. It is the help command, and will give help menus for the BBS, according to the setting in CREATE.CNF. This "?" value cannot be changed.

XON/XOFF CONTROL. During long listings of text on the BBS, such as the help menus, messages, bulletins, etc., the output can be paused or aborted. This is XON/XOFF. The ASCII standard for Xon/Xoff is CTRL-S to stop a listing, CTRL-Q to resume once stopped, and CTRL-C to abort at any time. Some BBS programs involve sub-listings that require additional list control. For example CTRL-P on some systems will place a user into a sub-command prompt,, while CTRL-C would place the user at the main command prompt. On this BBS, there is only one place where an extra list control key is used. This occurs in the message section during continuous read mode. Refer to that section for how the two abort keys function with respect to the message reading commands. The defaults for this BBS are:

"S" to stop a listing, "C" to continue once stopped, and "A" to abort at any time. In addition, the "Q" key will abort continuous read mode in the message section. You can modify these list control keys with CREATE.CNF.

UNIVERSAL COMMANDS. There are 4 commands that can be executed from any of the five sections, as well as from the main command prompt.

G: Logoff. This is the only command by which a user can logoff the system. The user will be given the option to confirm his logoff, then the logoff system file (+sy.3) may be diplayed, if enabled. The logon and logoff times will follow after this, then the final logoff prompt. The system will then reset for the next caller.

CHAT: Page The SYSOP For CHATting. As mentioned, you have the option to engage chat mode with the F7 key while a user is on-line. Alternately, a user may page you for a chat by using this command. If you disable the chat active status with C= C in the local modes, then the user will be notified that you will be unavailable for chatting. If you do have the chat command enabled, you will be paged. Paging involves the chat page sound call. This is a series of "chiming" sounds, looped 10 times to give you enough time to either hear that the user wants to chat, or let the user know you are not around to respond. The sound parameters for the chat pager were defined with CREATE.CNF, and can be modified with the aid of EDIT.CHAT. During a chat page, the user on remote will see the numbers from 1 to 10 appear in a self-deleting sequence, appearing something like : ....1 (deletes over this) ....2 (deletes over this) ....etc., until the number 10 is reached. The duration of the pager is determined by the the gate delays. See CREATE.CNF for how to set gate delays. During this time, the user can abort the pager by entering "A" to abort. The deleting number sequence uses Xon/Xoff, so that you can control it with S, C, and A like a text file listing. If you fail to respond to the chat pager, the user will be notified that you will not answer, and be placed at the previous command prompt. During the pager sequence, you can engage chat mode with the F7 key as before. Note that the chat paging sound and the keyclick sound may interfere with each other. Therefore, you may notice a sound distortion if you enter the F7 key in the middle of the paging sound.

While in chat mode, you and the user will be isolated from the normal BBS operating process, so feel free to type to each other whatever the keyboard allows. Note that Control-W, the "write user record" command, is still active, as is local mode 2. Therefore, you can go into local mode, modify the user on-line, and use CTRL-W to update his record, all while the user is chatting with you.

The F2 user defined modem key is active here, as it is while waiting for a call. See the TECHNICAL NOTES for information on programming this function key. F6 will toggle the chat mode between color and ASCII. This key is designed for use with the F8 key, which toggles between full and half duplex. While in half duplex, you will not be able to see what you are typing. This is the duplex setting terminal programs use to log onto a BBS. This key was provided to enable you to change over to terminal mode duplex (half), and then manually dial out to other BBS systems. There is no autodial feature for this mode. You will have to dial out by voice, then throw the data switch when carriers are locked. If you are using a 1670, then you can directly enter the AT commands. Remember that while in color mode, the AT commands should be sent as lower case, and while in ASCII, the AT commands must be sent as upper case. For example, log on locally with F5 or F7, go into chat mode with F7, use F8 to set the duplex to half, then set the state of this "terminal" mode to color or ascii with F6. You may now dial out to any BBS (Note: this BBS operates under 1 Stop Bit, No parity, and 8-bit wordlength. This is the usual standard for Commodore telecommunications. There is no provision to change these values). Note that carrier detect is disabled while in CHAT mode. So, it is possible to "Go Voice" with the remote user without the BBS resetting with the carrier loss. If you exit chat mode while in half duplex, modem suppression will be enabled to prevent the BBS from sending out any text data to a BBS you may be connected to. You must set the color status with F6 to match the version of the BBS you are using (ie. use F6 to set for color if running the CBBS before using F7 to exit). No full terminal mode is provided on this BBS, for lack of memory. That's why DarkTerm 4.0 is there. Use IT if you need a full terminal mode.

#BAUD: Change Baud Rate. This command will not work when the user is connected at 1200 baud. It is not possible to make a smartmodem switch baud rates while a carrier is present. This command is for a 300 baud connection. The baud rate factor can be any value from 250 to 600 baud. The baud rates may not go as high for certain modem types, and may cause garbled data transmission if the phone line is particularly noisy. This is one of several commands on the system that can use appended numerical references. For example, entering BAUD by itself will give the promot (200-600):, at which point the user will have to enter the new baud rate. You can do this is one step by entering BAUD450 to set the baud rate to 450. For all system commands that allow numerical references, the "#" symbol will appear next to the command. The actions of the BAUD command behave in a different manner than any other system command. When a remote user changes baud rate, he will need to access his terminal program in order to set the baud rate on his end. For this reason, the BBS will not respond to the BAUD change until after TWO RETURN characters are entered after the new BAUD rate. For example, the user will enter BAUD500, then RETURN. The BBS will then wait for a second RETURN, then will change the baud rate over. During this time, the user can change his baud rate for his terminal program. After hitting the second RETURN, there will be a 2 second delay. This is used by the BBS to adjust the BAUD rate to match the user's without any garbage characters appearing. Sometimes, a user may enter a baud rate too high for his modem to handle. In this case it may be necessary for the user to disconnect from the system, since the BBS will be unable to accept the user's garbled input.

#VOTE: If you enable the voting section in CREATE.CNF and make a vote file (+vo) with CREATE.VOT, then you will be able to use this command. The voting section works as follows. VOTE by itself, without any number, will display the list of voting topics, which is stored on a SEQ text file called "+vt.00". This file should act like a bulletin description file, giving the vote topic descriptions, and their respective numeric values. For example, you want to have these 3 vote topics:

1. What modem type do you use?
2. Do you think the 1581 is a good disk drive?
3. Do you agree that the ST is a better machine than the Amiga?

The responses to these 2 questions can have more values than a simple YES/NO/NOT SURE type result. For this reason, the voting section allows a response number, 0 to 9, to be used as the valid responses. Thus, you could use this in the 3 questions as follows:

What modem type do you use?

0. 1650 1. 1670 2. Total Tel 3. Mitey Mo 4. Westridge 5. 6420 6. 6470 7. Hayes 8. Master Modem 9. Other

Do you think the 1581 is a good disk drive?

1. Yes 2. No 3. Not Sure 4. Don't Care

Do you agree that the ST is a better machine than the Amiga?

0. Agree 1. Disagree 2. Abstain

As you can see, you can use the voting results in many ways, and you do NOT have to use all the numbers in a voting ballot. In fact, you can use this section for multiple choice quiz questions like:

What is the capital of Ontario?

1. Toronto 2. Montreal 3. Ottawa 4. Togoland 5. All Of The Above

As you can see, designing the voting topics and the outcomes takes a little imagination, but is quite easy to use. You will have to set the number of vote topics you want in CREATE.CNF, then use CREATE.VOT to make the "+vo" voting result table file, which is REL type. Set this to the MAXIMUM number of topics you feel you will ever need on your system. If you feel 30 topics is the most you'll ever use, then set CREATE.CNF for that many topics. If you only use 3 topics for starting out, yet you have the program set for 30, don't worry. You MUST have a vote topic data file created for each topic. Any attempt to vote on a topic number that you have as yet to create will not be allowed. To VOTE on a topic, you enter the VOTE command, followed by the number of the topic (if you define V as the vote command with CREATE.COM, then you would enter V followed by the topic number). Let's use the above example to make up the +vt.00 description file, so it looks like this when displayed:

Vote Topic	Enter
Modem Types	VOTE1
The 1581 As A Disk Drive	VOTE2
The Amiga Vs. The ST	VOTE23

Okay, so you see the VOTE23, instead of VOTE3. But we have the number of topics set to 30 maximum, so we can use any topic number from 1 to 30. If you wanted to, you can add or remove any topic at any time from any of the 30 positions, as they become outdated. What if a user enters VOTE without a number? Then the user will see the above vote description file. VOTE0 is not allowed. If we have a limit of 30 topics, then VOTE31 through to VOTE99 will not be allowed (Vote topics can go as high as 99). If a user tries VOTE4, then the BBS will search for the topic data file, which has the name "+vt.04" (see CREATE.VOT for setting file names). Since you have not created *one*, the BBS will return the user back to the previous command prompt. In essence, without topic 4 installed, the command was ignored. This is why you can jostle the voting section around at will. If you want to rearrange the vote topics, you will have to clear all vote results in the vote table file, as vote results are fixed into their own vote topic record. If you do need to move the vote topics around, clear ALL vote results by using CREATE.VOT to create a new VOTE topic file. Every time you use CREATE.VOT, all vote topic results are cleared. The V command in the editor section allows you to clear individual vote topic results.

Now, if the user enters "VOTE2", then the file "+vt.02" will be displayed. In this file, which is a normal sequential text file, like +vt.00, you must include the vote topic question (or whatever), and a description for each vote response you want. Thus, use the question/responses from the previous page in the +vt.02 file.

After a vote topic description file is displayed, the user will be asked to respond to the topic. The input can be a number from 0 to 9, which is of course a response to the vote topic, or the user can enter RETURN by itself. If RETURN is entered, the user will not vote, and will abort back to the previous command prompt. If the user does enter a number from 0 to 9, the results for that topic will be accessed in the result table. If the BBS finds that the user has previously voted, he will NOT be allowed to vote again. He will then either be given the voting results, or be returned to the previous command prompt, depending on the vote result flag status that you set in CREATE.CNF. If you set the flag so that users are allowed to see vote topic results, then the user will see results like this:

0:3
2:7
3:14
4:8

Note that the numbers in the left colum are the vote result numbers, which can range from 0 to 9. The numbers in the right column are the number of votes in favour of each of the vote responses (ie. vote response 4 has 8 votes in favour of it). Where is vote response 1, or responses 5-9? If a response has no votes in favour of it, then it will not be counted in displaying the results. So something like 1:0, meaning no votes for response 1, will not show up in the ballot results. There is no provision to stop a user from entering a response that is not part of the set you define for a topic. For example, using the Amiga Vs. ST topic on the previous page, a user could enter a response of "9", which is not part of the 3 possible responses you wanted. This is the only drawback. The user can use the full response set, even if you do not want him to.

After a user responds, provided he did not vote before, the BBS will notify him that his ballot has been cast, and then will either place the user at the previous command prompt (user vote result display in CREATE.CNF disabled), or give the current voting results for that topic, as is the above case with the user who already voted. Therefore, if a user wants to see a current vote result on any topic, he will have to attempt to cast a ballot by entering any response number from 0 to 9, even if the user has already voted before. This, of course, is only possible if you have enabled user vote result display in CREATE.CNF. Otherwise, all balloting results are kept for the sysop and the V command in the editor section.

To remove an old vote topic, you should delete the "+vt.XX" data file, update the "+vt.00" file, and clear the vote results for that topic with the V command in the editor section. The description for the V command will explain the last few points about the voting section. Note that the voting section requires a lot of disk space. The "+vo" result file takes up 1 block for every topic wanted. So 30 vote topics is 30 blocks. Also, you need the "+vt.00" file, and a "+vt.XX" file for each of the voting topics. So a 30 topic vote section will occupy a maximum of 30 files, one data file for each topic. Think of the voting section as a bulletin section with user responses. If you want a BBS with POLL messages, then you will have to settle for this voting section. This BBS does not support POLL messages.

CONTROL-Z. There is no TIME command in the BBS commands list to give the user the current time and connect time. CTRL-Z is accessible from any prompt on the BBS, where input is required. For example, you are in the the message section, in read mode. You want to know how much time you have used, and what time it is. Enter CTRL-Z and the user's connect time and current time will be dipslayed by "popping out" of the current prompt, waiting 2 seconds, then "popping into" the prompt, by deleting over itself. This is like unrolling the time by pulling it out from the current prompt, then pushing it back in. For example, the user is at the message command prompt "Command (M):". The user would enter CTRL-Z, and the time would roll out. When extended, it would appear as "Command (M):(34/12:45a). The (34/12:45a) part is the connect time/the current time, enclosed in brackets.

THE MESSAGE SECTION.

The message section is a fully integrated message base, with the ability to send public and private messages to any user, to all users, or to the sysop. All messages, public and private, share the same message base, which is directed through the message base header file, "+me". The capacity of this message base is defined in CREATE.CNF. The message base also uses reference numbers to allow a secondary way to access messages. Reference numbers appear in the message header next to the message number. These numbers "grow" upwards, and are always unique. Because they are unique, a message can be stored on disk with this reference number, and will never appear twice in the message base with the same name. As an example, 100 messages in a message base would be numbered from 1 to 100. Say there are some messages that have been deleted, like message numbers 12,13,23,56, and 72. Also say that the message base is at capacity. So now you have to recycle messages using the QUICK CYCLE program or the C command in the editor section. As a result, there are now 95 messages, 1 to 95. Message 13 becomes 12, 14 becomes 13, etc., as the empty spaces are filled. So you can see that the message number itself is not unique. Message 13 has now been assigned to message 12, and all the ones after that have been re-assigned too. But the reference number for message #13 will remain the same, even when changed to message #12. The same applies to all other messages after 13. The reference number stays as is. This is why reference numbers are always at least the same value, or larger than the message number. While messages are deleted, and their numbers shifted, the associated reference number will move with it's current value. As new messages are written, the reference number will grow larger. Eventually, after 65536 messages, the reference number will reset to zero. At this point, you will have to set all high reference numbers in the user records back to 0 with EDIT.USERS. But, even if you had 20 messages added to the system every day, it would take about 9 years for the message reference number to roll over.

Each message has 2 parts. The header or title of the message, is stored in the "+me" file as a record, one for each message, along with some reference values for each message. The body or text of the message is stored as a SEQuential file, one for each message. This is why you need to keep a careful watch on the number of files you can hold on disk before "disk full" errors occur (ie. a 1541 holds 144 files). So remember the system file overhead, the number of bulletins, vote topics, etc., when deciding the upper limit on how many messages your system can store. The text file exists with a file name of "+m.XXXXX", where XXXXX is the reference number of the message, padded with leading zeroes. For example, message #14 with reference #1843 would appear on disk as "+m.01843", and the header would appear in record number 14. Message text sizes can have a wide range. Usually the CBBS has larger text files in messages because of the potential allowed using full cursor and color control. It is very rare that you will see large message text files, although with the option of virtual message sending with the "S" command, a message text file could be as large as the capacity of the disk.

MESSAGE SECTION COMMANDS.

CAT: List Message Categories. If you enabled message categories in the configuration file, then this command will list all the categories you created with CREATE.CAT. If you set the number of categories to 0, then this command will be disabled from operation.

SCST: Set Scan State. The scan commands in the messages section will perform various summaries on the message headers. The normal state for each logon is "long state". This will give a full message header summary for each message scanned. In short scan state, each message will be summarized using various "one-liners" of text. SCST will toggle the state between long and short. The commands ST and SF in the message section require a short scan state for the global user scans (see ST and SF).

S: Send A Message. For any validated user, sending a message is the same, regardless of access. If a guest user is on-line (system mode 4), then that user will be sending his message to the "+va" validation file, where it will be appended to the current contents of that file. Guest users are not assigned an ID#, therefore, this command will not be allowed for the guest user. Instead, the S command sends a "validation" type message to the sysop in the +va file, consisting of the user's name, date the message was sent, and the text of the message itself. If you are running the CBBS, the color editor cannot be used by the guest user to send a validation message, since appending color files to the +va file will corrupt it's contents.

Note that at least 20 blocks must be free on the message drive for a message to be sent. The message drive was configured with CREATE.CNF to be any device and drive, so message text may be saved on a drive other than drive 8:0, the files disk drive. The message header file "+me" would be the only file stored on drive 8:0 that is used by the message section. There must also be 20 blocks free for every time a user wishes to continue a message using the continue after saved command in the text editors. This margin of 20 blocks ensures no disk full error will occur.

The first part of setting up a message is setting up the message header data. The user will be prompted for certain items of information, in the following order:

1. Editor Selection. This option is only provided in the CBBS. The ASCII version has only the one line editor, but the CBBS has 2 editors, the line editor, and the color editor. The user may use either one on the CBBS. If you are using the ASCII version, skip this and go on to item 2. If you set editor lockout in CREATE.CNF, then the user will be forced to use the color editor. Only a level "s" user may use both editors if you lock out the line editor from normal users. The idea of not allowing users to use a line-origneted text editor is based on the situation where the sysop may want a system full of color, without reduntant, boring, text only messsages. Using the line editor allows only printable characters to be used (including the left hand graphics symbols), but does not allow any other characters as input, while the color editor does.

2. Public Or Private. The user will be asked if the message is to be a private message, or a public message. Private messages are for the sendee's eyes only, whereas public messages are intended for everyone to read them. Responding NO to the prompt will make the message public, a YES response will make it private.

3. Send To. A message can be sent to a user, to all users, or to the sysop. To send a message to a user, you enter the name of the user, or the ID# of the user you want to send the message to. To send a message to all users, then use the word "ALL". You may only send to ALL, not ALL + SYSOP or ALL + the name of a user. The BBS does not allow you to append words to ALL. To send a message to the sysop, the user enters ID# 0, the sysop's ID number, or the name of the sysop. The default name is set to "SYSOP", usually the standard for the name of the system operator. However, if you change the name with CREATE.NAM, the user would have to send the message to the name you created. Using an invalid name, or the name of a user that isn't in the user list will result in an invalid user prompt being given, and the user will have to try again. Note that hitting RETURN at any of the message setup prompts will abort the message send command. A user can not send a message to ALL and have it be a private message at the same time. Sending a private message to ALL doesn't make sense.

4. Access Level. The access level prompt will be ignored by the BBS if access level mode was disabled in CREATE.CNF. In this case, the access level will be set to 1, so all may read the message. If the message was set as a private message, then the access level prompt will again be ignored by the BBS. All private mail has a preset level of 1, so that the user the mail is inteneded for can always read it, regardless of access level. If access levels are enabled, and the message is a public message, then the user will be prompted for an access level. Access levels can range from 1 to as high as the user's access level. If the user is at level 6, and tries to send a level 7 message, the BBS will not allow it. Once the level is set, the users who read it must have access levels equal to or higher than this set level to be able to read the message.

5. Category Number. If you have set categories to 0, then this prompt will be passed over, with the category number being preset to 0. With categories enabled, the user must enter a category number, from 0 to the maximum number of categories, minus 1 (ie. for 5 categories, the numbers allowed are 0 to 4). When the user enters the category number, it will be echoed after the prompt. Sometimes, one cannot remember what the category definitions are. For this reason, the user can enter "?" at the category prompt to get a list of categories, as if the CAT command were used. Then the user will be prompted once again for a category number.

6. Message Subject. A message subject can be any any string of characters from 1 to 25 characters in length. Hitting RETURN here will not give a null subject; instead, the send command will be aborted.

Message Header Display. Before going onto writing the text for the message, the user will see the message header previewed. The message header will generally have a format like:

```
Message : #CCC/MMM LX (RRRRR) : type 1
Message : #CCC/MMM P (RRRRR)  : type 2
Message : #CCC/MMM (RRRRR)    : type 3
```

This is the first line of a message header. Type 1 is what you would see for a message sent public, with access levels enabled. Type 2 is a private message, and would appear the same whether access levels are enabled or not. Type 3 is a public message, without access levels. The variables correspond as follows:

CCC: current message number. MMM: number of message in the message base. RRRRR: the reference number for the message. LX: X is the access level of the message (ie. L6 for level 6). For example:

```
Message : #43/87 L4 (1427). This is message #43 of 87, level 4, reference number 1427.
Message : #12/26 P (17). This is message #12 of 26, a private message with reference number 17.
Message : #127/176 (12671). This is message #127 of 176, a public message on a system without access levels,
                            and a reference number of 12671.
```

Note that there may also be a replied indicator in this first header line. If a message was replied to with the "R" reply command in read mode, the number of replies will appear at the top like this:

```
Message : #43/87 L4 (1427) R:3. This "R:3" means the message was replied to 3 times.
```

The rest of the message header appears like this:

```
Category : (category text)
Sent By  : (user name)
Sent To  : (ALL, or a user name)
Sent On  : (date and time)
Subject  : (message subject)
```

There are also 2 optional header additions after this. If a message was forwarded with the F command in read mode, then the line : Forward : (user name) will be added, the user name being the name of whoever forwarded the message. Also, if the reply command is used, the message that was replied to will appear like this :
Reply To : (reference number of the message replied to). Note that the category line will NOT appear if message categories are set to 0 in CREATE.CNF. Now for some examples:

```
Message  : #43/87 L4 (1427) R:1      Message  : #88/88 L1 (1501)
Category : GENERAL                   Category : GENERAL
Sent By  : 0-SYSOP                   Sent By  : 43-JOHN DOE
Sent To  : ALL                       Sent To  : 0-SYSOP
Sent On  : 23/09/86-12:34a           Sent On  : 30/09/86-01:23p
Subject  : System News               Subject  : What News?
                                     Reply To : (1427)

Message  : #12/23 P (19)             Message  : #178/188 (63245)
Sent By  : 242-JOHN SMITH            Sent By  : 56-JIM BROWN
Sent To  : 43-JOHN DOE               Sent To  : ALL
Sent On  : 15/09/86-04:34p           Sent On  : 25/12/86-00:00a
Subject  : This Is A Test            Subject  : Nothing
                                     Forward  : 43-JOHN DOE
```

Note that the ID# of the user is always placed before the name of the user. Message headers will always appear different on different system setups.

Writing Text. BBS V3.0 is "user-friendly", in the sense that message writing time is free. When a user enters the either text editor to write a message, his connect time will stop, and will not restart until the message has been sent. Message writing is contributory to the BBS, and we felt that stopping a user's connect time would be a beneficial idea, both for the user, and for the BBS. You might think a user can stay in the message editor for ever, since connect time will never reach the user's time limit, This is not the case, since the 2 minute idle delay timer is still running, so that a user will still be logged off the system after 2 minutes of idle time, even in the message editors.

The color editor operates EXACTLY like the editor in EDIT.COLOR. Look back in the manual for help in composing color messages. Once you have a color message done, use CTRL-C or STOP to exit the edit mode. The REVIEW option is the same as that in EDIT.COLOR, so no information on that will be repeated here. Remember that reviewing all the text will place you back into edit mode, so that you can resume where you left off. Also, remember that the review option in this editor uses the Xon/Xoff control of the BBS to control the output. One important note of difference. Every time you enter the color editor, the current screen colors are placed at the start of the edit buffer. So, if you have a red border, white text, and black background, the characters : (F1)(RED)(F3)(BLACK)(F5)(WHITE) will be put at the start of the edit buffer. This insures a color continuity, since the screen colors of the reading commands may not match those of the message send command. For example, the "S" command has the above 3 colors. The R command for reading messages has red border, white background, and black text. If the user FORGETS to enter the screen colors in the edit buffer at the start of his message, the result might be white text on white background. It is always important to set the screen colors at the start of any message. The BBS does this automatically, the EDIT.COLOR program does not. BUT you may want to not use the default colors, nor have them in the buffer at the start. So, when you continue a color message after saving, or start writing a new one, use CONTROL-D SIX times to erase the screen colors from the start of the buffer.

The capacity of the color editor is 4096 bytes, or about 16 blocks of text. This is a far cry from the 45K allowed with the EDIT.COLOR program. So, to allow larger messages, you can continue adding text after saving the current buffer. Continue after saved is allowed at any time, even if there are only 10 bytes in the buffer. The reason why it doesn't appear only when the buffer is full may not be apparent at first. When writing a color message, you make a picture using the Commodore graphics primitives. The cursor is at row 10, column 10, and the buffer is full! No more room to complete the picture. Well, you can continue after saving, but after doing this, you are given a clear screen, with the cursor at position row 1, column 1. How do you get the cursor back to row 10, column 10, where you left off at? How do you get the picture back you saved? The BBS could review the entire previous buffer, but it doesn't. In fact, there is no way to resynchronize the screen to get the old picture back to finish it off. This is why you should write color messages 1 screen at a time, and always complete a screen before you know the buffer is going to empty. It's better to save messages of length less than 4K, and do it a few times, rather than break up a screen when the buffer becomes full at the wrong time.

The line editor does not support right justification. Something like that does not appear very appealing when you have only 40 columns of text, and do not have proportional spacing. So, no right justification. The line editor has a capacity of 100 lines, with the option to add more text after 50 or more lines have already been entered. The line editor will sound the bell tone every time the 35th column on each line is reached. This is similair to the warning bell on a typewriter when you get to the end of a line.

Enter RETURN by itself to exit line entry mode. At this point you will be asked for an editing command. Enter "?" at the editor command prompt to get a brief summary of the available commands, which are described in detail here as follows:

#R: Read Message. This will display your message, as it will appear in it's final format when saved to disk. The "#" symbol indicates you can use a numerical reference (ie. R34 will read all text starting at line 34).

#L: List Message. This is just like R except that the line numbers are given for each line of text. Once again, you can use numerical references.

S: Save Message. This will save the current buffer. If you have entered 50 or more lines, or the buffer is full, you may add more text after the current buffer is saved.

A: Abort Message Editing. This will abort any command that requires use of the editor command. For the S command in the message section, aborting here aborts the S command as well. If this is a continued buffer, where a previous buffer has already been saved, then aborting here will only abort the current buffer. The previous buffer will have already been stored, and in the case of a message, the message will have been sent.

#C: Continue Editing. You may resume at the line you left off at by entering C by itself. If you add a number to C, then you will edit INTO the current contents of the edit buffer. For example, you have 56 lines of text in the edit buffer. Entering C alone would place you at line 56 (line numbers run 0-99, not 1-100). Typing C34 would place you at line 34. There is already text at line 34, as there is at lines 35-55 as well. When you write in text now, you will overwrite the current line with the new line. Thus this option lets you edit a range of text at one time. Enter RETURN by itself at the line you want to stop editing at. Entering return without changing any lines will abort the command.

#E: Edit A Line. Entering E by itself will force the BBS to ask for a line number to edit. Entering E34 would edit line 34, and would skip the prompt asking for a line number. When you select the line to edit, the old line will be displayed, and then you will be prompted to enter the new line. Entering RETURN alone will abort the command, and leave the line unchanged.

