QSD UTILITY DISK
FOR
LDOS©

MODEL 1 ONLY

* *

Increase your programming power and productivity when using LDOS! Here is a VERY brief synopsis of each program contained on the enclosed disk:

BINHEX/CMD by Tim Mann - a powerful Binary/Hex converter.
CHANGE/CMD by George Somers - re-route your drive logic.
CLONE/CMD by George Somers - "clone" files :1 :1 or :2 :2.
DCAL/CMD by Tim Mann - calibrate and check your drive speed.
DVORAK/CMD by Kim Watt - gives you the famous DVORAK keyboard layout.
LCOPY/CMD by Kim Watt - Super fast copies and mass transfers!
LPURGE/CMD by Kim Watt - a "mini" Super Utility(c) for LDOS! (This program has 23 seperate functions!)
LZAP/CMD by Frank Luff - the most sophisticated "zap" program ever written, and this is JUST FOR LDOS! Very powerful, yet keeps the beginner in mind with a "HELP" command. Works in single & double density. 35,40,77,80... whatever # of tracks! Even 8" drives! Hard disk too!!
RESCUE/CMD by Kim Watt - intelligently recovers "killed" files.
RESIDELD/CMD by Earle Robinson - lets you "SYSGEN" system files for faster execution and increased disk space.
STAT/CMD by Skip Taylor - gives you a "status" report on your system.
VDISK/CMD by Skip Taylor - a disk verify routine to check for bad sectors, etc. Gives you an idea if it's the disk or hardware at fault! Works in double density also.

- -

* also included are several drivers and "fixes" to enhance the operation of LDOS 5.0.2.

- -

LDOS is a trademark and is copyrighted by Logical Systems, Inc.

QSD Utility Disk #1 for LDOS 5.0.2 (Mod I only)

Copyright 1981 - All Rights Reserved

Table of Contents

Program Name	Page #
BINHEX/CMD	1
CHANGE/CMD	2
CLONE/CMD	4
DCAL/CMD	6
DVORAK/CMD	7
FILTER DOC includes: UPCASE/FLT and ADDLF/FLT	8
LCOPY/CMD	9
LPURGE/CMD	12
LZAP/CMD	16
PATCHES and FIXES DOC	24
RESCUE/CMD	30
RESIDELD/CMD	31
STAT/CMD	34
VDISK/CMD	35

DISCLAIMER OF WARRANTIES & LIMITATION OF LIABILITIES

We have taken great care in preparing this product to ascertain its effectiveness. We make no express or implied warranty of any kind with regard to this manual or to the Utility Disk. In NO event shall we be liable for incidental or consequential damage in connection with or arising out of the performance or use of any of these programs.

QSD Utility Disk #1 Copyright 1981 by:
Quality Software Distributors

All rights reserved. No part of this documentation may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by an information storage retrieval system, without expressed permission in writing from the publisher. (QSD)

Thanks to all the authors involved in this project. They are all outstanding! Also thanks to Bill Schroeder and LDOS Support for their help on various subjects.

A special thanks to Kim Watt, Frank Luff, and Earle Robinson.
Documentation edited and enhanced by Dennis Brent.

LDOS is a registered trademark of Logical Systems, Inc.
TRS-80 is a registered trademark of Tandy Corp.

BINHEX/CMD by Tim Mann* (c)1981

---

* transposed to BASIC/S compileable format by:
  Bill Stockwell and Renato Reyes.

Purpose of program:

BINHEX/CMD is a conversion program to go from BINARY to HEX and back again. It is quite useful in data communications. It is completely self prompting and quite easy to use. This program appeared in BASIC format in the first LDOS newsletter.

To run BINHEX/CMD type:

BINHEX <enter>

from then on, just answer the questions!

CHANGE/CMD Version 1.Ø by George Somers (c)1981

BRIEF DESCRIPTION:

CHANGE/CMD is a utility written exclusively for the LDOS operating system that provides the user with the ability to ROUTE drives in a manner somewhat similar to ROUTEing of devices like *PR and *DO, etc.

It is thoroughly error-trapped and is intelligent enough to know when the program is being used under the LOPO interface (LX8Ø interface users are completely supported in this respect). It will also error trap an attempt to ROUTE a device that is not already ENABLEd in the DEVICE listing. All errors result in no change to the DOS environment and subsequent exit is to DOS at its transfer address of 4Ø2D Hex.

A typical syntax:

------> CHANGE 1 Ø

This will cause Drive 1 to be removed from the DEVICE table and replaced by Drive Ø. This means that all output and input intended for Drive 1 will now be sought from or sent to Drive Ø. A DIR :1 will reveal an actual Directory of Ø. Similarly any and all activity involving Drive 1 will be sent to Drive Ø.

The possibilities resultant from such an ability are too numerous to list! For instance, 1 drive users may now run many BASIC programs intended (and written) for multi-drive systems. All data writes and reads for drives OTHER than Ø will be ROUTEd to Drive Ø. Assembly language programs that work within the DOS system or use DOS and ROM file READ and WRITE code will be interpreted in a like manner. Some assembly language programs feature DOS-independent code. These programs will be unaffected by the CHANGE command, as indeed they are by any alterations to the SYSTEM configuration.

Another feature would permit commands like COPY to no longer protest about Destination and Source diskettes being the same. For instance, if Drive 2 has been routed to Drive 1 (CHANGE 2 1), the following syntax: "COPY FAKE:2 :1" will cause LDOS to over-write FAKE with FAKE (all on Drive 1!). Think about it for a second. Updating date stamps is possible in this way.

Thus, the CHANGE feature moves the LDOS operating system one step closer to total device independence by permitting each drive to be viewed more as an individual device, instead of just as one component in a composite device.

Of course, with such added power comes the ability to make powerful errors. Be careful!

The CHANGE command will NOT permit you to ROUTE a drive that has not already been ENABLEd in the DEVICE table. Remember that any number of non-existent drives from 1-3 (1-7 for LX-8Ø users) may be

present in the DEVICE table and subsequently re-ROUTEd by the CHANGE command. Some commands may appear strange to the user after some creative drive ROUTEing. For instance, if only one drive is present in the system but 3 other drives have been ROUTEd to Drive Ø, a simple DIR will read Drive Ø four times in succession. Similarly for DEVICE or FREE, etc.

CHANGE/CMD will also not allow the user to ROUTE I/O intended for Drive Ø to another drive (which would make all calls for SYSTEM overlays reference a drive down-line). Of course, even such a strange route as this could be useful in some limited way, but it would demand such a great amount of care on the user's part, that its possible gains are outweighed by its considerable potential for utter confusion (possibly, even disaster!). Thus, an example syntax of CHANGE Ø 2 will abort to DOS level with NO CHANGE.

