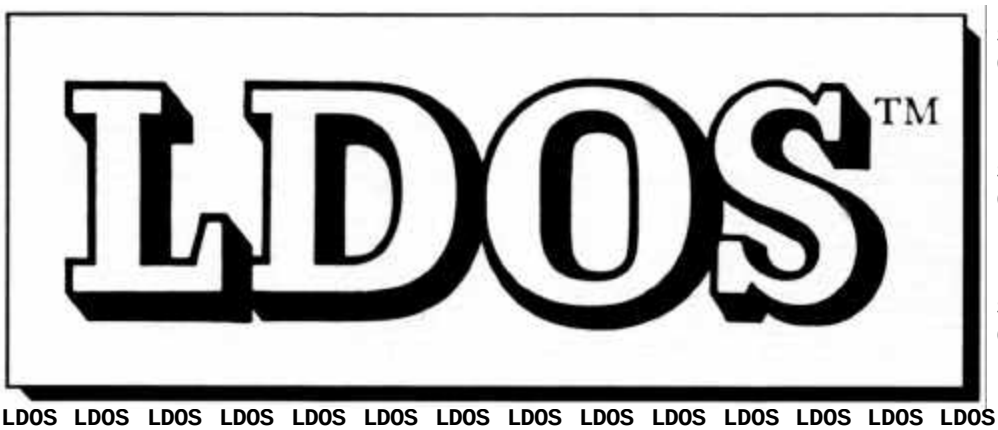
LDOS™

THE LDOS QUARTERLY July 1, 1981 Volume 1, Number 1

## VIEW FROM THE BOTTOM FLOOR

### ANNOUNCING PERCOM DOUBLER SUPPORT - SEE THE UPDATE NEWS FOR DETAILS!!

The "Percom Doubler II" seems to be working well and will be supported by LDOS effective June 15, 1981. This does not mean we are supporting the Doubler I, the LNW Doubler, or any other doubler that claims to be software compatible. Although our software may run these other doublers, it was designed and tested only on the Percom Doubler II. If you use another doubler and experience problems, don't call us - call the manufacturer of your doubler!

Along with the Doubler II support will be double sided support on the RS interface. Only three drives may be used with this double sided support because the Expansion Interface uses the "side select" line for "drive select 4". Also, a cable without pins missing is required and all drives to be used must have the ability to be set internally to a specific physical drive number.

If the above two new features are not enough, we have made other major modifications to the system. The RS-232 driver and other drivers that need to be "initialized" may now be SYSGENed, and will reinitialize themselves on each boot-up. Revised RS-232 drivers are included in this release. File space allocation is now more random, and several small bugs have been cured.

This new version, of LDOS is designated 5.0.2 and will be ready for shipment around June 15th. It will come complete with explanatory documentation and manual inserts. Note: All updates will be returned in a disk mailer! If you send us a library case, a shoe box or a refrigerator box, you will still get only a disk mailer back.

The ownership of the rights to LDOS has changed hands. LDOS is now owned by:

Logical Systems Inc. (LSI)
11520 N. Port Washington Rd.
Mequon, Wisconsin 53092.

LSI is a new company formed to develop, manage, market and support the LDOS product line. LSI is a Wisconsin Corporation, the principals of which are those originally involved with the creation of LDOS.

Some changes were needed in company policy because of this new arrangement. The most unpleasant is that updates to the LDOS product (5.x.x) will now be $5.00, and there will be no exceptions. This is required due to the costs of clerical work, repackaging,, mailing and the update itself. This new update fee will just cover the actual costs of creating the update, notifying owners, documenting changes, performing duplication and returning your master disk - we will not be making a profit on it!! The original fee or $1.00 was set up because a hardware company, LOBO Drives, was subsidizing the costs. This is no longer the case and the charge had to be raised to $5.00.

**You may disagree with this new fee, but we can not perform services at a loss. Tandy, Apparat, DOS+, Ultrados and all other vendors known to us do not correct problems or release enhanced versions as fast or as often as we do. We consider this to be one or our most important selling points. But we can't continue doing so if we lose money every time we try to give you a better software package (we'd go out or business).**

**You may think that we should have made plenty of money when we sold LDOS to you at $149.00, and that would be correct! But in most cases we did not receive anywhere near the retail price from you. The vast majority of LDOS's are sold through dealers and are substantially discounted to them. Many hundreds of you paid only $40.00 to receive an LDOS as a VTOS 4.0 replacement. LDOS costs about $30.00 per package to produce with royalties, printing, binders, disk, duplicating, mailer, packaging and handling. To amortize the actual cost of development and enhancement we have to sell over 5000 LDOS's per year just to break even. Authoring the LDOS manual alone cost over $25,000.**

**LDOS owners should realize that this is not a HOBBY or PART-TIME product for us. There is a full time staff of 5 programmers and 2 support persons to provide and maintain this product. This is in contrast to other companies providing operating systems for the TRS-80 MOO I & III (even Tandy doesn't have as many system programmers working for you).**

**As charges and services change from time to time, please bear with us as we will always try to give you the best possible product or service at a fair (not losing) price.**

**The 800 line service will remain but the hours of service will be modified temporarily. The new hours will be 10-12am and 4-6pm Central time, Monday through Friday. This is to minimize the interruptions in our programming department. Should these time slots become inadequate, they will be expanded as needed. You will be notified in this newsletter and on the MicroNET LDOS board of any changes in these hours.**

**If the label on your MASTER disk does not show ".2" in the Update row, send in your MASTER disk along with $5.00 as soon as possible for your 5.0.2 update. Even if you don't need or want the new features, there have been several small bugs repaired in this version which may keep you from having a problem. As usual, send your disk packaged carefully, as the replacement of a damaged master disk will cost you an additional $5.00. Address your disk to LDOS Support at the same address given above for LSI. Make checks payable to Logical Systems, Inc.**

**LDOS owners in England or the Common Market Countries should send their master disk with $10.00 to MOLIMERX at:**

**LDOS Support**
**c/o MOLIMERX Ltd.**
**1 Buckhurst Rd.**
**Bexhill, Sussex, England**

**Phone: (0424) 223636**

**Note: If you are sending to Molimerx for an update, make your check payable to Molimerx Ltd.**

**Molimerx is our distributor in England and Western Europe and will be providing update services and customer support. They have our full support and we maintain very close communications with them.**

**For any of you that may be able to help, we are actively seeking a distributor to represent us in Japan. If you can recommend a firm, please let us know.**

**==========**

## UPDATE NEWS - LDOS 5.0.2

**LSI is happy to announce the release of LDOS 5.0.2. This update version will be available June 15th. Among the enhancements is double density support for the Percom Doubler II. As with the original maintenance release, changes made to the operating system will be detailed in the sheets that are returned with your updated Master disk. To get the update, send in your disk with the $5.00 fee. The address is in the customer service section of the manual.**

**If you don't want to wait for your 5.0.2 disk to arrive, you can find the object code for the Doubler driver in our MicroNET database and in the program section of this newsletter. You can download it and convert it with BINHEX/BAS, found in the database and also reprinted in this newsletter.**

**Note: Checks or money orders for all updates should be made out to "Logical Systems, Inc.". Overseas updates must be accompanied by an international money order for $10.00**

**==========**

## Items Of General Interest

**Model III LDOS should be ready for release at the end of June. The price will be $149.00 and will include the Master disk and a full manual containing operating instructions and technical information.**

**The passwords for LDOS files were left out of the manual. Refer to this list:**

| | | | |
|---|---|---|---|
| System files (/SYS) | WOLVES | Filter files (/FLT) | GSLTD |
| Driver files (/DVR) | GSLTD | Utility files (/CMD) | RRW3 |
| Basic overlay (/OV$) | BASIC | LOBO/DCT | RRW3 |
| CONFIG/SYS | CCC | | |

**User contributions to this newsletter have been almost non-existent. We receive many suggestions about new features that many of you would like to see in future releases of LDOS. Many of you have written routines that add these features as separate programs or high memory drivers. Why not contribute them to your LDOS Newsletter so that all LDOS owners can use them? To those of you that have already contributed to the newsletter or the MicroNET bulletin board database - many thanks from all who have made use of your programs.**

**For those of you who are legal owners of more than one LDOS, you can refer to these as "your LDii" not "LDOSes".**

**Want to clear the screen inside of a JCL? Try the command MEMORY (GO=X'1C9').**

**This short program will cause a Top of Form to be sent to the lineprinter. Use the BUILD command with the Hex parameter to build the program "T/CMD" as follows: BUILD T/CMD (HEX). Then enter in the following string 010700523E0CC33B0002020052. The byte 0C is the actual character sent and could be changed to some other control character to perform other printer functions.**