#D: Delete A Line. Works like E, but removes the line specified.

#I: Insert A Line. You cannot insert a line if the buffer is full. Entering I by itself will ask for a line number. Entering I34 would insert a blank line at line #34, and display line #33, and the line after that, which would be the old line #34, but which is now line #35. Then you will be able to enter the new line #34. Entering RETURN alone does NOT abort insert. Instead, a blank line will be left at that line position. You will have to use D to remove it.

Note that during the listing (L) and reading (R) commands, the listing may be controlled with the BBS Xon/Xoff keys.

When you send a message to a user, the "messages sent to" counter in that user's record increases by one. This also applies if you reply to a message that was written by that particular user. Also, the "messages written" counter will increase by 1 in the record of the user who wrote the message.

#DM. Delete A Message. A user may delete a message if the message was written by him, to him, or the user is a level "s" user. The BBS will ask for a message number if DM is used by itself. DM34 would delete message #34 without the BBS asking for a message number. The level "s" user can delete any message on the system.

#NEW. Read New Messages. With the reference number system, it is very fast and easy for the BBS to locate all new messages for any user. The hi reference message number in the user record is updated "on-the-fly", so that the BBS always knows where new messages begin at. For example, if the user types NEW, and starts at message #12, reads through to message #18, then aborts reading, he may still use NEW again during the same logon period, and resume at message number 19. The only commands that update the high reference number are the NEW command and the R command for general read mode. The other read commands, like MAIL and MINE, do not change the high reference number, so it is possible for a user to read his MAIL first, without disturbing the high reference number and any new messages. At logoff time, the high reference number of the last message the user read with NEW or R will be updated in the user's record. The "$" next to NEW above indicates that this is a continuous mode command. Continuous mode commands are those that allow you to optionally use continuous reading, as opposed to using the read mode command prompt after every message. Note that because of this, you are not able to use the read mode commands in the message section any more. Reading is done on a straight beginning to end basis, without any stopping. The continuous mode is entered by adding a "+" or "-" to the end of any command that allows continuous reading. For example, NEW by itself will not read continuously, and will place you at the read mode command prompt after each message is read. However, NEW+ will read all new messages continuously. During the reading of messages, use the Xon/Xoff list control keys to pause or abort output. Entering the abort key "A" (default value) will place you at the read mode command prompt if in normal reading mode, or will abort to the next message if in continuous reading mode. So, another abort key must be used to abort continuous reading entirely. This is the "Q" (default) key, and it will place you back at the message command prompt. The "+" is used to read on an old message to new message basis. The "-" key is used to read on a new message to old message basis. Thus, NEW- will read all old messages, continuously. Read mode commands are explained in the next command, R.

R#: General Read Mode. Read mode is termed for the way in which the message section operates. There are five separate reading commands, each one reading through the message base under different modes of message selection. This command is marked with a "#" and the "#" symbol because you can use numerical references and/or continuous reading. If you select R without any number reference, the BBS will scan from the newest to the oldest messages. As a general example for use in describing this command, we'll assume that there are 60 messages in your message base. When you enter R, the BBS will start at message 60, and look for the first readable message, going from #60 to #1. A message is not readable if:

-the access level is beyond that of the user who tries to read it, provided that access level mode is enabled.
-the message has been deleted from the system.
-the message is private, not written to or by the user who tries to read it.

A level "s" user may read any message, provided it's not deleted. Any attempt to read a message that's unreadable will be ignored without notice (ie. there is no prompt "Private Message" when one tries to read a private message). If, for example, all 60 messages are unreadable, no messages will be read. If you enter a command like R-, All messages from 60 to 1 will be read contiuously. Entering R45 will read message #45 directly, unless message #45 is unreadable, in which case the next lowest message will be searched for reading. The command R+ will read all messages continuously, reading from the first message. R34+ will read all messages from 34 onwards, continuously. When you do not select continuous mode, you will be given the read mode command prompt, which appears as (CCC/MMM:?). The CCC is the current message number, the MMM is the number of messages in the message base. The "?" is included in the prompt to remind the user that the commands summary for the read mode is available by entering "?". The following read mode commands are supported:

R: Reply To Current Message. Replying to a message is almost the same as sending a message. In this case, the name of the user to send the message to is already known, so you will not have to enter the name of the user to send the reply to. The "Reply To : " prompt will appear under the message header, followed by the reference number of the message you replied to with R. As well, the replied counter in this message replied to will increase by 1, and this will be indicated in this message's header. Other than that, follow the same guidelines for writing messages as you do with the S command. When done replying, you will be placed back at the read mode prompt. You will remain in the confines of read mode during the entire reply procedure.

F: Forward A Message. Cases may arise where a message is written to a user as a private message, and when the user reads it, he finds that the message is so good that he wants everyone to be able to read it. Forwarding a message involves re-directing the message to another user, or to ALL provided that the message was written to the user who uses the F command. For example, the user has a message like this:

```
Message : #43/87 P (1234)
Sent By : 0-SYSOP
Sent To : 43-JOHN DOE
Sent On : etc.....
```

User #43 wants everyone to read this message, but it was written privately to him, for his eyes only. When the F command is entered, the "Sent To :" line will change, and the line "Forward : 43-JOHN DOE" will be added to the message header, indicating that user #43 forwarded a message that was written to him originally. When using forward, the user is asked for the name of the person to forward the message to. As with S, the user may forward the message as a public message to ALL or a user, or a private message to a user. The access level may be changed if the message is public and access level mode is enabled. The rest of the header information will not be changed. The subject, sender, category, and message text remain unchanged. The forwarded message will have a new reference number, and be added to the end of the message base, and will have the current date marked in it. The message will move from the old position to the end of the message base, thus deleting the message from the old spot. The user will be shown the new message header, and asked if it should be forwarded. If the user responds with NO, then the forward command will be aborted. The message text file will be renamed on disk, to match the new reference number, and the message header record will be moved to it's new position.

Note that a level "s" user can forward ANY message. A message addressed to ALL can even be forwarded elsewhere by a level "s" user. A message directed privately to another user can also be forwarded by the level "s" user. Note that after the forwarding process is complete, the messages will be scanned in the set direction until another readable messsage is found.

R#: Read a message by reference number. A message can be read directly by entering R followed by the reference number of the message (this is not the same as R by itself, which is the reply command). For example, R6274 will read the message with reference number 6274. If the message is unreadable, the command will be ignored.

XXX, Where XXX Is Any Number. Entering a number by itself will cause the BBS to attempt to read the message of that number. For example, to read message number 56, a user enters "56". If the message is unreadable, the command will be ignored.

+: Continue Reading Forward. Entering "+" by itself will allow you to read the next message AFTER the current message. The BBS will search all messages after the current one, until a readable message is found. If none is found, then the BBS will return the user to the message command prompt.

-: Continue Reading Reverse. The "-" command is the reverse of the "+" command above, and will search all messages BEFORE the current message for one that is accessible.

C: Read Current Message. This is in case you want to read or re-read the current message you are positioned at in read mode. This will be necessary if you delete a message.

D: Delete Current Message. This is the same as the DM command, except that D will delete the message you are currently positioned at. The message cannot be deleted if it's not addressed to you or was not written by you, unless you are a level "s" user, in which case anything can be deleted. After a message is deleted, the next available message will be searched for, at which time you are placed at the read mode prompt again. You should use "C" to read the message at this point, as it usually will be a new message.

RETURN will abort read mode, and place you back at the message command prompt. The R command and NEW are the only two that will keep track and set the user's high reference number as new messages are read.

*#RCAT: Read Mode (Category Mode). RCAT will only read messages in the selected message category. The numeric reference is the category number (ie. RCAT1 reads messages which have category #1). The BBS will prompt the user for a category number if RCAT is used alone. While reading messages, the user operates read mode exactly the same way as for the R command. But this time, messages are unreadable if the category number is not that of the one selected. These messages will be ignored. Note that if categories are disabled with CREATE.CNF, then this command is disabled from operation on the BBS. The "+" and "-" continuous mode characters can be added to the command for continuous category reading. RCAT2+ would read all messages, low to high, with category number 2.

*RALL: Read Mode (Messages To ALL Mode). RALL reads only those messages addressed to ALL. RALL+ will read messages to ALL from low to high, while RALL- reads messages to ALL from high to low, continuously.

*MINE: Read Mode (Messages By You). This command will search for and read all messages that were written by you. You may delete any of these messages, since they were written by you. If "+" or "-" are added to MINE, the continuous mode will be used.

*MAIL: Read Mode (Messages To You). This is the command to use for reading messages anyone sent to you. These messages may be either public or private. Mail is not self-deleting. You will have to delete all mail to you, or it will remain on the system until someone else deletes it, like the SYSOP. This is where the forward command may be used to redirect the message to a user or to ALL. After each letter is read, you are given the read mode command prompt again. You must use "+" to go through all your mail. The mail is not read automatically for you, with the exception of the first message. You may however, use MAIL+ to read all mail continuously, but you will not be able to use the read mode commands now, and therefore be unable to reply. MAIL- does not have any effect on the BBS.

Remember that MAIL, MINE, RCAT, and RALL do not affect the high message reference number. Only R and NEW will do that.

#SC: Scan Messages. This is the general scan command. Without using a number, all messages from new to old will be scanned, giving a full header if scan state is long, or just the message number, reference number, and message subject if short scan is used (see SCST). Use the Xon/Xoff characters to control output. Using a number like SC45 will cause the SC command to begin scanning at message #45, scanning all messages from #45 to #1. Any message that is unreadable will not appear during the scan.

#SCAT: Scan Messages By Category. Only messages with the selected category number will be scanned. The short/long scan state give the same results as SC above. The number reference is the category number, and operates in the same fashion as RCAT with respect to category selection. This command is not active on the BBS if the message categories were disabled with CREATE.CNF.

SALL: Scan Messages To All. This scans messages written to all, and gives the same information as the above 2 commands.

SF: Scan Messages From A User. This command will ask for a user name or ID#. Enter the name or ID# of the user whose messages you want to scan. All messages found that were written by that user will be scanned for, giving the same results as the above 3 commands. This command also allows a global user scan. Enter RETURN by itself when the BBS asks for a specfic user name / ID#. The scan will now give the message number, reference number, and the names of the users each message was written by, provided that the scan state is set to short (see SCST). If the scan state is set for long display, then this command will operate exactly like the SC command above.

ST: Scan Messages To A User. This works just like the ST command, except that the users listed are the those who the messages were written to, not by. Entering the name or ID# of a user, when the BBS asks for it, will give the same results as all the above commands, either a full header, or the message #, ref# and subject. Using the global user option by entering RETURN by itself will give the names of the users the message was addressed to. If the message was addressed to ALL, then the "*" symbol will appear instead of the user's name. Remember that unreadable messages are skipped by all the scan commands.

THE FILE SECTION.

The file section was explained in detail already, so no reference will be made here as to how it works. This BBS uses new PUNTER protocol, and a unique multi-file transfer system. Xmodem was not supported because of lack of memory, and we needed to make the ABBS and CBBS as compatible with each other as possible. There was no need to put Xmodem in the CBBS, as it will not work with any computer other than a C64. During file transfers, a carrier loss will result in the partial file being saved to disk, and the user being logged off. File transfers can be aborted from the keyboard at any time by pressing the STOP key. Remember certain points about the file section:

1. There can be from 1 to 8 directories for both physical directory and relative directory storage formats, numbered from 0 to 7. If only one directory exists for either type of storage, then the file section will not need number references. For example, if there is only one relative directory, LIST will list that directory, UPL, DOW, and DEL will upload, download, and delete files from that directory. LIST1, UPL1, DOW1, and DEL1 are not necessary.

2. Unless you are basing the entire file section on multi-file transfer, exclusively using the MD and MU commands, then you will have to make all file names in lower case format, which is the usual state of file names anyways.

3. Files with the "+" character as the first character require access in the user's record for those files to be listed or downloaded.

4. File transfers will not abort in the middle of the transfer if the user's time limit expires. Only after that file transfer will the user be logged off the system.

5. If there are 0 relative directories, the "+pr" file may be scratched, and the LIST, DOW, UPL, and DEL commands will no longer be active on the BBS. If there are 0 physical directories, then the C, D, U, MD, and MU commands will no longer be functional.

6. It does not matter what file transfer command or format is used to acknowledge a change in the upload and download status of the BBS (see UD command).

7. The file section may be locked out from a user, either through the user record, by shutting down the file section with the local mode command C= F, or by setting both physical and relative directory limits to 0.

8. You should be aware of how Punter Protocol works to understand the transferring of files on this system. Information on this can be found most anyplace. The file transfer system on this BBS uses on-screen buffering, so that all data sent or received can be witnessed by you as the transfer proceeds. Note that the block size can range from 40 to 255. This is only important on downloads. The block size is the number of data bytes to send for each data block. The smaller the block size, the slower the transfer, and the more data blocks to send. A 255 byte data block size will send 247 bytes of data from the disk at a time, which is 7 bytes short of a full sector on disk.

9. The BBS will acknowledge whether the data is being sent from BBS to USER or USER to BBS during every file transfer. The data block rate at block size 255 is about 6 blocks per minute at 300 baud, or 21 blocks per minute at 1200 baud.

UD: Upload/Download Accesses. For every file transfer, or multiple file transfer, the upload and download counters in the +sf system file are updated. The upload and download counters will increase by one for single file transfer commands, or by multiple amounts for multiple file transfers. The counters reflect the actual number of programs uploaded or downloaded.

#LIST: List A Relative Directory. If you use relative directories, then LIST will give a listing of each program stored on the selected directory. LIST1 would list directory 1, while LIST by itself would cause the BBS to prompt you for a directory number. The LIST command and the C command both share common directory header descriptions. The default file BBS.TXT sets these to simple "Directory : X" prompts, but can be changed to any text description you want. This allows you to categorize your directories in much the same way as messages are done in the message section. An example file entry would appear like this:

```
(034) #TEST FILE        P-123 A:002 000.
(035)  PROGRAM 1        S-043 A:000 043.
```

In the 2 file examples, the first number in brackets is the file reference number. As mentioned before, the relative directories can use reference numbers for downloading as well as the file name. See the DOW command for how to download using the reference number. The next item is the file password indicator, which is the "$" symbol. If a password exists on the file, then that symbol appears next to the file name. If no password is placed on the file, it will appear without the symbol, as the second example file shows. The file name follows this, then the file type, which is P (program), S (sequential), or U (user). The file size is next, 3 digits. The A:### is the system accesses indicator, which increases once for each download. The last number (0 and 43 in the above 2 examples) is the ID# of the user who uploaded the program, which is entered manually, or automatically when a user uses the UPL command.

#DOW: Download A Program. You may download a file by either using the file name, or the file reference number. The reference number was given when listing the directory. When the BBS prompts you for a file name, enter the ":" character, followed by the reference number. For the LIST example, the response ":35" would download the file PROGRAM 1. Using reference numbers gives very fast downloads, since the BBS does not have to search for a file name. A file password may be required for programs that use it.

#UPL: Upload A Program. Uploads are time credited on this BBS. Uploads are considered of great value to a BBS, so it would be nice to provide incentive to the user to contribute uploads. When the UPL, U, or MU commands are used in the file section, connect time stops in the same manner as it does in the text editors in the message section. The connect time will resume after the file transfer is completed. The 3 uploading commands also require 50 blocks of disk space minimum for uploading, or the uploads will not be allowed. It may occur that a 120 block program could be uploaded when only 80 blocks remain on disk. Nevertheless, the upload will take place, but the BBS will automatically abort the file transfer when a disk full error occurs. Thus only 80 blocks will be sent, and the file will be corrupt. A reference number will not be allowed as an upload (makes sense). Files cannot be uploaded if they exist already (see MU for the exception). Files uploaded preceded with a "+" will be under the "+" file protection scheme of the file section. Illegal characters will not be allowed by any of the three upload commands. The UPL command only checks the disk itself to see if the program exists. This is done to avoid searching the "+pr" relative file. This means that the file could be duplicated on the relative directory if you forget to place the corresponding file on the disk. When an upload is completed, the BBS will ask the user for a security password. If the carrier was lost, the system will reset without one being set, and the program will be stored "outside" of the relative file "+pr". The user will have 1 minute to enter a security password. After this time, or if the user enters RETURN by himself prior to this, the BBS will not place a password on the file. The password can be anything from 1 to 8 characters. Once set, the file will be updated to the relative directory, placed at the END of the directory. This BBS orders the relative directory from old to new, not new to old.

#DEL. Delete A Program. A user may delete any program if the sender ID# of that program is his ID#, or if the user is a level "s" user. Note that both the relative directory entry AND the program itself will be deleted. You may use reference numbers in the same way as the DOW command does.

#C. Catalog A Disk. This gives a physical disk directory of all files on the drive specified. If a file has a "+" as the first character, and the user does not have access to "+" files, then the BBS will respond with "**NO ACCESS!", after which this prompt will delete itself into the left edge of the screen. These file names will be kept hidden from the user. The directory header will be given first, corresponding to the text descriptions placed in the BBS.TXT or your own custom text file. After this, either the files themselves, or the disk header will appear. If you enabled disk header display in CREATE.CNF, then the disk name and disk id will follow, then the file names. The file display is a slightly different one from the one you get using LOAD"$",8:LIST. The file type, size, and name are given, in that order. The blocks free follow at the end. Standard Xon/Xoff list control applies to both DOW and C.

#D. Download A File. Just specify a file name, and let the BBS do the rest. If a user specifies a "+" type file, and doesn't have access, then the BBS will respond with "file not found".

#U. Upload A File. This works just like UPL, but will upload directly to the disk. A user may upload a file with a "+" as the first character, regardless of whether that user has "+" file access or not.

#MU: Multi-File Upload. Multiple file transfers use a unique protocol that was developed for use with the DarkTerm series of terminal programs. As such, other terminal programs that offer multiple file transfers will not be compatible with the BBS. Attempts to use a different type will be ignored and rejected by the BBS as non-standard. This method for multiple transfer is combined with the Punter protocol to give a superb, error free method to send as many as 144 files for BBS-Terminal transfers, and 256 files for Terminal-Terminal transfers at one issue of the transferring commands. The PAL source code for the multi-transfer routine may be obtained, along with the source code for the complete color terminal protocol used in DarkTerm and the BBS, can be provided on disk by sending $15 (US) to the address on the warranty page. You will be allowed to use both the color terminal protocol and file transfer protocol for your own programming purposes. You may also give away copies of the the source codes for anyone to use as they see fit. Only DarkTerm series 3 and 4 terminal programs use this multi-transfer method. This new version will send, for each file transferred, the current file name, size, type, files left, and blocks left in the transfer, giving both the BBS and terminal end user enough information to determine the time it will take to complete the transfer. The difference between single file transfers and multiple file transfers can be noticed on the BBS by the additional line "F:XXX B:XXX", which is the files left and blocks left indicator. Note that only the lower 3 digits are displayed during file transfers. The fourth digit, which occurs on files of size 1000 blocks or more, will be truncated from the display.

You should be cautious in assigning MU command access. MU will save and replace any file on the upload drive if it matches any file that is currently being uploaded. So, if a user was given MU access to drive 8:0, a user could send dummy files like +sf, +me, +pr, etc., and crash the system by replacing the system files with garbage files. The MU command will not allow bad data strings during multi-file transfer, such as bad file name characters, bad file size characters, etc., and attempting to do so will result in the command being aborted by the BBS. Every time a new file is to be uploaded, the BBS will make sure at least 50 blocks are free, so that the disk will not get full during the middle of a multi-file upload. If the disk space does run low with more files to go, the BBS will abort the MU command. After all files have been uploaded, the user's upload counter and system upload counter will increase by the number of files sent. MU is a time credited command, like UPL and U are. There is no limit on the amount of files to be uploaded, so a user could conceivably send more files than the directory can hold. As you can see, the MU command can do as much harm as it can good, but will tend to be very beneficial when it's access is given to the outstanding users on your system.

#MD: Multiple File Download. This command will not corrupt the disk that the programs are stored on. Since programs are read, and not written, feel free to assign MD access to any and every user. Only the first 144 files on a disk can be downloaded with MD. The MD command loads the contents of the directory into memory prior to the file transfer. The BBS only has the capacity to hold 144 files in memory at once, which is the capacity of a 1541. So, for an SFD drive with 224 files on disk, the last 80 files can never be accessed with the MD command. After the directory is placed in memory, the user will be allowed to respond YES, NO, ALL, or DONE for each file. YES and NO are obvious. ALL will select all REMAINING files in the list for download. DONE will stop selection at the current file, and allow the user to download all files selected up to this point. Note that the user may only download as many files as his time limit allows. After each file is downloaded, the BBS checks to see if the user's time limit has expired. If this is the case, the BBS will log the user off without completing the rest of the file transfer. The BBS will send the end of multiple file transfer signal ahead of time to notify the user that the remaining files could not be sent. If a user does not have access to "+" files, then any program found in the directory with a "+" as the first character will NOT be loaded into memory with the rest of the files.

Note that for all the physical and relative directory commands, a user may not have access to a particular directory, depending on the physical and relative directory access values you assigned to the user (which are seen on the bottom status line). If the user is unable to access a directory, he will be notified that access is restricted to the directory.

For those who tend to swap disks while the BBS is running, take note that it can be done with this program too, with the exception of drive 8:0.

THE BULLETIN SECTION.

The principles of both relative and primate text mode bulletin storage were explained in complete detail in the sections EDIT.BULLETINS and CREATE.CNF. Only the commands will be summarized below.

L: List Bulletins. If the bulletin section is running in text mode, this command will display the "+bt" text file, which contains the descriptions and names for all main bulletins. In relative mode, L will list all main bulletins with a primary (first) link number of 0. Use the standard Xon/Xoff list control characters to control output. When listing in relative mode, the reference number of each bulletin is placed next to the bulletin description. A typical listing would be like:

```
REF# Description              Enter
---- ------------------------ --------
001  Modems And Maniacs       MODEMS
002  System Help              HELP
010  Jokes And Humor          HAHA
```

Note that the post command requires you to specify a link number to sub-bulletins. Say that the above example HELP bulletin is a main bulletin linked to 6 sub-bulletins that describe procedures on various operations on the BBS. By listing alone, you do not know the second link number, the one that links the main bulletin to one or more sub-bulletins. Therefore, you can enter L followed by any digit from 0 to 9 (ie. L5) to list the link numbers for each of the main bulletins. The listing above would appear something like this:

```
REF# Description              Enter
---- ------------------------ --------
L000 Modems And Maniacs       MODEMS
L001 System Help              HELP
L002 Jokes And Humor          HAHA
```

The REF# field is now replaced by L and then the link number to the sub-bulletins. In this case, the HELP bulletin uses a second link number of 1. When posting a sub-bulletin in the HELP sub-section, entering 1 for the second link number would place the sub-bulletin in it's proper sub-section. All main bulletins are listable, yet they may not all be readable, depending on their individual access levels.

R: Read Bulletins. When you enter R to read bulletins, you will be placed in a loop. After reading each bulletin, you will always be placed back at the read prompt. Hit RETURN to abort to the bulletin command prompt. In bulletin text mode, just enter the name of any bulletin that is on disk, whether it is a main bulletin, or a sub-bulletin. The bulletin section is only a sequential file reader in this case, so the bulletins reader will be able to read any sequential file stored in bulletin name format.

If you are using relative mode, you may specify a reference number instead of a name. When the BBS asks for a bulletin name, enter the name, OR enter ":" followed by the reference number seen with the L command. In the above example, ":1" would read the system help main bulletin. If the bulletin read is a main bulletin linked to a series of sub-bulletins, you will be given a sub-listing of the sub-bulletins, which will look exactly the same as the main bulletin listing using the L command. Using the above example, reading the HELP bulletin might yield the following sub-bulletin listing:

```
REF# Description              Enter
---- ------------------------ --------
003  Help On Writing Messages HELP0
004  Help With File Transfers HELP1
005  Writing A Color Message  HELP3
```

Then, to read the sub-bulletin HELP0, the user enters "HELP0", or ":3" to read the text file for that bulletin, provided the access level for that bulletin AND the access level of the main bulletin HELP are equal to or below the user's access level.

A sub-bulletin that is hidden can be read in text mode by just guessing the file name. A sub-bulletin that is hidden while in relative mode (links both 255) can be read only if the access level is at or below the user's, and the user can guess the name or reference number of the bulletin.

P: Post A Bulletin. Posting bulletins follows the 20 block free rule for text editing (see the S command in the message section). The procedure for posting a bulletin was outlined in EDIT.BULLETIN$, but there is a slight difference here. When posting a bulletin, you must first have access to posting bulletins, assigned to each user in their individual user records. If access is given, the user is asked whether the bulletin is to be a main bulletin or not. If not, then the bulletin will be a sub-bulletin. The BBS will then ask for the second link number. Here are some possible combinations:

Main Bulletin	Link #2	Result
Yes	0	The bulletin must have a text file, and is not linked to any sub-bulletins.
Yes	2	No text file, linked to all sub-bulletins that have link number values of 2.
No	2	Needs a text file, and is a sub-bulletin linked to the main bulletin with a link number 2 value of 2.
No	255	This is a hidden bulletin, one that cannot be linked to any main bulletin. A text file is, of course, required.

Remember that the bulletin name can only be alphanumeric characters, 1 to 8 characters long, with optional blanks spaced into the name. To check if the bulletin already exists, the BBS will only search the bulletin drive, not the "+bu" relative file. Therefore, it is possible to have bulletin names duplicated if you are not careful. If no text file is required, as is the case with a main bulletin linked to sub-bulletins, then the main bulletin will be placed into the "+bu" file, and you will not need to use the text editor. Once again, you have the choice of line or color editors, and must also supply an access level if access levels are used. Access levels serve the same purpose here as they do in the message section. Remember that the bulletin text file name will always be 16 characters long, padded with blanks. Use the $ command in the editor section to examine the bulletin files to be sure of this. Note that the P command is not active on the BBS when the bulletin section is running in text mode.

D: Delete A Bulletin. This command only works when the bulletin section is running in relative mode. Enter the name or the reference number of the bulletin or sub-bulletin to delete. Only a level "s" user may use this command. Access is not given to a normal user, nor is there provision for it. This command will also delete the bulletin text, in addition to the description record in the "+bu" file.

THE USER SECTION.

There are very few commands to the user section. This section will be very brief in it's explanation.

ST: User Status. This command will give a short summary of a user's level of participation on the system. The ST command will give the following items on-screen:

-User Name And ID#.
-Access Level.
-Time Status. This will be either NORMAL, DAILY, or UNLIMITED (see CREATE.CNF and EDIT.USERS for information on time status).
-Time Limit. This is the user time limit inn minutes. This line will not be displayed if the user has unlimited time (note that the "minutes" prompt following the time limit can be changed to any unit of time in the text file for use in developing "theme" boards).
-System Logon. The number of calls the user has made to the system since being added to the user list.
-Total Uploads.
-Total Downloads.
-Messages Written To The User.
-Messages Written By The User.