Final note:

As the types of changes caused by CHANGE/CMD are drastic (albeit in one sense very superficial and high-level), it is suggested that, when a particular drive-route scheme has served its purpose, the user re-BOOT to establish the normal drive I/O configuration. The LDOS command RESET will NOT restore the drive I/O configuration to that previous the CHANGE! You MUST RE-BOOT!!

As with all device ROUTE-ing, it is possible for a user to create such a tangled web of disk ROUTE-ing, that the operating system may become totally confused. Think out the CHANGEs before you issue commands. This may prevent some freak tragedy from occuring on that one-of-a-kind file some day.

CLONE/CMD Version 1.3 by George Somers (c)1981

PROGRAM DESCRIPTION:

This utility offers owners of more than one floppy disk drive the ability to perform file copies on a drive other than drive Ø as both source and destination.

For instance, the hypothetical program BLITZ/CIM is on data diskette # 1. The user however desires to transfer it to data diskette # 2. The LDOS COPY command cannot be used in this case, and the XFER command requires that you keep removing and inserting the SYSTEM diskette, which can be quite a nuisance. CLONE/CMD will copy this file with only two insertions and one removal necessary.

```
.............................................................
: VALID SYNTAX :  The syntax examples presented below       :
:               assume the hypothetical filespec BLITZ/CIM. :
:                                                           :
: ------> CLONE BLITZ/CIM:1                                 :
:         (Copies BLITZ/CIM from Drive 1 to Drive 1)        :
:                                                           :
: ------> CLONE BLITZ/CIM:2                                 :
:         (Copies BLITZ/CIM from Drive 2 to Drive 2)        :
:                                                           :
: ------> CLONE BLITZ/CIM:3                                 :
:         (Copies BLITZ/CIM from Drive 3 to Drive 3)        :
:                                                           :
.............................................................
```

INVALID SYNTAX (and WHY!):

```
------> CLONE BLITZ/CIM:Ø
        No Drive Ø copy permitted. Use XFER command.

------> CLONE BLITZ/CIM
        Because Drive Ø is assumed this is not permitted either.
        Use XFER command.

------> CLONE BLITZ/CIM:Ø TO :1
        CLONE/CMD does not perform multi-drive copies and, besides,
        this will appear to CLONE merely as a CLONE on drive Ø, a
        function that is not supported.

------> CLONE BLITZ/CIM:1 TO :Ø
        CLONE will treat this as a copy of the program on drive 1.
        What follows is simply ignored.
```

NOTE:

If the DESTINATION disk happens to contain the identical filespec as the SOURCE disk, CLONE/CMD will not abort its copy; it

will ask the user if he (or she) desires to over-write the file. If the user responds "N", then the copy will abort. If the user responds "Y", then the copy will proceed and over-write the DESTINATION file.

CLONE/CMD is also useful in adding a time stamp to a file that has none (or an old one). Simply issue the "DATE mm/dd/yy" command and CLONE the file allowing the identical disk to be both SOURCE and DESTINATION. CLONE/CMD does not reject the disk for this reason, as both COPY and XFER do.

Postscript:

CLONE/CMD was originally written for the VTOS operating system. LDOS has now implemented the XFER function for all drives. In this light, CLONE may seem a bit redundant. Why bother??? (you might ask)

(1.) XFER will not permit the user to COPY a file over itself (useful in correcting or adding a time stamp to a file).

(2.) XFER is slower (more disk-swapping)

LIMITATIONS:

Drive Ø copying is not supported by CLONE/CMD. This function is already performed adequately by the XFER command. You will abort to DOS and be told to use XFER. By the same token, multiple drive copies are not supported either.

This utility has one other limitation; that of memory size. Should the source file's length be beyond the capacity of memory, CLONE/CMD will inform the user of that fact and abort the copy function. In this case, some other means of copying the file must be chosen (if possible).

Final note:

The use of CLONE/CMD is NOT restricted to the VTOS and LDOS operating systems. It works on EVERY DOS released as of June 1, 1981.

Because it uses only "standard" DOS calls (like OPEN, INIT, and CLOSE) it should work on all future DOSs that follow these very established standards.

DCAL/CMD by Tim Mann* (c)1981

---

* transposed to BASIC/S compileable format by:
  Bill Stockwell and Renato Reyes.

Purpose of the program:

DCAL/CMD is a disk drive calibration utility to inform you of what speed your drives are turning at. It is accurate enough to adjust timing with, and if there is a speed problem, you will be informed of this fact. It shows exact speed, and a "Smoothed" speed, which is exact speed averaged out.

To run DCAL/CMD type:

DCAL <enter>

From then on, you will be prompted for all info. You will be asked for the drive you want to check. Answer with a number, hit <enter> and the rest is automatic! This program appeared in BASIC format in the first LDOS newsletter.

DVORAK/CMD by Kim Watt (c)1981 - Breeze Computing

This is the famous DVORAK keyboard layout! It is VERY easy to implement, and toggle back and forth between QWERTY and DVORAK. If you are a touch typist and are looking for ways to pick up extra words per minute... then DVORAK is the way to go! For more information see the cover story from the December 1980 issue of 80-Microcomputing Magazine. The history and logic behind the DVORAK keyboard layout is explained in full detail there.

Just a few tidbits from the article mention that DVORAK has been around since the early 40's and that DVORAK users commonly type 100 wpm while the average QWERTY typist does good at half that speed.

There are several ways to load the program. The easiest and most direct is by typing in:

DVORAK <enter>

The default here is: DVORAK layout

You may also type:

DVORAK Q (to load the driver, but not have it active.
QWERTY is still in operation here.)

To toggle from one to the other type:

<clear> and "0" at the same time.

To escape completely from DVORAK/CMD, and reclaim memory used, hit the <clear> key and the <break> key at the same time. A message will be displayed showing you that this indeed has happened.

* This is NOT the same program that was in 80-Micro. That was the "classis" DVORAK keyboard. The version on this disk is an ENHANCED layout and includes brackets < >, and all other keys you need. If you have the "old" style keyboard, then you may "lift" off your keytops with a paper clip, and re-arrange them DVORAK style. DO NOT DO THIS IF YOU HAVE THE ALPS (new style) KEYBOARD!! If this is your situation, then you may purchase a sheet of dots at an office supply and just stick them on the keytops.

Filters for use with LDOS by Tim Mann (c)1980,81

UPCASE/FLT

UPCASE/FLT translates lower case to upper case.
Works on either INPUT or OUTPUT devices

Activate by typing:

FILTER <devspec> USING UPCASE <enter>

ADDLF/FLT

This is a filter to add a line feed after every carriage return when output is done to a device.

Line feeds sent to the device are also translated into <CR> <LF> sequences.