**An interesting feature not documented in the manual concerns the system's type ahead feature and the SYSTEM (SYSGEN) command. When doing a SYSGEN with type ahead on, anything you type on the keyboard up to the time the "User configuration built" message appears on the screen will be saved in the configuration file and executed each time the system is powered up or booted. If you don't want this to happen, don't type ahead when the SYSGEN is occurring.**

**Some of you have indicated that you dislike the use of the term CYLINDER rather than TRACK. Let's clear the air. A cylinder is all tracks of the same number, on all surfaces of a drive. If the drive is a single sided floppy, there is ONE track per cylinder and the terms are interchangeable. But LDOS supports much more then single sided floppies. When we say "cylinder" on a SA-506 hard drive we are referring to the imaginary cylinder that is created by the same track on each of that drive's FOUR surfaces. So this drive has four track number X's, but only one cylinder X.**

**It's true - Tim Mann did get his hair cut. Dick Konop also shaved off his beard. Guess we are all joining the Establishment.**

**==========**

## ON THE TRAIL OF THE ELUSIVE PARITY ERROR

**This section of the newsletter stems from users asking LDOS Support "Why does LDOS create a parity error?". Well, LDOS does NOT create parity errors – it only reports them. A parity error is the sole creation of the Floppy Disk Controller (FOC) chip in whichever expansion interface you are using. When the FDC formats a disk, it writes CRC bytes to use as a checksum for the sector ID's and the data written to the sectors. When the FDC reads data from a disk sector, it computes a checksum for that data and compares it to the CRC that was created when the data was written to that sector. If there is a difference, the FDC returns a CRC error to the operating system, and "Parity error during read" appears on the screen.**

**When attempting a write, if the FDC detects a difference in the Sector ID checksum and its CRC, a "Parity error during write" message will appear.**

**Now for the next obvious question - "Why does my system get parity errors ?". Well . . . . The only reason for the parity error message is a CRC error from the FDC. If a poor data separator circuit is causing incorrect data to be read from the disk, an error occurs. If the disk heads are dirty, the data written to the disk may not have sufficient amplitude when read back, and an error occurs. Some drives have internal adjustments that control data pulse timing. If they are out or tolerance, an error occurs. All disk media is not created equal - using marginal media produces marginal results, especially with double density! What it all boils down to is that a parity error means something is not quite right with the hardware or media.**

**The data sheets on the Western Digital 1771 and 179x series FDC chips contain more information on the use of the CRC bytes and what sequence of events will cause a CRC error.**

==========

## DATA ADDRESS MARKS

---

**Everything you always wanted to know about Data Address Marks (those damn DAMs), and other assorted trivia. by Roy Soltoff**

**This issue's technical topic concerns itself with a subject destined to become one of the hottest tidbits of thingamajigs, scroillions, and zwortjhinks of recent fame, beat out only by the delectable doubler debacle. This DAM topic covers the least understood principle of operation in the TRS-80 operating systems and is associated with the ubiquitous Floppy Disk Controller. Since you are an avid reader of this newsletter, you must be using LDOS (unless you snuck a copy of the NL and now wish you were using LDOS). One thing required of all disk operating system users is that they have an expansion interface - which IS used for a little more than a means of interconnecting five cables in a junction box!**

**The E/I, as it is commonly known, contains the FDC, which is used to interface the operating system with the disk drives. The FDC used in the TRS-8O is manufactured by Western Digital. This Large Scale Integrated (LSI) chip (a 1771) handles all serial data transfer to and from the diskette, converting parallel data between it and the CPU. The CPU controls and handshakes with the FDC via commands and data values passed to the FDC, and status information returned to the CPU from the FDC.**

**The 1771 is compatible only with soft-sector formatted media. In this type of diskette operation, the location of the space occupied by a sector is denoted by special data patterns and identification fields written on the diskette during a formatting operation. This is in contrast to hard-sectored media, which has a physical sense hole on the outer or inner rim of the diskette to mark the physical beginning of each sector. The soft-sectored format was defined by IBM with the introduction of eight inch disk drives. A variation of this format is used in the TRS-80. Without going into too much detail on the exact format, suffice to say that it consists essentially of an index mark, track and sector identification, a data address mark, a sector of data, and checksum fields (actually called cyclical redundancy checks - CRC's).**

**This information exists for every sector on the diskette. There is also dead space, called gaps, between the sectors. The dead space is used by the FDC as a buffer zone to aid in separating the end of one sector from the identification field of the next.**

**The data address mark was defined by IBM to be one of two values: an X'FB', used to indicate that the respective sector contains data, or an X'F8', used to indicate that the data in the sector was deleted. The value of the DAM is controlled on writing, not as a discrete byte of information passed to the FDC, but rather by means of a bit value in the write sector FDC command. It is detected by a specific pattern in the status register of the FDC which is read by the CPU. When Western Digital was designing the 1771 chip, they had an unused bit position in the write sector command byte. Now if two DAMs are good, then four DAMs are better - or so they thought. WD defined X'F9' and X'FA', generating the four DAMs via a two-bit pattern:**

**00 => X'FB'**
**01 => X'FA'**
**10 => X'F9'**
**11 => X'F8'**

**So far, so good. The original design of TRSDOS made use of two of the above DAMs. In order to be able to differentiate a system sector (i.e. any one of the sectors constituting the directory) from an "ordinary" sector, system sectors are written with an X'FA' DAM while all other sectors are written with an X'FB' DAM. In reading any sector, by checking the status register of the FDC, the operating system can determine if the sector just read was in the directory or not. Before we get far down the road, it is best to clarify a printing error in the WD data sheet for the 1771. The DAM status bit configuration is as follows and NOT what appears on the WD data sheet. Pay close attention to the revised column headings - that was the error:**

| Status bit 6 | Status bit 5 | Data AM |
|---|---|---|
| 0 | 0 | FB |
| 0 | 1 | FA |
| 1 | 0 | F9 |
| 1 | 1 | F8 |

**The plot thickens when double density enters the scene. The 179X series of FDC chips support the transfer of double density data, which certainly helps to increase the storage capacity of disk drives. This was not done at the sacrifice of single density - WD used an external input to the 1791 chip to control the switching between single and double density. But at the same time, WD wanted to add other features, including the ability to check whether the correct head had been selected when reading from a two-sided disk drive. This required additional bits in the read and write sector commands. Where did these bits come from? One bit was taken from the two used to denote the DAM. This results in the selection of only two distinct DAMs. You guessed it. The 179X can generate only an X'F8' or an X'FB' when writing, and furthermore, it CANNOT differentiate between an X'FB' and X'FA' when reading, returning a "0" status for both, or between an X'F9' and X'F8' returning a "1" status for both in bit position 5 of the status byte. Essentially, if you examine the above table, the FDC 179X series returns in status bit 5 what was returned in status bit 6 of the 1771, while status bit 6 of the 179X is unused! Since the 1771 data sheet always had columns 5 and 6 reversed, this change appears more than coincidental, and I suspect that Western Digital engineers were the victims of an error in their own data sheet.**

**Consider a diskette made on a system with a 1771 (like the TRS-80). Put the diskette on a Model III which uses a 179X series FDC, and the DOS cannot "find" the directory because the FDC doesn't know that the X'FA' is any different than an X'FB'. It can't tell them apart. This is the main reason why the Percom Doubler for the Model I uses both a 1771 and a 1791 FDC chip - so that its single density operation would be compatible with other TRS-80s. Otherwise, your disk could not be used easily by other systems nor could you read diskettes made on other systems. What would be required would be to have a utility to regenerate the proper data address mark for your system on the diskette. This is quite easy to do but would be cumbersome. There is another solution.**

**Since the 1771 can both generate and detect an X'F8' DAM, LDOS for the Model I actually doesn't care whether the DAM used in the system sectors is an X'FA' or X'F8' - either is acceptable because LDOS ignores bit 6 of the status. Thus an X'F9' is equivalent to an X'FB' and an X'F8' is equivalent to an X'FA'.**

**TRSDOS, and possibly other DOSs, do not ignore bit 6 but would suspect that some unknown disk error occurred if it read a sector preceded by an X'F8' or X'F9'. For this reason, up to this time, LDOS has generated an X'FA' for the directory sectors.**

**Other systems, however, are already using 179X FDCs. The double density modifications for the Model I use 1791s. The LX-80 E/I uses a 1791. The Model III uses a 1793 (equivalent to the 1791). In order to be able to exchange media between these machine configurations and the stock Model I, it would make sense to have all machines accept each other's DAMs. Then diskettes would be directly usable on each machine. This would result in less confusion and frustration and improve tremendously the operating convenience of users dealing with multiple systems.**