A level "s" user will see the prompt "I/N:" when using the ST command. The BBS is prompting the user for an ID# or name of any user on the system. This will allow level "s" users to get a status report of any user on the BBS, including himself. The SYSOP cannot have a status report because the sysop is not part of the user list file. The guest user cannot use this command either.

MOTD: Message Of The Day. This command will display the daily bulletin, in case the user accidently aborted it when logging onto the BBS.

LOG: Displays User Log. This gives a list of users who have logged onto the BBS since the last time this file was cleared. This is the "+ul" file, which is updated every user logoff. A guest user will be denoted with the ID# of "GST" in the log. The normal user will be listed in the log, with his ID#, and his date of logon, and date of logoff. There is a special symbol that may appear next to the user's name in the log. This symbol will be one of the following. If there is none, it means the user is a normal user logging off normally.

"*" : The user is a level "s" user, with special access.
"-" : The user hung up on the system, or vice versa, and carrier was lost.
"#" : The user was connected at 1200 baud, not 300 baud.

Of course, not all 3 can appear at the same place, so there is a level of priority. The "*" symbol has lowest priority, while the "#" symbol has highest priority.
A level "s" user will be prompted to clear the log after reading it. If the user replys YES, then the log will be cleared, and a single carriage return will be placed in the file. The normal user will not see this prompt.

PASS: Change User Password. Guest users and the sysop can not use this command. All other users can change their passwords at any time with PASS. The old password will be displayed first. Then the user will be asked to enter the new password, TWICE. Each time, the password will be echoed as "*" symbols to hide the identity of the new password. The password must be entered the same way both times. When the password change is complete, the user will be notified that the change was successful. If the user did not enter it twice in a row, then the PASS command will be aborted.

#U: List System Users. This command will give a complete list of all the current users of your system. It display the user's ID#, name, and date of last logon. If the user is a level "0" user, then the "*" character will appear next to the user's name. Entering U without a number will start listing users at ID# 1. U56 will list users starting with the user who has ID# 56. On an ID# system, you may want to list large user lists with some type of alphabetical order, to make finding names easier. For this reason, U supports alphabetical references as well. To use an aplha reference, enter U followed by ":" and a letter that will be the first letter in each of the user's names. For example, U:J will list all users whose names start with J, like JOHN SMITH. A numerical reference may be used at the same time, so that U144:Z would list all users whose names started with a "Z", and start at the user with ID# 144.

The next 2 commands are for the ASCII version of the BBS only. They are not available in the CBBS.

FF: Toggle Formfeeds. Formfeeds have ASCII value 17, and are used to clear the screen on some TTYs. Many terminal programs allow clear screen to be used with the formfeed character. This command will toggle the formfeeds on and off.

BT: Toggle Bell Tone. The BELL is ASCII value 7, and is another common feature of many TTYs and terminal programs. Toggle the bell on and off with this command.

The next 2 commands are for the CBBS only. They are not available in the ASCII version.

COL: Toggle Mono-Color Mode. As mentioned under CREATE.COL, the CBBS has 33 commands that will cause the screen colors to change to new values. COL will toggle between the two modes. Mono-color mode will leave the screen colors in one state only. Full color mode changes the screen colors to the values set in CREATE.COL.

CCOL: Change Mono-Color Mode Colors. While in mono-color mode, you will normally be left with the first set of screen colors that are found in the color table. After logon, you can toggle into mono-color mode, then set the values of the border, background, and text colors that you want to use throughout the BBS. There will be 3 prompts, one for each of the above three screen colors. Use the values from 0 to 15, the standard Commodore color code values.

THE EDITOR SECTION.

The editor section is where system maintenance operations take place. Through the 16 commands available here, you are at liberty to adjust most things on the BBS to your liking, all from remote. Some things will still require off-line editor support however. Only the level "S" user may access this section. The SYSOP is the only one who may access all 16 commands. The "$" and "K" commands are for the sysop's use only, and are never allowed to be used by anyone else. Because of the power of the editor section, you must be careful as to who can access this section. In the wrong hands, your system could be destroyed, unless you have backups. The global commands VOTE, G, BAUD, and CHAT will not work in this section. You must be outside this section to access those commands. There is no help screen or file for this section. From the "E)" prompt, enter "?", and you will see a summary of all 16 commands. The text for the editor section is built into the main program, and cannot be changed. It all exists in a very short keyword abbreviation setup, to save memory.

S: Change User Status. Specify a user name or ID# at the prompt I/N. As in EDIT.USERS and the modify command, enter RETURN alone in any of the fields to not change the value. After all changes, the new record will be saved back. You may only change a few of the user's record fields:

Abbreviation	Field
NAM	User Name
LVL	Access Level
PWD	Password
TLI	Time Limit
TUS	Time Used
TST	Time Status
RDA	Relative Directory Access
PDA	Physical Directory Access
SAC	System Access

M: Change Message Status. BBS V3.0 only allows 2 things to ever be changed in the message header. The category number and access level are the only 2 items that can be changed. The BBS will prompt you for a message number, then will ask for a new category number (CAT:), then the new level (LVL:). Enter RETURN alone to not change either value.

B: Change Bulletin Status. This command allows you to change the first and second link numbers, and the level of any bulletin. This command is not active if the bulletin section is running text mode. Enter the 2 link numbers (LK#1, LK#2) as outlined in EDIT.BULLETINS. Use RETURN by itself to not change a link number or the access level.

#F: Change File Status. This command is not active if there are no relative directories. You may change the directory number (DIR:) of any program, and either add or remove a system password from the program (PWD:). Entering RETURN alone at the PWD prompt removes a password that is on a program while any 1 to 8 character word can be entered to put a password on the program. You may use a numerical reference like "F4" to modify a program in directory 4. You may also use a reference number at the NAM: prompt, like ":23" to modify the program that has reference number 23.

A: Add A User. This is similair to the add command in EDIT.USERS. Enter all data as you would with that command and the S command above. The date of last logon for the user will be set to the current date on the BBS, and all system counters will be set to 0. Specify an ID# of a user that is not currently in use.

#P: Add A Program. This will add a file to a relative directory. This command is not active if there are no relative directories. The prompts are as follows:

NAM: Program Name. DIR: Directory #. SID: Sender ID#. SIZ: File Size. PWD: Password. TYP: File Type.

The system accesses for the program are preset to 0. Enter RETURN alone at the PWD: prompt to not use a password on the program.

Note that with the P command, you can use a directory number reference, like P2.

D: Delete A User. Just specify the name or ID# of the user to delete, You may not delete yourself, and cannot delete the sysop. If you try to delete yourself, you will end up removing your name and ID# out of memory, but not off disk. This means that you will not exist on the BBS until the next BBS system bootup and I&FS reset.

$: Disk Commands. This is a sysop-only command. As soon as you enter "$", the "$)" prompt will appear. This command sets the default work drive to drive 8:0. To change the drive you want to access, enter:

"b" : Bulletin Drive.
"m" : Message Drive.
"f" : Files Disk.
"1" to "8" : System drive 1 to 8.

Enter one of the above letters/numbers instead of a disk command to change drives. The numbers 1 to 8 correspond to the 8 system drive device and drive numbers you set for the PHYSICAL directories, not the relative directories. Use "@" to read the disk error channel, "$" to read the disk directory, or enter the disk command. You MUST use a drive number of "0" after every command, even if the drive is drive "1" of an 8050 dual drive. The BBS will change "0" to the proper drive number before the command is sent. For example, use "s0:file name" instead of "s:file name". Doing the latter will send a bad command command to the drive.

E: Edit A File. You may load, edit, or create any file, as long as it is a SEQ type file. You may save the edit buffer to a file, or append it to the end of any seq file on disk. Writing to an existing file will save and replace the file ONLY if that file was a SEQ file. This is to ensure that the system relative files and +sf program file can not be damaged. The continue after saved option in the text editors have no bearing on this command. The W/A: command for appending text allows the same thing without having to rely on the editors. When the command asks for SDRV:, enter the system drive value, as you did for the "$" command above (ie. enter "f" for files disk). You may choose to create the file, in which case you will go straight to the editor you selected. Otherwise, the BBS will ask for a file name. Enter the name of the file to edit, the system drive, and the BBS will check for it, and determine if it is a sequential file. If it is a SEQ file, you will be asked for a starting line number to begin editing the file at (ST-LN:). Enter "0" if you want to edit the file from it's start. Otherwise, enter a number that will be the first line the BBS will begin loading the program in at. An example of this would be a 225 line text file. The line editor only supports 100 lines at a time (note that starting lines are only allowed with the line editor, not the color editor). Say you want to copy the file from the files disk to the bulletin disk. So it would go like this:

Pass 1: Starting line 0. Write to system drive "b" as "file name".
Pass 2: Starting line 100. Append to system drive "b" as "file name".
Pass 3: Starting line 200. Append last 25 lines to drive "b" as "file name".

In this way, the entire file can be copied. If the file is a color file, then you can still copy it, even though you will use the line editor. The line editor will "hold" the color file in memory. You will not be able to edit this buffer though. You must save it as another file as soon as it has loaded into the buffer. When you select the save option from the editor, you will be asked for write or append (W/A:). Select the mode you want, then the file name, then the system drive.

K: Kill System. This command will hang up the BBS through the hangup routine in the modem file. After logging the user off, the system will "hang" in an endless loop, and never reset the modem for answering. In this way, the BBS is no longer active and waiting to take incoming calls. The board is essentially "dead", as is the phone line. To get out of this hangup, press the RUN/STOP and RESTORE keys at the same time, or shut the computer off.
This command is for the sysop only.

H: Toggle CHAT Command. This is the remote version of the local mode C= C function, and toggle the active state of the CHAT command between active (C-Op) and de-active (C-Cl).

I: Toggle File Section. This is the remote version of the local mode function C= F. It will toggle the file section between open (F-Op) and closed (F-Cl).

#V: Read Votes / Validations. This command has a dual role. When you enter V, the BBS will ask you for a vote read or a validation read (1:VA 2:VT). Enter 1 for validation read, or 2 for vote read. If you select validation read, the BBS will display the contents of the validation file "+va". After reading this file, you will have the option to clear this file. Respond YES to clear the "+va" file.

If you selected "2" for vote read, you will have had to enter the number of the vote topic you wanted to read the results on (see VOTE, CREATE.VOT, and CREATE.CNF). For example, enter V30 at the "E)" prompt, not V, to read the vote topic results for topic #30. After selecting vote read, you will be given the choice of whether to read a vote summary or tabulation for that topic (1:SU 2:TB). Enter "2" for topic tabulation. The tabulation is the same as that in the vote command, shown to users if the user vote display status is enabled (see VOTE command). If option "1" is selected, you will be given a summary of the voting topic. The summary consists of the user vote response (0 to 9), followed by the user ID# and name. For example, "5:43-JOHN DOE" would mean that user #43 has responded with a "5" to the vote topic selected. All users who voted on this topic will be displayed. Use the Xon/Xoff characters to control the output of the summary.

After a vote summary or tabulation, you will be asked if you want to clear the vote results. Entering YES will set all user vote responses to nulls again, including the sysop's. This will ONLY clear the vote results for the topic selected. To clear ALL vote topic results, use CREATE.VOT to erase all voting results.

R: Read A File. This allows you to read any SEQuential file off any drive on the BBS. Enter the drive value in the same manner used with "$" and "E" at the SDRV: prompt.

U: Go To User. With this command, you can log onto the system as another user (SYSOP excluded!) without having to really logoff. For example, you logged on as the sysop, and want to know what JOHN DOE has for mail on the system. You could easily check the message section, but you could also log on as JOHN DOE. Enter "U" here, and you will be asked for the name or ID# of a user. Enter JOHN DOE, and you will log onto the system at the point where the password was just accepted by the BBS. The password of the user you change over to need not be known. The U command will place the password into the system at the point where it's required. You will then begin logon procedures at the point of normal logon (ie. daily bulletin displayed). What happens to the record of the user who used the "U" command? Well, the user's time and actions on the BBS are thrown away, and not used, as are any actions that would affect the user's system counters. It would be as if the user were never on the system. This "U" command is what you should use for validating a guest user. With a guest user on-line, use the sysop emulate local function C= S to emulate the sysop, use "A" in the editor section to add the guest user, then use "U" to go to that new user.

#C: Cycle Messages. This is the remote message recycler. If you have a lot of deleted messages on your message base, you will likely have a lot of "empty" places where read mode can't read. To crunch down the message base and remove the empty records, use this command. The time it takes to recycle message depends on the number of messages and the type of disk drive. A full message base with NO deleted messages will not recycle, since there are no empty records; the message base is already crunched. An IEEE drive will recycle messages very quickly. A 1541 will take from 1 second to a few minutes to recycle a message base. The mean time is about 1 to 2 minutes for a 100 record message base. Sometime, the Commodore DOS fails to delete certain messages because of a bug in the 1541 DOS. It will be necessary to periodically (once a month) check the message disk to see if there are any "unwanted" messages left that should have been deleted. Note that the QUICK CYCLE program on side 2 will do the same thing as this command, but at a faster pace. If you add a numerical reference to the end of the "C" command, then the recycler will delete all messages up to the value of that number. For example, C15 will delete all messages from 1 up to 15, and then begin the normal recycling process.

TECHNICAL NOTES.

MODEM FILES.

Modem files are small 1-2 block files that contain the necessary routines to allow the BBS to work with various modem types. There are only 3 modem files on the standard BBS system, 1 for the 1650 compatible modems, and 2 for the 1670 modem. These files, and their associated PAL source code files, are found on side 2 under the file names:

```
modem.1650   - Source Code : m1650.pal
modem.1670   - Source Code : m1670.pal
modem.1670.2 - Source Code : none
```

Designing a modem file involves using machine language and an assembler program. The source code for the 2 modem files on side 2 were written with PAL, a very good assembler program. You may use any assembler, or a monitor if you wish, to design the modem file. The modem file has the following structure:

```
Start Address     : $9600 Hex - 38400 Decimal
Maximum File Size : 512 Bytes in the range $9600 to $97FF inclusive.
Routines Needed   : $9600 - Answer Routine JMP.
                    $9603 - Off-Hook Routine JMP.
                    $9606 - On-Hook Routine JMP.
                    $960C - Modem Hangup Routine JMP.
                    $960F - User Defined Routine JMP.
```

These 5 routines MUST start with a jump table at $9600, so that the modem file always starts off as:

```
*=$9600
JMP ANSWER
JMP OFFHOOK
JMP ONHOOK
JMP HANGUP
JMP USERDEF
```

You may also access the 2 printing routines in the main BBS programs, through the following JSR calls:

```
JSR $290C (10508 Decimal) : Print the current character in the accumulator.
JSR $290F (10511 Decimal) : Print the text string following the JSR call.
```

```
Example: lda #"a" ; print "a" to the screen
         jsr $290c ; print routine
         more code....

         jsr $290f ; call line print routine
         .asc "Text String" ; this stuff goes to the screen
         .byte 0 ; terminate with a null
         more code....
```

The print character routine will print any character through the BBS custom screen editor. Do not use the KERNAL output routine CHROUT ($FFD2). That routine should not be used to print characters to the screen. The line print routine at $290f will send all characters immediately following the call to the routine. As in the above example, the characters "Text String" will be printed to the screen. A null byte (0) MUST always be used for terminating a text string. After this null byte, your code can continue again. The M1670.PAL code uses both these routines. Print routines are provided in case you want one or more of the modem routines to send some type of acknowledgement message that a routine has been called.

The answer routine must reset the modem, and setup to answer the phone for the next call. The routine must also set the .X 6510 index register to indicate what type of connection was made to the BBS.

.X = 0: If the register is set to zero, then no call was made to the BBS. Rather, a key on the keyboard was pressed. Use the $FFE4 GETIN KERNAL routine to check for a keypress. If a key is pressed, leave it's value in the accumulator, and set .X to 0. This key is checked to see if it is one of the LOCAL MODE 1 function keys.

.X = 1: This will acknowledge that a 300 baud connection was made to the BBS.

.X = 2: This indicates that a 1200 baud connection was made. Only the X register need be set for 300 or 1200 baud connect to be acknowledged by the BBS.

The off-hook routine and on-hook routine are straightforward. You should know what these terms mean if you are considering writing your own modem file.

The hangup routine must disconnect the caller from the BBS and after doing so, must ensure that no calls will be answered by the modem during the BBS reset period. The 1650 code will leave the modem off-hook during the reset period to keep the line busy until the answer routine disconnects and resets for the next call. The 1670 routine will send the default "+++" escape sequence to disconnect, then will either send an "AT S7=0 D" or "AT S1=0" command to prevent incoming calls.

The user defined routine is called by entering the F2 function key while waiting for a call, or while in CHAT mode. This routine can do anything you want it to do. For example, if you were to make a Mitey Mo modem file, you can use F2 to toggle the modem between answer and originate modes. This routine is not used in the 1650 version. The 1670 version will use this routine to toggle modem responses on and off.

All routines must assemble into the 512 byte area set aside. Any larger modem file will cause the BBS to crash. The source codes for the modem files are provided to give sufficient information to understand how to make or modify the files. If you do not understand assembly language, and are unable to make a modem file, you will have to send us information regarding how your modem operates, and how it would perform the above 4 routines the BBS needs. You may provide a BASIC listing, pseudo-code, or even an assembler listing of the modem routines, and the peeks, pokes, and AT commands that make your particular modem function. Write to us at the address found on the warranty page. We will attempt to write the modem file based on the information you give us. We cannot guarantee that the program will work, if we do write it. We will send a BASIC-loader listing of the modem file on printout, so that you can type in the listing, and run it to create the file.

Modem files may be modified and created without your need to notify us. We would appreciate knowing if you do create a modem file that works, so that we can provide better compatibility to other users who run our program.

GameLink-3. Some of you may be aware of the fact that the C-Net bulletin board program allows overlay programs to be used to provide on-line games, quizes, etc. DarkStar BBS V3.0 allows something similar to that. The problem with this system is that it is written in machine language, and access to the main system code is not allowed. The BBS will allow game program files, utilities, or even networking programs to be loaded "on top" of the main BBS program. These programs are all part of GameLink-3. This feature IS already incorporated into THIS version of the BBS. However, we are keeping it's structure, and the method by which is is enabled, hidden from you until game modules can be created. This feature is not documented, since it is not guaranteed that the GameLink-3 system will go through. It's in the program, but we haven't used it. It is our job to make these modules, since they will need to patch into all the major input/output routines in the BBS, and you are not permitted access to the operation of the BBS. Also, they must be written in machine language, not BASIC. GameLink-3 modules can be anywhere from 1 to 50 blocks long. A 50 block game in machine language, running on a BBS, will prove to be quite an experience.

When you purchased this program, you were placed on our customer list, and added to the DarkStar BBS node list, which means that your BBS system gets national advertising through a monthly BBS listing we distribute. Because we will have information about you, we will be able to send additional documention on GameLink-3, and any modules that we design. These programs will be placed on disk, and shipped to you for a minimum fee. We will also take suggestions on game ideas for GameLink-3. If you want something better than poker, or roulette, or simple mind games, then feel free to notify us of any good ideas. We will design special games for the CBBS as well, to take advantage of the color potential of this BBS.

RECORD STRUCTURE.

The 5 relative file structures are described here in detail for your convenience.

Messages. File Name: +me - Record Length: 45 bytes

Byte 0: Message access level. Value: 0-9. if a message is deleted or not present in a record, then the first byte will contain the "#" character, to signify an empty record.

Bytes 1 And 2: Message reference number. This is a 16 bit binary number, in the usual 6510 lo byte / hi byte format.

Byte 3: ID# of who wrote the message. Value from 0 to 255 are allowed. Note that using ID#s here saves much space, since the name of the user doesn't have to be stored in the record.

Byte 4: ID# of who the message was written to. Values are 0 to 255, like above.

Byte 5: Public / private status. If a message is public, this byte has a value of 0, and if the message is private, this byte has a value of 1.

Byte 6: ALL status. If the message is addressed to ALL, then this byte has value 0. This byte has value 1 if the message is addressed to an individual user. If the message is addressed to all, then byte 4 above is also 0.

Byte 7: Message category number. Values range from 0 to 9. If categories are not used, this byte will still contain 0, which is category # 0.

Bytes 8-32. These 25 bytes are the message subject.

Bytes 33-38. Date message was sent on. This date is stored in a compressed 6 byte format. The first 3 bytes are the day, month, and year, stored in binary format. The hour and minute are the next 2 bytes, each stored in binary coded decimal format (BCD). The last byte is the Am/Pm character, which is an "a" or a "p".

Byte 39: Forward flag. If a message was forwarded, this byte will have value 1. If not, it's value is 0.

Byte 40: Forward user ID#. If the previous byte was set to 1 and the message forwarded, this byte will contain the ID# of the user who forwarded the message.

Byte 41: Replied counter. This is a count of the number of times the message has been replied to. If it's value is 0, then the replied counter will not be shown in the message header.

Bytes 42 and 43: Reply to reference number. If the message was a reply to another message, then this 16 bit value will contain the reference number of the message that was replied to. If this message is not a reply, then both these bytes will have value 0.

Byte 44: Carriage return. All records must have a carria
data sometimes.

If a user is deleted on the BBS, the message base has a unique way of handling this. A message that was written by a deleted user, or to a deleted user will appear like this:

```
Message : #43/87 L6 (1427)
Sent By : 43-(User Deleted)
Sent To : 23-(User Deleted)
Etc.....
```

The user's name is removed from the message header display. The user's ID# will still be stored here in the record, but the BBS will still acknowledge a deleted user. A reply cannot be made to a message that was written by a deleted user. As you can see, if the name of the user is changed, or if the deleted user was replaced by a new user, any message in the message base that involved the old deleted user now becomes the property of the new user.

By using internal user storage, the size of the relative files was reduced by 100%. But as you can see, shifting ID#s of users by sorting them would cause a headache when examining the message base, seeing that the messages have all changed ownership.

Users. File Name: +us - Record Length: 59 bytes. The size cannot change from 255 records (61 blocks).

Byte 0: Access level of user. Ranges from 1 to 9, or the character "s" for sysop level users. If there is no user at this position, then this value will be the "#" character, as was the case with messages, and is for bulletins and programs as well.

Bytes 1-25: User name. 25 characters in length, letters A-Z, spaces, and the period (.) being the only characters allowed. All letters are upper case.

Bytes 26-33: User password. 1 to 8 characters, padded with spaces if less than 8 characters in length, any "printable" character value, upper case letters only.

Bytes 34-39: Date of last logon. Compressed format.

Byte 40: User time limit. 1-59 minutes, stored as binary coded decimal (BCD).

Byte 41: Time used. Ranges from 0 to 59 minutes, BCD format.

Byte 42: Time Status. Bit 1: 0=daily time limit. Bit 0: 0=unlimited time. Bits 2 to 7 are not used.

Byte 43: Relative directory access. All on bits indicate no access, all off bits (0) indicate access for the directories 0 to 7.

Byte 44: Physical directory access. Has the same format as the previous byte.

Byte 45: System accesses. All off bits (0) indicates access is given.

Bit 7: Access to "+" files	Bit 6: Access to DOW command	Bit 5: Posting bulletins.
Bit 4: File section access	Bit 3: Physical directory commands	Bit 2: Multi-file download.
Bit 1: Multi-file upload	Bit 0: Access to view restricted file "+re"	

Bytes 46 And 47: Hi reference number of last message read (16 bit).

Bytes 48 And 49: Total uploads (16 bit).

Bytes 50 And 51: Total downloads (16 bit).

Bytes 52 And 53: Messages to user (16 bit).

Bytes 54 And 55: Messages by user (16 bit).

Bytes 56 And 57: Messages to user (16 bit).

Byte 58: Carriage return.

Bulletins. File Name: +bu - Record Length: 37 Bytes.

Byte 0: Access level (0-9) or "#".

Byte 1: Link number 1 (0-254).

Byte 2: Link number 2 (0-255).

Bytes 3-10: Bulletin name. Alphanumerics and spaces only, upper case.

Bytes 11-35: Bulletin Description. Any 25 characters.

Byte 36: Carriage return.

Programs. File Name: +bu - Record Length: 32 Bytes.

Byte 0: Directory number (0-7) or "#".

Byte 1: Sender ID# (0-255).

Bytes 2-17: File name. Upper case letters only.

Bytes 18-25: File password. Length 1 to 8 characters, padded with blanks, upper case. If there is no password, then this field will contain 8 lower case "a" characters.

Bytes 26 And 27: Download accesses (16 bit).

Bytes 28 And 29: File size (16 bit). Only the first 3 characters are used for a 0-999 block range.

Byte 30: File Type. An upper case P, S, or U.

Byte 31: Carriage return.

The VOTE command relative file structure is very simple. For each topic, there are 2 records. Record 1 contains the voting responses for users 0 to 127, plus the return character. The second record contains user responses for users with ID#s from 128 to 255, plus the return character. Each response byte will be a null (0) byte if the user has not voted on the topic. Otherwise, the byte will contain a value from "0" to "9".

INDEXED SEQUENTIAL FILE STORAGE.

The ISFS system the BBS uses involves taking the key fields from the relative files, and storing them in sequential lists in memory. This type of storage is sometimes called Indexed-Sequential, since the files on disk exists in a sequential format, while the key field lists are indexed into memory. Although relative files are fast, and are non-sequential format, a BBS will tend to access the records in a sequential fashion when searching for data in a specific record. This involves using the DOS pointer command and a lot of record input from the drive. Since serial drives are so slow, and IEEE drives are only 5 to 8 times faster, there should be a way to search for data instantaneously, requiring zero search time. The ISFS system stores all key index lists in memory, 8 lists for the messages, with a preset size of 200 bytes per list, 2 lists for the bulletins, each with a size of 200 bytes, 1 700 byte list for the programs, and 1 6400 byte list for the users. The 6400 byte user list uses the entire user name for the key index field, so that makes 25 bytes X 256 users = 6400 bytes. By storing users in memory, the BBS can compress the relative files down to very small sizes, and can respond instantly to locating any user when a user's name is requested on the system. The amount of internal storage may seem a little high, but the space vs. speed tradeoff was worth the effort. Besides, the savings in disk space by holding users in memory pretty well offset the loss in internal memory.

Also, with lists in memory, operations can be performed on them with instant results. The number of messages to a user, for example, can be summed up internally along one of the key index lists, and stored elsewhere in memory. This eliminates the need for counters in records, since the ISFS system can keep count of everything in memory, and adjust values in memory AND on disk at the same time. Because the BBS makes full use of all memory from $0000-$F000, including part of the stack, RUN/STOP RESTORE had to be disabled from use. Attempting to warm start reset the C64 would wipe out all data from $0100-$07FF, and this can not be allowed.