Activate by typing:

FILTER <devspec> USING ADDLF <enter>

LCOPY/CMD by Kim Watt - (c)(p)1981 Breeze Computing

LCOPY/CMD is a multiple file transfer program that allows you to move MANY files simutaneously from drive to drive with little effort by the user. It only works on single density diskettes, but it is VERY fast. It is MUCH faster than a BACK-UP BY CLASS. Do NOT use it, however, to create LDOS system disks.
Here is how it works:

You may load in the program by typing:

LCOPY/CMD <enter>

from LDOS READY to be prompted for all parameters.

First prompt is the SOURCE drive from Ø-3 (No need to hit ENTER)

Second is the destination drive from Ø-3 (they may be the same)

Next is the paramater line:

You will see this line displayed on your screen:

Parameters (H,V,I,S,P,A,O,X,Ø-3) ?

You may now hit the command letters one by one. If you hit a wrong one, then just hit the correct one. A new command supercedes the previous one if the same. When you are through entering commands, then hit <enter> to continue with the program. "H" is the "HELP" command.

A common command entry might be:

1 (for 12 msec. copy)
A (for ALL files) -remember that VISible is the default-

Now hit <enter>
You will be prompted the rest of the way.

Or when you see the command line, just hit enter, and you will be prompted the rest of the way through the program.

Here is an explanation of each letter and what it does:

Your commands may be any or all of the following:

= = = =

X = RESTART the program
A = copy ALL files as specified, with NO file prompting
O = OVERWRITE files if found on destination disk... NO prompting
Ø-3 = set the step speed to 12, 12, 20, 40 ms respectively.
H = HELP!! Displays this list on your monitor:

<H>elp, print this text.
<V>isible files.
<I>nvisible files.
<S>ystem files.
<P>rompt for disk mounts.
<A>ll, copy all files indicated, no prompting.
<O>verwrite files if found on destination disk, no prompting.
<X> Cancel and restart program.
<Ø-3> set step speed to 6, 12, 20, or 40 milliseconds.

After answering the source and destination prompts, and giving your commands, the drives will be checked to see if there is enough room on the destination disk for everything. If there isn't, then you will be informed of this situation, and will be prompted file by file for the copy. After answering all the questions, the transfer will happen 1-2-3. (Quick!)

Examples of Syntax once you "learn" the program:

Once you get used to the program and become a "pro", you may also specify the parameters from the LDOS command level. You will also notice that it doesn't matter if you are using upper or lower case, or even a combination of them.

Examples:

LCOPY Ø 2 (copy visible files from Ø to 2, no prompting)

lcopy Ø Ø (copy visible files from Ø to Ø, with prompting)

lcopy Ø 3 V i 2 (copy visible, and invisible files from Ø to 3 using step speed of 2Øms)

lcopy Ø (copy visible files from drive Ø, all other parameters are prompted)

LCOPY (all parameters are prompted)

lcopy,Ø,2,V,Ø,I,S (copy ALL files from Ø to 2, using step speed of Ø, copy visible AND invisible)

LCOPY,Ø 2,I,P (copy only invisible files from Ø to 2, all disk mounts are prompted)
note:
You will be prompted for each file to be copied. Answer Y or N (enter alone implies N). If the file exists on the destination disk, you will be prompted if it is to

OVERWRITE or not.

LCOPY Ø,1,Ø,1 source Ø, dest 1, step 1, (Ø is overwritten)

LCOPY 3 Ø V P 2 W source 3, dest Ø, visible files, prompt for disk mounts, step speed 2, prompt for more params (W)

LCOPY,2,3,3,W,V,I source 2, dest 3, step 3, prompt for more params, (the V and I are not executed, the W command forces a prompt for params)

LCOPY Ø12PVI this form may be used too... source Ø, dest 1, step 2, mount prompts vis and invis files (everything but system)

LCOPY Ø1 ,I,, P Ø,,,,,,,,,,,,,V
this is just to show you that LCOPY COPY is not picky about the parameters!

LCOPY Ø1, IVP,,,,,,,,,,22222222222PPVVW,I V
This example will also work...
source Ø, dest 1, vis, invis, mount prompt and step speed 2. Each time a command is issued it will supercede the last command. In this example, the I and V at the end would be ignored because they follow a W, but they were already set in the beginning of the line.

The program runs with interrupts on, so the ALIVE, CLOCK, and whatever else is going will be active. The program will also honor TOPMEM at 4Ø49H, and uses the DOS's internal stack area. LCOPY provides a very FAST copy. There is nothing that will come close to it's speed on a 2 drive copy. On a single drive copy, you are always limited to speed by the number of files, but the speed of this program is SOMETHING else!

Note:
If you specify the P option for prompting, you may turn it off at ANY prompt by pressing N for "no more".

Also:
You may exit the program by hitting the <BREAK> key.

One final note:

You may ask yourself," Why does it copy so fast, and is that necessarily good? What's happening here?"

Good question! When using LDOS (or any other system) to COPY files, the data is copied, and the directory is updated, back and forth, etc. Kim transfers ALL data first and updates the directory at one time at the end of the COPY. There is nothing wrong with this, and will indeed save you quite a bit of time. The directory is still 1ØØ% LDOS compatible.

LPURGE/CMD by Kim Watt (c)(p)1981 - Breeze Computing

Enter the program by typing:

LPURGE <enter>

Next enter the drive which contains the TARGET diskette. It may be located on ANY drive. The directory will be loaded from the disk and its track will be displayed. The directory is kept in a buffer and the changes to it are IN- MEMORY. Only when YOU are ready, and instruct the program to do so (option <K>), will the actual directory be written to and changed. Whenever the directory is being displayed you may press <ENTER> to display the menu selections. Whatever kind of cursor you are using (if any) will carried over to LPURGE.

The directory is displayed as follows:

All of the 64 available entries may be displayed at the same time. Four entries per line for 16 lines.

Immediately following the name of the file will be a space, and the following two letters (if they apply):

S - will be displayed if the file is a SYSTEM file.

I - will be displayed if the file has the INVisible attribute.

After these letters, all files will show a number. This is the PASSWORD PROTECTION LEVEL of the file. If the number is zero (Ø), then there is NO ACCESS RESTRICTIONS.

Menu Commands:

The following will discuss the different options that are available from the MENU display. The MENU options will appear on your screen after telling the program what drive you want to work with, and hitting <ENTER>.