**The bottom line is that effective with release 5.0.2, LDOS will generate only the X'FB' and X'F8' data address marks, although it will still be able to read disks written with X'FA' data address marks when the disk controller is a 1771. As time permits, we will try to provide patches to other operating systems so that they will also be able to use X'FB' and X'F8'. All that is required is to have the DOS mask and ignore status bit 6 on sector reads and write an X'F8' data address mark on directory sector writes. The modifications are simple to implement, and will result in a more powerful and convenient operation for all. We trust that our users will be understanding of this need and appreciative for our willingness to discuss this problem with you.**

==========

## THE 800 NUMBER - ATTENTION ALL USERS

**This is being written almost exactly 3 months after the first LDOS 5.0 was sent out into the world. Since that time, we have taken what seems like a year's worth of calls on our LDOS Support 800 lines. On a busy day, our customer service department will deal with over 100 people, via updates, registrations, and phone calls. Although there is a section in the manual that deals with when and how to use the 800 number, it seems that more will need to be said about it here.**

**As stated in the "VIEW FROM THE BOTTOM FLOOR" section, the hours on the 800 lines are being changed. The new hours will be from 10AM to 12 NOON, and from 4PM to 6PM, Central time.**

**We have been receiving many calls from people who have not read the manual, and call for answers to questions that are explained in detail in the manual. Please DO NOT call until you have looked in the manual for an answer to your question!**

**Customer Support needs to know what LDOS version and hardware you have if they are to help you. That means you MUST have your serial number in hand when making the call. The procedure that we use is:**

**1) Verify the name and serial number from our list of owners, and check the version number. We can immediately tell what version you have by the list. If we have a more recent update version, we can inform you of it.**

**2) Starting immediately, if you don't have your serial number when you call, you will be told to call back when you have it. We can't help you if we have incomplete information.**

**3) Only those owners whose registration cards are received will be allowed to update their disks, use the MicroNET bulletin board, and receive the newsletter. It is extremely difficult to mail update notices and newsletters to unknown owners.**

**We will try to answer any questions you may have about the operating system features and commands. However, certain types of questions are impossible for us to answer. PLEASE DO NOT CALL AND ASK IF A CERTAIN PROGRAM WILL RUN ON THE LDOS SYSTEM - we have not tested any programs other than Electric Pencil and Scripsit. The only answer we can give you is "try it and see". Your best bet would be to call the software vendor or the author.**

**Technical questions that cannot be answered using information in the user's manual probably cannot be answered directly on the 800 line. We will try to answer any questions immediately, but if finding the answer would entail searching the source code or talking with an LDOS systems analyst, the following rules will apply.**

**1) Detailed technical questions may be answered over the phone by our system's analyst if an appointment is made in advance. The fee for phone time is $50.00 per hour, 1 hour minimum, with the user initiating the call.**

**2) Technical questions submitted in writing will be answered free of charge as long as they do not entail large amounts of research or detailed explanation. However, any technical questions, no matter how complex, will be fully answered for a fee that will be dependent on the nature of the questions.**

**In other words, we will answer any questions about any part of the LDOS operating system, but like any other service, compensation will be required for questions outside of the normal user support.**

**Please do not call in with suggestions for new features or enhancements to LDOS. Leave us a note on the bulletin board or send us a letter. Making suggestions by phone ties up the line for those who have need of our services.**

**Please do not interpret this section and the shortening of 800 service hours to mean a decrease in the quality of the customer support. We will continue to cheerfully provide operating help and system explanations to all of our registered owners. What we hope to accomplish is an optimization of our 800 line service and new feature/product development time. Remember, the Support personnel are the same people who create, test, and document all of the enhancements to the LDOS family of products.**

**==========**

## NOTES ON LBASIC

In the first edition of the manual, there were several points that were left out, and several points which needed clarification concerning LBASIC and its features. In this section of the newsletter, we will try to clear up some of these areas.

1) Although it is not stated in the manual, the LDOS command -- LBASIC * -- does exist, and does work. The user is warned NOT to perform any commands that may affect HIGH$, or alter any memory above X'6000', if he wishes to use LBASIC * to reenter LBASIC. Doing so may cause the destruction of any LBASIC program that is in memory. It is much safer and wiser to perform any LDOS commands or functions from LBASIC using the CMD"command" format.

Also, at this time, LBASIC * will NOT work with the LX80 interface after the system has been rebooted. However, LBASIC * will work with the LX80 interface after a CMD"S" command has been issued.

2) The LBASIC function -- LOC -- will return the most recently accessed record of a random file which is currently open. The syntax for this function is:

LOC(n)

where n is the buffer number corresponding to the file. If the LOC function is used and the buffer number specified does not relate to an open file, the error message "BAD FILE NUMBER" will appear.

3) If the LBASIC command -- OPEN"EO" -- is given, but the file specified does not exist, a FILE NOT FOUND error message will be generated.

4) Any LBASIC program may be protected with an "Execute Only" password. This protection status is assigned using the ATTRIB Library command.

What this means is that the protected program may be RUN, but any attempt to LOAD, LIST, LLIST or examine the program will not be allowed. If an attempt is made to break the program during execution, the program will be erased from memory. Finally, the DEBUGger will be disabled during the execution of a protected program.

5) If you use the comma (EDIT .) or the period (LIST .) abbreviations, these characters must be the FIRST character entered in a line for the command to be issued. For example, suppose you finish entering a program line, type in a line number, and realize that the line you have just entered needs to be edited. If you backspace to erase the new line number and press <,>, you will NOT be placed in the Edit mode. (The comma was not the FIRST character to be entered in the line, as a line number and backspace characters were entered.) Using the above example, if <ENTER> is pressed after the backspacing has been performed and before the <,> command is given, you WILL be placed in the edit mode.

6) If you intend on using CMD"X" or CMD"N", the files BASIC/OVX and BASIC/OVN must be present on some disk in the system. (BASIC/OVX is the overlay which contains the cross reference utility; BASIC/OVN is the overlay which contains the renumber utility.)

**Also, if you wish to use either of these utilities, you must NOT have renamed LBASIC to BASIC. Doing so will cause the error "PROGRAM NOT FOUND" to appear after the operation has been completed. In addition, you will leave LBASIC, and control will be returned to the operating system.**

**7) The proper syntax for the CMD"X" command is as follows.**

**CMD"X devspec/filespec parameter,<title>"**

**Note that there is no comma required between the devspec/filespec and the parameter, as was shown in the first edition of the manual.**

**8) NEVER use CMD"X" or "N" if your program contains a line #0! Doing so will cause unpredictable results, and your program may be erased from memory!!!!**

**9) When using CMD"X", variables in a PRINT statement that are "packed together" may not appear in the variable listing. For example, suppose the following line appears in a program.**

**100 PRINT A$B$C$D$E$F$**

**Because the variables are packed together (i.e. not separated by semi-colons or spaces), some may not appear in the cross reference listing. To assure that all variables are recognized in a cross reference listing, proper punctuation must be used.**

**10) When using CMD"X" or CMD"N", the line number associated with the RUN command (i.e. RUN400) and the ERL command will be ignored. That is to say, if the CMD"X" command is given, line numbers used with the RUN and ERL commands will not be displayed in the cross reference listing. If the CMD"N" command is given, line numbers used with the RUN and ERL commands will not be updated, and thus may not be correct.**

**11) Default values for the parameters in the CMD"N" command are as follows.**

**aaaa=l bbbb=20 cccc=20 dddd=65529**

**12) There may have been some confusion as to what the last parameter in the CMD"N" command represents (i.e. "dddd"). The following two paragraphs will try to explain how this parameter is to be used.**

**Suppose you have a program whose beginning line number is 10, and whose ending line number is 2000. In addition, you have a block of code within your program that you wish to renumber, and the line numbers in this block range from 600 to 800. The next line number in your program following line number 800 is 1350. After renumbering takes place, line 800 will be reassigned the line number 1260.**

**A correct value for the last parameter would be a number which is between 1261 and 1349, NOT 800. That is to say, the last parameter must always be greater than the value of the last line number in the block to be renumbered (AFTER renumbering occurs), and it must be less than the first line number outside of the block to be renumbered. Using the above example, the easiest way to assign a value to the last parameter is to assign to it a number which is one less than the value of the first line number outside of the block to be renumbered (i.e. 1349).**

**13) When using the CMD"N" command, do NOT use the (!) parameter unless you are absolutely sure no errors exist. If you do not specify the (!) parameter, a full scan for errors will be done before renumbering starts. If errors do exist, no lines will be changed, and any errors may be corrected at this time.**

**If you do specify the (!) parameter, any error found will abort the renumbering. However, all internal line number references will have been changed up to the line number that caused the error.**

**14) For many Model I owners, the idea of "Blocked Files" may be new. The following examples will draw a comparison between the code needed to access conventional Model I random files, as opposed to the code needed to access LBASIC Blocked Files.**