Using ISFS and IEEE drives together, you will have a BBS that runs faster than any system currently available for the C64.

FINAL NOTES: You will notice a few things about the BBS during the input of commands and text strings to the system. For one thing, all input is converted to lower case, and echoed as upper case in most areas. This is normal to the system, and gives a better appearance on the BBS. Upper case terminals can log onto the BBS, and use it the same as if it were in lower case. The BBS does not need input to be case-specific.

Some areas of the BBS where input and output is used will not check for carrier loss. These are areas where the BBS will not be allowed to reset, such as the logoff prompts, and password input after an upload with UPL. The BBS will "time-out" in these places, and reset will occur after a certain time period, or after the BBS is done with that area of input/output.

Most data in the BBS exists in upper case format, such as user names and passwords. This is done to make the parsing routines work better and faster.

The BBS uses hotkeys for the keywords "YES" and "NO". Certain prompts will require this Yes/No response. As such, you should be aware of this fact when modifying the text file.

Although password input is echoed to the remote user as asterisk characters ("*"), password input will always be echoed to this end as the input character values.

REFERENCE SHEET #1 : STATUS LINE

```
 1   2                      3   4
-------------------------------------
000-8Y5QP                 -9-YYYYYYYY
12:34a 12-12 CFMQ YY0059YYYYYYYYNNNNNNNN
-------------------------------------
  5     6   7  8 9 A   B      C
```

1: Current User ID#
2: Current User Name
3: Current User Access Level (1-9 or S).
4: Current User System Access:

Position 1: Access To "+" Files.
Position 2: Access To Downloading With DOW.
Position 3: Access To Posting Bulletins.
Position 4: Access To The File Section.
Position 5: Access To Physical Directory Commands (MD,MU,U,D,C).
Position 6: Access To The Multi-File Download Command MD.
Position 7: Access To The Multi-File Upload Command MU.
Position 8: Access To View The Restricted File "+re".

5: Current Time.
6: Cursor Row And Column Position.
7: System Status Flags:

C: CHAT Availability (Inverted Is NO).
F: File Section Status (Inverted Is Closed).
M: Modem Input/Output Suppression (Inverted Is Suppressed).
O: Carrier Detect Flag (Inverted Is No Carrier Detect).

8: Time Status:

Position 1: Does User Have A Daily Time Limit?
Position 2: Does User Have Unlimited Time?

9: Connect Time (Minutes).
A: Time Limit (Minutes).
B: Relative Directory Access. Positions 1 to 8 are for access to directories 0 to 7, respectively.
C: Physical Directory Access. The access format is the same as that of the relative directories.

REFERENCE SHEET #2 : LOCAL MODE FUNCTION SUMMARY

LOCAL MODE 1: Waiting For A Call.

F1: Change System And/Or SYSOP Password.
F2: User Defined Modem Function Key.
F5: Local 1200 Baud Logon.
F7: Local 300 Baud Logon.
F8: Exit To BASIC.

C= C: Toggle CHAT Availability.
C= F: Toggle File Section Status.
C= M: Toggle Modem Input/Output Suppression.
C= O: Toggle Carrier Detect Status.
C= U: Toggle Upper Case Mode.
C= K: Toggle Keyclick.
C= 1: Change Border Color.
C= 2: Change Background Color.
C= 3: Change Text Color.

LOCAL MODE 2: On-Line. Use SHIFT & RUN/STOP To Toggle Local Mode 2 On And Off

C= C/F/M/O/U/K/1/2/3 All Operate The Same As Local Mode 1.

C= T: Reset User Connect Time.
C= D: Delete User.
C= X: Logoff User.
C= @: Toggle Daily Time Limit.
C= $: Toggle Unlimited Time.
C= 4: Increase Connect Time 1 Minute. C= 5: Decrease Connect Time 1 Minute.
C= 6: Increase Time Limit 1 Minute. C= 7: Decrease Time Limit 1 Minute.
C= S: SYSOP Emulate.
C= L: Change User Access Level.
C= +: Take Modem Off-Hook.
C= -: Put Modem On-Hook (Disconnect).

1-8: Toggle System Accesses.
SHIF 1-8: Toggle Relative Directory Access.
CTRL 1-8: Toggle Physical Directory Access.

WHILE ON-LINE AND NOT IN LOCAL MODE 2:

While At Any Point Of Input: CTRL-W : Write user record to disk with any temporary changes made permanent.
F7 : Engage CHAT mode.
While In CHAT Mode: F2 : User defined modem function key.
F6 : Toggle between color mode and ASCII mode.
F7 : Disengage CHAT mode.
F8 : Toggle duplex.

```
0 "                "
1     "configuration"    prg
1     "categories"       prg
1     "a.commands"       prg
1     "c.commands"       prg
1     "color table"      prg
1     "sysop name"       prg
1     "modem.1650"       prg
2     "modem.1670"       prg
13    "bbs.txt"          prg
1     "timelok"          prg
0     "----------------" usr
40    "create.cnf"       prg
9     "create.dsk"       prg
4     "create.vot"       prg
3     "create.cat"       prg
13    "create.col"       prg
12    "create.com"       prg
11    "create.txt"       prg
2     "create.nam"       prg
7     "create.tim"       prg
0     "----------------" usr
25    "edit.bulletins"   prg
5     "edit.chat"        prg
12    "edit.color"       prg
26    "edit.programs"    prg
18    "edit.text"        prg
24    "edit.users"       prg
0     "----------------" usr
17    "charset 0"        prg
17    "charset 1"        prg
17    "charset 2"        prg
17    "charset 3"        prg
17    "charset 4"        prg
17    "charset 5"        prg
0     "----------------" usr
18    "copy.disk"        prg
9     "copy.file"        prg
8     "copy.rel1"        prg
6     "copy.rel2"        prg
0     "----------------" usr
7     "fix relfiles"     prg
3     "fix +sf file"     prg
2     "color test"       prg
6     "quick cycle"      prg
0     "----------------" usr
37    "text.pal"         prg
10    "m1650.pal"        prg
15    "m1670.pal"        prg
0     "----------------" usr
158   "darkterm docs"    seq
0     "----------------" usr
4     "+he.0"            seq
5     "+he.1"            seq
3     "+he.2"            seq
2     "+he.3"            seq
2     "+he.4"            seq
2     "modem.1670.2"     prg
31 blocks free.

ready.
```

OVERLINK LIBRARY #1

DarkStar CBBS Modular Expansion ToolKit

The New Era of Open Architecture BBSing

Copyright Notice:

Our warranty policy follows that outlined on the first page of the BBS V3.0 manual. Any defective media may be returned at a replacement cost of $7.50 (U.S. funds) for either the OverLink disk or the V3.1 upgrade disk, or $15.00 (U.S. funds) for both disks, provided that you follow the guidelines of the warranty policy outlined in the V3.0 manual.

The OverLink - Contents

PUBLISHER'S NOTE

We Hope You Enjoy This First Library/Expansion Disk. We Are Breaking New Ground With Our Unique Approach! We Invite All Comments andSuggestions Relating to Ideas That You Would Like to See Implemented in Future Library Editions! If You Can Provide Us With a Basic Version of Your Idea, We Will Make Every Reasonable Effort to Convert It to ML For the BBS!

If You Would Like to Become a DarkStar Distributor in Your Area, then By All Means, Drop Us a Line and We Will Provide You With Further Details.

Andrew Leaver
President, D.S.S.

DarkStar BBS V3.1

Version 3.1 is a slightly revised version of 3.0, with only a few minor changes being made, along with one major change. Although the manual stated that version 3.0 would work with the GameLink expansion system, this is not the case anymore. The original GameLink design was dropped due to problems with the basic routine calls in the V3.0 program. Major changes were made internally to V3.1 to accomadate the new OverLink, and also the new CWI. A full explanation of the OverLink system is given later.

There was one minor bug that was corrected to the system with respect to forwarding messages on a dual drive. If you had placed your message text on drive 1 of a dual drive, you would find that forwarding messages did not rename the text file. This has been corrected.

The basic screen editor system of DarkTerm/DarkStar BBS has been modified to allow the cursor to be plotted on the screen, so that it will be able to "jump" from one (row,column) position to another. On the BBS V3.1 disk #1, you will find a file called DARKTERM 4.C. This file is a boot that installs the plot routine on DarkTerm 4.B. To use it, just type LOAD"darkterm 4.c",8,1. When you see the status line on the terminal now, it will indicate that you are now using version 4.c. NOTE: you cannot use the EDITLINK with the 4.c boot. For this reason, do not get rid of your V4.b boot. You must use V4.b to use the edit link. The plot routine is in effect automatically when using the BBS.

The plot function operates as follows: enter CTRL-V. This puts the print routine into plot mode. Follow CTRL-V by entering a 2 digit number for the column for the cursor to jump to. Follow this with a 2 digit number for the new row for the cursor. You can see that this makes up a 5 character string. It MUST be 5 characters in length. Thus, if you want to put the cursor at column 1, you must enter a "01", not "1". For example, to move the cursor to column 20, row 10, enter "CTRL-V2010", where CTRL-V is the key you press with control and v at the same time. To move the cursor to position 1,1, you would enter CTRL-V0101.

Cursor plotting will be used in future on-line games for the CBBS, so become familiar with it; you'll need it. If a terminal other than DarkTerm 4.c is used, then the plot will not go through. Instead, the user would only see the number strings. The plot routine ensures that improper plot calls will not work. For example, CTRL-V9856 would not move the cursor to position 98,56. Remember that the cursor tracker in the BBS and terminal status lines displays the cursor position in row,column format, whereas the plot routine requires a column,row input.

V3.1 uses a different copy protection scheme than V3.0 uses. The new method requires that your disk drive that you boot the BBS from be in proper alignment. This protection type is not one of those infamous "head-banging" schemes, so relax. If the program does not load, then you should a) lift the drive door, remove the disk, put it back in, then close the door again (the head will not bang if you do this), b) have your drive aligned, or c) don't use a backup copy, which won't work at all (at this point in time). Only disk #1 is copy protected. Disk #2 is the same disk used with the V3.0 system. Read the BBS V3.0 manual for information regarding that disk. The program will load off most 1541 compatible drives, not just the true blue 1541.

If you look carefully at the other manual you have, you will see that it is a V3.0 manual. BBS V3.1 is EXACTLY the same as V3.0, EXECPT for the few changes which are mentioned in this manual. The BBS structure has not changed, nor have the commands and their functions.

NOTE: You must not have a printer connected to your computer, even if it is not turned on. Make sure you do not have anything plugged into the cassette port on the computer. When you boot the BBS up, be sure to turn the computer off then on, so that the memory is reset to its power-on state.

A second change to the BBS was made to allow the OverLink system to better communicate with the BBS. Both the E (Edit File) and $ (Disk Commands) command in the editor section access the disk by selecting a system drive to operate with. The V3.0 BBS used F,M,B and the 8 physical directories (1-8) to specify a drive. With OverLink, the BBS can operate on many more drives, which may not be set to one of the 11 drives above. To remedy this, V3.1 uses a drive table that is created with INSTALL DRIVES, which can be found on your OverLink disk. This small BASIC program installs up to 32 drives on the system that can be accessed by E,R and $ by entering the 2 digit number of the drive you want to access. For example, when you get the "$)" prompt with disk commands, you would enter "01" to access drive 1 in the drive table. Remember to use 2 digits! Entering "1" will only give you a disk error #31. If you attempt to access drive 56 by entering "56", this will cause the BBS to set the current drive to the device of drive #2 in the drive table. This is to prevent illegal drive table access.

Setup procedures for V3.1 basically follow those of V3.0. Working with OverLink is a separate task altogether. That will be explained later. Just note that you should perform both the BBS AND OverLink setup procedures before you run the BBS. The order is not relavent.

Setup procedures for V3.1 are outlined below.

1. Create a customized version of the BBS setup using the utilities on disk #2. This is outlined in the V3.0 manual. As a suggestion, take the files the BBS needs to run with from disk #2, and place them on a new disk. This will make working with the BBS a little faster and easier. The files the BBS needs for execution are :

CONFIGURATION	: The BBS configuration created with CREATE.CNF
CATEGORIES	: The message categories file created with CREATE.CAT
?.COMMANDS	: The BBS commands file. The value of "?" will be a C for color, or an A for ascii, created with CREATE.COM
COLOR TABLE	: The CBBS command color table, created with CREATE.COL.
SYSOP NAME	: Sysop name file, created with CREATE.NAM.
MODEM FILE	: The modem file you want to use (ie., MODEM.1650 for 1650's).
TEXT FILE	: The BBS text file. This defaults to BBS.TXT, but can be changed to any name and style with CREATE.TXT.
TIMELOK	: The restricted time period file, created with CREATE.TIM.
DRIVE TABLE	: The V3.1 drive table, created with INSTALL DRIVES.

All of the above are set up with the V3.0 system, except for DRIVE TABLE. See the V3.0 manual for setup procedures. Copy INSTALL DRIVES to your disk #2 backup, so that it will reside with the other setup files.

INSTALL DRIVES. This very short BASIC program sets up the drive table. Load it, and change the data statements in lines 1060-1090 and lines 1130-1160. There is data for 32 device numbers. Do not use device numbers that don't exist. Repeat the device numbers in the table if you have fewer than 32 drives. The 4 data statements for the drive numbers are grouped into strings of 8 numbers each. Set the values in the strings accordingly. For example, the 6th digit in the 3rd data statement is the drive number for disk drive #22. Remember that only the E,R and $ commands in the editor section make use of this drive table.

A note about OverLink: The OverLink system genrally requires more disk space than one drive can afford. There is no restriction on running the system with one drive, other than the fact that you will have a very minimum system. It is recommended that you have at least two drives, with 3 being suitable for a large system using dozens of modules, in case of more OverLink library disks. As you will see from the OverLink SETUP files, the system allows for usage of large disk storage capabilities (ie. multiple SFDs, etc.).

2. Load the BBS. The BBS has two visible files in the directory, Just type load "?$",device,1 to load the first file. Make sure you copy the DarkTera files to another disk before you use the terminal program. Follow the prompts as you would with the V3.0 system, outlined in that manual. The BBS will prompt you for disk #2 at the end of the load. Place your working copy of disk #2 in the boot drive. The BBS will attempt to load all the aforementioned files. If any are missing, you will find that you will not get any farther into the startup procedures. The BBS MUST load from a 1541 or compatible drive.

The order in which the setup works is slightly different from V3.0.

SYSTEM DRIVE. Normally with V3.0, the SYSTEM disk (or FILES disk) was set to drive 8:0, and could not be changed. This is the drive where the relative files, and system files reside. You now have the option to move this disk to any drive on the system. If you want to leave the system drive as 8:0, just hit RETURN. Otherwise, enter the device and drive number of the new system drive in the format "XX:Y", where XX is the device number (TWO digits) and Y is the drive number (ie. use 08:0 instead of 8:0). Make sure your files disk is on that drive when you are prompted for the files disk at the end of the start-up procedures.

LINK FILE. This option allows for external program chaining to the main BBS for expansive and radical changes to the operation of the BBS program. Details of this option can be found with any module that makes use of this feature, No explanation is given here. So, just enter RETURN.

MODEM FILE. See the BBS V3.0 Manual.

USE INVERTED CARRIER?. Some modems, such as a Hayes compatible (ie. GVC 1200) have a reversed carrier detect signal on the user port. This means that it would normally detect a carrier where other modems would detect a carrier loss. For all modem files written for this BBS, a note will be added to indicate whether you should use inverted carrier detect or not. If you use a Hayes clone, just enter a "y". A sure way to find out if you have inverted carrier is to use the normal carrier detect, and wait for an incoming call. If the BBS resets from carrier loss, you should use inverted carrier detect. Enter "n" to set carrier detect to normal operation.

CHARACTER SET. See the BBS V3.0 manual. Note that V3.1 relocates character sets independant of the load address, unlike V3.0, which required at $E000 (57344) load address.

TEXT FILE. See the BBS V3.0 manual.
SYSOP PASSWORD. See the BBS V3.0 manual.
SYSTEM PASSWORD. See the BBS V3.0 manual.
DAY,MONTH,YEAR,TIME. See the BBS V3.0 manual.

OVERLINK. Enter RETURN to NOT use the OverLink feature. In this case, the BBS runs like a V3.0 system. Enter "n" to use NUMERICAL format. Enter "w" to use WORD format and the command word interface (CWI). Both of these modes are explained later.

At this point, you will be prompted for the master disk (#1). Put this in the boot drive, and hit return. This will complete the start-up procedure. You will be prompted for the files disk, then the BBS will begin operation. From this point on, reference the V3.0 manual.

The OverLink Expansion System

In your BBS manual, the technical notes section mentioned that the BBS has the facility for expansion via the GameLink Module System. The term "GameLink" was dropped, and is henceforth denoted "OverLink".

The OverLink is a system by which multiple independant programs can run in addition to the normal commands found in the basic BBS system. The OverLink is so termed because of the 2 ways by which the BBS can interface to this facility. The system will work either by using overlays or program linking, or in some cases, will use both methods at the same time. Overlays are done before the BBS begins system initialization, and are used for replacing or modifying entire sections of the BBS BEFORE it is run. In this way, some parts of the BBS that you have little use for can be replaced with something the basic system does not have.

Generally, program linking is done AFTER the BBS has run. These are the individual OverLink program modules that are loaded into memory via the OverLink command on the BBS. These link files may be anywhere from 1 to 50 blocks long, and may be chained together to give very large programs. The typical link module is usually only 1 to 10 blocks in length. The link modules may or may not modify the main BBS while it is running. If it does do so, then the module will tend to have overlay code in itself that overlays parts of the main system. This is a very powerful feature. A BASIC BBS can not modify itself while it is running, unless files are overlayed of course. This involves too many restrictions on that type of system. On BBS V3, the code will be changed while it runs, and will usually change back to original format prior to the next incoming call.

The primary use of OverLink was originally intended for the use of making on-line games that can be played over the modem. Some of you may have heard of a BBS program called C-Net 10, which allows on-line games to be used with overlays. These games are restricted to a very small size (<16 blocks), and are written in BASIC which we know is slow. With our system, all games will be written entirely in machine language, and can be as large as 50 blocks without secondary chaining of the main BBS. Because of the nature of the OverLink, it is not possible for you, the user, to write your own modules. However, if you have an interesting idea, or even a program you wrote for C-Net that is PD, then don't hesitate to let us know. We are always open to suggestions.

The OverLink disk you purchased is not copy protected, and you should make up many backups, and keep the original disk itself untouched.

All OverLink modules are (C), whether it is based on your ideas or ours. OverLink disks will be sent to you for a minimum fee, as they become available. You will be notified of further OverLink disks, provided you are a legal owner of the BBS, and are on our Node list of DarkStar BBS3 systems. Appropriate documentation will be supplied with each OverLink Library disk.

The first library disk does not really contain a wide array of game modules. This is because of :

a) OverLink must first make an impact on the majority of users of this BBS program. The time and effort required to create game modules that go beyond the simple levels of BASIC game routines on other BBS programs depends on just how much they will be used, AND how much time the actual users of your BBS system will devote to these game programs. Once OverLink has made on impact and users are interested enough to the point of requesting games for us to create, we can be assured that it is worth our effort.

b) Time did not permit us to start writing any game modules. We were only able to get the first 14 modules completed by the official release date. Game modules are of less importance than the other system modules on the O-L disk, and as such were not considered mandatory for the first disk. Of course, the release of future game modules depends on just how well the sales of the initial V3.1 OverLink system go.

The 3 games modules on the first disk are written for the color system only. In particular, BlackJack was written with color in mind. Cursor plotting is a definite must with this module. Black Box is a simple game, a "timewaster", used as an introduction to just how effective cursor plotting can be. The games were not written in ASCII, since most users of the BBS run a color system, and we just are not aware of who is running an ASCII system. Please let us know if you would like these modules converted to ASCII. Because the modules are not copy-protected, appropriate BASIC binary file encoder programs can be sent to you on listing for typing in special modules or for bug patches on current modules.

OverLink Expansion System

The OverLink routines exist on the BBS even if you run the system without the OverLink enabled. Obviously, there must be a way to access OverLink modules. The OverLink command will replace the VOTE command in the BBS, once installed. The VOTE command will not be disabled. VOTE will move from a main BBS command to an OverLink module, which happens to be module #1. See the docs on OLINK Module 1 for more information. As such, the VOTE command definition should be changed from VOTE to something more appropriate (ie. EXP for system expansion command). If you consult the BBS manual under VOTE, you will find that the OverLink (henceforth termed O-L) works in a similar manner. We will use the EXP command definition to represent the O-L interface.

When a user enters the command EXP from any of the main command prompts, the BBS will display a sequential description file for current O-L modules. This file is named "+gl" and must be placed on the files disk (drive 8:0). If a user enters EXP followed by a number from 1 to 99 (ie. EXP8), then the BBS will load and execute the O-L module designated by the number the user input. O-L modules have a file name format of "+ga.XX" for the ASCII BBS, and "+gc.XX" for the CBBS. For example, EXP8 would load and execute the module "+gc.08" when using the color BBS, and "+ga.08" when using the ASCII version. Load times depend on the length of the module.

Let's try an example O-L structure for the BBS. We have 5 current O-L modules, and we can use any number sequence we want. That is, we do not have to use 1 to 5; we can use 1, 4, 8, 56, and 99 if we want to, but we'll use 1 to 5 in our example. We'll be using the CBBS, so we'll use the color BBS "+gc.XX" modules. The following steps are done to setup our system:

1. Place the O-L modules on the OverLink disk. You may use the SETUP files to change the O-L module before it is placed on the O-L disk. Refer to the individual O-L module docs. The OverLink disk is the system drive that physical directory #8 uses. In the CREATE.CNF program, you had to set up the physical and relative directory device and drive number tables. The last one (the 8th) is now designated as the OverLink BBS disk. You may share other system files on this disk. If you are using the 8th directory, then you will have to share it with the O-L modules. Remember that this is physical directory #8, NOT relative directory #8. Set the device and drive numbers to the drive you want. For our example, we'll use drive 9:0 as the O-L disk. Use CREATE.CNF to perform this step, then use the SETUP files on you O-L module disks to change the modules, or simple copy them as is to the O-L drive. O-L modules, as they are when you first see them on the module disk, have a file name format of "olink.mXXX" or "olink.mXXX.a". The latter name is the ASCII module format, the former is for the CBBS. You can NOT interchange them. You should always make any changes to the modules on a backup disk, then copy them over to the O-L drive and change the file names to the appropriate O-L number. The XXX in the file name is the module volume number. For example, the VOTE command O-L module is module #1, and appears on the O-L module disk as "olink.m001" (color) and "olink.m001.a" (ascii). The setup file is called "setup.m001". On the O-L BBS drive, we will have to rename the file "olink.m001" to "+gc.01", once copied to the drive. Note that we do not use "+olink.m001.a", since we are not using the ascii version in our example.

2. Setup the "+gl" description file. Our file will appear as follows. Remember to write this to drive 8:0.

Expansion Command	Enter
Voting Section.....	EXP1
BBS Advertising Section.....	EXP2
Las Vegas Blackjack.....	EXP3
Hangman.....	EXP4
Trivia Query #1.....	EXP5
This List.....	EXP

The above O-L modules would have file names of +gc.01 to +gc.05. If we ran the ascii version, the file names would be +ga.01 to +ga.05. Remember that the VOTE command is now the EXP command (or whatever you want for the command word, using CREATE.COM).

OverLink Expansion System (P3)

We now have the overlay files on the O-L drive, have changed the VOTE command word to EXP, and have created the "+gl" description file (remember that our overlay modules are on drive 9:0). We now boot up the CBBS using the normal procedures outlined in the manual.

Once the BBS is running, a user would enter "EXP" and would be given the description file on drive 8:0. If the user enters EXP5, then the BBS will load and execute "+gc.05", which should be the trivia module, from the O-L drive, which is 9:0 in our case. If a user enters an invalid number, such as EXP89 when there is no "+gc.89" file, the BBS will still attempt a load, but will not execute anything if there is no file. Therefore you do not have to have 99 modules running to use the O-L. Note that the BBS uses the Kernal LOAD routine, and only checks the routine exit status and not the disk error channel, so you will see the error light flash if a module is not found. Ignore this; it is harmless, and will disappear upon the next access to that disk drive.

Note that the OverLink module numbers do not have to correspond to the EXP numbers. For example, the VOTE module is module #1, but it can be accessed with EXP56 with a file name of "+gc.56" if you want it as such. Some modules can even be duplicated on the same disk, like the trivia module, which allows as many trivia sections as you want using muptiple copies of the same module via it's setup file.

That is about all there is to the actual OverLink system itself. Because it is always expanding, documentation will appear as more modules are created. Each module itself will also contain all necessary information for its use on the BBS. To finish up, you should take a look at the Command Word Interface, which follows this. It explains how to change the O-L routines so that you can use real command words instead of numbers (ie. TRIVIA instead of EXP5).

The Command Word Interface (CWI)

The CWI is an overlay program that allows the OverLink to run using command words, rather than the numerical selection mentioned previously. There is 1 file on O-L disk #1, named "cwi.com", that you will need to create a CWI command file.

This is not a run time overlay, and as such is installed along with the OverLink at bootup time. Refering back to the section on OverLink setup, you should install the O-L when the "OverLink:" prompt shows. As said before, you enter either "n" for numerical operation, or "w" for using the CWI system.

The "cwi.com" program is written in BASIC, and is very small and simple. It is used to create the command word data file. This program MUST be run on the OverLink Drive disk, the one you place the overlay programs on, which would be the physical directory #8 drive. If you intend to place a lot of other system files on the O-L drive, then you should rearrange the directory so that the command word data file is at the top of the disk directory. A program called Yellow Pages has been placed on O-L disk #1 for your use. It is a P.D. directory organizer, and does a great job of rearranging. For example, if your O-L drive is drive 8:0, then you should place the CWI data file just under the system counter file, "+sf". The CWI data file is named "++xc", and in this case, it would be the second file in the directory.

Load and list the CWI.COM program. Line 1040 sets the number of O-L commands, which would be the number of data statements used, one for each command. In the example given, VOTE and BBS are the only 2 listed. There are 2 commands, thus NC has a value of 2. As you add more commands, increase the value of NC as necessary. The syntax for command word definition is the same as that listed under CREATE.COM, except (1) the command word length can be from 1 to 8 characters, and (2) there is no length data byte in the CWI.COM program. All commands are divided by a chr$(128), and saved in sequence.

When writing overlays to the O-L drive disk, you do not save them in the format "+ga.XX" or "+gc.XX". You now save the file in the format "+-COMMAND". For example, VOTE would be stored as "+-VOTE", and BBS as "+-BBS". Remember! Only 1 to 8 characters. Going beyond this limit will cause problems.