Here is what will appear on your screen...

```
** QSD LPURGE Utility  **

<A>  SELECTIVE purge
<B>  Kill all VISIBLE files
<C>  Kill all INVISIBLE files
<D>  Kill all INVISIBLE, NON-SYSTEM files
<E>  Kill all  /CMD  files
<F>  Kill all  /BAS  files
<G>  Kill all SYSTEM files
<H>  Kill ALL files EXCEPT BOOT and DIR
<I>  Zero out all UNUSED DIRECTORY ENTRIES
<J>  Zero out all UNUSED DISK SECTORS
<K>  WRITE UPDATED DIRECTORY back to disk
<L>  Display current DIRECTORY
<M>  MORE Routines
<N>  RESTART program
? _
```

Explanation of Commands:

Option <A>

will allow you to look through the directory and SELECTIVELY kill any files. An option is also provided to "jump" over the remaining entries to quickly bypass a long directory. (press the letter "N" for "NO MORE")

Options <B> through <H>

are self explanatory

Option <I>

When a file is killed with LPURGE, only the FIRST byte of the DIRectory entry is zeroed. Use option <I> to ZERO OUT ALL current UNUSED ENTRIES in the buffer. This will make it hard to locate ANY killed files. When a file is "killed", its DATA is left on the disk, only the DIRectory is changed.

Option <J>

Use OPTION <J> to zero out ALL current unused disk sectors.
When using this option, the changes are IMMEDIATE and IRREVERSABLE. Combine this with option <I>, and a "killed" file will indeed be gone!

```
.............................................
:  MAKE SURE THAT THE DIRECTORY IS EXACTLY AS :
:  AS YOU WANT IT BEFORE USING THIS OPTION!!  :
.............................................
```

When all unused sectors have been zeroed (in-memory), a "ROUTINE COMPLETED" message will appear. Press <ENTER> to write the DIRectory back to the disk. If you want to zero the sectors, but do not want the directory written back to the disk, hold down the <CLEAR> key when you press <ENTER>.

Option <K>

Writes the updated directory to disk. Once done, this is irrevocable! Check the directory before hitting <K> to make sure it appears just as you want it!

Option <L>

will display the CURRENT DIRectory that is in the buffer.

OPTION <M>

M means MORE in this case! More Menu Selections...
When you choose option <M>, these is what you will see:

Note:
All of these commands are self-explanatory and self-prompting.

-SPECIAL ROUTINES-

1. Make all files VISIBLE
2. Make all files INVISIBLE
3. REMOVE PASSWORDS from all files
4. Change disk NAME, DATE, and MASTER PASSWORD
5. Change AUTO command
6. RANDOM PASSWORDS to all files
7. COMPUTE PASSWORDS for all files
8. Compute Passwords for SINGLE file
9. Change Passwords on SINGLE file
Ø. Return to MASTER MENU
?

OPTION <N>

Restart the program. The Buffer will be lost and the directory will NOT be written back to the disk. This way you may cancel any changes to reload the existing directory as it was when you started.

ALL OF THE ABOVE COMMANDS ARE TEMPORARY EXCEPT OPTIONS <J> AND <K>.
Be Careful!

LZAP/CMD by Frank Luff (c)1981

Introduction

LZAP/CMD was written specifically for LDOS and will only run under the LDOS operating system. LZAP uses the resident disk driver of LDOS and hence, enjoys its flexibility and device independence. It makes extensive use of the the DCT (Drive Code Table) and will dynamically configure itself to whatever the drive capacity or diskette format. LZAP/CMD will support drives from Ø to 7, any cylinder count to 255, and any sector number to 255. It supports both single and double density recording formats automatically. Various drive sizes and densities can be intermixed and LZAP/CMD will page between them without error. Lower case input is supported.

```
.................................
:  LZAP works BEST when using   :
:  TYPE-AHEAD!  Please use it   :
:  that way for best results.   :
.................................
```

Command Mode:

1. (B)ackup

Enter B to make a mirror image of the target diskette for data recovery. The destination diskette must be formatted identically to the source diskette or Backup will abort with the error message "Source and Destination disks are NOT compatible" (e.g. if the source diskette is double density 41 tracks, the destination diskette must be double density 41 tracks). Backup will support disks with up to 36 sectors. The copy is performed a full cylinder at a time. NOTE: It is possilbe to backup a diskette to itself. I haven't found a use for this, but backup will do it correctly.

2. (R)ead

This is the primary display mode. After entering R, you will be prompted to "Enter drive # (Ø to 7)". If the drive entered is not ready or nonexistent an error message will be posted and you will again be prompted to enter drive number. When the drive number is entered, the drive is issued a Check Drive command. This will update the DCT (Drive Code Table) to the configuration of the inserted diskette. If the diskette has a damaged BOOT/SYS sector Ø or DIR/SYS sector Ø it will not initialize properly. See below examples of data recovery for how to correct this.

After the drive number has been entered, you will be prompted to "Enter Cylinder #". This prompt will accept any hexadecimal number up to FF. Pressing < ENTER > will default to ØØ, but only when this

rountine is entered from Command Mode. A check is made of the DCT and any entry greater than the drive capacity will cause the error message "Requested cylinder GREATER than drive capacity", and again issue "Enter cylinder #".

Next, the sector is asked for. This prompt behaves the same as the cylinder prompt.

The program will then enter the Display Mode.

3. (D)ebug

Entering D will cause immediate entry into the LDOS Debug utility. You may return to LZAP, by entering G7000, if the memory occupied by the program has not been altered.

4. (E)xit

E will terminate LZAP, clear the screen, and return to LDOS Ready.

5. Error Dictionary (Y)es or (N)o

The Error Dictionary refers to the error traps in LZAP and within LDOS. Y will turn it on and display "ON". N turns it off and will display "OFF". With it off it is possible to read diskettes ordinarily not readable, such as diskettes with a damaged BOOT/SYS or DIR/SYS and some protected diskettes that use a read lock or deleted record address mark. Although it may become necessary to turn it off to recover some damaged diskettes, it is highly recomended that it normally be left on. Some of the automatic feartures such as G (go directory) and F (find file) will be disabled with the dictionary off. If you turn it off, you are on your own and you better know what you are doing.

Display Mode:

Display mode is entered by the (R)ead command. The screen will display, from left to right: the first verticle column of one digit equals the current drive address; the next two digit column is the current cylinder; followed by the two digit current sector; the next two digit verticle column is the relative byte number of the starting byte of that row in the hexadecimal and ASCII display fields; the next 16x16 two digit per byte block is the hexadecimal representation of the sector data buffer; this is followed by a 5 character wide, blank verticle bar, refered to as the message area; next is the 16x16 byte ASCII display field.

The message area will be used to display the current mode, error messages, and the hash code.

The following commands are availible:

The following commands are availible:

1. <A> ASCII modify

The keyboard performs like a typewriter to enter ASCII data into the sector buffer. The full ASCII set supported by the keyboard may be entered, including lower case. The byte in the sector buffer and display buffer is changed on entry of the character and the cursor is advanced automatically.