**In all examples below, let us assume we are dealing with records whose Logical Record Length (LRL) is 25 bytes. Also, each logical record is divided into four fields. Field #1 is 5 bytes long, Field #2 is 4 bytes long, Field #3 is 8 bytes long, and Field #4 is 8 bytes long.**

**Let us assume that we wish to represent the variables in our field statement as an array. Example 1A will represent a conventional Model I field statement, while Example 1B will represent a field statement using LBASIC's Blocked Files. Note that for every 10 logical records, six bytes are wasted in conventional Model I Basic, as logical records cannot span sector boundaries.**

***** Example 1A *****

```
FOR L=1 TO 10:FIELD 1, ((L-1)*25) AS DM$, 5 AS AR$(1,L), 4 AS
AR$(2,L), 8 AS AR$(3,L), 8 AS AR$(4,L):NEXT L
```

***** Example 1B *****

```
FIELD 1, 5 AS AR$(1), 4 AS AR$(2), 8 AS AR$(3), 8 AS AR$(4)
```

**Let us now assume that we wish to perform a GET of a logical record, and utilize some field of that particular record. The variable LR contains the record number we wish to GET, and the variable NF contains the number of the field that we wish to access. The following examples will show how the logical record in question will be accessed. Example 2A demonstrates how access is performed using conventional Basic, while Example 2B demonstrates how access is performed using LBASIC Blocked Files.**

***** Example 2A *****

```
PR=INT((LR-1)/10)+1:SR=LR-((PR-1)*10):GET 1,PR:XX$=AR$(NF,SR)
```

***** Example 2B *****

```
GET 1,LR:XX$=AR$(NF)
```

Hopefully, the above paragraphs and examples will clear up any remaining questions that exist concerning LBASIC and its use.

==========

## LINKING TO LDOS IN ASSEMBLY

Linking to the LDOS Command Interpreter from a Machine Language Program - by Roy Soltoff

One of the more specialized operating system functions available to the assembly language programmer using LDOS is access to the command interpreter. This function is activated via the @CMNDI system call noted on page 1 of the "Tech Info - Entry Points - Control" part of the LDOS reference manual. The @CMNDI vector is similar to the @EXIT vector in that it returns to LDOS Ready. It is different in that it passes LDOS a command line to be interpreted and executed prior to the LDOS Ready prompt. In some cases, this can be quite useful. In other cases, it would be desirable to return to the running application after the command is executed rather than exiting to the LDOS Ready prompt. This can be accomplished by specific code in the assembler program, once certain limitations are understood.

First, the @CMNDI processor can execute ANY command that could be entered in response to LDOS Ready. That includes all LDOS library commands, all LDOS utilities, and all other user /CMD files. If you want the system to return to your program, you will have to ensure that the command being executed will not clobber your program. As a general rule of thumb, by restricting your application to a memory region above X'6000', all LDOS library commands can be accessed. You are on your own when it comes to /CMD files.

An experienced Assembler programmer should be able to make sense out of the LINKER routine shown below. This routine, if included in your program, will enable the program to access the @CMNDI interpreter while providing linkage code back to the program. If you do NOT understand the LINKER routine please do NOT call the LDOS Support Center for assistance as you will definitely have difficulty in other aspects of LINKER's use and we cannot operate as a training center. Good luck to the experimenters.

```
00100 ;LINKER/ASM - Roy Soltoff
00110 ;*=*=*
00120 ;       On entry to this routine, an LDOS command line
00130 ;       should be located in a buffer area. The address
00140 ;       of the buffer should be contained in a word
00150 ;       labeled CMDPTR. The routine should be entered
00160 ;       by a CALL LDOSCMD instruction. The following
00170 ;       code is assembled with your program.
00180 ;*=*=*
00190 LDOSCMD LD      HL,@EXIT        ;point to @EXIT & @AB0RT
00200         PUSH    HL              ;save pointer
00210         LD      DE,LDOSSV       ;point to our save area
00220         LD      BC,6            ;init to move 6 bytes
00230         PUSH    BC              ;save count
00240         LDIR                    ;save @EXIT & @ABORT
00250         LD      HL,NEWVECS      ;point to new vectors
00260         POP     BC              ;recover 6-count
```

```
00270          POP     DE              ;point DE to @EXIT
00280          LDIR                    ;move in new vectors
00290          LD      HL,(HIGH$)      ;p/u high memory ptr
00300          LD      (HIGHSV),HL     ;& save it
00310          LD      HL,NEWHIGH      ;this value would be
00320          LD      (HIGH$),HL      ;below our program
00330          LD      HL,(CMDPTR)     ;pt to command string
00340          LD      (SAVSP),SP      ;save our stack pointer
00350          JP      @CMNDI          ;let LDOS interpret it
00360 LDOSRET  LD      HL,LDOSSV       ;pt to saved vectors
00370          LD      DE,@EXIT
00380          LD      BC,6            ;move the saved vectors
00390          LDIR                    ;back into @EXIT & @AB0RT
00400          LD      HL,0            ;p/u saved HIGH$
00410 HIGHSV   EQU     $-2
00420          LD      (HIGH$),HL      ;& restore it
00430          LD      SP,0            ;restore stack pointer
00440 SAVSP    EQU     $-2
00450          RET                     ;reenter program
00460 ;*=*=*
00470 ;        Save area for @EXIT and @ABORT vectors
00480 ;        during command execution
00490 ;*=*=*
00500 LDOSSV   DS      6               ;vector save area
00510 ;*=*=*
00520 ;        The following instructions are the vectors back to
00530 ;        your program from @EXIT and @ABORT. You may wish to
00540 ;        treat @EXIT differently from @AB0RT.
00550 ;*=*=*
00560 NEWVECS  JP      LDOSRET         ;@EXIT vector
00570          JP      LDOSRET         ;@AB0RT vector
```

## USING CLOCK SPEEDUP KITS WITH LDOS

by Tim Mann

One of the lesser-known features of LDOS is its ability to work with TRS-80s that have been modified to speed up the CPU clock. Unlike most other operating systems, LDOS does not require any patches to work at a clock speed as high as 3.54 MHz, or possibly higher. It is also unnecessary to run a wire into your expansion interface to slow down the clock during disk I/O, as some speedup board manufacturers recommend. If you have a clock speedup in your TRS-80, or are thinking of putting one in, this article will help you understand this LDOS feature, and guide you around some of the possible pitfalls.

Why does a disk operating system need to do something special to work with a speeded-up clock? As you might have expected, there are certain time delay loops used in disk operations that will not function properly at increased CPU speed. There are three areas in which some operating systems use delay loops or "time wasting" instructions. First of all, when the motor on a 5" floppy disk drive is started, it is necessary to wait until it comes up to speed before you attempt any I/O. This waiting time is typically 1/2 second to 1 second, depending on the specifications set down by the drive manufacturer. Secondly, some operating systems use a delay loop to allow a certain amount of time for an index hole to go by before they decide that

**there is no diskette in a drive. But most importantly, after any command is passed to the TRS-80's disk controller chip (FDC), there is a certain settling time required before the status bits that are returned to the CPU are valid. This can be as long as 100ms, according to the Western Digital 1791 data sheet. The need to allow for this settling time accounts for the PUSH HL/POP HL or EX (SP),HL instructions found in several places in TRSDOS and other operating systems.**

**How does LDOS deal with the need to increase its timing loops when running with a fast clock mod? This is done by means of the SYSTEM (FAST) and SYSTEM (SLOW) commands. When you execute a SYSTEM (FAST), LDOS sets a system flag bit informing the disk drivers that the CPU clock has been speeded up. It also outputs a 1 to port X'FE', which seems to have become the standard method for switching clock speedup mods on. The speedup mod is left on until you execute a SYSTEM (SLOW), at which time LDOS resets the flag bit and outputs a 0 to port X'FE'. While the fast clock flag is on, LDOS doubles the length of the delay loop used when a disk drive motor is started. The other two problems mentioned above do not occur under LDOS at all, since it bases its index hole detection on the system's 25 ms heartbeat counter instead of a delay loop, and the delay loop used to allow for settling time is already long enough to work at the fast speed without change. When you do a SYSTEM (SYSGEN), the state of the fast clock bit is saved in your CONFIG file, and the proper value is output to port X'FE' after the CONFIG file has been loaded in.**