From our previous OverLink example, we used EXP as the command redefinition for the O-L interface. With CWI, we get the same result when we type "EXP". We get a listing of the commands to be entered. Consider this a help file, like any other (the "+gl" file, that is). No need to enter EXP1 for the voting section; now we just enter VOTE, and we get the same result. The big difference is that when we enter an O-L command through the CWI, the BBS will access the "++xc" file from the O-L drive, and search for the command, and if found, load in the overlay and execute it, based on the name of the command. This is the reason why you place "++xc" at the top of a directory. Access times will be very quick when doing it this way. As you add more commands to "++xc", the access time will go up, but not enough to take notice. The most important thing to remember is that ALL O-L commands accessed via the CWI can only be accessed from the MAIN MENU prompt. You can not type "TRIVIA" at the main message command prompt, and expect to get the trivia module.

Well, that's about it for the CWI. Additional information may be given with each individual OverLink module. The CWI does not have a module catalog number like normal O-L modules.

Module #001 : VOTE Section Replacement
Files : SETUP.M001 - OLINK.M001 - OLINK.M001.A

With the OverLink interface replacing the VOTE command in the BBS, the first module to design would logically be one that replicates the VOTE command of the BBS. Thus, we have the VOTE section module. Funtionality is the exact same as that with the normal BBS VOTE command. Refer to the BBS Manual for details.

The SETUP file will allow you to change the drive to which you will place the "+vt.XX" files on. The "+vo" result file is still on drive B:0, and the +VT.XX files can be on any drive this time around. The following lines are the only ones you may modify in the setup file:

100. DV : Device number of drive to place +VT.XX files on.
DR$: Drive number of drive to place +VT.XX files on.
V : Version of module to modify. Set to 0 to modify the color module, or 1 to modify the ascii version.
120. DF$: Destination file name. This usually defaults to the original file name, in which case a modification will result in the original file being scratched and saved over with the new file. You may change this to any file name you want, and if you use the CWI, you should change it to the command word, preceded with the "+-" characters. You should also change the destination drive too (see below).
130. Some characters in prompts can't be expressed in text strings, so it is necessary to use variable strings. Such is the case with q$ and r$, for quotes and the return character. Other modules may have more of these things found on this line.
140. Vote Selection Prompt. All modules will have at least one prompt to redefine. Unlike the CREATE.TXT program, you must change the string value here. Keep it less than 255 characters. You may have to concatenate if you use more than 80 characters. This modification should be fairly obvious to you; if not, don't modify the prompt.
250. This line sets the device and drive number for the destination drive, where the modified module is to be written to. You can change the drive and device number if you want; it will save placing things on the original OverLink library disk (always modify modules on a backup, never on the original disk purchased!).

You will then run this program. The old module is read in, the destination file is scratched, then the modified file is written to the destination drive. Be careful not to change the data statements. They are the machine language routines for reading and writing the module quickly from and to the disk.

Note that the original VOTE modules have defaults as you see them in the setup. You don't have to use the SETUP program if you do not want to change anything.

Changes to the VOTE command: you will remember that the command VOTE by itself reads the "+vt.00" description file, and VOTE12 would read and execute the vote procedures for vote topic number 12 ("+vt.12"). Well, in the O-L version, the prompt states the change. Entering "?" will give the "+vt.00" file, whereas entering a topic number directly does the same thing as the old command+topic number. For example, instead of having to use VOTE12 as in the old command, the O-L version would require only "12" as input.

All tabulations are done through the OverLink module as well. The results, and the summary that you saw with the "V" command in the editor section is rerouted into the module. The "V" command in the editor section is no longer active. Immediately after the vote topic is selected, and the ballot made, the level "s" user will be placed into the "V" command of the editor section automatically.

Module #002 : TRIVIA Section (P1)
Files : SETUP.M002 - QLINK.M002 - QLINK.M002.A

This is the first of 2 trivia modules provided on the library disk. This module does not have to function as a trivia module however. The basic design behind this program is that a user is given an introduction to a questionaire and allowed to answer a number of questions, the results of which are added to an answer file. For description purposes, we will express this module as a trivia section.

Like many of the modules you see on the library disk, this is the first of modules which can contain "cloned" sub-sections. This means that if you ran just 5 sub-sections, then each sub-section would operate as if you were only using one section with no-subsections. The only difference between one section and multiple sections is that you have to go one extra step to creating a sub-sectioned module.

The trivia section can be considered "cheat-proof" by the way answers are stored. A user's name and ID# are saved every time the module is accessed, provided the user agrees to answer the questions. Thus, you can see if a user attempts to answer questions more than once by looking in the answer file for the user's name to occur more than once. The basic operation of the module works like this:

1. If there is more than one trivia section, then a master menu appears describing each of the trivia sub-sections. This file is named "+t/menu", and is a standard SEQ text file you create. The user then enters the number of the section to access. Entering "?" re-displays the main menu. Entering RETURN exits to the main BBS.

2. The introduction file for the trivia section is displayed. If there is only one section, this file will appear in place of the above menu. This file is named "+ti/xx", where xx is the trivia section number, from 1 to 25. If you use one section only, the section number defaults to 1. For example, the intro file for section 5 would be "+ti/05". This file should outline the nature of the questions the user is to answer.

3. The user at this point has 2 choices. The prompt "Continue?" will be displayed. If the user replies yes, then his name and ID# are added to the trivia answer file for that section. The user will then proceed to answer the questions. If the user replies no, then he will be returned to the main BBS without any change being made to the answer file. If the user is a level "s" user, he will be asked to see the answer file for that section, then he will be placed back into the previous menu. Note: if you use 1 section only, exiting the trivia section exits the module, and you are returned to the main BBS. If you have more than 1 section, then you are returned to the main menu.

4. The question file will now be displayed. All questions are stored in one file, and are separated by a delimiter character, which is the "@" character. This character is used by other modules too, so become familiar with it. As such, this character cannot appear in the questions themselves. Similarly, the "&" character cannot be used either. This character marks the end of the text file. Always end your file with the "&" character. Always place an "@" at the end of each question. A description of character delimiters will not be given. The following would be what a typical question file looks like:

```
What Is Your Name?@
What Is Your Quest?@
What Is Your Favorite Color?@
This Is The End. Bye!&
```

As you can see, the "@" is used after each question, and the "&" is used at the end of the file. Remember, only 70 questions maximum per section.

Because the screen clears after each answer and question, you can use a lot of color in your question files, provided you use the color version of the BBS.

After each question is displayed, a user is allowed to enter from 1 to 39 characters for an answer. If the user hits RETURN alone, the question file is aborted, and all answers made up to that point are saved to the answer file. If no questions were answered, the section is aborted altogether. But the user's name is still saved to the answer file to let you know that he attempted to answer the questions. Note that the question file is named "+tq/xx", where xx is the trivia section number. The format is like the intro file, except a "q" is in the file name rather than an "i".

Module #002 : TRIVIA Section (P2)

5. Once all questions are answered, the answers are saved to the answer file, which is named "+tr/xx", where xx is the trivia section number. Note that for each section, you must have an intro file, a question file, and a result (answer) file. The result file should be created with any starting text you want. When answers are stored, they are appended (added) to this file. Thus the file grows and grows, and if you are not careful, you could end up with a very large answer file. The module prevents any access to the trivia section if less than 10 blocks are free on the disk. The user will be notified of the low disk space and will be returned to the main menu (or the BBS). The answer only is saved to the file. If a user answered 5 questions, the following would be written to the answer file:

(User ID#) User Name

Answer 1
Answer 2
Answer 3
Answer 4
Answer 5

Make sure your questions are simple. Only 39 characters + return are allowed per answer, so multiple answer questions would not be a good idea. The module does not verify correct answers. This is your task. Use the other trivia module if you want the answers to be checked if they are right or wrong.

6. After the answers are written, the user will be returned to the main menu (or the BBS), unless that user is a level "s" user. If so, he will be asked to view the result file. The option to clear the answer file is also available. You will have to clear the file frequently if you have limited disk space.

Using The SETUP File:

100. V : Version of the module to modify. 0=color 1=ascii.
120. DF$: Destination file name.
300. Set the device/drive for writing to another disk.

Type LIST5000- to see the DATA statements that have to be set for this module. Most modules have their parameter data from lines 5000 onwards.

Lines 140-190 contain the text prompts for the 6 prompts this module uses.

5000. Connect Time Flag. Set to 0 if you want a user's connect time to stop during his stay in the trivia module. Set to 1 otherwise.
5005. Number of trivia sections. Select a value from 1 to 25.
5010/5020. Device and drive number for the main menu. If you use more than 1 section, then this is for which drive the "+t/menu" file appears on.
5030-5080. Device/drive numbers for each section. Each trivia section can have its question, result, and intro file on its own drive. This allows for systems with large disk capacity. The device and drive numbers are set in the same manner like the table the INSTALL DRIVES program uses, except that it is only for 25 drives, not 32. you CAN and should try to keep all sections on the same drive for readability.

Module #003 : Advertising Etc.
Files : SETUP.M003 - QLINK.M003 - QLINK.M003.A

This section has no pre-defined use. Originally, the first idea for this module was to use it for advertising BBS numbers. The concept of this section is fairly simply. Each sub-section of this module has a file, which grows as data is added to it. For example, when users compile a BBS numbers list, they will add the BBS name, number, etc. to this file. All data is appended for chronological ordering.

A user basically has 2 options when accessing a sub-section. The user may use (R) to read the data file, or use (A) to add to the data file. In either case, one or the other option will be accessed in turn until the user enters RETURN. The data file is a simple SEQ text file, that must initially contain at least one character. Generally, you should add a simple description of what the file contains, and what the user should add to the file if he should choose to do so.

This module sub-divides into 25 sub-sections in the same manner as the previous trivia module does. You may have up to 25 sub-sections, or just one sub-section. As with the trivia section, you will have to create a master menu if you want to have more than one section. This file is named "+a/menu", and the context of this file should follow along the lines of "+t/menu" in the trivia section. See module #2 docs for information on how the sub-section system works.

For each sub-section (or for just one section if chosen), there is only one file. This file is named "+++bbs.xx", where "xx" is the section number. For example, bulletin ad section 1 would be named "+++bbs.01". Always create this file intially with at least one character contained within it, to prevent the module from adding to a null file. From the main menu list, the user may enter "?" to re-display the section numbers and their descriptions given in the "+a/menu" file. If you have one section, you will proceed straight to section number 1. The user will then be given 2 options: (R) to read the ad file, or (A) to add to the file. Entering RETURN places you back to the main menu, or to the main BBS if only one section is used.

The READ option is fairly simple, so it won't be explained. The ADD option involves adding new text to the file. Depending on your needs, the sections could be used for adding BBS numbers to a numbers list, or having users add wanted/for sale ads to others. The choices are yours. When adding text to the file, you will be allowed to add only so much text at a time. Twenty blocks must be available on the disk drive for the current section to be able to add to the +++bbs.xx file. If you are using the color BBS, the module will allow you to enter text with either a color editor or a line editor. The ascii version only allows line entry. In either case, the editors are one-shot only, meaning you can only enter text as you go, and not re-edit upon exit.

When using the line editor, a RETURN alone on a line exits editing. The color editor uses STOP to exit. When text has been entered, the user may abort at this point. If not, the text will be added to the data file for that section. Then user will then be placed back to the point where he may re-read or re-add to the file. The remainder of this module is explained under the setup procedures.

The SETUP file. Most of the parameter data exists from lines 5000 onwards, with a few exceptions.

100. Set V=0 for the color module, or V=1 for the ascii module.
120. DF$: Destination file name.
140-190. These are the text prompts for this module.
300. You may change the destination device and drives numbers here.

5000. Connect Time Flag. Set to 0 for users' connect time to stop during the module access. Set to 1 for normal time.
5010. Number Of Ad Sections. A number from 1 to 25.
5020/5030. Device and drive number that the "+a/menu" file resides on. Ignore this for one section usage.
5040-5060. Access Levels. The ad module runs separate from the BBS in terms of access levels. You may set the level for each of the 25 sections. Any level may be used, from 1-9, or "s" for a SYSOPs only section. There are 25 numbers in the data, one for each section, ordered from left to right, top-down.

Module #003 : Advertising Etc. (P2)

5070-5090. Text Entry/Display Mode. Each section can have from 1 of 4 settings to it's mode of operation. The following values allowed are:

0: Line Entry, Non-Continuous. All text is entered as line input only. No color graphics are allowed. Non-continuous refers to the actual reading of the text with the (R) option. The "@" delimiter character is added to non-continuous files to allow the display to break at the point an "@" shows up. In line entry mode, the user will be prompted to hit RETURN to go to the next ad. With color mode section, the cursor will flash between ads and wait for the user to hit RETURN. The "@" character may NOT be used in text entry by a normal user. The module will prevent this from occuring. The level "s" user may use "@" however, when entering text. This allows for forced breakpoints in the data file. Note that the only way to break out of the ad file display is to use the Xon-Xoff character for list display abort. For example, the user enters "a" to abort during the file display, not between ads. Every file displayed in all modules of OverLink will be abortable using the standard list control keys, unless otherwise stated.

1: Color Entry, Non-Continuous. Color entry allows you to enter text in the same manner as the BBS color editor does. You do not have any editing capabilties however. When the number of characters entered exceeds the set limit (see limits below), the cursor will flash to let you know you have entered the limit, and you will have to hit RETURN at this point to save the text. Non-continuous mode works as described above.

128: Line Entry, Continuous. Same as 0, but with continuous mode enabled. This allows the file to be read without any stops. The "@" delimiter is not used in this case, and the user is free to use it in the text itself.

129: Color Entry, Continuous. Same as 1, but with continuous mode.

5100-5120. Line/Character Limits. For each section, you must set a pre-defined limit to the amount of text that a user may input. The maximum limit for color entry is 2048 characters. For line input, it is 50 lines. The number for each limit in the data statements will be either a character count, or a line count for the limit. Be sure that you align your mode values with these so that color modes align with character limits and line modes align with line limits. For example, the default sets all 25 sections for line mode with a maximum line input limit of 5. The SETUP file will split the number into hi/lo byte values, so that entering a value of 1560 for a character limit in color mode is valid.

5130-5150. Add Name Mode. You may choose to add the users name and ID# to each ad file, be it color or line. This will be added to the +bbs.xx file before the text is written to it. A value of 0 will add the user's name and ID# to the file, a value of 1 will suppress it.

5160-5210. Device/Drive Numbers. As with the trivia module, each section has it's own drive and device number. See the trivia module (#002) for details on drive table setups.

Module #004 : Bulletin Section Expander
Files : SETUP.M004 - QLINK.M004 - QLINK.M004.A

This module is provided as a faster more simple way to archive your bulletin section. If you have a lot of bulletins that you want to make as permanent fixtures on your BBS, then use this module and place them in one of it's sub-sections. It is possible to store an un-limited number of bulletins with this module. The program acts as a simple file reader, with no limit on how many bulletins can be accessed. There is no relative file for this module. As with modules 002/003, this one can be split up into 25 sub-sections. each with it's own disk drive.

When the user enters this module, the master menu will be displayed IF there is more than 1 section enabled. For 1 section use, this part will be skipped, and the user will proceed to section 1. The master menu is named "+b/menu", and follows the same guidelines as the previous 2 modules do. Entering "?" will re-display this file. Entering the section number will allow the user to proceed to the appropriate section. Entering RETURN will exit back to the main BBS.

Each section has it's own main menu, which is named "+b.xx.menu", where "xx" is the section number. For example, section 20 has a main menu name of "+b.20.menu". The main menu is a SEQ text file as are all bulletins. Because there is no structure to this BBS module, like there is with the bulletin section in the main BBS, you are free to divide and sub-divide your bulletin section at will, using text files for sub-sub-menus etc. The manual mentions how private text mode in the bulletin section works. This module operates the same way, except that you may have 25 "discrete" sections as well. Once in a section, "?" will display the main menu. RETURN will exit to the master menu, or to the main BBS in case you only use 1 section.

Bulletin names can be up to 10 characters long, and can use any characters you like, except for upper case. The bulletin names are set up so that one section's bulletins will not interfere with another's. That is, you can have a bulletin with the same name in different sections. Bulletins exist in the format "+b.xx.NAME", where "xx" is the section number, and "NAME" is the actual name of the bulletin. For example, a bulletin called "intro" in section 4 would be named on disk as "+b.04.intro". The important thing to note here is that unlike the BBS bulletin section, there is NO padding of the bulletin name with spaces to make it length 16.

The SETUP File.

100. Set V=0 to modify the color module, or to 1 to modify the ascii module.
120. DF$: Destination File Name.
140-150. Text Prompts For This Module.
260. Line to change destination drive.
5000. Number Of Bulletin Sections. A number from 1 to 25.
5010-5030. Access Levels. Set to 1-9 or "s" for each section.
5040-5060. Connect Time Flags. Set 0 for connect time to stop, 1 for it to run normal, during module access. Note that you must set connect timers for each section.
5070-5090. Display Main Menus. For each section, you may have the main menu file displayed as soon as the user enters the section. This is analogous to the user entering the section and typing "?". Use a value of 0 to display the opening menu, or 1 to disable it.
5100-5110. Device And Drive Number For The Master Menu. This is for the file "+b/menu". For one section only, this file is not used, nor is this setting.
5120-5170. Device And Drive Numbers For Each Section. See module #002 for details.

NOTE: A late addition was made to this module. If a user appends the asterisk (*) character to the bulletin name, the BBS will initiate a Punter file transfer of that bulletin. There is no provision for Xmodem bulletin transfers. For example, if a user normally enters BBSNUMS as a bulletin to read, BBSNUMS* would cause the file to be downloaded as a single file Punter transfer, rather than a normal bulletin read. This feature can be handy for a user who wants to archive your bulletin section.

DarkStar BBS V3.0 - The OverLink - (C) 1986 D.S.S.

Module #005 : On-Line File Copier (LOCAL MOD)
Files : OLINK.M005/A

This is an exact duplicate of the 2 drive file copier on disk #2 of the BBS, named COPY.FILE, with one exception. The source and destination device numbers can range from 6 to 19, not just 8 or 9.

See the BBS manual under COPY.FILE for more information on how to use this program. Note that this module will not be executed if the user on-line is not the SYSOP (ID#0). Although the program module will load if the command is entered, the program will abort back to the main BBS if any other ID# is found. This module runs in local mode, and cannot be executed from remote, this being why the SYSOP-only feature was added.

Since this program must store the files to be copied in memory, it was necessary to find an area that wasn't in use. Unfortunately, the BBS uses virtually all of memory, so the only way to make room was to temporarily use part of the ISFS data tables. The user name table is used as the files buffer, which means that the users are erased from memory during the operation of the copier. When you enter "X" to exit the copier program, the "Users." section of the ISFS setup routine will be executed to restore the users to memory. Once complete, you will be returned back to the main BBS.

As you can see, there is only one file for this module. The /A ending on a module file name indicates that it is to be used with either BBS. It contains auto-detect routines to determine which BBS is running. There is also no setup file for this module. It is a straight copy from the COPY.FILE program with the aforementioned modifications.

This file copier allows you to copy from drives that have a device value as low as 6 or as high as 19. You CAN have a disk drive with device 4-7, if you do not have a printer hooked up.

DarkStar BBS V3.0 - The OverLink - (C) 1986 D.S.S.

Module #006 : Relative File Maintenance Manager
Files : QLINK.M006 - QLINK.M006.A

You may have read in the manual under relative file editing that some fields in the records of the files could not be changed by any means other than a disk doctor. This program allows you to change every byte of every record of all 4 system relative files (NOTE: Please read the section in the BBS manual under record structure to understand how to change each field in the records of the files). This program is restricted to level "s" users only. With any of the level "s" OverLinks, it's best to make the numbers of the OverLink command (or the command value using the CWI) unreferenced to keep the normal users from knowing these sysop commands exist.

The end of this documentation summarizes all the field descriptions and definitions for all 4 files. The program consists of 2 menus. The first menu consists of the relative file selection. Due to relative file "bugs", the files will be opened and closed for each record operation. Selecting the file will place you in the second menu. The second menu consists of 4 basic operations: ADD/DELETE/MODIFY/SUMMARY. Each command loops around, waiting for records to work with until you enter return to exit to the main menu.

When adding a record, you must enter all data for each field, or the command is aborted. When using the modify command, the current field values are echoed, and new input is required after that, UNLESS you enter a return by itself. By entering a return at any prompt using the modify command, that field will remain unchanged, and you will be able to continue onto the next field. For delete, modify, and add, you will be always given a second chance to confirm your options, in case you make a mistake.

Since all operations are done on the disk level only, it is necessary to reset the data to memory. So when you exit the program, the ISFS tables will reset. Some records are stored in upper case format. This does not mean you have to hold the shift key down. All input is properly converted regardless of your input. Sometimes the input allows you to go beyond the norm. That is, it is possible to add erroneous data as input. Memory was at a premium, and it was not possible to verify all input to be correct. Just like with using a disk doctor, you can add the wrong input. If this happens, just abort or re-edit. The following is a list of all fields for the records of the 4 files, with descriptions of the abbreviated 3 character prompts, and the input expected.

Relative File #1: Messages

LVL: Message access level. Input: 1-9.
RF#: Message reference number. Input: 1-65535.
FID: User ID# of message writer. Input: 0-255.
TID: User ID# of who the message is addressed to. Input: 0-255.
PVT: Message public/private status. Input: 0 for public, 1 for private.
ALL: ALL status. Input: 0 if message addressed to all, 1 if addressed to a user.
CAT: Message category number. Input: 0-9 (or the number of categories you have, minus 1).
SUB: Message subject. Input: 1-25 characters.
DAT: Date message was posted. Input: DDMMYYHHMMx. DD: Day MM: Month YY: Year HH: Hour MM: Minute x: "a" - am or "p" for pm. You must make sure the length is 11 characters. For example, Jan. 5, 1987, 12:34 am is "0501871234a".
FWD: Forward flag. Input: 0 if message not forwarded, 1 if it is.
FWI: Forward ID#. Input: 0-255, the ID# of the user who forwarded the message.
REP: Replied counter. Input: 0-255, the number of replies made to the message.
RRF: Replied reference number. Input: 1-65535, the reference # of the message that was replied to, OR a 0 to indicate that the message was not a reply to any other message.

DarkStar BBS V3.0 - The OverLink - (C) 1986 D.S.S.

Module #006 : Relative File Maintenance Manager (P2)

Relative File #2: Users

LVL: User access level. Input: 1-9, or "s" for sysop level.
NAM: User name. Input: 1-25 alphabetical characters, plus spaces or the "," character.
PWD: User password. Input: 1-8 characters.
DAT: Date of last logon. Input: See DAT under message relative file description.
TLI: User time limit. Input: 1 to 59 minutes.
TUS: Connect time used. Input: 0 to the user time limit byte above (This byte applies to daily time limit only).
TST: Time status. Input: "YN" binary input, consult manual under the EDIT.USERS program for details.
RDA: Relative directory access. Input: the 8 "Y"/"N" binary input string. See EDIT.USERS and CREATE.CNF.
PDA: Physical directory access. Input: Same as RDA.
SAC: System accesses. Input: Same as RDA and PDA.
RF#: Hi message reference number of last message read. Input: 0-65535.
TUP: Total Uploads. Input: 0-65635.
TDN: Total downloads. Input: 0-65535.
MTO: Messages written to user. Input: 0-65535.
MFR: Messages from user. Input: 0-65535.
TLG: Total logons. Input: 0-65535. NOTE: On page 58 of the manual, this byte was erroneously set to the definition of the MTO byte. The TLG definition is the correct one.

Relative File #3: Programs

DIR: Directory number. Input: 0-7.
FID: ID# of user who uploaded the program. Input: 0-255.
NAM: File name. Input: 1-16 characters.
PWD: File password. Input: 1 to 8 characters for a password, OR the left arrow symbol for no password ("aaaaaaaa" will indicate no password. The left arrow key places 8 "a"s in the field).
DAC: Download accesses. Input: 0-999.
SIZ: File size. Input: 0-999.
TYP: File type. Input: "p", "s", or "u".

Relative File #4: Bulletins

LVL: Access level of bulletin. Input: 1-9.
L#1: Link number 1. Input: 0-254. See EDIT.BULLETINS.
L#2: Link number 2. Input: 0-255. See EDIT.BULLETINS.
NAM: Bulletin name, Input: 1-8 alphanumeric characters, plus spaces.
SUB: Bulletin description. Input: 1-25 characters.

DarkStar BBS V3.0 - The OverLink - (C) 1986 D.S.S.

Module #007 : Terminal Link V3
Files : OLINK.M007 - OLINK.M007.A

This program provides a terminal linkup so that you can log onto other BBS systems while remaining in the BBS operating environment. This module's functions are a subset of those in DarkTerm 4.0. The autodialing feature could not be provided, as there is no place to load a modem file into. You will have to manually dial out, or send out AT dialing commands manually for smartmodems. The function list here has the same definitions as those of DarkTerm 4.0. Therefore, only the command summary is given here. If you want further details on the command functions, you should consult the DARKTERM DOCS file. If you know how to use DarkTerm 4.0, you will have no problem with this terminal link. If you don't, then read those docs, as they will not be re-printed here.

Status Line:

```
12:34a  R:01 C:01 6144 ASCII COM B:H:I:U
  1       2      3    4    5  6 7 8 9
```

1: System clock.
2: Cursor row/column.
3: Buffer count. There is a 6K buffer in the term, not exactly the greatest, but when you still have a BBS in memory, you do what you can.
4: Terminal status. Will read ASCII or COLOR (as opposed to A and C in DT4).
5: Command mode indicator. Reads COM for command mode, or blank otherwise.
6: Buffer open/closed status.
7: Hide output status.
8: Modem I/O inhibit status.
9: Upper case lock status.

Functions:

```
Shift Run/Stop: Toggle color/ascii mode.
Shift Return  : Toggle command mode.
SPACE         : Pause/resume output.
STOP          : Abort output.
C= B          : Buffer Options (<P> and <E> not available).
C= C          : Change terminal options:
                P: Toggle protocol B: Change block size D: change disk device number #: change disk drive #.
C= D          : Disk commands.
C= F          : File options (only <R> and <T> supported).
C= H          : Hide screen output.
C= I          : Toggle modem I/O inhibit.
C= L          : Load character set ("rom" for rom set).
C= M          : Modem options (2400 baud not supported).
C= O          : Toggle buffer open/closed.
C= S          : Set terminal colors.
C= T          : File transfer options.
C= U          : Toggle upper case lock.
C= X          : Clear buffer.
C= (Pound)    : Send ascii delete.
```

DarkStar BBS V3.0 - The OverLink - (C) 1986 D.S.S.

Module #007 : Terminal Link V3 (P2)

NOTE: Run/Stop Restore is still tied into the BBS, so a computer cold start will occur if you use it. This means that you should be very careful about not getting disk errors, or you'll have to re-boot everything.