The arrow keys may be used to position the cursor over any location. Entering a <CTRL> M will escape to the Hex Modify mode. This will allow toggle between ASCII modify and Hex modify. A <CTRL> W will home the cursor and exit to Display mode. A <CTRL> X will escape to Command mode.

2. <C> Compare

Entering a C will clear the screen and issue the prompts "Compare to:", "Enter Drive# (Ø to 7)", "Enter cylinder #", and "Enter sector#". The same checks as made under the (R)ead command will be made at each entry. Default to ØØ is not allowed. The compare may be made to any valid drive, cylinder, and sector.

After the entries have been accepted, the previously displayed sector will be compared, byte for byte, with the just entered sector number. In the case of a non-compare, the comparing will halt, the byte in error will be displayed and the option will be given to continue or terminate.

Terminating will return to the previously displayed screen, continuing will continue the compare operation. Upon completion of the compare, "ALL DONE!" will be displayed, and the program will return to Display mode, displaying the previous sector.

3. <D> Debug

This entry will cause immediate entry to the LDOS Debug utility. Entering <CTRL> D will exit LZAP, load the extended Debug and return to LDOS Ready. Type G7ØØØ to re-enter LZAP, if the program has not been altered.

4. <E> Exit

E clears the screen, exits LZAP, and returns to LDOS Ready.

5. <F> Find file

Position the cursor anywhere in the directory entry of the desired file and press F. The next display will be relatve sector Ø of that file. This command is enabled only in the directory cylinder. It is also disabled if the Error Dictionary is turned off.

6. <G> Go directory

Pressing G will cause the display to go to relative sector Ø of the directory cylinder (DIR/SYS). This command is disabled if the Error Dictionary is off.

7. <H> Hash code

Position the cursor anywhere in the directory entry of the desired file and press H. This will calculate the hash code of the file for entry in the HIT (Hash Index Table). The HIT sector will be read and the hash code will be inserted in the proper location. The HIT sector will be displayed with the cursor positioned over the new hash code. This location is called the DEC (Directory Entry Code). The hash code will also be displayed in the message area of the screen.

It will be necessary to write the HIT sector back to the disk to make the change permanent.

8. <L> Locate

After pressing L, the keyboard will be scanned for two hexadecimal digits. When they are entered, the cursor will be positioned over that relative byte number in the display buffer. At each of the keyboard scans, a check is made for the character X to abort Locate and return to Display mode.

9. <M> Hex modify

The value of any byte in the sector data buffer may be changed in the Hex modify mode. The keyboard will be scanned for two hexadecimal characters at each cursor position. After the first character is entered, the display buffer will be changed and the cursor will be frozen until after entry of the second character. After entry of two characters, the sector data buffer will be modified and the cursor will advance automatically.

Normal exit is <CTRL> W after entry of the second character for that cursor position. The cursor will be homed and the program will return to Display mode. Entering a <CTRL> X will return to Command mode. Entering a <CTRL> A causes entry into ASCII Modify mode, allowing toggle between Hex and ASCII modify. The arrow keys may be used to position the cursor over any byte in the buffer, if you are not between characters modifying a byte.

If you exit between the first and second characters, the program will escape the Hex modify mode but, the sector data buffer will remain unchanged for that byte.

1Ø. <P> Screen print

Entering a P will cause the display buffer to be printed on the parallel port line printer. A check is made for printer present. Pressing P without a printer attached to the system will be ignored. Pressing < BREAK > will abort the print out and return to Display mode. If you SPOOL *PR before running LZAP/CMD control will return almost immediately.

11. <R> Remove password

Position the cursor anywhere in the desired directory entry and press R. This will do four things: allow full access to the file, change the password to all blanks, make the file visible, and if the file is inactive (i.e. killed), make it active. Other attributes of the file are not changed. Note: You must write this sector back to the diskette for it to become permanent.

12. <S> Search

The keyboard is scanned for two hexadecimal numbers to use in a byte by byte compare, from the current cursor position to the end of the sector data buffer. If the byte is found, the cursor is positioned over the byte in the display. The program will return to Display mode. If the byte is not found from the current cursor position to the end of the buffer, "NOT FOUND" will be displayed, the cursor will be homed, and the program will return to Display mode.

At each of the keyboard scans, a check is made for X, to abort Search and return to Display mode.

13. <W> Write

W causes the sector buffer to be written to the diskette. The option will be given to write to the current drive, cylinder, and sector, or to change the destination. If you elect to change the destination, you will be prompted to "Enter Drive#", "Enter Cylinder#", "Enter sector#". The usual checks are made for valid entries. Default to ØØ is not allowed. At completion of the write command, the destination drive, cylinder, and sector is displayed.

14. <X> Escape

X Returns to Command mode.

15. <Z> Zero

The sector data buffer and display buffers will be zeroed.

16. <Ø> to <7> Page Drive

Read the current cylinder and sector from that drive number. The drive will be issued the Check Drive command to update the DCT. A

compare of the current cylinder and sector will be made with the new drive to ensure compatibility. If they are not compatible, an error message will be displayed and the program will return to Display mode displaying the previous drive, cylinder, and sector. After completion of the read, the program will return to Display mode, displaying the new drive's sector.

17. <ARROWS> Cursor Controls

The arrow keys will move the cursor in the indicated direction over the Hex and ASCII display fields. The left and right arrows will wrap around, treating the display fields and data buffer as a contiguous string of 256 bytes. Pressing <SHIFT> Left Arrow will home the cursor.

18. <;> (or +) and <-> (or =) Page Sector

Pressing ; (or +) will cause the sector to be incremented and display the next higher sector. If the display is the highest numbered sector the drive is capable of, the cylinder will be incremented and the Ø sector will display the next higher cylinder. If the current cylinder and sector being displayed is the highest the drive is capable of, the pressed key will be ignored.

Pressing - (or =) will cause the sector to be decremented and display the next lower sector. If the currently displayed sector is Ø, the cylinder will be decremented and the highest sector of the next lower cylinder will be displayed. If cylinder Ø and sector Ø are currently being displayed, the pressed key is ignored.

19. <@> Control key

This is the <CTRL> key. It is used as a super-shift in the ASCII and Hex modify modes for control functions. It is also used in conjunction with the D key to enter the extended debugging utility.

2Ø. <SPACE> Menu toggle

Pressing the space bar, while in display mode, will cause a summary of the available commands to be displayed. Press the space bar again to return to display mode.