**Despite the careful provisions that have been made for handling clock speedups in LDOS, there are still some problems that can crop up. The most common complaint is that "SYSTEM (FAST) seems to cause a SYSTEM (CRASH)." If you have a Radio Shack expansion interface, the usual reason for this is flakey memory in the interface. The memory seems to run fine at 1.774 MHz, but you get random bit errors when you switch to the higher speed, which cause any device drivers that may be in high memory to crash, causing a reboot or system lockup. If this is your problem, you will probably notice that SYSTEM (FAST) works fine if you boot up without a CONFIG file (hold down the (CLEAR> key while booting). It is also possible that the Z-80 CPU in your keyboard will not function at the higher speed, as the Z-80's used by Radio Shack are only guaranteed to run at 2.5 MHz or below.**

**Again, if you have a Radio Shack interface, you may have difficulty in booting. The usual symptom is that the system boots fine on power-up, but using the Reset button or the BOOT command after the fast clock has been turned on causes a DISK ERROR or SYS ERROR message, or a crash during the bootstrap procedure. The reason for this appears to be that the system boot loader (BOOT/SYS) does not delay quite long enough after passing the disk controller a read command to allow for a clock speed of 3.54 MHz. If you are having this problem, there are two solutions open to you. You can modify your clock speedup board to reset itself to the slower speed whenever you boot up, allowing it to be switched back in by your CONFIG file, or you can patch BOOT/SYS to provide an increased time delay. On my own system, I found it more satisfactory to make the hardware modification, but I will explain both methods here and let you take your choice.**

**In my own system, I have the Archbold speedup board, which seems to be the most popular one. The version of the board I have is an older one which is no longer being made, but there were quite a few of this type sold. Mounted on the board are a resistor and capacitor that form a power-up reset circuit.**

**This automatically resets the board to the slow speed when you turn on the power to the keyboard. The modification I use is very simple. Remove the resistor and capacitor from the board by cutting their leads on the component side. Then run a wire from Z2 pin 6 on the Archbold board to Z37 pin 1 on the TRS-80 CPU board. This causes the board to be reset to slow whenever the SYSRES* line goes low, which occurs on powerup, when the Reset button is pressed, and when the BOOT library command is used (which causes the CPU to execute a HALT instruction).**

**The nice thing about this mod is that it enables you to boot on any disk, including a foreign operating system with no provisions for handling a fast clock.**

**I am told that if you have the new version of the Archbold board, which does not have a resistor or capacitor on the board, you can accomplish the same thing by removing the wire that connects to Z53 pin 12 on the CPU board and connecting it to Z37 pin 1 instead.**

**If you do not wish to make this change to your speedup board, the following patch may also eliminate your bootup problems:**

```
. FASTBOOT/FIX
. Apply with the command PATCH BOOT/SYS.WOLVES USING FASTBOOT
D00,BD=DF DF DF DF
D00,CB=DF DF
. End of patch
```

**Use the BUILD command to create a file containing the above patch, and then use the indicated command to apply the patch to each of your system disks (NOT to your master!!). The patch is not carried over when you do a backup, so you will have to reapply it to every system disk you create. This is strictly an optional patch, and it will not be made an official part of the LDOS system. It does appear to work quite well, nevertheless.**

**In the last few paragraphs, you probably noticed that I qualified most of my statements with the words "if you have a Radio Shack interface." Well, what if you have a Lobo LX-80? In that case, things are not quite so nice, but it IS possible to use an LX-80 with a clock speedup. There seem to be two components of the LX-80 which do not function at the fast clock speed -- the boot ROM and the SIO chip. The SIO chip is better known to LX-80 owners as the "RS232 option." It is what drives the two serial ports in the LX-80, if you have them. The $10 chips supplied by Lobo are only rated at 2.5 MHz, and generally do not work at 3.54 MHz. The only cure I know of for this problem is to buy a 4 MHz SIO chip. These are available, but expensive. I get around the problem by doing a SYSTEM (SLOW) every time I enter LCOMM, which is the only time I use my serial ports.**

**The LX-80 boot ROM presents a more serious problem. It is used during booting, and also contains all the code for the LDOS/LX-80 disk driver routines. If you attempt to access this ROM when operating at the fast clock speed, you will read back invalid data, and typically cause a rather dramatic crash. Once again, SYSTEM (FAST) becomes SYSTEM (CRASH). Fortunately, there is a way around this, too. Of course, you could run a wire into the LX-80 to slow down the clock during disk I/O, but this negates much of the benefit of having a clock speedup, since this type of mod actually slows down the clock whenever the drive motors are running, even if there is not actually any I/O going on.**

First of all, you must make the hardware change I described above to allow rebooting after the speedup has been turned on. (The software patch will have no effect on the LX-80.) Secondly, you need to use a program I have written called FAST/CMD, included in the program section of this newsletter. This program acts as a "front end" to the disk drivers, slowing down the clock before entry to the ROM routines, and speeding it up upon exit if SYSTEM (FAST) was active. When you execute this program, it relocates itself to high memory, protects itself, and patches itself in between the system and the ROM disk drivers. It then automatically executes a SYSTEM (FAST).

After this you can freely execute SYSTEM (SLOW) or SYSTEM (FAST) commands to turn your speedup on or off as needed. Under LDOS 5.0.2, you will be able to do a SYSGEN to save FAST/CMD after you have called it in. (Under earlier versions of LDOS, you must execute it as a direct command or AUTO command only--it will not SYSGEN properly.)

Well, that about wraps it up. If you follow the above procedures, you should be able to run quite well with a speedup mod. Mine is so convenient that I am hardly conscious I have it--but I really notice it when I am running on a computer without one (it's S-L-O-O-O-O-W!). If you have any further comments or questions about speedup mods, feel free to write LDOS Support. (Please do not call on the 800 line to discuss this subject.)

==========

## THE LDOS MANUAL STORY

The LDOS user's manual contains over 250 printed pages of information, and the text takes up over half a Meg of storage on an 8" double density diskette. It was written with Model I Scripsit, running on the LDOS operating system. The manual was started during the last week of December, 1980, and went to press at the start of the second week in February, 1981. In that 6 week period, approximately 1800 man hours went into the writing and proof reading of the manual. Since LDOS was also being "fine tuned" during that period, revisions to the text were constantly being made. The writing, test printing, proof reading, re-writing, etc. used up over 6000 sheets of paper and 2 dozen printer ribbons. The normal work week for the people involved was 9 AM to whenever, at least 6 days a week. But, we feel that the end result was well worth the work.

Your comments on the manual have been for the most part favorable, and the manual seems to be serving its purpose rather well. However, it appears that a few errors did slip by the proof readers during some of their late night sessions. What follows will be a list of corrections for the first edition. Those of you with serial numbers ending higher than 1000 will probably have the second edition (check the bottom of the Table Of Contents for the "second edition" message).

This list of corrections will not correct any typographical errors or misspellings. Most of these were hopefully corrected in the second edition, and had no real bearing on the clarity of descriptions or examples. This list will be arranged by section and page number, starting from the front of the manual.

**SYSTEM DEVICES - PAGE 4**

**The *CL device section refers to a "KSR/CMD" utility program. This program was a stripped down version of LCOMM, and was not included on the release disk. It was felt that since LCOMM contains all the functions of KSR, nothing would be gained by placing KSR/CMD on the disk.**

**LIBRARY COMMAND <COPY>**

**Using the LRL parameter will always cause the destination file to have the specified LRL, regardless of the destination file's LRL before the copy.**

**LIBRARY COMMAND <DO>**

**No commands that require removing the system disk from drive 0 may be included in a file for DO processing.**

**LIBRARY COMMAND <LOAD> - PAGE 1**

**The addresses for the LOAD command should be changed from X'5200' and X'5300' to X'51FF' and X'52FF', respectively.**

**LIBRARY COMMAND <SYSTEM> - PAGE 7**

**In the SYSGEN parameter section, item 3 states that any routines loaded into memory and protected with HIGH$ will be saved in the configuration file. It should also say that ALL memory from the physical top down to HIGH$ will be saved in the configuration file.**

**LIBRARY COMMAND <VERIFY>**

**All disk writes are automatically verified during any type of BACKUP.**

**LIBRARY COMMAND <XFER>**

**The source and destination disks used for the XFER command must have different PACK ID'S (disk name and master password) or the XFER will abort.**

**UTILITY <BACKUP> - PAGE 1**

**The X parameter is only valid for backups with drive 0 as the source drive.**

**Backup (X) and single drive backups cannot be included in a JCL file.**

**UTILITY <FORMAT> - PAGE 3**

In the section on double sided formatting, note that either a special driver routine or a hardware modification will be necessary to use a double sided drive as a single drive on the Radio Shack expansion interface.

**UTILITY <LCOMM> - PAGES 2,6**

The *KI device control on page 2 should say <CLR><1> rather than <CLR><&><1>.

The PEOF on page 6 should read <CLR><SH><6>. The <CLR><&> shown is in effect the same thing, but does not follow the format used for the other control functions.