This module can be used by the sysop only (ID#0), and since it is a local mode module, you should not make public notice that this module is present.

The C= X exit function will do 2 things. First, the program looks for the text file on the OverLink drive. This is the same text file you input when setting up the system initially. The terminal program overwrites most of the text prompt storage area, so it must be put back into memory. The text file must be named "+bbs/txt" and be put on the OverLink drive. After this is loaded back into memory, the IFSF tables will reset. The terminal uses the data table memory for the 6K buffer and for the screen swap storage area, so this was necessary as well. After this, you will be returned to the main command prompt.

OverLink modules like this one stress the importance of having a multi-drive system. The overhead here is 32 or 34 blocks for the module, and that used by the text file, which will be at least 12 or so blocks. The OverLink was not designed for a uni-drive system, and you will realize this when more modules are created.

Module #008 : File Section Xpander (P1)
Files : SETUP.M008 - OLINK.M008 - OLINK.M008.A - ++-+M008 - +-+M008.A - LINK.M008.C - LINK.M008.A

This is one of the largest modules on the library disk. It serves to supplement the standard file section by adding several new commands, as well as to re-organize the current commands and directory structure of the existing file section. This module will link into the main BBS and take control of several of the main BBS input/output routines. Because of it's size, it is located in a different area of memory, which happens to be the user data table. During entry to this module, the load time will typically run about 16 seconds on a 1541. When exiting the file section, the BBS needs to reset the user name table, which is one of the 5 steps in setting up the ISFS data. This setup time depends on the amount of users you have. Typical times range from about 25 seconds (no users) to 45 seconds (255 users) on a 1541. If you are using an IEEE drive as the system drive, you can expect a significant decrease in reset and load times.

The new features of this section include:

-Xmodem Sum And CRC Protocols, Both ASCII And Binary With Auto-Pad Stripping.
-Optional Upload/Download Logs.
-Restricted Time Periods For Uploading/Downloading On Any Directory.
-Additional 8 Physical Directories.
-Multi-Downloading From Relative Directories.
-Forced Message Writing/Download Option.
-Information Files.
-The Ability To Read A File From A Directory.
-New Combined Directory Access Structure.

All these features will be explained in detail according to the new commands that use them. Note that the new module command definitions replace those used by the old file section.

Combined Directory Structure.

With the old file section, the physical and relative directories each have their own commands, and the listing, uploading/downloading commands all worked with 2 distinct directory formats. With this new section, both formats are grouped into one combined directory list. The relative directories will occupy the lowest directory numbers, and the physical directories will occupy the highest directory numbers. Certain commands for physical and relative storage will be shared with this new format. For example, say you have 6 relative directories and 5 physical directories. The new format organizes the directories as follows:

Directory	0	:	Relative	directory	0	Directory	6	: Physical	directory 0
"	1	:	"	"	1	"	7	: "	" 1
"	2	:	"	"	2	"	8	: "	" 2
"	3	:	"	"	3	"	9	: "	" 3
"	4	:	"	"	4	"	10	: "	" 4
"	5	:	"	"	5				

Module #008 : File Section Xpander (P2)

The module will set the directory limit as the sum of the number of physical and relative directories, in this case being 11. The base directory is still 0, not 1. The module will be able to recognize the directory as physical or relative and assign the file commands appropriately. If you have no physical or no relative directories, the module will still operate with just one format as well as it would with two.

Xpander Quirks.

If a user drops carrier in the file xpander, the system will not disable auto-answer if you use a smart modem. During the reset period as the user table is read back in, a caller could log into the system. This is not a flaw. The carrier loss is detected before the system can get back to the main menu. Thus, anyone on the system during the user reset will be disconnected immediately without ever gaining real access to the system.

Xmodem downloads will time out to the file section main command prompt after 6 seconds if the user has not hit a key.

There is no multiple file upload command for relative directories. It is not possible to add multiple records to a relative directory at one time. Similarly, it is not possible to update download accessed for 50 files at one time. So an MRD command will not update the download access counter in the files downloaded.

It is a good idea to not disable the old file section. The idea behind this is based on a multiple sysop system. Say, for example, you had files that assistant sysops wanted to upload or download files that they do not want to let the users know about. If you have an upload or download log enabled, and have them set for user access, then those files will update the logs after the file transfer. The only way to get around this is to go to the old file section, perform the transfer, then exit. The logs do not update. Think of the old file section as a "quick" access file section for priveledged users. You should change the old file section command definition to a value that normal users will not be aware of. The Xpander resets the old file section to it's own mode of operation once a user exits from it.

The help file for the file section is still used by the new one. You are responsible for adding the new commands into the help file yourself, and also removing/modifying the old ones in the "+he" text file.

When using the D command to display the available directories, up to 16 descriptions are displayed. The old file section shared 8 descriptions with both directory formats. Physical directory 0 shared the same description with relative directory 0. The text prompts for the new file section include the 8 directory descriptions for the 8 physical directories. The 8 in the standard text file are now solely for the relative directories. You may have to change those prompts in the text file if necessary.

Also, the descriptions do NOT appear anywhere except when using the D command. This gives a cleaner appearance to the file section. You will not see the descriptions when typing U or D etc..

Module #008 : File Section Xpander (P3)

File Expansion Commands:

(D) Display Available Directories: Entering D alone will display the directories available and their descriptions. If a user does not have access to a directory, then it will not be displayed. This is a nice feature not found in the old file section. The user will be asked for a new directory number. When selected, the new directory becomes the current directory. Entering D followed by a directory number will set the current directory immediately without displaying the directory descriptions. If you have the "show-directory" flag enabled, the current directory number will appear next to the main file section command prompt. Attempting to switch to a restricted directory will not be allowed. Note that most commands can switch the current directory automatically without using the D command at all.

(PRO) Set Xmodem Protocol : The default Xmodem protocol must be set by you with the SETUP file. You may set it to one of the 4 values set with this command, and that setting will be effective as soon as the user enters this module. The protocols available are:

-Sum Mode, Binary
-Sum Mode, Ascii
-CRC Mode, Binary
-CRC Mode, Ascii

Details of Xmodem transfer can be found on most bulletin board systems. A detailed discussion of what Xmodem is will not be given.

This module uses automatic Xmodem file transfer. That is, the last block, which usually contains padding characters to make the block 128 bytes, is handled in a special way. This padding (usually CTRL-Z else 0) is written to disk along with the file. This can create problems with text files and binary files. The module will double-buffer the data so that the padding is removed before the block is written to disk. During an Xmodem transfer, you will see that downloads use 2 128 byte on-screen buffers. The upload command does not need to double-buffer, so only one screen buffer will appear.

As with Punter transfers, you may abort the transfer with the STOP key at any time. Unlike Punter transfers, corrupt files are removed from the disk if they should occur. This happens when a user disconnects during an upload. The file would normally be left on disk during a Punter transfer as a partial program. Xmodem will delete this file in the event of carrier loss, so that only complete files are uploaded. Note that Xmodem also has a timeout feature that will cause transmissions to be aborted in the event of bad data or line noise that can't be corrected. CTRL-X repeatedly entered after a manual abort will usually stop an Xmodem transfer.

(BBS) Return To Main BBS : It is very likely that a user may hit RETURN one too many times. This would have the effect of reseting the user data and dumping the user back to the main BBS. To remedy this, a separate exit command is used. If a user were to hit RETURN now at the main file command prompt, a prompt would be displayed to indicate how to get back to the main BBS.

(T) Toggle Directory Banks : This allows you to use expanded physical directories. Using the second bank, you can use up to 8 more disk drives for directory storage. But there is a catch. The new bank will act just like the old bank, except that it uses a new physical directory drive table created with the SETUP file. Think of the second bank as an extension to the current directory. For example, physical directory 0 would have 280 files of storage if 2 1541s are used, one for bank 1, and one for bank 2. Be careful of the bank distribution. If you have 10 physical directories you can place 8 of them in bank 1, and 2 in bank 2. But the physical directory size set with CREATE.CNF will be set to 8, so that the system will think that bank 2 will have 8 directories, when in fact you only would have 2. You then must set directories 2-7 in bank 2 to those of directories 2-7 in bank 1, so that they will not point to null directories.

Module #008 : File Section Xpander (P4)

A more logical way to handle the situation of 10 physical directories would be to group 5 in directory bank 1, and 5 in bank 2, then set the physical directory size to 5 in CREATE.CNF. This will not give any null directories. If you used 10 1541 drives for the directories, then each physical directory would have 288 files and 1328 blocks, split up between the 2 banks. The T command is really intended for a large system. If you don't want to use it, then set the length byte to 0 for the T command in the SETUP file. Note that T does not toggle the relative directories. Those will always remains as is. The module will also reset the bank to one upon module exit, in case it was left as bank 2. This ensures that the main BBS is using the correct physical directory drive table that was created with CREATE.CNF.

(DIR) List Disk Directory : The C (Physical) and LIST (Relative) commands are now combined to this command. The module will be able to list either a physical or relative directory with this one command. The function of this command is that of both C and LIST. See the BBS V3.0 manual for details. Note that adding a directory number after the command will switch the current directory and then list the directory for the new current directory. For example, using DIR6 is the same as using D6, then DIR. This command replaces both LIST and C.

(DOW) Punter Download : This command replaces both the DOW and D command of the old section. Once again, the 2 commands are combined. After a download, the download log may or may not be updated (see DL).

(UPL) Punter Upload : This command replaces both the UPL and U command of the old section. After the file transfer is complete, the upload log may be updated (See UL). Both DOW and UPL can change to a new directory by adding the new directory number after the command.

(MD) Multiple File Download (Punter Only) : This command replaces MD in the old section. It also adds the multiple download option to the relative directories as well. When downloading from a relative directory, the stored file information is derived solely from the records in the +PR relative file only. You MUST be sure that all files match to those on the disk itself, or the BBS WILL hang somewhere. The download access counters in the records will NOT update after a multi-rel download. Be aware of this fact. You will have to use the download log as an indicator instead of the counters if you need to know how many times a file has been downloaded. The download log itself will update after either type of multi-download, in a specific way that distinguishes it from single downloads (see DL). The mode of operation for the multi-rel download is the same as that of the physical method. They only differ in where they each get their directory information from.

(MU) Multiple File Upload (Punter Only) : This command replaces the MU command in the old section. There is one major difference in the new version. As each file name is received during a transfer, it is checked to see if it already exists on disk. If it does, then the file transfer is aborted. Thus, you now have an MU that can NOT crash the BBS. The module will check for bad characters, insufficient disk space, and file existence. You may now allow anyone to access this command without worrying about losing precious files. Note that there still is no multi-rel upload command for reasons explained earlier. Add a directory number to MU or MD to switch the current directory. The Xmodem commands below can also switch the current directory. Note: When setting up user access and directory setup for the file section, interpret the setup with respect to the old file section commands. It will make understanding the new file section easier. For example, the DOW command has 2 separate user access bits in their record, one for physical, and one for relative. DOW in the new section will interpret this based on both the DOW and D commands of the old file section.

Module #008 : File Section Xpander (P5)

(DL) Download Log :
(UL) Upload Log :

Both the download and upload logs have the same method of operation, except that one is used in conjunction with uploading, the other for downloading. They will be explained as one unit. Both logs are optional. You may run one or the other, or both. Similarly, you can have either (or both) logs be user accessible or sysop level ("s") accessible only. The SETUP file will allow for your own configuration.

Upload and download logs will update when using the UPL, DOW, XDOW, XUPL, MD, and MU commands, provided the logs are enabled. When a single file upload or download has completed, the user will be notified that the log is being updated. You will have to create the log files initially, so that at least one character is in it for the upload and download data to append to it. Both logs have their own device and drive numbers for the file locations, which you specify in the SETUP file. The upload log is named "+u.log" and the download log is named "+d.log". Once again, these are simple SEQ text data files.

For single up/down-load updates, the user's ID#, the name of the file, and the date transfered are stored in one line in the format: FILE NAME / USER ID# / DATE. The date does not include the time of day, only the date is stored. Their is only one log for uploads and one for downloads. All directories, physical and relative, share the same upload and download logs. If you have files that you don't want to appear on the logs, then you will have to hide the files under new file names, or allow the elite group of users to access the original file section. Or, you can disable the logs from general user access by restricting the UL/DL commands to level "s" users only. In this case, when a normal user enters UL or DL, the BBS will ignore the command.

For multiple file uploads and downloads, all files transferred will be written to the log as one unit. The way to tell if a multiple file transfer has occured is to examine the file for the lack of an ID# and date next to a group of files. A multiple file update to either log writes the user ID# and date only to the FIRST file, and not the remaining files that were transfered. Thus, you would get a format that looks like this:

FILE #1 NAME / USER ID# / DATE
FILE #2 NAME
FILE #3 NAME
FILE #4 NAME
Etc....

When the next file transfer occurs, the user ID# and date will show up again. Upload and download logs can be quite large on a BBS that is file transfer intensive. Using the user ID# and not the user name, and using the user ID# and date only once in multiple file transfers saves quite a bit of space and update time. The UL and DL commands do not have the option to clear the logs as does the LOG command in the user section. You will have to use the edit file and DOS wedge in the editor section to manually clear the logs while on-line.

If you disable either or both logs, then you do not have to have the files on disk. The BBS will not update anything after a file transfer, and will perform in the same manner as the old file section. The commands of the new file section are provided for the SYSOP who wants to take advantage of a file section that has large amounts of disk space to offer.

Module #008 : File Section Xpander (P6)

(XDOW) Xmodem Download : This command allows for single file Xmodem transfers. It is important that both the terminal and BBS use the same transfer protocols when up/down-loading (ie. CRC to CRC, and SUM to SUM, not CRC to SUM or SUM to CRC). If the protocols are incompatible, or if the user attempts to use Punter transfer, the BBS will time out after a dozen seconds or so, and abort the command. As mentioned before, a manual abort of the Xmodem transfer can be accomplished by hitting STOP and then waiting for the terminal/BBS user to enter CTRL-X enough times for the other system to recognize the cancel request. There are no multiple file transfers with Xmodem protocol. It should also be noted how the ASCII transfer of Xmodem works. For downloads the file stored on disk must be a PET ASCII file. If the file is stored as ASCII, you should select a binary tranfer, so that the data will not be translated. If the file is PET ASCII, it will be converted to ASCII on the fly and sent to the terminal-end user as ASCII.

(XUPL) Xmodem Upload : This command is similar to XDOW, but applies to single file uploading. When performing an ASCII upload, the file sent must be an ASCII file. If the file is PET ASCII, send it using binary protocol. If the file to send is an ASCII text file, send it using ASCII protocol, and it will translated to PET ASCII on the fly, and stored on disk as PET ASCII. This is important to note. For Xmodem ASCII transfers, the files stored on disk should ALWAYS exist as PET ASCII files. The ASCII protocol is used to translate TO ASCII for downloading, so that the user gets ASCII if he wants it, and TO PET ASCII for uploading, so that the user's ASCII file he is uploading will be saved as a standard PET ASCII file on the BBS.

(DEL) Delete A File : This command replaces the DEL command in the old file section, plus adds one extra feature. This extra option is the ability to kill files straight from the physical directories, something that was lacking in the old file section. When using DEL on physical directories, you can set the priority in the SETUP file so that only you, the SYSOP, can use the command, level "s" users can use the command, or no one can use the command, in which case it is disabled. With DEL on physical directories, you can avoid having to use the DOS wedge in the editor section. By adding a directory number to the end of the DEL command, you can switch the current directory, in the same way as the aforementioned commands.

(F) Free Disk Space : This command merely prints the blocks free on any disk directory, in case a user wants to know. This command, as well as I and R below, also allow you to switch directories by adding a directory number to the command.

(I) Information File : One of the features of a Punter BBS system, among others, is the ability to have long directory listings that contain all sorts of information for each file in a directory, along with a short one line description. When you have something like that, then you are dealing with large records, which mean slower access, harder maintenance, and large relative file size. The approach taken with the I command is to store only important information for each directory in a SEQ file, and leave the non-descrip files alone. Each directory can have it's own description file, or you may disable an information file for any directory using the SETUP file. You are free to put anything you want in an information file, in any style. It's nothing more than a bulletin that appears in the file section. If you only want one information file to describe all your directories, disable all other information files in the other directories, and use just one information file on one directory of your choice. When an information file is inactive, the request for information on that directory will be ignored, as if the command never existed. Information files exist in 2 file name formats, one for physical directories, and one for relative directories. "+r.x.inf" is the format for relative directories, and "+p.x.inf" for physical directories, where "x" is the directory number from 0-7. For example, the information file for relative directory 3 would be "+r.3.inf".

Module #008 : File Section Xpander (P7)

(R) Read A File : If you are a member of CompuServe, you may be aware of the fact that you can read a file when browsing a data library. The R command allows you to read a file from any directory, and type does not matter. You can read a program file, as well as a sequential or user file. You cannot buffer a program file with the R command and save the buffer, run it, and expect it to work. The reason being is that null bytes (0's) are converted to spaces (chr$(32)) when any file is read throught the BBS.

(UD) Upload / Download Status : This is the same command as the UD command in the old file section. See the BBS V3.0 manual for details.

User Accessibility.

With the new commands in the module being added, a way had to be designed to allow the user access bits in the user's record to work with these new commands. The following table summarizes user accesses for the new commands. Refer to the BBS V3.0 manual for user access definitions.

New Command	Directory Format	Old Section Command Access Equivalence
PRO	R/P	None/C,D,U
BBS	-/-	None/None
T	-/P	-/C,D,U
UL	R/P	SETUP file sets access
DL	R/P	SETUP file sets access
I	R/P	+$DOW/C,D,U
R	R/P	+$DOW/C,D,U
F	R/P	$None/C,D,U
UD	-/-	UD/UD
D	R/P	$None/C,D,U
DOW	R/P	+$DOW/D
UPL	R/P	+$None/U
XDOW	R/P	+$DOW/D
XUPL	R/P	+$None/U
DIR	R/P	$None/C
DEL	R/P	$DEL/SETUP file sets access
MD	R/P	+$MD/C,D,U,MD
MU	-/P	+$-/C,D,U,MU

As an example, MD shows that it exists both in physical (P) and relative (R) formats, the relative MD format requires access to the MD command, set in the user access byte, whereas the MD physical command requires MD access AND requires physical directory access, which is denoted above by the commands C,D,U. C,D,U are the standard physical directory access commands. Where none is mentioned, this means that only standard file section access is required. Where "-" appears, it means that there is no analogy for the directory format (ie. MU has no relative format version). Where "$" appears next to the equivalence column, this means that the command requires additional access based on the individual access to each directory. For example, if P DIR 2 is restricted, all commands relating to it are as well. Where a "+" appears, this indicates a command based on upload/download restriction times (See next page). A few commands have access set in the SETUP file for this module. This will be explained later.

Module #008 : File Section Xpander (PB)

File Transfer Restricted Time Periods.

Each directory, physical or relative, can have it's own restricted time period. Uploading has a restricted time period, as well as downloading. The 2 are distinct. In the SETUP file, there are 64 bytes to set for time period restrictions. 32 are for uploading, 32 for downloading. Of the 32 for each, 16 are for the start time of the period, 16 are for the end time. Of the 16 start times and end times, 8 are for physical directories, 8 are for relative directories. This can be summarized below:

```
64 Time Values : 32 Upload   : 16 Start Time : 8 Physical
                                               8 Relative
                             16 Stop  Time : 8 Physical
                                             Relative
                 32 Download : 16 Start Time : 8 Physical
                                               8 Relative
                             16 Stop  Time : 8 Physical
                                             8 Relative
```

Each value represents an hour of the day. Thus, a value can be from 0 to 24. The numbers have the following meanings:

```
0     : Time restrictions disabled.
1-11  : 1 Am to 11 Am
12    : 12 Pm (Noon)
13-23 : 1 Pm To 11 Pm
24    : 12 Am (Midnight)
```

The start time and stop time have the following relationship:

-The ending hour represents the hour at which time the restricted period ends. The starting time represents the time at which the restricted period starts. For example, a start time of 11 and an end time or 15 will result in a restricted period from 11 Am UP TO 3 Pm (Time values are military: 1500h is 3 Pm). This means that at 3 Pm, the period ends. This does not work like TIMELOK, where 3pm would be an inclusive hour. 2:59.59 is the last second in which the restricted period exists.
-A start value or 0 AND a stop value of 0 disables the restricted time period.
-A start value equal to the end value (But NOT 0) cause the period to run a full non-stop 24 hours.

Examples: The numbers given are the start, then the stop values.

15-14 : 3 Pm - 2 Pm / 1-1 : 1 Am - 1 Am (24 hours) / 0-0 : Period disabled / 24-12 : Midnight - Noon
24-1 : Midnight - 1 Am / 1-23 : 1 Am - 11 Pm / 24-24 : Midnight - Midnight (24 Hours)

Module #008 : File Section Xpander (P9)

SETUP Procedures.

The SETUP for this module is quite different than that of other modules. There are 2 files to the Xpander module. The file OLINK.M008 is the boot file for the module. This is the file OverLink loads on the BBS, then executes. This file will check the user's file section access, and verify the message write/download feature (see below), and then proceed to load in the main module if the user has access to the file section. This main module is named either +-+M008 or +-+M008.A. The files LINK.M008.A and LINK.M008.C are the linking programs to create the modified modules for the file section. Both SETUP.M008 and LINK.M008.C are part of the setup procedures. The 2 module files +-+M008 and OLINK.M008 will go onto the OverLink disk on your BBS AFTER it is modified. In short form, this is the way you make the OverLink module for the expanded file section:

1. LOAD, LIST, Modify, then SAVE the SETUP file on your library disk backup copy. Make sure ALL the module #008 files are on drive 8:0. This is IMPORTANT. Both LINK files, OLINK files, +-+M008 files, and the SETUP file must be on unit 8:0.

2. RUN the modified SETUP file. During this step, two temporary data files are written to the disk that contain the custom data you set in the SETUP file. Once those 2 files are created, the LINK file will be loaded on top of the SETUP file, hence replacing it. It will be run, at which point it will load the OLINK and +-+M008 files into memory. Then those 2 files on disk are removed, and the files in memory are merged with the data files on disk to form the new modules.

3. The data files will be scratched from the disk. During this process, the following willl be displayed to let you know that the process is operating correctly:

```
Linking File 1...+
Linking File 2...+
Removing Data Files...
Done!
```

4. You may now transfer the new OLINK and +-+M008 files to your OverLink drive. If you are running ASCII, then the modules are named +-+M008.A and OLINK.M008.A. Otherwise, they are as above, without the .A suffix. The OLINK file must be renamed to the OverLink command word you are using (ie. for "FSECT", name OLINK.M008 as "+-FSECT". The +-+M008 file must NOT be renamed. Put it on the disk as it is named, so that the secondary load will find the right file. If you are not using the CWI, then you will have to assign the correct module number to it (ie. "+gc,15" for CBBS O-L module #15). You do NOT have to put the LINK or SETUP files on your O-L drive. Those are part of the setup procedures only.

The next page(s) will describe how to go about modifying the SETUP file itself. It is a fairly lengthy process, similar to setting up the CREATE.CNF file.

Module #008 : File Section Xpander (P10)

The SETUP File : Line numbers are given next to each parameter to change.

1070-1330. These are the 22 prompt strings for this module. String lengths can only be up to 255 characters, so be aware that you can't concatenate long paragraphs into one string. In the case of prompt string A2$, the protocol select string could not be represented in one physical line number, so it was necessary to concatenate A2$ a few times. If for some reason, your string length were to exceed 255, then you will have to modify the program to write out the extra string variables that hold the extra length. For the modest BASIC programmer, this shouldn't be too difficult. Remember that B4$-C1$ are the 8 PHYSICAL directory descriptions. Make a trial run of the file xpander, until you are sure of where each prompt string appears, before you make any drastic modifications to the text strings.

1350. Version Of The Module To Modify : Set to 0 for color, 1 for ascii. Color uses the files OLINK.M008, LINK.M008.C, and +-+M008. The ascii version use OLINK.M008.A, LINK.M008.A, and +-+M008.A.

1360. Protocol Default : This is the default Xmodem protocol that will be enabled whenever the user enters the module. It may be change with PRO at any time. Set to:

0: SUM, Binary 1: CRC, Binary 128: SUM, ASCII 129: CRC, ASCII

1370. Carrier Signal Status : Set to 0 for normal carrier detect logic, set to 1 for inverted carrier detect. See The intro to BBS V3.1 at the start of this manual for an explanation.

1380. Display Directory Flag : Set to 0 for the current directory number to show next to the main file command prompt. Set to 1 to hide the current directory number.

1400-1570. These are the 18 command definitions for the new file section. They must be 4 characters long, padded with blanks if necessary. These definitions take priority over the file commands in CREATE.COM, unless you access the old file section, where the old command structure still exists.

1590-1600. Command Lengths. The command strings you make above must have their lengths set here. This differs from the CREATE.COM file, where the command strings and their lengths were put side by side. Make sure you align the length bytes with the strings. For example, the 5th value in the first set of DATA statements is a 2, which is the length for the 5th command, which is set as "DL ". For some commands, like T, that you don't want to use, set the length byte to a value of 0.

1620-1630. I Command Display Flags. For the information (I) command, you can enable or disable the file display for any of the 16 directories. The first set of 8 "Y"s is for the relative directories, the second set is for the physical directories. If you want the I command to be active, use a "y" for that directory, otherwise use an "n". For example, "ynynynyn" enables I for directories 0,2,4, and 6, but not for directories 1,3,5, and 7.

1650-1660/1680-1690/1710-1720. Border/Background/Text Colors : 11 of the new file section commands use their own colors accessed through this small color table. The old commands, which have been combined into the new section, will still access the old color table. See CREATE.COL for details on color table setup. This table is exactly the same, except that there are only 11 values. The command correspondence is:

1: PRO	2: BBS	3: T	4: UL	5: DL	6: I
7: R	8: F	9: D	10: XDOW	11: XUPL	

1740. DEL (Physical) Command Status : Set to 0 for level "s" and SYSOP access, set to 1 for SYSOP ONLY access, set to 2 to disable DEL from deleting files on physical directories.

1750. Upload Log Status : Set to 0 for general user access, set to 1 for level "s" only access, set to 2 to disable the upload log entirely.

1760. Download Log Status : Same as upload status above, but applies to download log.

Module 8008 : File Section Xpander (P11)

The SETUP File : Line numbers are given next to each parameter to change.

1780-1790. Upload Log/Download Log Drives : The first value in line 1780 is the device that the upload log resides on, the second is for the download log. Line 1790 is the drive numbers for the upload and download logs, respectively.

1810-1830. Bank #2 Drive Table : This is the drive setup table for physical directories, and is toggled in with the T command. The drive numbers are assigned individually. There are 8 device numbers and 8 drive numbers, 1 pair for each of the 8 directories.