Examples of use:

The primary purpose of this program is data recovery. Other uses will be found: such as removing passwords from protected files, direct patching of programs, as a tutoral to disk file organization, and any time it is desired to see and/or modify the actual data on the diskette. The following will give a few examples of how to use this program. The byte number references are those used in the "LDOS User's Manual". There is not room in this document for a complete description of the directory and file organization. Two excellent

publications exist and are highly recommended:

1. "LDOS User's Manual", Tech Info Section; contains a excellent, concise description of the directory and file organization

2. "TRS-80 Disk & Other Mysteries" by H.C. Pennington; this is the "Bible" of disk data recovery.

Note: Before proceeding with data recovery, it is good practice to write-protect the target diskette and make a backup copy for use in the data recovery operation. The backup facility of this program is provided for that purpose.

Example #1 Recovering a killed file

This method will only work if the space occupied by the file has not been reassigned to another file. If it has been reassigned, the file is irrevocably lost.

Locate the directory cylinder by pressing G. Page through the directory, using the ; and - keys, until you find the filespec of the killed file. The filespec will be located at relative bytes DIR+5 to DIR+15 in the directory entry. If DIR+0 bit 4 is 0, it indicates the file has been killed and is no longer active.

Using the arrow keys, position the cursor anywhere in the directory entry and press R. This will make the file active, visible, and unprotected. Press W to enter Write Mode. Answer Y to the question "Write to this location ?". The attribute byte and passwords of the file have now been changed on the diskette.

Position the cursor anywhere in the directory entry and press H. The hash code and DEC (Directory Entry Code) for the file will be calculated and the new HIT sector will be displayed. Press W to enter Write Mode. Answer Y to the write to this destination prompt and the new HIT will be written to diskette.

Press E to exit LZAP/CMD and return to LDOS Ready. A DIR command should now show the file active. Immediately COPY or XFER the file to another diskette. The file should once again be KILLed on the target diskette because the GAT (Granule Allocation Table) was not corrected to show the space occupied by the file. It is far easier to copy the file to another diskette and then reKILL it and let the system take care of the GAT.

Example #2 - The clobbered BOOT/SYS or DIR/SYS

This is one case where it is necessary to turn off the Error Dictionary. The Check Drive command that is issued when the drive number is entered will fail and indicate that the drive is not ready or the program may even hang with the drive trying to locate the DIR/SYS on a nonexistant cylinder.

With the Error Dictionary turned off, look at the BOOT/SYS on cylinder Ø sector Ø. If it has been clobbered, relative byte 'Ø2' may not indicate the correct cylinder for the DIR/SYS. Page through the diskette to locate the directory cylinder. This is the cylinder that is formatted with 'ØØ' pattern instead of the normal 'E5' or '6BD6' in double density.

Sector Ø of the DIR/SYS is necessary for the diskette to initialize correctly. If it is clobbered, read in sector Ø of DIR/SYS of a good diskette, press W to enter Write Mode and at the prompt "Write to this location ?" answer N. Change the destination to the target diskette, the cylinder for the directory just found, sector Ø. Now return to the BOOT/SYS and change byte 'Ø2' to point to the cylinder of the directory. If byte 'Ø2' of BOOT/SYS points to the directory cylinder and the GAT of DIR/SYS is readable the diskette should initialize correctly.

If the HIT is damaged, it will be necessary to calculate the hash code for each file to be recovered and to write each newly calculated HIT sector to the diskette.

Using this procedure, you may be able to recover some or all of the data on the clobbered diskette, but if the directory entry for the file has been damaged it will require a complete rebuilding of the directory entry. See one the references mentioned above and good luck.

As in the above example no attempt has been made to correct the GAT. The recovered files must be copied to another diskette before they can be used.

Example #3 Changing passwords

Position the cursor anywhere in the target file directory entry and press R. Press W to write the directory sector to the diskette.

The file is now visible, unprotected, and the password is all blanks.

-Frank

PATCHES TO LDOS 5.0.2

NOTICE:

These FIXES are not really "fixes" to LDOS. LDOS is an outstanding operating system, and we are in NO way implying that there are "bugs" that we are "fixing". These are ENHANCEMENTS to the system. Do not (repeat... DO NOT) apply these to your LDOS 5.0.2 MASTER DISK. You will void your LDOS warranty. Apply these ONLY to a BACKUP of 5.0.2. They have been fully checked out and should present NO user problems... only enhancements to the fine system you already own.

When applying these "fixes" make sure that 5.0.2 is in Drive 0, and that PATCH/CMD and the FIX files are on line also. If you get a FILE NOT FOUND error, then chances are something you need is NOT online. (It could be PATCH, the /FIX file, or the object file you are trying to patch.)

The /FIX files are included on your disk ready to patch. We have listed them at the end of each description also, for your viewing pleasure.

STATUS/FIX by Earle Robinson

After application of STATUS/FIX and SY6RESET/FIX, the user will have the status of the system displayed at the bottom of the screen after calling DEVICE. Included are the presence of JKL, TYPE-AHEAD, VERIFY, the system files RESIDEnt, DEBUG, the extended debugger (DEBUG (EXT)), PR/FLT, and SPOOLER. Also, if the BREAK key is disabled or the FAST clock (if installed) is set, these facts are displayed. If you are running LDOS 5.0.2 with the double density driver installed, this information is also present. Once the KSMSTAT/FIX is applied, STATUS will also show if KSM is part of the system. Once the patches are installed, each time the system is initialized, STATUS will show what is present at the moment it is called.