**UTILITY <PATCH> - PAGE 4**

The command line patch example PATCH MONITOR/CMD should have parentheses around the text (X'E100=C3 66 00 CD 03 40). The parentheses must be used in all command line patches.

**DEVICE DRIVER <RS232,RS232L> - PAGES 1,2**

On page 1, the PORT= command for LOBO LX-80 owners should show PORT=0 for serial port A, and PORT=1 for serial port B.

On page 2, the default parameters for the LX-80 should show PORT=0.

The fourth paragraph describes Line Condition parameters. The correct description is as follows:

> If CTS is turned ON, the driver will wait until a Clear To Send signal is received before sending a character.
>
> If CD is turned ON, the driver will wait until a Carrier Detect signal is present before sending a character.
>
> If DSR is turned on, the absence of the Data Set Ready signal will cause the driver to wait (Radio Shack interface only).

In the example using the RS232L driver, the "Port 1" should be changed to "Port 0".

**JOB CONTROL LANGUAGE**

Page 1, paragraph 2, should end with the sentence "Also, no line in a JCL can exceed 64 characters in length.".

**Page 2, first paragraph in the "JCL USED WITH LBASIC" section should note that only those commands that do not alter high memory can be in a JCL used from LBASIC.**

**Page 4, paragraph 4, describes the logical operators available with JCL. When using these operators, no space should be left between the operator sign and the token.**

**LBASIC - See the LBASIC section of the newsletter**
**.....**

**TECH INFO <DRIVE CODE TABLE> - PAGE 1**
**.....**

**Section DCT+3, bit-4, had page formatting problems. The correct text is:**

> **A "1" will cause the selection of the disk's second side. The first side will be selected if this bit is a "0". The bit value will match the side indicator bit in the sector header as written by the FDC.**

**Section DCT+9, last paragraph, second sentence. The "highest numbered cylinder" should read "highest numbered sector".**

**TECH INFO <DIRECTORY RECORDS> - PAGES 7,11**
**.....**

**Page 7, GAT+X'CD', last sentence, should read "Bits 2-0 contain the number of granules per cylinder -1."**

**Page 11 shows the HIT positions assigned to LDOS SYStem files. The proper HIT positions for SYS6-SYS13 should be positions 20 through 27.**

**TECH INFO <ENTRY POINTS>**
**.....**

**Page 1, @ADTSK, should note that the HL register pair is used. The correct text should read:**

> **Note: The DE register is a pointer not to the location of your task driver, but to a block of RAM called the TCB, which contains the address of the task driver entry point. Upon entry to your task routine, the register IX will contain the TCB address.**

**Page 10, the @DODIR information was missing. This text should go between @DATE and @DSPLY.**

> **@DODIR .... (Vector = X'4463')**
> **This routine will read visible files from a disk directory. The screen display will be in a 4 across format. The buffer will contain directory record bytes 0-15, and bytes 20-21. An X'FF' will indicate the buffer end.**
>
> **B => Option (0=Screen display, 1=Buffer)**
> **C => Drivespec**
> **HL => Buffer address (if selected with B=1).**

**Page 14, @PAUSE should say 14.67 microseconds, not milliseconds.**

**Page 15, GETDCT should say "This routine will obtain the address of the drive code table for the requested drive.". The IY register description should read "the drive code table address".**

**TECH INFO <MEMORY MAP>**
**.....**

**X'4315'-X'4317' text changed to "used with DEBUG (do not use)".**

**X'4424', @OPEN text should read "Open an existing file or device".**

**X'4428', @CLOSE was missing from the original memory map. The text should read "Close a file or device".**

**X'4442' and X'4445', the text should refer to LOGICAL rather than PHYSICAL records.**

**GETDCT is located at X'478F', not X'468F' as stated. The text should read "Get Drive Code Table address".**

**TECH INFO <RAM STORAGE ASSIGNMENTS> - PAGE 6**
**.....**

**JFBC$ should be named JFCB$.**

**JFCB$ should be named CFCB$.**

**These corrections should take care of all the missing and incorrect data in the manual. If you have found other errors, please send us a letter detailing the section, page, and what the error is (don't use the 800 number).**
**==========**

## USER CONTRIBUTED PROGRAMS

**This section of the newsletter contains some Filter programs and other files contributed by LDOS owners. Most of these files are also in the data base of the LDOS MicroNET bulletin board. Please remember that these are not "official" LDOS programs - any questions, comments, etc. should be addressed to the person that contributed the program, not to LDOS Support. In this issue, most of the programs have been contributed by LDOS Support people (especially Tim Mann). This section of the newsletter is specifically meant for interchange between LDOS users. We are always looking for contributions for this section.**

**Note that the hex program listings must be put into the proper load module format before they can be used. There is a program included in this section called BINHEX/BAS, which will do the conversion for you. To use the program, follow these steps:**

**1) Use Scripsit, Electric Pencil, or the BUILD command to create an ASCII file containing the hex code. DO NOT leave spaces between the hex characters - the spaces were put in for readability only!. The ASCII files must NOT contain more than 254 characters (127 byte pairs) per line.**

**2) Enter LBasic, and Run the program BINHEX/BAS, choosing the "Hex to Binary" mode.**

**BINHEX/BAS**
**----------**

**Following is the listing for the LBasic program, contributed by Tim Mann.**

```
10 REM -- Hex to binary/Binary to hex file converter
20 REM -- Tim Mann
30 CLS:PRINT:PRINT"Hex to binary/Binary to hex"
35 PRINT"   file converter" PRINT
40 CLEAR 5000
50 GOSUB 58000
100 PRINT "Type 1 to convert a binary file to hex"
110 PRINT "     2 to convert a hex file to binary"
120 PRINT:INPUT D
130 PRINT
140 ON D GOTO 400,200
150 GOTO 100
200 LINE INPUT "Hex file name: ";HF$
210 LINE INPUT "Binary file name: ";BF$
220 OPEN"I",l,HF$
230 OPEN"O",2,BF$
240 IF EOF(1) THEN 320
250 LINE INPUT#1,D$
255 IF D$="" OR D$="OK" THEN 240
260 FOR I=1 TO LEN(D$) STEP 2
270    PRINT#2,CHR$(FND2(MID$(D$,I,2)));
300 NEXT I
310 GOTO 240
320 CLOSE
330 PRINT:PRINT"Done":PRINT
340 GOTO 100
400 LINE INPUT "Binary file name: ";BF$
410 LINE INPUT "Hex file name: ";HF$
420 OPEN"RO",1,BF$,1
430 OPEN"O",2,HF$
440 FIELD 1,1 AS F$
450 FOR     1=1 TO 30
455    IF EOF(1) THEN 505
460    GET 1
470    PRINT#2,FNH2$(ASC(F$));
480 NEXT I
490 PRINT#2,
500 GOTO 450
505 PRINT#2,
510 CLOSE
520 PRINT:PRINT"Done":PRINT
530 GOTO 100
58000 DEF FNH1$(X)=MID$("0123456789ABCDEF",(X AND 15)+l,1)
58010 DEF FNH2$(X)=FNH1$(X/16)+FNH1$(X)
58040 DEF FND1(X$)=INSTR("123456789ABCDEF",LEFT$(X$,1))
58050 DEF FND2(X$)=FND1(RIGHT$(X$,1))+16*FND1(RIGHT$(X$,2))
58070 RETURN
60000 END
```

**PENCTL/FLT**
**----------**

**This keyboard filter program will allow Electric Pencil owners to use the LDOS keyboard driver and the original Pencil control key. This program was contributed by Tim Mann.**

```
05 06 50 45 4E 43 54 4C 01 02 00 52 D5 1A F5 21 5F 52 CD 67 44 F1 FE 08 CA 50
52 FE 10 CA 5A 52 CB 47 CA 55 52 DD E1 DD E5 DD 6E 01 DD 66 02 22 44 53 22 47
53 2A 49 40 01 15 00 AF ED 42 22 49 40 23 EB 21 43 53 D5 ED B0 D1 DD 73 01 DD
72 02 C3 2D 40 21 FC 52 CD 7B 44 C3 30 40 21 0C 53 18 F5 21 1E 53 18 F0 21 32
53 18 EB 50 45 4E 43 54 4C 2F 46 4C 54 20 2D 20 46 69 6C 74 65 72 20 74 6F 20
61 63 74 69 76 61 74 65 20 50 65 6E 63 69 6C 20 63 6F 6E 74 72 6F 6C 20 6B 65
79 20 75 6E 64 65 72 20 4C 44 4F 53 0A 43 6F 70 79 72 69 67 68 74 20 28 63 29
20 31 39 38 31 20 54 69 6D 6F 74 68 79 20 50 2E 20 4D 61 6E   6E 20 2D 20 56 65
72 73 69 6F 6E 20 31 2E 30 0A 52 65 70 72 6F 64 75 63 74 69 6F 6E 20 70 65 72
6D 69 74  74 65 64 20 66 6F 72 20 70 65 72 73 6F 6E 61 6C 20 75 73 65 20 6F 6E
6C 79 0A 0D 50 61 72 61 01 5A 00 53 6D 65 74 65 72 20 65 72   72 6F 72 0D 44 65
76 69 63 65 20 6E 6F 74 20 61 63 74 69 76 65 0D 4E 6F 74 20 61 6E 20 69 6E 70
75 74 20 64 65 76 69 63 65 0D 44 65 76 69 63 65 20 69 73 20 72 6F 75 74 65 64
0D D2 00  00 CD 00  00 F5 3A 80 38 E6 10 28 05 F1 E6 9F 37 C9 F1 C9
```