1850-1900. Information (I) Command Drive Table : The I command gets it's information files from the drives pointed to by this drive table. The first 8 device numbers and drive numbers are for the relative information files (+r.X.inf) and the remaining 8 device and drive numbers are for the physical directory information files (+p.X.inf).

1920-1990. Restriction Time Period Table : These 64 bytes were explained in detail 2 pages previously. The order of the DATA statements is organized as follows:

1920 : Upload Time Restriction - Relative Directories 0-7 - Start Time
1930 : Upload Time Restriction - Relative Directories 0-7 - End Time
1940 : Upload Time Restriction - Physical Directories 0-7 - Start Time
1950 : Upload Time Restriction - Physical Directories 0-7 - End Time
1960 : Download Time Restriction - Relative Directories 0-7 - Start Time
1970 : Download Time restriction - Relative Directories 0-7 - End Time
1980 : Download Time Restriction - Physical Directories 0-7 - Start Time
1990 : Download Time Restriction - Physical Directories 0-7 - End Time

The next few setup parameters are for the small OLINK boot module. This is part of the second data file that is created for the linking process.

2040-2050. These are the 2 text prompts for the small boot module.

2090. Master Message Write Toggle : Set to 0 to disable the ALL-WRITE process, set to 1 to enable it.

2100-2190. User ID# Table For Message Write/Download Process.

Both lines 2090 and 2100-2190 set up a procedure that will require users to write a message in the message section before being allowed access to the file section. This was implemented as an extreme measure to take against certain users who have compulsive downloading habits. The BBS will check to see if any user is in the ID# table you create, and if found, he will be told to go to the message section and write a message; otherwise, he may not access the file section. The master toggle in line 2100 requires that EVERY user write a message before they can go to the file section. This is a drastic option to get your message section going. This master toggle takes priority over the ID# table, no matter what you have it set to. The ID# table contains a list of from 0-100 user ID# of those users that HAVE TO write a message. If you find that you have more than 100 of these "bad" users, you will have to use the master toggle, or else resort to other means. Note that the SYSOP and level "s" users are exempt from this option no matter what you set for the master toggle or ID# table. That is why ID#0 in the table is a null ID#. Just key in all ID#s you want for forced message writing. They can appear anywhere in the table.

After this table, there is no more parameter data to set. Save this custom file AS any file name, then RUN it. You may modify the file anyway you want. Note that the LINK.M008.x files expect to use drive 8:0 at all times. Don't try to use 2 drives for this module or a drive other than 8:0, unless you want to disassemble the LINK files.

Module #009 ; Alternate TRIVIA Section (P1)
Files ; SETUP.M009 - QLINK.M009 - QLINK.M009.A

This is the alternate module for trivia, as opposed to module #002. This is more of a "true" trivia section, in that all questions are answered and verified as to whether they are correct at the time the user enters his answer. The basic structure of this section follows module #002 exactly, except for the mode by which questions are answered.

There can be from 1 to 25 sub-sections, and if there is only 1 sub-section, it will be treated as a single section only, without a master menu for accessing multiple sub-sections. The master menu is named "+u/menu", as opposed to "+t/menu" that module #002 uses. The outline is the same: set it up as a description of all sub-sections, as you would do with the master menu in module #002. If you have only one section, then this file is not needed, and the user would proceed to section 1 without having to make a selection. Entering "?" will re-display the "+u/menu" file, entering RETURN will exit to the main BBS.

Once in a section, the intro file is displayed the same as would be in module #002. You should create this file so that is describes the current trivia section. This file is named "+ui/xx", where "xx" is the section number. For example, the intro for section 9 is "+ui/09".

The user will be asked if he wants to continue and attempt to answer the questions. If so, the user's name and ID# are written to the result file (named "+ur/xx", where "xx" is the section number), as is the case with module #002. If the user responds no, then he will be placed in the master menu again, or back to the main BBS if you only have 1 section. If the user is a level "s" user and responds with a no, then he will be asked if he wants to see the results for this section, and then asked if he wants to clear the result file.

The question file is named "+uq/xx", where "xx" is the section number. This question file is where the difference in trivia structure comes into focus. You have to be VERY exact in creating this file, or the trivia module will go off into neverneverland. The format for this file is:

-Question Data, With Answer Choices
-An "@" delimiter to indicate the end of the question.
-The number of choices that you have in the multiple choice selection.
-The number that represents the correct answer.
-The text data that will be displayed if the user is correct.
-An "@" delimiter to indicate the end of the above data for a correct answer.
-The text data that will be displayed if the user is wrong.
-An "@" delimiter to indicate the end of the above data for a wrong answer.

-The start of the next question, or end of file text + the "&" delimiter to indicate the end of the file.

The above 8 parts make up one question block. Here is an example:

What color is not part of the basic visible light spectrum:

1) Blue 2) Red 3) Violet
4) Yellow 5) Pink@
55hat Is Right!@That Is Wrong@

No Carriage Return.

The above show the question with possible answers, the "@" delimiter, the number of choices (5) and the right answer (5). This is followed by the text for the right answer, the "@" delimiter, the text for the wrong answer, and the "@" delimiter. This is one question block. The end of file text can be simply an "&", or can be a final comment ended by an "&". Otherwise, you make up the next question block.

Module #009 : Alternate TRIVIA Section (P2)

The limit on the number of questions you can have per section is 256. The answer file will be much smaller than that of the first trivia section because only 1 character answers are stored. The basic format of the question file is multiple choice. Answers can be any digit from 1 to 9. You specify the number of possible answers for each question as you go along. This number is stored as part of the question file to make things easy. The answer is stored right after that. The idea of having text displayed after a right or wrong answer was added for emphasis on the questions. You can make sly remarks about wrong answers, or casual remarks about the right answer. If you do not want remarks for the right or wrong answer, then you must place at least "null" equivalent character in the proper text area. You could add just a blank space instead of a comment. This will not affect output if you do this.

Note that the module itself displays the prompt that asks for an answer to be input; you need not enter this in every question. Also, the module displays a prompt for the right answer and the wrong answer. This prompt is the same each time a question is answered. This is why additional comments can be added to the question file.

As users answer questions, the results are stored in memory. When a user chooses to quit (entering "x" at the choice prompt does this, or the last question will too), the results are saved to the answer file. The user will then be given his results: the number of questions answered, and the number he answered correctly. The results stored in the answer file would look like this:

User ID# - Name
Y122Y2Y3Y2YYYY2Y21Y
(10/19)

The first line is obvious. The second line is the actual answers stored in order. Leftmost is the answer to question 1, rightmost the answer to question 19. A "Y" in any position indicates that the user got the question right. A digit in any position is a wrong answer. This digit is the user's choice for what he thought was the right answer. Since it is not a "Y", you know it is a wrong answer, and this digit represents it.

The last line is the number of questions correct / the number of questions answered). As you can see, this occupies much less memory than the other trivia module. However, you are restricted to a multiple choice system. Ten free blocks must be available for a trivia section to be accessed.

The SETUP File.

100. Set V=0 for color, set to 1 for ascii.
120. DF$: Destination file name.
140-245. The 12 text prompts for this module.
350. Change this line to modify the destination drive.
5000. Connect Time Flag : Set to 0 for connect time to stop during module access, set to 1 for normal run time.
5005. Number Of Trivia Sections : A number from 1 to 25.
5010-5020. Drive for the "+u/menu" file.
5030-5080. Drive table for the 25 trivia sections. See module #002.

Module #010 : On-Line Ordering Section (P1)
Files : SETUP.M010 - OLINK.M010 - OLINK.M010.A - CREATE.ORD - ORDER.RUN - ORDER.DLM

This module allows you to run a system with the capability of operating very simple "on-line shopping", similar to one of the services (stores) of CompuServe's Electronic Mall. Although most of the users of this BBS will not use this module, it is still here for the few who might have a business where things may be sold using the BBS as another outlet for product purchasing. This module is fairly lengthy, therefore the steps for setting up this section will be given in the order in which you should follow.

A. Using CREATE.ORD. This program is used to create, modify, and maintain your item files. You may have up to 25 sub-sections, the standard number for multi-section modules. Item files are relative files, each file containing from 1 to 195 items. The module sets the file to a fixed record count of 195, which amounts to 33 blocks per item file. Item files are named "+or.xx", where xx is the section number. For example, "+or.08" is the item file for section number 8. If you are using a single section module, the item file section number defaults to 1. These realtive files must be created with the CREATE.ORD program. Creating a new module erases the items and file of any previous item file you may have created. For the first time setup, you must create the item files for each section.
The record structure for each item is as follows:

Bytes 0-34 : Item Description. Each item can have from 1 to 35 characters for its description. This description is shown as the main title for the item when the user browses the items for each section while on-line. Thus, the standard description would be the item name only.
Byte 35 : Item File Number. The item file number represents the item description file number (See Below).
Byte 36 : Item File Position. This is the position value within the item description file.

Bytes 35 and 36 work together to specify where the BBS is to find the description text for each item in the item file. Description files are completely managed by you. They are simple SEQ text files that contain a short paragraph or two about the item. This description should contain the price and title of the item. If you run the color BBS, do NOT use special characters like cursor values and clear screens. Try to keep the size as small as possible if you intend to link descriptions.
Descriptions can be linked in the same file by separating each one by the "@" delimiter, the same character used in other modules. The item file position byte (36) is a number from 1 up to 99. An example linked description file would be like this:

(Text for item #m)@
(Text for item #n)@
(Text for item #o)@

Byte 35 is the description file number. This number can be a value from 1 to 195. Description files have the format "+des.xx/yyy", where xx is the module section number, and yyy is the description file number mentioned above. For example, description file 34, section 23, would be named "+des.23/034". As was said before, you are responsible for maintaining these text files.
In the example above, the "@" character must appear at the end of each description. Even if you are only using one description per file, you must end it with "@", since it is also used as an end-of-file marker, in addition to a delimiter. Item m,n, and o would have position numbers of 1,2, and 3, respectively. This is the position value in the record (byte 36). So, if byte 35 has a value of 34, and byte 36 has a value of 3, then this would correspond to the above example file name, and item #o in the above description text. The values m,n, and o can have any numerical value. For example, the items 11,189, and 134 could be assigned to m,n, and o respectively. The numerical assignments are left up to you.

Module #010 : On-Line Ordering Section (P2)

Bytes 37-39 : Item Price. Prices are stored in this record as a 24 bit binary value. This allows for a price range from 0 to 16,777,215. Because of the necessity for dollars and cents values, the price has a fixed decimal point 2 positions to the right. Therefore, prices run from $0.00 to $167,772.15. This should be quite adequate for your ordering section, provided you do not use it for dealing with real estate. Since this represents the maximum value for pricing, the total cost of a user's order cannot exceed the maximum value without causing a price overflow.

Bytes are stored in low, high, extended high byte order, where extended high byte is the value DIV 65536. The value MOD 65536 is left for the standard low and high bytes.

An added carraige return to the record makes a 41 byte record. CREATE.ORD maintains these records for you. The standard way of interfacing the text description files with the item relative files is to modify the item records when changing the structure of your items. For example, if you have a description file with descriptions for 5 items, and you decide to remove item 3 from this text file, you do not have to remove the text description from within the text file if you wish to add a new item. It is easier to change the item file numbers and position values in the item records to point to new description files than it is to move blocks of text in a text file. Always set up the relative files before making the description files.

CREATE.ORD Options. When you first run CREATE.ORD, you will not have a data file for your ordering section. This data file, named "+o/data", contains information for the ordering module to use during the BBS-User interaction. At any time the program requests the disk for the data file, you may enter F1 to create a new data file. The values that are placed in the new data file depend on what they are currently in memory. There is an option in the program to modify the data file's parameters, explained later. Once a data file has been created or loaded, you will be given the main menu, with the following options (NOTE: CREATE.ORD may be loaded from ANY drive 0 device you want):

CHANGE SECTION, CREATE ITEM FILE, LOAD ITEMS, SAVE ITEMS, SORT ITEMS, ADD ITEMS, DELETE ITEMS, MODIFY ITEMS, ITEMS SUMMARY, PAGED ITEM LIST, PRINT ITEMS, CHANGE DATA FILE, QUIT.

(1) Change Section. Before you switch to a new item section, you should always save your current section items to disk. The program only works with one section at a time, memory wise. Switching to a new section clears out the items of the old section. Section numbers range from 1 to 25.

(2) Create Item File. This option will create an item relative file for that section. If you already have an item file for this section, it will be replaced with the new one, and all items in the old item file will be lost. Thus, to clear an item file quickly, just re-create it. Note the following rules for file placement:

a) The ordering data file and the module data file (+ord/inf and +o/data) must reside on the same disk.

b) For each section, the description files, and the item file must reside on the same disk drive.

(3) Load Items. When you switch to a new section, or run CREATE.ORD the first time, the item count will be set but the item data (records) will not be in memory. Before you work with any section that has at least one item, you MUST load the items into memory. Otherwise, you will be working with garbage memory.

(4) Save Items. Any time you wish to reset the "+o/data" file, even when you don't want to save items, you should use this option. Saving the items will place all the records in memory in the item file, AND will also cause the data file to be updated. Always use this option before you switch sections, provided that the section you switch from has at least one item.

Module #010 : On-Line Ordering Section (P3)

(5) Sort Items. Sorting items will shift the item numbers, but will NOT change the item description file numbers and positions (bytes 35 and 36 in the record). This is why bytes 35 and 36 are there: to allow you to have your items in alphabetical order without worrying about having your description text files in the same order. When you add a new item to a a file, you simple assign it a new position and file number that you haven't used yet. The sort algorithm is a standard bubble sort. You may notice a slight delay if you have 195 items to sort, but still, you won't have to wait an eternity.

(6) Add An Item. Add, Delete, and Modify options will "loop". You will use that option over and over until you enter a RETURN to exit back to the main menu. When adding an item, the item number will be displayed, and you will be required to enter the following input:

DES:Item Description, 1-35 characters
ITF:Item File Number, a value from 1-195 (ie. 7 will be for file "+des.xx/007", where xx is the section number.
POS:Item File Position, a number from 1-99 (see NOTE below).
PRC:Item Unit Price, a value from $0.00 to $167,772.15. You must always add the cents value (ie. .XX, as in $.88).

You will be asked to confirm the addition at the end. NOTE: the ordering module must do a sequential search for the description text for an item that is not the first item. Thus, to display the text for an item in the 5th position in the description file, the BBS must read in and skip all characters that make up the first 4 descriptions. For example, if your first four descriptions contain 500 characters each, the BBS will have to read in 2000 characters before it finds the text for the item in position 5. On a 1541, this will take about 6 seconds. If you have 95 items in one description file, the BBS would have to read in 94*500=47000 characters, which would take about 3 minutes. Obviously, you must use as few descriptions per text file as possible in order to save time. Five is a good number. The reason why you should link descriptions in one file is so that you will save file space on your directory. If you use 1 description per text file, and you have 195 items, that means that you will use 195 directory entries, more than a 1541 can handle. If you use 5 descriptions per text file, you will only take up 39 directory entries. IEEE drives like an SFD are capable of handling probably 25-30 descriptions per file, thus occupying maybe only 7 files on disk. Of course, this all depends on just how much text you put into each description.

(7) Delete An Item. You are allowed to specify the item's full description name, or the item number (that is, its position in memory), given when you use the Summary option. NOTE: All items MUST have a description of at least 4 characters in length.

(8) Modify An Item. When you modify an item, you will be given the current value of the field in the record. If there is a field you do not want to change (ie. price), then just enter RETURN by itself, and you will skip to the next field in the record. Enter RETURN at all input prompts to abort the modification you may have erroneously started.

(9) Item Summary. This option will display the data for all items, as you entered them, starting at item #1. The listing will pause at the end of each item. Hit RETURN to continue scrolling through the items. Enter F1 while the output is paused to abort to the main menu.

(0) List By Page. This option lists all items exactly as they would appear to the user when on-line with the ordering module. Items are displayed in pages of 15 entries. Enter X to abort the listing after any page. Enter RETURN to move to the next page.

(A) Print Items. Output is not formatted with this option. Set your printer to top-of-page. The output given, from left to right, is: Item #, Description, Item File Number, Position In The File, and the Price. Use STOP to abort the printout.

Module #010 : On-Line Ordering Section (P4)

(8) Change Data File. This option allows you to change the current parameters of the "*o/data" file stored in memory. You must use the Save Items or Create Item File options to update this data file on disk. In addition, using F1 to create a data file when you are given the option will also update the data file. The parameters you may change are as follows:

1. Billing Methods. When dealing with on-line ordering, options like cash and cheque are not really of value, since orders are usually pre-paid before they are shipped. Thus you have the other options: credit cards and C.O.D. You are only allowed to specify up to 4 methods of payment for the user to make. The methods are definable by you. Enter the "1" key to cycle the value from 1 to 4.

2. Billing Queries. When a user is prompted for billing information, you are required to tell the user just what you want to know. For each of the 1-4 billing methods you have, you may specify from 0-5 input responses that the user must make. For example, if you use VISA as a billing method, then you might ask the user the following:

Card Number.
Expiry Date.
Issuing Bank.
Cardholder's Name.

The above 4 options are required input for the user. The actual text defining each response is left up to you by modifying the ORDER.DLM text prompt file. When you enter "2" for that option, you use the cursor up/down key to change the first number, and the cursor left/right key to change the second number. For the X:Y values shown, X is the billing method currenty selected. It runs from 1 to the number of billing methods set with option 1. Y is the number of user inputs you want. The above example would require a value of 4. Note that you can specify a value of 0. For example, C.O.D. usually does not require any information other then the user's name and address, and phone number. If you use a value of 0, the user will not be given any inputs. The upper limit is 5 for each method. You must change ORDER.DLM so that your text prompts correspond to what you want as input from the user. Enter RETURN when you are done with option 2.

3. Address Responses. The module will always ask for a billing address and a shipping address. If the user's shipping address is the same as his billing address, the user will not have to enter the information twice. A usual setup for address input is Name, Address, City, State/Prov, Postal Code, and Telephone Number. This would be 6 input responses. You are allowed from 1 to 10. Once again, you must change the ORDER.DLM file if you want your own text prompts. Enter "3" to cycle the number values.

4. This option sets whether a user has registered his billing information with you. Sometimes, a user will give his billing/shipping address information, and his billing method. This information may not change for a long time period, and if this user is a regular customer, you can set it up so the user does not have to go through the lengthy process of entering billing information every time he wants to order something. Enter "4" to select option 4. The first 3 digits are the user ID numbers. Note that the SYSOP (you) has a number as well, even though you may never use it. Use the cursor up key (SHIFT held down) to move DOWN through the user ID numbers, and use the cursor down key to move UP through the user ID numbers. There are 256 ID numbers, one for each user. You use the cursor left/right key to toggle the second value between a "y" and an "n", which indicate YES and NO. A "y" (YES) value means that the user can use the billing information you already have for him. The BBS will then always ask the user if he wants to use current billing information. He still has the option to use new information at any time.

In addition, the F1,F3 and F5 keys, entered from the main menu, will change the screen colors. Once you have set up all your item files, you will proceed to create the description files.

Module #010 : On-Line Ordering Section (P5)

B. Creating Description Files. This requirement was already discussed earlier, so only a short summary is given. Description files are SEQ text files, with the format "+des.xx/yyy", where xx is the item section number, and yyy is the item file number (ITF in the CREATE.ORD program). Each description must end with the "@" character, even if you only have one description per file. Obviously, you may not use the "@" character in the actual body of the description text, since it is a delimiter. If you only use one item description per file, the position number for every item (POS in CREATE.ORD) will always be 1. If you want to link descriptions in the file, just add another description, ending with another "@" character. Position number limit is 99, but will usually only be about 5 for a 1541 drive, a little more for an IEEE drive. Try to avoid using excessive screen control characters (clear screen) in your description text if you run a color system. Always place your text for each section on the same drive as the corresponding item file for each section.

C. Modify ORDER.DLM. .DLM files are delimiter files. These files contain the "@" character as a delimter character, used to separate text prompts. The BBS text file (BBS.TXT) has been converted to a .DLM file, so that you may make changes to it without having to use CREATE.TXT. Normally, text prompts on the BBS system are separated by a NULL character (value 0). NULL characters cannot be used as text, and there is no way to generate nulls on a text editor, since a null represents "no-key-pressed". Other than that, the text files the BBS uses for text prompts are standard SEQ text files that a word processor can handle. The program CONVERT.DLM on your OverLink disk will take these .DLM files and turn them into the standard text prompt program (PRG) files, replacing delimiters with nulls as part of the conversion process. (See CONVERT.DLM in this manual for details).

In the case of ORDER.DLM, you do not use CONVERT.DLM to create a proper text file. Instead, the SETUP.M010 file will do this for you. Nevertheless, the format for the text file is still .DLM. For reference, you should print out the ORDER.DLM file to see how it is set up, or else load it into your favorite word processor. There are 57 text prompts that make up this file. Size is very important. You must make sure that your text file does not exceed the limit such that when the module has been setup with SETUP.M010, the end address of the load does not pass $CC00 (52224), or else your text will overwrite the screen memory, and possible the C64's I/O memory block. The current size of ORDER.DLM leaves about 8 more blocks of room (about 2000 bytes), which should be plenty.

Text prompts are numbered from 0-56, although you won't see the prompt numbers as you would in TEXT.PAL. Before the start of each new prompt, you will see the character "@", followed by a carraige return. This character pair (@+<CR>) is the delimiter between each successive text prompt. When you run the SETUP file, ORDER.DLM will be read into memory, and the character pair (@+<CR>) will be removed, and replaced with the NULL (value 0) character.

.DLM files must always have this format. So, the chain would be @+<CR>, text prompt 0, @+<CR), text prompt 1, @+<CR>, text prompt N-1, @+<CR>,@@. Note the way a .DLM file ends. You must end the file with the four characters: @, RETURN, and then 2 more @ characters. If you do not do this, the computer will go into an endless loop looking for these four characters. Note that we ended with prompt N-1. Prompt values always start with base 0 instead of 1.

You may freely modify the text any way you want, then save it back as a SEQ test file, with the name ORDER.DLM. No other name is acceptable, since SETUP.M010 expects this file. You must make sure that you have the exact number of text prompts. Do not remove any. You will note that where we have indicated that none are in use, we put the word "Empty". You can use anything you want for a prompt that you won't use. Even a single RETURN character will suffice.

To help you find out where certain prompts are in the file, and what numerical value they represent, a summary of portions of the ORDER.DLM file is given.

Module #010 : On-Line Ordering Section (P6)

0: Section Number Input Prompt.
4,6: Ordering Information And Summary Headers.
7: This prompt can be anything you want (order excludes tax etc....).
9: Given If User Has Registered Billing Information (See CREATE.ORD).
10: Ordering Form Header (User will always be given these 3 options).
11-13: Billing Address, Shipping Address, And Billing Method Headers.
14-17: Billing Method Descriptions.
19-28: The 10 Prompts For Billing/Shipping Addresse.
5,29,30: Quantity, Description, and Price Prompts Displayed During Ordering Summary.
31: Total Order Cost.
32-36: The 5 Prompts For Billing Method #1.
37-41: The 5 " " " " #2.
42-46: The 5 " " " " #3.
47-51: The 5 " " " " #4.
52: Displayed After All Is Said And Done. Users May Not Turn Back If They Say YES Here!
53: Prompt To Confirm Shipping Address Same As Billing Address.
54: BBS Return Prompt.
55: Displayed After User Order Saved To Disk.
56: Displayed Under Summary Of Shipping Address IF Same As Billing.
1: Displayed At The End Of Each Page Of Items. X Exits/Orders. # To Get Item Description. RETURN For Next Page.
2: Displayed At The End Of Item Description. 0 Puts Item In User's "Shopping Cart".
3: Displayed If User Tries To Overfill "Shopping Cart" Without Emptying First.
8: General IS-THIS-CORRECT, OK, ALRIGHT Prompt.

D. Using SETUP.M011. This file is a 2-stage setup module, as is the file expansion module. After the SETUP program creates the data file ORDER.DAT, the program ORDER.RUN is loaded in and executed. This small ML program will merge the OLINK.M010 or OLINK.M010.A module, the ORDER.DAT file, and the ORDER.DLM file together, to create a new OLINK.M010 or OLINK.M010.A file. Always use a backup of your OverLink disk for processes like this, because a power out during a module setup like this could leave you without any module to modify. The ORDER.DLM file will be converted to standard text prompt format during the merge process. If the computer resets during the SETUP run, you are missing a file.

The data to be modified is contained in the 5000 line range like the other modules. Type LIST 5000- for the module parameters.

5000. Version to modify. 0 for color, 1 for ASCII.
5010. Drive that the main menu ("+o/menu") resides on. Ignored if only one section used.
5020. Number of ordering sections, 1-25.
5030-50. Section device table (see previous module setup procedures for details).
5060-80. Section drive number table.
5090. Drive that the "+o/data" and "+ord/inf" files reside on.
5100. Connect time flag. Set to 0 if you want users' connect times to stop during their stay in the module. Set to one otherwise, for normal running time.

NOTE: The file "+ord/inf" is where the user's orders are stored when they make a purchase. Details will be given in the next section.

E. Using The Ordering Module. Once you have run the setup file, you must put the OLINK file on your OverLink drive. It is the only file that you put on it. This is a large program, like the terminal link for the BBS. As such, this module writes into the storage area for the standard BBS text file. You must therefore place your BBS text file on your OverLink drive, and name it "+bbs/txt". When a user exits the ordering module, the text file will reload. If you have already done this by installing the terminal link, then you don't have to do it now.

You must create the "+ord/inf" file. This is a SEQ text file, and does nothing more than hold user's orders. You may initially place any data in the file you want (ie. 2 RETURNS, a header description, etc.). As orders come in, they will be appended to the this file. Place the file on the same disk drive that you put the "+o/data" file on, which you set in the SETUP data.

You are now ready to use the ordering section. When a user enters this section, the contents of the "+o/data" file are read into memory. Then, if you are running more then one section, the ordering section main menu is displayed. This file is named "+o/menu", and is analogous to the other master menu files in other modules (ie. +t/menu in trivia module), in that you put in text describing each sub-section. If you are using one section, the module sets the section value to 1, and the user proceeds to the items display. Otherwise, the user enters an item section number. However, at this point, the module checks to see if the user has placed any orders. If the user tries to hit return to exit the module (in case of more than one section), or the user attempts to exit by entering 'X' from any page in the item display (for one section), the module will see if an order for any items has been made. If so, the user will proceed to the ordering information section.