Apply as follows:

```
PATCH SYS6/SYS.WOLVES STATUS <enter>
```

next apply:

```
PATCH SYS6/SYS.WOLVES SY6RESET/FIX <enter>
```

Once you have applied these two patches a DEVICE command will result in this appended display:

```
LDOS Ready
DEVICE
:Ø  5" Floppy #1, Cyls= 41, Sden, Sides=1, Step= 6 ms, Dly= 1 s
:1  5" Floppy #2, Cyls= 41, Sden, Sides=1, Step= 6 ms, Dly= 1 s
:2  5" Floppy #4, Cyls= 41, Sden, Sides=1, Step=12 ms, Dly= 1 s
:3  5" Floppy #8, Cyls= 4Ø, Sden, Sides=1, Step=12 ms, Dly= 1 s
*KI <=  X'FDD3'
*DO <=> X'FDB5'
*PR <=> STATUS/PCL:Ø
*JL  =  Nil
*SI <=  *KI
*LO <=> *DO
Status: <Type-ahead><JKL>
<Resident overlays 5,1Ø,3,2,1>
LDOS Ready
```

```
. . . . . .
.Status/fix - Ø7/19/81  version 2
.Upgraded to tell if the double density driver is installed.
.apply to SYS6/SYS
.
L61
X'53BE'=ØØ 55
X'55ØØ'=21 DB 55 CD 67 44 3A 1F 44 CB 4F 21 E4 55 C4 BD 55 21 F1 55 CB
5F C4 BD 55 21 ØF 56 CB 57 C4 BD
X'552Ø'=55 21 15 56 CB 47 C4 BD 55 3A 3A 44 FE E1 21 3A 56 CC BD 55 3A
Ø2 47 FE 46 21 1F 56 C4 BD 55 3A
X'554Ø'=CB  43 CB 7F 21 68 56 C4 67 44 3A ØE 4Ø FE 4C28 3C Ø8 21 54 56
CD C7 55 DD 2A ØD 4Ø DD 7E Ø4 E6
X'556Ø'=ØF C6 2E FE 3A 38 Ø9 Ø8 3E 31 CD 33 ØØ Ø8 D6 ØA CD 33 ØØ DD 6E
2F DD 66 3Ø 7C FE 4C 28 ØA E5 DD
X'558Ø'=E1 3E 2C CD 33 ØØ 18 D4 3E 3E CD 33 ØØ 3A ØF 43 CB 5F 21 2D 56
C4 BD 55 21 FA 55 CB 67 C4 BD 55
X'55AØ'=21 46 56 CB 7F 28 13 CD BD 55 3A 11 43 FE 4C 21 4D 56 C4 67 44
3E 3E CD 33 ØØ C3 2D 4Ø Ø8 3A 2Ø
X'55CØ'=4Ø E6 3F FE 3Ø 38 ØF E5 3E ØD CD 33 ØØ 3A 2Ø 4Ø C6 Ø8 32 2Ø 4Ø
E1 CD 6744 Ø8 C9 53 74 61 74 75
X'55EØ'=73 3A 2Ø Ø3 3C 54 79 7Ø 65 2D 61 68 65 61 64 3E Ø3 3C 5Ø 52 2F
46 4C 54 3E Ø3 3C 42 72 65 61 6B
X'56ØØ'=2Ø 6B 65 79 2Ø 64 69 73 61 62 6C 65 64 3E Ø3 3C 4A 4B 4C 3E Ø3
3C 53 7Ø 6F 6F 6C 65 72 3E Ø3 3C
X'562Ø'=44 64 65 6E 2Ø 64 72 69 76 65 72 3E Ø3 3C 46 61 73 74 2Ø 63 6C
6F 63 6B 3E Ø3 3C 56 65 72 69 66
X'564Ø'=79 2Ø 6F 6E 3E Ø3 3C 44 65 62 75 67 Ø3 2Ø 28 65 78 74 29 Ø3 3C
52 65 73 69 64 65 6E 74 2Ø 6F 76
X'566Ø'=65 72 6C 61 79 73 2Ø Ø3 3C 4B 53 4D 3E Ø3
....End of patch
```

```
. Now apply this patch to SYS6/SYS
. SY6RESET/FIX  - 5.25.81 -
. This patch will permit RESET to also
. eliminate traces of RESIDELD.
.
L63
```

```
X'52D6'=CD 6Ø 54
X'546Ø'=21 31 4C 22 ØD 4Ø 21 4A 53 C9
...End of patch
```

KSMSTAT/FIX by Earle Robinson

Allows the status of the KSM driver to be shown in the device listing. Requires that STATUS/FIX and SY6RESET/FIX be applied also. If you use KSM frequently, then you will want to apply this patch.

Apply as follows:

```
PATCH KSM/FLT.RRW3 KSMSTAT <enter>
```

```
.............................................................
. KSMSTAT/FIX    - 5.25.81 -
. This patch is to permit flagging the
. the presence of a KSM driver.
. PATCH KSM/FLT.RRW3 KSMSTAT <enter>
.
X'5296'=C3 BØ 54
X'54BØ'=21 CB 43 CB FE C3 2D 4Ø
...End of patch
```

DIRLOWER/FIX by Earle Robinson

This patch is convert the directory entries to upper/lower case. It is totally optional, but to some, enhances the display of the DIRectory. For example, your utility disk directory would appear on your screen like this:

```
LDOS Ready
DIR :3
Free space=   50.0 K  Drive 3  QSDUTL#1 -- 07/28/81

Dcal/cmd +          Clone/cmd           Dvorak/cmd
Vdisk/cmd +         Change/cmd          Lzap/cmd +
Lcopy/cmd           Upcase/flt          Status/fix +
Binhex/cmd          Sys7resi/fix        Dirlower/fix +
Sy6reset/fix        Resideld/cmd        Addlf/flt
Ksmstat/fix         Stat/cmd +          Lpurge/cmd +
Recover/cmd +
```

Apply as follows:

```
PATCH SYS6/SYS.WOLVES DIRLOWER <enter>
```

Remember that this to be patched to SYS6/SYS, using the normal protocol. Do NOT patch it to your Master disk; it will void your warranty!

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
.DIRLOWER/FIX   -5/29/81-
. This patch is convert the directory
. entries to upper/lower case.
. PATCH SYS6/SYS.WOLVES DIRLOWER <enter>
.
L21
X'536E'=D0 57
X'538D'=D7 57
X'57D0'=F5 78 FE 08 28 06 F1 F6 20 C3 6F 55 F1 C3 6F 55
...End of patch.
```

MEM/FIX by Frank Luff

```
. Provides a C parameter to MEMORY library command
. to CLEAR memory from X'5400' to HIGH$.
. Deletes the A abbreviation for ADD.  Apply as follows:
.
. PATCH SYS6/SYS.WOLVES MEM <enter>
.
L1E
X'52AC'=C5 53
X'535E'=43 20 20 20 20 20 C6 53
X'53C5'=21 00 00 7C FE FF 28 0B 21 00 00 7C FE FF 28 19
X'53D5'=C3 2D 40 2A 49 40 11 00 54 37 ED 52 E5 C1 EB 11
X'53E5'=01 54 AF 77 ED B0 C3 2D 40
...end of patch
```

RESCUE/CMD by Kim Watt (c)(p)1981 - Breeze Computing

The purpose of this program is to restore "killed" files. RESCUE is not an ordinary plain-jane recover utility. It is "intelligent" enough to know whether or not the file is 100% recoverable or not. If it the only copy of an important data file that you have, and you accidently killed it, then by all means recover what you can. Usually if you haven't written anything to the disk since killing a file, it will be 100% recoverable.

To run the program, type:

RESCUE <enter>

You will be prompted for a drive. There is no need to hit <enter>.

The object directory will be read, as well as the disk, and the "killed" files will be displayed one by one.

If the drivespec is preceded by an OK, then all the data is still there, and you can recover it, no sweat.

If the filspec is preceded by NG, (no good), then that means at least SOME of the records have been over-written. You can go ahead and recover it and take your chances on how much is still there.

RESIDELD/CMD by Earle Robinson (c)1981

Purpose of program:

This utility is used to place frequently used system files into high memory, reducing wear and tear on disk drives, and speeding input to and output from disk files, or to hasten execution of JCL files.

This utility will also permit the user to place a DATA diskette in his drive #Ø, or if there is only one drive, in that drive. After booting up, the system disk may be replaced by any data diskette, if the requisite system files are RESIDEnt in high memory.

How to use RESIDELD:

Type from LDOS READY:

RESIDELD 1 <enter> (to load SYS1/SYS for example)

What SYS files to load:

SYS1, SYS2, SYS3, and SYS8 are ABSOLUTELY necessary to effect reading and writing to disk. If the user makes use of the DO command frequently, putting SYS11 in high memory will make execution MUCH FASTER. If you make many errors, then SYS4 might well be a candidate for inclusion, also. Bear in mind that there will be less RAM (random access memory), and though it may be tempting to put all system files into high memory, such a practice may cause problems when running a lengthy program. Since SYS6 and SYS7 are accessed somewhat differently than the other system files, they CANNOT be loaded into memory.

If you want to be able to "SYSGEN" RESDIELD, apply SYS7RESI/FIX to SYS7. It is included on your disk ready to patch in.

Apply by typing: PATCH SYS7/SYS.WOLVES SYS7RESI <enter>