**MX80/FLT**
**--------**

**This printer filter program will allow the MX-80 printer to directly print TRS-80 graphics with the JKL,GRAPHIC parameter. It was contributed by Ken Roser.**

```
05 06 4D 58 38 30 2F 46 01 82 00 52 1A E6 02 28 33 D5 21 41 52 CD 67 44 DD E1
2A 49 40 01 11 00 AF ED 42 22 49 40 23 DD 7E 01 32 06 53 DD 7E 02 32 07 53 F3
DD 75 01 DD 74 02 EB 21 F7 52 ED B0 FB C3 2D 40 21 D7 52 CD 7B 44 C3 30 40 0A
4D 58 2D 38 30 20 46 49 4C 54 45 52 20 2D 20 4B 65 6E 20 52 6F 73 65 72 20 2D
20 34 2F 31  31 2F 38 31 0A  0A 54 68 69 73 20 66 69 6C 74 65 72 20 77 69 6C  6C
20 73 68 69 66 74 20 54 52 53 01 82 80 52 2D 38 30 20 67 72 61 70 68 69 63 73
0A 73 6F 20 74 68 61 74 20 74 68 65 79 20 61 72 65 20 63 6F 6D 70 61 74 69 62
6C 65 20 77 69 74 68 20 74 68 65 20 4D 58 2D 38 30 0A 69 6E 20 74 68 65 20 73
74 61 6E 64 61 72 64 20 63 6F 6E 66 69 67 75 72 61 74 69 6F 6E 2E 0D 54 68 69
73 20 66 69 6C 74 65 72 20 69 73 20 66 6F 72 20 6F 75 74 70 75 74 20 6F 6E 6C
79 21 0D 38 0C 79 FE C0 30 07 FE 80 01 0A 00 53 38 03 C6 20 4F C3 00   00 02 02
00 52
```

**BEEP/FLT**
**--------**

**This keyboard filter will output a "beep" tone to the cassette port each time a key is pressed. It was contributed by Tim Mann.**

```
05 06 42 45  45 50 2F 46 01 02 00 52 D5 1A F5 21 5F 52 CD 67 44 F1 FE 08 CA 50
52 FE 10 CA 5A 52 CB 47 CA 55 52 DD E1 DD E5 DD 6E 01 DD 66 02 22 03 53 22 06
53 2A 49 40 01 31 00 AF ED 42 22 49 40 23 EB 21 02 53 D5 ED B0 D1 DD 73 01 DD
72 02 C3 2D 40 21 BB 52 CD 7B 44 C3 30 40 21 CB 52 18 F5 21 DD 52 18 F0 21 F1
52 18 EB 42 45  45 50 2F 46 4C 54 20 2D 20 4B 65 79 62 6F 61 72 64 20 61 75 64
69 6F 20 66 65  65 64 62 61 63 6B 20 66 69 6C 74 65 72 0A 43 6F 70 79 72 69 67
```

```
68 74 20 28 63 29 20 31 39 38 31 20 54 69 6D 6F 74 68 79 20 50 2E 20 4D 61 6E
6E 20 2D 20 56 65 72 73 69 6F 6E 20 31 2E 30 0A 0D 50 61 72 61 6D 65 74 65 72
20 65 72  72 6F 72 0D 44 65 76 69 63 65 20 6E 6F 74 20 61 63 74 69 76 65 0D 4E
6F 74 20 61 6E 20 69 6E 70 75 74 20 64 65 76 69 63 65 0D 44 65 76 69 63 65 20
69 73 20 72 6F 75 74 65 01 35 00 53 64 0D D2 00  00 CD 00 00 C8 D5 C5 F5 3E 01
32 E4 37 06 05 16 0F 3A 0F 43 CB 5F 28 04 CB 20 CB 22 3A 3D 40 F6 01 4A EE 03
D3 FF 0D 20 FD 10 F6 F1 C1 D1 C9 02  02 00 52
```

**UPDATE/BAS**

---

This LBasic program will allow you to update your LDOS working disks after you have sent in your Master disk for an update. It will copy only those programs which already exist on the working disk. It was contributed by Tim Mann.

```
10 REM--UPDATE/BAS
20 REM--Updates all LDOS files present on a disk
30 REM-- by copying them from a disk containing
40 REM-- the latest release.
50 REM--Tim Mann 03/25/81
55 CLEAR 1000
60 DEF FNS$(X$)=LEFT$(X$,INSTR(X$+" "," ")-1)
70 DEF FND$(X)=":"+RIGHT$(STR$(X),1)
100 REM--Initialize
110 CLS:PRINT
120 INPUT"Source drive";D1
130 INPUT"Destination drive";D2
133 IF D1=D2 PRINT "Source and destination drives cannot be the   same!":GOTO
120
135 IF D2=0 PRINT "Destination drive cannot be drive 0!":GOTO 130
140 PRINT
150 D1$=FND$(D1):D2$=FND$(D2)
200 REM--Loop through directory
210 OPEN "RO",1, "DIR/SYS"+D1$,32
220 FIELD 1,1 AS XF$,4 AS XD$,8 AS XN$,3 AS XE$
230 FOR I=17 TO LOF(1)
240    IF I=17 OR I=25 THEN 350  '17 is BOOT/SYS, 25 is DIR/SYS
250    GET 1,I
260    IF (ASC(XF$)AND&H90)<>(&H10) THEN 350  'Go if dead file
270    N$=FNS$(XN$)+"/"+FNS$(XE$)  'Form name
280    ON ERROR GOTO 500    'Skip if not on destination drive
290    OPEN "I",2,N$+".P3UF"+D2$
300    ON ERROR GOTO 0
310    CLOSE 2
320    C$="COPY "+N$+" . RS0LT0FF"+D1$+" "+D2$
330    PRINT "Copying file ";N$
340    CMD C$   'Do COPY command
350 NEXT I
360 PRINT
370 END
500 RESUME 350
```

**FAST/CMD**
**--------**

**This program will allow LX-80 interface owners to use a fast clock modification without hardware modification of the LX-80 interface. See the article in this newsletter for more information. It was contributed by Tim Mann.**

```
05 06 46 41 53 54 3A 32 01 59 00 52 21 56 52 ED 5B 49 40 01 24 00 ED B8 ED 53
49 40 13 21 01 47 06 08 73 23 72 3E 09 85 6F 10 F7 21 25 52 C3 05 44 53 59 53
54 45 4D 20 28 46 41 53 54 29 0D E5 F5 21 0F 43 7E CB 9E 67 AF D3 FE F1 E3 CD
8A 30 E3 F5 7C 21 0F 43 CB 5F 28 06 3E 01 D3 FE CB DE F1 E1 C9 02   02 00 52
```

**USRFREE/CMD**
**-----------**

**This is a short utility that can be called as a USR routine from LBasic, and will return the amount of free K on a disk. An empty or missing disk will return a zero. To make a program module, use the DEBUG command to enter the program into memory at the desired location, and then use the dump command to save it to disk. Do not use the BINHEX/BAS program on this module! To call this USR routine, use the syntax "X=USR1(drivespec)". This routine was contributed by Chuck Jensen.**

```
CD 7F 0A 4D 7D CD B8 44 20 3D CD 8F 47 FD 46 06 04 FD 56 09 21 00 42 1E 00 CD
45 4B 11 00  00 78 06 08 CB 46 20 01 13 CB 0E 10 F7 3D FE 00 28 03 23 18 ED EB
FD 7E 08 3C E6 1F CD 8F 4B 65 6F 3E 04 CD A9 4B C3 9A 0A 21 00   00 C3 9A 0A
```