When the user enters a section, pages of items are displayed, 15 at a time, giving the item number, and it's description (the one you made with CREATE.ORD). After each page, a user may enter "X" to exit, a number that corresponds to an item number, or RETURN to go onto the next page. In addition, a level "s" user may enter "r" to read the current ordering result file (+ord/inf). If a user enters "x", he will either exit back to the main menu in the case of multiple sections, or he will be asked for ordering information IF he placed any orders. If not, the user will be returned to the main BBS. When selecting an item number, it does not have to be one that is on the current page. For example, if the user is on page 1 (items 1-15), and he enters 35 (for item #35), he will still get to see the description for item #35. After seeing the description, the user will be placed back at the current page, not at the one for item #35 (which would be page 3). The module remembers the current page number at all times, independant of item selection. A user may not select an item number of 0 or one greater than the number of items you have. After all pages have been displayed, the user will be returned to the previous menu.

When a user enters an item #, the module will search for the item's file number for that section, and when found, will initiate a search within the file for the position of it's description. If the position within the file is the first one, no search is necessary, and the description is immediately displayed. During a search, for each item description that has to be read past, a period (.) will be displayed to the user to let him know that a search is being made. Generally, for small descriptions (<half a page), this will go too fast for the user to notice.

After a description is displayed, the user will be given the chance to order this item. Entering "O" places the order. Hitting RETURN (or anything else) places the user back to the current item page. As orders are placed, they are stored in an item order list. This list has been limited to 25 items per order. This does not mean a user can only order 25 items. It simple means that a user must proceed to complete the order for the current items selected. After that, the use may order 25 more items.

Module #010 : On-Line Ordering Section (PB)

Placing The Order. If a user has ordered one or more items, the ordering sequence will begin when a user attempts to exit the module. The sequence of events are as follows:

A. Quantity Entry. Each item's description and price are given. The price is the one in the item record, not the item description file (they should be the same though!). The user must enter how many of the current item he wants. A user may order up to 99 of any item. The module keeps a running total of the user's order (like a cash register). Each item ordered will have to have a quantity input by the user. After all items are done, they are reviewed for him. During the output of ANY text in the module, the list control keys are active (S,C, and A for list control), so that the user may pause the review if it is going by too quickly. After all items are done, the total for the order is displayed. The user now has the option to confirm this. By not entering a quantity, a user will abort the ordering process, and his order will be cancelled.

B. Billing Address. The user must answer all questions you give for all remaining prompts, or the order will be cancelled. The billing address info is setup as previously mentioned. Up to 25 characters are allowed for input for each question.

C. Shipping Address. The user will have to enter more address info, unless of course he selects same as billing address. If so, this section is skipped.

D. Billing Method. The user must select one of your billing methods. When one is selected, he will then have to answer any questions you created (from 1-5). If you set the queries to 0 with CREATE.ORD, then this part is skipped too.

E. Information Summary. Each of the user's inputs from parts B,C, and D are displayed to the user in summary format. After each of the 3, the user may abort the order. If aborted, the user will start with part A again. He will not be returned to the main menu in this case. If all 3 inputs are verified okay, the user is asked to confirm the order. If the user responds yes at this point, then he can not cancel it. The order will be saved to "+ord.inf", and the user will be notified of this. Then the user will be placed into the start of the module, where he may order more items.

NOTE: If a user has registered billing information (set with CREATE.ORD), then the user will be asked if he wished to use his current billing information BEFORE the start of part B. If the user responds Yes, then parts B-E are skipped, and he will simply be aksed if he wants the order placed. If so, the prompt "*** Current Billing Info***" will be placed in the +ord/inf file instead of the actual ordering information.

Storage Format. What is placed in the +ord/inf file for each order is as follows:

User ID#-User Name-Method # Example: 034-JOHN DOE M:1

Item Entries. For each item ordered, the following information is displayed: Section Number, Item Number, and Quantity. Example - S:13 I:144 Q:3. Item number is IMPORTANT! In this case, the item number is the actual position of the item in the current item file. Changing the item file at any time can change this number, so always process ALL current orders before you change an item file. Use the LIST BY PAGE option in CREATE.ORD to find out what the user has ordered. After the item information, the ordering information is displayed. Only the user input is stored, without the text prompts. You will have to match the input to your prompts to determine the user's input. Each of the 3 blocks of input is separated by a blank line. If a user selects current billing information, these 3 blocks of data will not be shown. Also, if you have no inputs for the billing method the user selected, that block (the 3rd one) will not be shown. The 3 input blocks, in order from top down, are Billing Address, Shipping Address, and Billing Method Information. You are resposible for clearing this ordering result file any way you want. The module does not include a clear result file option.

Module #011 : BlackJack
Files : SETUP.M011 - OLINK.M011 - +BJ.INST - CREATE.M011

This is the first of 3 simple game modules you will find on this disk. More will be added later, provided that OverLink is met with suitable response to allow the creation of additional OverLink disks.

The file "+bj.inst" is the instruction file. It is a color file that you should place on your BBS system as the module's help file. There are no ASCII versions of the 3 game modules, for reasons mentioned earlier. You should read the color help file to see if you think modification is necessary. All the instructions for playing the game are in this file. Therefore, we will skip the play mechanics, and just mention the setup procedures.

This module uses cursor plotting for the game dispaly, as does module #12. Because of this, DarkTerm 4.c must be used by the terminal end user. Plotting is built into V3.1 already.

BlackJack saves the user's current cash value and his high score to disk on a relative maintenance file. High scores represent the highest cash value the user ever reached up to his current session. The current scores represent just what the name implies. How much money they have currently. Only you can change the user's high score and/or current cash status. Both variables always update when a user completes a session.

SETUP file variables:

100. V : Version variable is ignored.
120. DF$: Destination file name.
140-190. These 6 prompts are written to disk as one prompt. Because of the size of this text prompt block, it was split into 6 variables, a0$-a5$.
200-210. The second prompt is split into variables b0$ and b1$.
320. Destination drive and device values.

5000. Connect Timer. Set to 0 for connect time stops, set to 1 for standard running time.
5010-20. Drive that the relative file "+bj/hi" goes on.
5030-40. Drive that the helpfile "+bj.inst" goes on.
5050. Number of decks of cards to use. You can use from 1 to 6 decks of cards at once. Casinos generally use at least 2 decks. Multiple decks will prevent card counting, used by cheaters.
5060. Shuffle marker. The marker is the card number where re-shuffling occurs. Casinos usually place the marker two-thirds of the way into the card stack. The module limits the card marker to values from 1 to 255. Do not use a marker value that exceeds the number of cards in the deck. Casino rules specify that top cards are not "burned".
5070. Dealer stand value. Usually the dealer will stand on 17. Some casinos use 16 for dealer stand. You can use any value from 1 to 21. Using a stand value above 21 will be somewhat unpredictable.
5080. Minimum bet. The bet values are dollars. The module allows bets to run from $1 up to $32,000. Try to use a narrow range to keep the users's cash value from jumping up and down like a yo-yo.
5090. Maximum bet. Make sure it's at least as large as the minimum.
5100-20. Border, background, and text colors. The values stored are the string values of the color to use. For example, white would be CTRL-2, in quotes.

Module #011 : BlackJack (P2)

Once you run the SETUP file and place the module on your OverLink drive, you must make up the user high score file. CREATE.M011 will create and maintain your high score file. But since it is just a high score file, their is no provision for in-depth record control that a relative file manager module would have. Load up and examine the CREATE file.

The first data value is the mode specifier (line 1010).

Value 0: The high score file will be created from scratch.
Value 1: The cash values for all 255 users are re-initialized. The high score values are not affected, however.
Value 2: The cash value and the high score value for one specific user are changed.

The two data values in line 1020 are the high score initial value, and the inital cash limit values for all users if mode 0 or mode 1 above is used. Cash values and high score values are signed 24 bit dollar values. This means that a user's cash value and high score value can be as high as $8,388,607. If a user's cash value goes above this, very strange results will occur, but will not be lethal to the BBS. High score values may not be negative values. The lowest high score a user can have is $0. However, the cash value can go as low as -$8,388,608. A user will have to make up his debts unless you re-initialize his starting cash value with mode 2.

It is suggested you make the high score and initial cash values the same when starting the high scores for the first time. The last three data values in line 1030 are for mode 2 change for a single user. The first value is the ID# of the user you want to change. The second and third values are the user's new high score and new current cash value, respectively. NOTE: You (ID# 0, SYSOP) do not have a value in the high score table. If you want to participate in the high score runs, you must add yourself in as a normal class user.

The CREATE program is small. Examine it closely to see just how the program works. Either run this on your disk that you want the high score table on, or use COPY.REL1 to copy the file to the drive you want it on.

Each record consists of only 7 bytes. The first three are for the high score, the next three for the user current cash value, and the last byte is a carraige return. Bytes are stored in the format : lowest 8 bits, middle 8 bits, high 8 bits.

Module #012 : BlackBox
Files : SETUP.M012 ~ OLINK.M012

This program was written to take advantage of the cursor plotting of the color BBS. There is no analogous program for the ASCII version. This game is a very simple one. The full instructions for this program are found on the OverLink disk, under the file name "+box.inst". This is the same file name that you will place on your system, as the help file for the users. Users of the system must use DarkTerm 4.C or later.

SETUP file:

120. DF$: Destination File Name.
230. Destination device and drive.

5000. Connect Timer. Set to 1 for time to run normal, 0 for user connect time to stop, while in this module.
5010-20. Drive that the file "+box.inst" goes on.

Note that whenever the cursor stops and sits at any point on the screen, a user is required to enter some input (ie. the number of atoms to use).

Module #013 : Polling Section
Files : SETUP.M013 - QLINK.M013 - QLINK.M013.A - CREATE.POL

If you have used the voting section on V3.0, you will be aware of the fact that each voting topic requires 258 bytes of storage. That means that 100 topics requires more than 100 blocks on disk. Also, there was no way to sub-divide the topics as is the case with the bulletin section. The polling module allows up to 10 sub-sections of up to 50 topics per section, for a limit of 500 topics. The section numbers run from 0 to 9, not 1 to 10. Remember this! As another option, a user may be able to write a poll message on the system. However, the module defers its addition to the polling topics by placing written topics into a poll storage file. This file must be broken down, and topics extracted, through the user of a wordprocessor. You must also add the description part of the user's topic to the description files for each section.

The section operation part of the module is the same as would be for the Trivia modules or the other multi-section modules, except for the fact that you only have from 1-10 sections (not 25), and the section numbers run from 0-9, not 1-25. If you are using one section, you will be using section number 0 as the default. The master menu will be skipped in this case, and the user will not have to select a section number. If you have more than one section, then the file "+v/menu" will be displayed, and the user will have to enter a section number. This master menu file should be set up like other multi-section master menu files.

Once the user enters a sub-section, that section's description file is displayed. This description file contains the main titles for all of your topics for that sub-section. Because it is a standard SEQ text file, you may place anything in this file you want. Sub-section menu files are named "+po/menu.x", where x is the section number (from 0 to 9). For example, the description file for section 3 would be "+po/menu.3".

After the description file is displayed, the user will be able to : enter "?" to re-display the description file for that section, enter "w" to write a poll topic, enter "r" to read the topic file, or select a topic. To select a topic, the user enters the topic number he wants. If there is no associated topic data file, or the user enters an illegal topic number, the option will be aborted. This is similar to the way in which the voting section handles topics not yet implemented. If you have 15 topics in section 3, and the user types 16, the module would get a file not found error, and would abort the option, since the topic is not available.

Topic data files are SEQ text files that function in the exact same way as the voting section of the BBS. The file must contain a list of all possible voting selections. See the VOTE command for vote topic file organization. After the file is displayed, the user must select a topic value, or enter RETURN to not vote. If the user enters "?" instead of a voting value, the current results will be displayed to the user, based upon priviledged access settings for this module. The possible values are 1-9 for poll casting, which is the same as that for the voting section in the BBS.

Each of the sub-section value files (the relative files that contain the user's poll selection values (1-9)) occupies only 3 blocks on disk. This is because only 10 bytes are used to store the results for each topic, compared to the 258 bytes for the VOTE command. This dramatic drop in disk space has a price though. A user may vote on any topic as many times as he wants. There is no way the module can check if the user has voted at a previous time on the same topic. This allows for "fixing of the ballot box". It may just happen that users may illegally vote to change the proper outcome of a poll. Thus, you should realize that without honest users, you may end up with very inaccurate results. In this case, you should go back to the VOTE command.

The relative files that store the results of a poll are named "+poll/x", where x is the section number. For example, the result file for section 0 would be "+poll/0". The program CREATE.POL will create these result files for you. The text files that store the data and result outcomes for each topic are named "+po/y.xx", where y is the sub-section number, and xx is the topic number for that sub-section. For example, the topic file for topic #34, section 9, would be "+po/9.34".

Module #013 : Polling Section (P2)

The "r" option for reading the module topic file is only available to a level "s" user. When a user selects "w" to write a topic, all of his input is written to this file, or rather is appended. This file contains all topics written by all users from all sub-sections. You must maintain this file, extracting each topic individually with a text editor. This file is named "+poll/res", and is a SEQ text file that you must initially place on one of your BBS drives. It may contain any starting text you want to use. All topics written with "w" will be appended to this file.

The user may or may not be able to use "w" to write a topic. This option is left up to you. The level "s" user will always be able to write a topic, though. The standard line oriented text editor of the main BBS is used for writing topics. The range has been reduced from 100 to 50 lines, but the option to continue after saved still exists, so there should be no size problems. Each topic is stored as follows:

User ID#-User Name
Module Sub-Section User Was In At Time Of Writing:Description User Wants
Date Written

(Text Data)

Example:

034-JOHN DOE
1:This Is My Description

This is my text data.

If a level "s" user reads a topic, and sees the results of the poll, he will have the option to clear the results. Poll results are displayed as they are with VOTE. For each topic value that has been voted for at least one time, its value, followed by a colon, and the number of votes for that value are displayed (ie. 4:25 indicates that the vote value 4 for this topic has been voted for 25 times). The only way to clear individual topics is by selecting the clear option. Creating a result file clears out all topic results for a specific sub-section. A disk doctor may be used if you need to make drastic repairs to the results of a major portion of a result file.

Use CREATE.POL to create the topic result files. Line 1050, PN specifies the sub-section number for the poll file to create. Value range from 0-9. This is a fixed record size file. Each file contains room for 50 topics. Lines 1060 and 1070 specify the drive to write the file to. This is in case you want to create the files on an SFD, but have the CREATE.POL file stored on a 1541 or some other incompatible drive, incompatible with the destination drive for the relative files.

Using the SETUP file:

100. V: Version of module to change. 0 for color, 1 for ascii.
120. DF$: Destination file name.
140-220. These are the 9 text prompts for the module.
330. Destination file device and drive number.
5000. Connect time flag. Set to 0 for connect time to stop, 1 for normal running time.
5010-20. Device and drive for the BBS drive that "+v/menu" goes on.
5030. Number of polling sections (1-10).
5040-5070. Device and drive tables for the 10 sections.
5080. If set to 0, poll results for topics will be displayed. Set to 1 to restrict results to level "s" users.
5090. If set to 0, all users may write poll topics with the "w" option. Set to 1 for level "s" users only.

Module #14 : SuperWumpus
Files : OLINK.MO14 + SETUP.MO14 + WUMPUS.OLM + WUMPUS.RUN + +WUMPUS.INST

This game is an adaptation of the game SuperWumpus, written by Jack Emmerichs for 6800-based machines. Originally published in 1978, it has been re-written and modified to run on the CBBS. There is no ASCII version of this module, for reasons outlined earlier.

The instruction file "+wumpus.inst" contains all the information you need to play the game. This file must be placed on the system for users to access. This is another multi-stage link module, organized in much the same way as the on-line ordering section. The file "wumpus.dlm" is a delimiter-type file. For information on delimiter files, see the instructions in the on-line ordering section. This file contains 39 prompts that you may modify. The second prompt named "NO PROMPT" is not used by the module, but must contain some text anyways. You should familiarize yourself with the module before you begin to change any of the text. When using the setup file, the file WUMPUS.RUN, WUMPUS.OLM, and OLINK.MO14 must alll be on the same disk as the setup file. The linking procedures follow the exact format as that of module #12. The setup parameters in lines 5000 onwards will now be explained.

5000-10. Device and drive for the help file "+wumpus.inst".

5020. Set this value to 0 for connect time to stop, 1 for normal running time.

5030. This value is the number of bottomless pits you want in the 20 caves. The value of 2 is optimum, but you may change this value. The sum of pits, bats, the wumpus, and you must never exceed 20, since there are only 20 caves.

5040. This is the number of caves that will contain the SuperBats. You must use a value of at least 1 for pits and bats.

5050-60. These 2 values specify the minimum number and maximum number of days of supplies the player will have for survival. You may set the 2 values as low as 1 or as high as 255. They may be equal to each other.

5070. This value is a binary divide factor that determines how much supplies a player will lose when he is picked up by superbats. The value of 2 means that 1/4 of the supplies will be lost. Other values that you may use are 1 for 1/2 supplies lost, 3 for 1/8 supplies lost, or 4 for 1/16 supplies lost.

5080. This is the number of arrows a player starts with. Values may range from 1 to 255.

5090. A value of 234 means that caves WILL BE scrambled at the start of each game. A value of 96 means that the maze of caves will remain the same from game to game. Any other value will crash the system. Remember, only 96 or 234. If you design a custom cave maze other than the dodecahedron, you should not scramble the caves, even though it is possible.

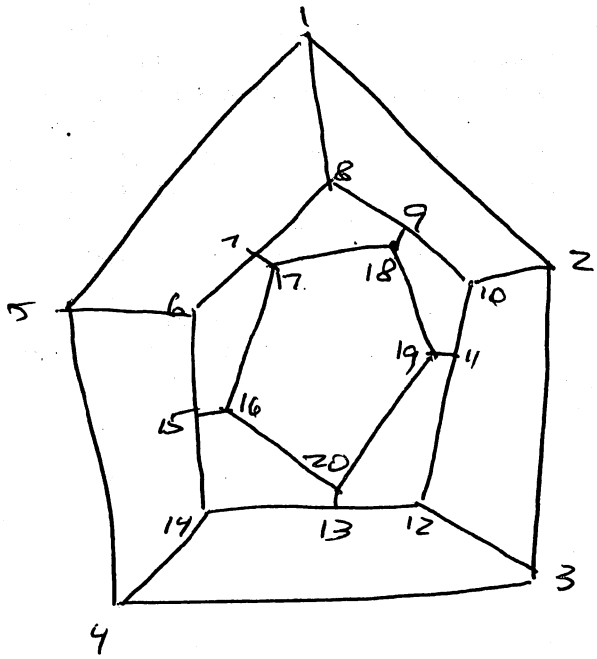
Lines 5130 to 5220 specify probability factors that determine whether certain events will occur during game play. All factors can have a probability from 1 percent to 100 percent by specifying values from 1 to 100. These events will be tested for occurence for every move the player makes during the game.

5130. Arrows shot will hit superbats.
5140. Arrow shot will fall into a pit (hint: tells where a pit is).
5150. Wumpus will move on next turn.
5160. Bats will migrate to a new cave.
5170. You find a throwing rock for which to kill the Wumpus.
5180. Rats invade the caves and eat half your supplies.
5190. Rock slide occurs and closes a tunnel.
5200. Player finds extra supplies.
5210. A rock thrown at a Wumpus will kill it.
5220. A shot arrow that goes wild will kill you if you're hit.

Module #14 : SuperWumpus (P2)

The 20 lines starting from 5270 are the connections for the 20 caves. Line 5270 represents the connections for cave 1, line 5280 for cave 2, etc., up to cave 20. For each cave, there are 3 tunnels that lead to other caves. The fourth value of each line in for each cave is always 0. The first 3 data statements on each line are the tunnels leading to other caves. When you make your own custom caves, you must make each tunnel connect to a cave number from 1 to 20. A cave CAN connect to itself. For example, cave 1 can have a tunnel lead to cave 1. In this case, you would be creating a dead-end tunnel, effective for making harder mazes. The normal cave pattern for this module is a squashed dodecahedron, which is a 12-sided cube. The graphical representation of the default maze is given below.

You are free to create any pattern you want. Just make sure all 20 caves can be reached by the player (ie. include all caves in your design). This game was released on the library disk after the official OverLink release. Please notify us of any problems with this module.



Miscellaneous Files (P1)

There are a few files on the OverLink disk that are not considered modules. Each will be explained, in the order they appear on the disk.

NOTICE. This program, when loaded ,8,1, gives the copyright notice for the OverLink.

YELLOW PAGES. This is a very handy public domain utility that will re-organize your disk directories. The program was not written by D.S.S. You should use it to re-organize your directories. A good idea would be to remove the "--" seperators from backup copies of your OverLink disk to make the directories shorter and faster. All instructions for that program can be found within the program. We recommend it as an excellent disk utility.

CWI.COM. This program is the one that makes up the "++xc" CWI file. The "++xc" file contains the command definitions for the CWI. The function of this file was explained at the beginning of this manual.

INSTALL DRIVES. This is the file that creates the DRIVE TABLE file, the one that is added to your disk #2 along with the standard V3.0 configuration files. Only "DRIVE TABLE" belongs on disk #2, not INSTALL DRIVES. The function of this file was outlined previously as well.

M.GVC.PAL -PAL source code file for MODEM.GVC 1200.
M.TTEL.PAL -PAL source code file for MODEM.TOTEL TEL.
M.1650SLOW.PAL -PAL source code file for MODEM.SLOW 1650.
M.1670-2.PAL -PAL source code file for MODEM.1670.2.

MODEM.GVC 1200 -Modem file for the GVC 1200 Super Modem.
MODEM.TOTEL TEL -Modem file for the Westridge/Total Telecommunications Modems.
MODEM.SLOW 1650 -Modem file for a Pocket Modem or 1650 compatible that has a slow response to answering calls.
MODEM.1670.2 -Modem file for the 1670, a new and perhaps(?) better version.

The source code was written with the ProLine PAL 64 assembler. With some changes, the source code will run on other assemblers, but we recommend this one. The object code files are your standard modem files, which you should have read about in the V3.0 BBS manual. The source code format follows that which was outlined in the Technical Notes section of the manual. If you find that your 1650/Pocket modem is answering the phone in strange ways with the old modem file, try the new one above. It introduces a slight delay after going on-hook before accepting incoming calls. If you find that your 1670 does not like to answer the phone frequently, try the above modem file. It sends the AT command sequence string twice, in case something garbled it the frist time it was sent. If you have a Hayes compatible modem, *like* the GVC 1200, you may use the above GVC modem file. Hayes modems use inverted carriers, and this file handles that properly. If your modem supports DTR, you can add a direct DTR disconnect to the hangup routine. For example, LDA #0:STA 56577 will disconnect the caller.

Miscellaneous Files (P2)

CONVERT.DLM - Delimiter file converter.
BBS-TXT.DLM - BBS V3.0 text file, in delimiter (.DLM) format.

Delimiter files, mentioned under Module #10, will be explained once more. The file BBS-TXT.DLM is, in essence, the BBS text file, which is named BBS.TXT on disk #2. The difference is that the BBS-TXT.DLM file has been put in a format that you can use with any text editor or word processor. The normal sequence of text prompts for a normal text prompt file, such as BBS.TXT, would be:

```
NULL
Text Prompt 0
NULL
Text Prompt 1
NULL
....
Text Prompt N
NULL,NULL,NULL
```

The NULL is a null byte, which is value 0. These zero bytes are used to separate prompts on the BBS. Each prompt starts with a NULL, and ends with the NULL of the next prompt, as you can see above. The very end of the file, after N prompts, would contain 3 NULLS, which tells the BBS that this is the end of the text prompt file. NULLS have no real value as input as the letter "a" would have, for example. Therefore, it is impossible to put these nulls in the file with any text editor. Delimiter files do not use nulls. Instead, the characters "@"+CARRAIGE RETURN make up the text prompt separator. The above sequence would thus be, in delimiter format:

```
@(RETURN)
Text Prompt 0
@(RETURN)
Text Prompt 1
@(RETURN)
Text Prompt 2
@(RETURN)@@
```

Note the ending of the text file: @+RETURN+@+@. These MUST always be the last 4 characters of the text file. To see a proper .DLM file, load up the BBS-TXT.DLM file in any text editor. Note that the files you make do not have to be named .DLM like a .ARC file with ARC. It's just there for notation. By using this format, you can bypass the CREATE.TXT program on disk #2. Also, you may add color to your text prompts without having to use TEXT.PAL and an assembler. CONVERT.DLM will convert your delimiter files to a more standard text file that the BBS can use. It will replace the @(RETURN) character pair with the NULLS it needs. Your delimiter file will be replaced with the proper file when it is run, so always use a BACKUP of your delimiter file with CONVERT.DLM.

Load Adress: 39424

Miscellaneous Files (P3)

RELATIVE COPY - This program is an update to the file COPY.REL1 on disk #2 of the BBS. This version only differs from the previous version in that you may specify a wider range of device numbers for copying the files.

CHARSET CONVERT - This small BASIC/ML program allows you to manipulate the character sets that the DarkTera series of terminal programs and the BBS use as character sets. These include the 6 character set files on disk #2 (CHARSET x). Any character sets you've used on DarkTera Versions 2.0, 2.1, 2.2, 3.0, 4.x (x=0,A,B,C,X) will work with this program. Using this program means that you do NOT have to use a machine language monitor to merge the character sets, or a disk doctor to set the load address to $E000 (57344). This program will convert character sets to DSS Terminal/BBS format or will convert DSS Terminal/BBS format files to normal character set format, in two 9 block files. Complete instructions are provided within the program. They will not be repeated here.

+HE.2 - This is a sample help file for the new expanded file section. Use it as a replacement for your old +HE.2 file, or use it as a reference when writing your own.

NOTE: Any late changes made to the OverLink disk (ie. late bugs, or added modules) will appear after this last page. These changes may or may not be documented in the table of contents. If you have any questions regarding this OverLink system, or the V3.1 system, don't hesitate to write us, or call our BBS Hotline ((416)-445-6788). As of this writing, we are currently working on a special high-speed version of the BBS that will run exclusively on the Xetec Lt. Kernal 20 megabyte hard disk drive. Stay tuned for further news.

CLOSING COMMENTS

Well This Wraps Up Our First Expansion Library Disk!!! This Should Keep You Busy For a Little While!!!

Development is UnderWay to Support the Xetec Lt Kernal 20 MegaByte Hard Drive! Once Done, We Will Tackle The ICT DataChief 20 MegaByte Hard Drive!!!

We Will ReDesign the Message Base, Incorporating NetWorking and AutoMessage Forwarding Between 1200 Baud Boards in the Near Future! HopeFully on OverLink Library #2!!!

Also, One Final Note to All Programmers, If You Have Written Any Modem Files For the BBS or Term, Please Send Them Along So That We Can Make Them Available to the Others!
All the Best in '87!!!
Andrew Leaver
President, D.S.S.