```
.SYS7RESI/FIX - 5/21/81-
.This fix is to permit sysgening Reside under Ldos.
.
DØ8,8B=Ø2 ØØ ØD 4Ø 48 ØØ B8
...End of patch
```

Some notes on the LIBRARY and how to create /CMD files from LIBRARY commands to save disk space - by Earle Robinson

---

These remarks are an attempt to explain how you may create CMD files, the only purpose for which is to have more disk space available for data files or whatever. Using CMDFILE you merely call for SYS6 or SYS7, depending where the routine you wish is located, and then give the overlay number when requested. These numbers may be found by reading a download of SYS1 with the extended debugger. To make it easier I am listing them below:

Append	31	Attrib	51
Auto	11		
Build	33	Clock	17
Copy	32	Create	13
Date	15	Debug	14
Device	61	Dir	21
Do	91	Dump	71
Filter	66	Free	22
Kill	18	Lib	19
Link	62	List	41
Load	81	Memory	1E
Prot	52	Purge	72
Rename	53	Reset	63
Route	64	Run	82
Set	65	Spool	A2
System	A1	Time	16
Trace	1A	Verify	1B
Xfer	73		

I'll leave it to you to figure out to which SYS file each of these belongs, reminding you that it is obligatorily SYS6 or SYS7. Let's assume you wish to create a CMD file of the COPY command. Merely enter the number 32 when you are requested. It is not necessary to change addresses or transfer addresses, since you will be using the command file exactly as you would have used the library routine. Save the file with a different name since if you call "Copy" the parsing routine will load the routine from SYS6 and not your COPY/CMD file. In any case, you would only wish to create these CMD files to replace SYS6 and SYS7, reducing the overhead on your disk. This is especially useful for users with only one disk drive. Since these routines, when changed into CMD files, will occupy a full gran of 5 sectors (6 on double density), it is clear that if you create too many such files, you will have defeated the purpose of eliminating the SYS6 and SYS7 files! As for names of the corresponding files, I should opt for names similar enough to the library commands to avoid confusion with yourself, CCOPY for COPY, DDIR for DIR, XXFER for XFER, etc.

Finally, for those who wish to better understand the principle employed with the library, I will explain how a routine is parsed and then loaded and run from its SYS file. When you enter a command, it is parsed by SYS1, and a table with the names of all the LIB routines, each followed by the index number shown above. Following that number is an 8Ø or a CØ, which tells the system what to look for in SYS6 or

SYS7. Within the SYS6 or SYS7 file a search is made for the index number. This number is followed by the starting address of the routine once loaded and the relative sector number and relative byte where the routine is located within the SYS file. The routine is loaded and executed.

I hope these remarks will assist any interested users in better understanding both my idea of creating command files to save space, and of the library routines themselves.
Earle-

STAT/CMD 1.3 by Skip Taylor (c)1981

Purpose of program: To report the current status of the system options that are active, and to give you a readiness report of all defined disk drives. Gives similar results as STATUS/FIX, but does not require any patching. This is a stand alone utility.

```
-------------------------------------------------------------
! Syntax: STAT  <enter>                                      !
! Abbreviations: NONE                                        !
! Parameters required: NONE USED OR ACCEPTED                 !
! Options checked:                                           !
!      Fast Clock          Break          Spool              !
!      Type ahead          JKL            PR/FLT             !
!      KSM (Earle Robinson's version)  Verify                !
-------------------------------------------------------------
```

The options mentioned above will only be shown on the display IF they are activated.

The number of disk drives that will be included in the display will vary, depending on what configuration you have currently defined. There are three messages that could be displayed for the disk drives. They are: NOT Ready, Ready, and Ready/WP.

NOT Ready indicates that the drive does not contain a readable diskette, or that there is no diskette in the drive, or that the door is not closed. Or that the drive is NOT powered.

Ready indicates the the drive is functional and ready to be used, and contains a formatted diskette.

Ready/WP indicates that the drive is functional and ready to be used, and that the diskette in that drive has been Write Protected.

VDISK/CMD by Skip Taylor (c)1981

Purpose of program:

To verify the readability of all sectors on a specified drive. It will NOT verify the accuracy of the data, only that the sector is readable by the system.

Example:
You keep getting a PARITY READ ERROR. Is your disk REALLY messed up, or is the hardware a little flakey, maybe at the outer tracks? VDISK will at least tell you if the disk is okay and readable, perhaps on another system.

```
.............................................................
: Syntax: VDISK  or  VDISK :d  (d = desired drivespec)      :
: Abbreviations: NONE                                       :
: Parameters required: Mandatory space after the filespec   :
:                      and before the drive number. The     :
:                      colon is optional. If drivespec is   :
:                      not included or not in system, it    :
:                      will request a drivespec from you.   :
.............................................................
```

The program will obtain and display the number of cylinders and the number of sectors per cylinder from the Drive Code Table (DCT), and will use these as the parameters for the verify operation.

When the program encounters an error, it will tell you the Cylinder number, Sector number = Extended error message. This information can be used with LZAP/CMD to repair the diskette. LZAP is included on this diskette, and for more information on this, check the documentation. VDISK will identify the presence of a READ PROTECTED sector. The message will include the Cylinder number, Sector number, and tell you that it is READ PROTECTED. This is usually the directory or a system information sector.

Upon program termination, you will be notified of the total number of errors encountered, if any.

Published by
Quality Software Distributors
11500 Stemmons Exp., Suite 104
Dallas, Texas 75229