**UPCASE /FLT**
**-----------**

**This is a general filter for use with any device (*KI, *DO, *PR, *CL, etc.). It will convert all characters to upper case. This should be useful for those without lower case conversions. It was contributed by Tim Mann.**

```
01 53 00 52 D5 DD E1 DD 6E 01 DD 66 02 22 3D 52 22 40 52 01 24 00 2A 49 40 AF
ED 42 22 49 40 23 E5 EB 21 2D 52 ED B0 E1 DD 75 01 DD 74 02 C3 2D 40 38 10 F5
79 FE 7F 30 06 FE 61 38 02 CB A9 F1 C3 00  00 CD 00 00 F5 FE 7F 30 08 FE 61 38
04 F1 CB AF C9 F1 C9 02  02 00 52
```

**P E R C O M  D O U B L E R  D R I V E R  (PDUBL/CMD)**
**------------------------------------------------**

**The disk driver program PDUBL is provided to allow you to use the Percom Doubler II double density modification with LDOS.**

```
  ========================================================
  !  PDUBL                                               !
  !                                                      !
  !      No parameters are required or allowed.          !
  !                                                      !
  ========================================================
```

**This command loads a special disk driver program which allows you to use the Percom Doubler II to read, write, and format double density disks.**

**If you have a Doubler installed, after you give this command, you can use either single or double density disks in any of your 5" disk drives. LDOS will automatically recognize whether you have a single or double density diskette in a drive, and react accordingly.**

**PDUBL also includes support for double-sided 5" drives on a Radio Shack expansion interface with the Doubler. Both sides of the diskette are treated as a single volume. The drives and cabling must be set up correctly for this feature to work.**

**The PDUBL driver is loaded into high memory and protected by lowering HIGH$. Logical drives 0-3 are set up to use this driver in place of the normal LDOS single density driver. Under LDOS 5.0.2 or later, you can use the SYSTEM (SYSGEN) command to save the driver in your configuration file, to be loaded automatically every time you boot. Please note that you CANNOT boot up on a double density LDOS diskette when using the Percom Doubler. You may, however, boot up on a single density diskette and exchange it for a double density diskette as soon as the bootstrap operation has finished.**

*** * * WARNING! * * ***

**The double density disk format used by Percom's DBLDOS is NOT compatible with LDOS! The LDOS double density format is the same in all versions of LDOS -- Model I with the Doubler, Model I with the Lobo LX-80, and Model III. It could not be changed to match DBLDOS.**

```
05 06 50 44 55 42 4C 20 01 02 00 52 21 5A 52 CD 67 44 DD 21 C4 52 2A 49 40 11
9D 54 B7 ED 52 44 4D 3E 18 DD 6E 00 DD 66 01 23 5E 23 56 EB 09 EB 72 2B 73 DD
23 DD 23 3D 20 E9 ED 5B 49 40 21 9D 54 01 AA 01 ED B8 ED 53 49 40 13 EB FD 21
00 47 06 04 11 0A 00 FD 75 01 FD 74 02 FD CB 04 F6 FD 19 10 F2 C3 2D 40 50 44
55 42 4C 20 2D 20 44 72 69 76 65 72 20 66 6F 72 20 74 68 65 20 50 65 72 63 6F
6D 20 44 6F 75 62 6C 65 72 20 49  49 20 2D 20 56 65 72 73 69 6F 6E 20 54 45 53
54 31 0A 43 6F 70 79 72 69 67 68 74 20 28 63 29 20 31 39 38 31 2C 20 4C 6F 67
69 63 61 6C 20 53 79 73 74 65 6D 73 20 49 6E 63 6F 72 70 6F 72 61 74 65 64 0D
2F 53 01 53 4B 53 65 53 8E 53 A0 53 A3 53 B7 53 CF 53 D8 53 DF 53 E6 53 F1 53
0D 54 19 54 2E 54 33 54 01 54 10 53 5A 54 98 54 BD 53 9B 54 7C 53 3A 0F 43 CB
5F 28 12 DB FE CB 87 D3 01 02 00 53 FE CD 0D 53 F5 DB FE CB C7 D3 FE F1 C9 78
CB 58 C2 A0 53 FE 07 28 40 FE 06 28 4A 3D 28 11 FD 34 05 FE 04 06 58 28 67 FD
36 05 00 06 08 18 5F CD 57 53 07 F5 FD 7E 03 E6 10 0F FD B6 04 E6 0F 32 09 43
32 E1 37 F1 D0 FD CB 03 56 CC 4E 53 C5 01 00 7F CD 60 00 C1 C9 3A EC 37 CB 47
C8 3A 09 43 32 E1 37 18 F2 CD 57 53 FD 7E 05 32 ED 37 FD 7E 07 E6 1F 3C FD CB
03 A6 93 ED 44 D5 FA 84 53 FD CB 03 E6 5F ED 53 EE 37 D1 FD 72 05 06 18 CD 2F
53 FD 7E 03 E6 03 B0 32 EC 37 06 08 10 FE AF C9 CD 57 53 CD 6C 54 78 CB 50 06
06 20 16 06 04 0E 88 FE 0A 28 08 06 0A CD D5 53 01 1A 02 CD D5 53 01 1A 7E 0E
A8 FE 0E 38 06 0E F4 20 02 0E A9 CD D5 53 09 12 0A E3 7E 23 32 65 54 7E 23 66
6F 22 29 54 E1 C5 CB 61 CC 65 53 D5 E5 21 EC 37 11 EF 37 CD 57 53 79 FE A9 20
08 FD CB 03 76 20 02 3E 01 A3 00 54 AB CD 97 53 CB 69 C1 C5 20 0A 7E E6 03 E2
0A 54 F3 1A 18 16 FB F3 7E E6 F6 CA 14 54 0A 12 CB 46 28 24 03 0A 03 CB 4E 28
FC 00 00 03 CB 4E C2 29 54 CB 4E C2 29 54 CB 4E 20 EF CB 4E 20 EB CB 4E 20 E7
CB 46 20 E6 FB 7E E6 7C E1 D1 C1 C8 CB 57 20 91 F5 E6 18 28 0B CB 67 C5 C4 7E
54 C1 F1 10 82 06 F1 47 3E 00 CB 08 D8 3C 18 FA FD 7E 03 E6 40 07   07 F6 FE 32
EC 37 3E D0 32 EC 37 C9 E5 FD 7E 03 EE 40 FD 77 03 21 09 24 CB 77 28 03 21 11
45 FD 75 07 FD 74 08 E1 CD 6C 54 C3 27 53 56 33 39 02  02 00 52
```

**DCAL/BAS**
**--------**

**This program will allow you to check your disk drive motor speed. It runs only on the Radio Shack expansion interface. Contributed by Tim Mann.**

```
10 REM--DISK DRIVE TIMING PROGRAM
20 REM--TIM MANN
30 GOSUB 8000   'LOAD MACHINE LANGUAGE ROUTINE
40 OUT 254,0    'ENSURE CLOCK SET TO 1.774  MHZ
50 CLS:PRINT:PRINT"TRS-80 DISK DRIVE SPEED CALIBRATION PROGRAM"
60 PRINT"INSERT A DISK IN THE DRIVE TO BE CALIBRATED":PRINT
70 INPUT "DRIVE NUMBER (0-3)";DR
80 IF DR<0 OR DR>3 OR DR<>INT(DR) THEN 70
90 DR=2[DR
100 DEF USR1=&H900C
110 FOR I=1 TO 100  'START DRIVE
120   POKE &H37E1,DR
130 NEXT I
140 POKE &H37EC,3   'RESTORE COMMAND
150 FOR 1=1 TO 100:NEXT
160 A=USR1(DR)
170 CT=PEEK(&H9092)+256*PEEK(&H9093)
180 RPM=60*1.774E6/(CT*35+112)
190 IF SBAR=0 THEN SBAR=RPM
200 SBAR=SBAR*.9+RPM*.1
210 PRINT USING "SPEED = ####.# -- SMOOTHED SPEED = ####.#";RPM,SBAR
220 GOTO 160
8000 REM--LOAD MACHINE LANGUAGE ROUTINE
8010 FOR I=(&H900C) TO (&H9047)
8020 READ B:POKE I,B:NEXT I:RETURN
9000 DATA 205,127,10,125,243,237,115,128,144,17
9010 DATA 225,55,221,33,144,144,6,2,49,0
9020 DATA 0,33,236,55,18,203,78,51,202,36
9030 DATA 144,18,203,78,51,194,43,144,33,0
9040 DATA 0,57,221,117,0,221,35,221,116,0
9050 DATA 221,35,16,220,237,123,128,144,251,201
```

**==========**

**The addresses of the contributing people are:**


| | | |
|---|---|---|
| **Tim Mann** | **Chuck Jensen** | **Ken Roser** |
| **4139 N. 78th Ct.** | **4869 N. 71st St.** | **1907 Arbor Lane** |
| **Milwaukee, WI 53222** | **Milwaukee, WI 53218** | **Union, NJ 07083** |