NORTHERN BYTES

Volume 5 Number 2

Welcome to the second edition of the "reborn" NORTHERN BYTES. I'd like to start out this issue by saying that if you are the editor of a TRS-80 User Group Newsletter, PLEASE make sure that we are on your exchange newsletter mailing list. The address for newsletter exchanges is: NORTHERN BYTES, c/o Jack Decker, newsletter editor, 1804 West 18th Street - Lot # 155, Sault Ste. Marie, Michigan 49783. Please do NOT send NORTHERN BYTES exchange newsletters to The Alternate Source, since they will just take longer to reach me. Also, you may already have us on your exchange list under 80-USER DIGEST or MICROCOMPUTER USERS INTERNATIONAL or even under my name (Jack Decker), you may want to change this to NORTHERN BYTES for uniformity. If you are sending two copies of your newsletter each month, you need not do so any longer, since 80-USER DIGEST is not being published (see NORTHERN BYTES, Volume 5, Number 1 for details). In short, please make sure that we are on your exchange newsletter list - once - and if it's not too inconvenient, under the name NORTHERN BYTES.

Just in case you had any doubts, user groups are still free to reprint anything appearing in NORTHERN BYTES in your user group newsletter. Please be sure to credit the original source of the article - and please note that WE often reprint from other newsletters. We have noticed that some clubs are pretty lax about acertaining the original source of an article - for example, we've seen reprinted articles credited to NORTHERN BYTES, when in fact we reprinted the article from another newsletter, and said so at either the beginning or end of the article. We really appreciate it when you credit the ORIGINAL author and the ORIGINAL source of the article - we really don't want to take credit where it isn't due.

TRS-80 user groups can skip this paragraph, but other user groups please note: If your group is a "general interest" computer club that does NOT serve TRS-80 owners, please let us know if you wish to discontinue our newsletter exchange. From here on out, NORTHERN BYTES will be a TRS-80 ONLY publication. We are happy to continue to exchange newsletters with any user group that desires to continue the exchange, however. If your group originally exchanged newsletters with Microcomputer Users International, and if you have Apple Computer users in your group and wish to remain in contact with the "Apple Users Group of MUI", you should contact Bert Williams, 47 Tuckett Street, Sault Ste Marie, Ontario P6A 4G8, Canada, as Bert is the current head of that group (which is the only remaining active user group of Microcomputer Users International).

One note for TRS-80 users that were members of Microcomputer Users International (or that might have become members had the organization continued) - I have received about 100 postpaid cards from Computer User (the magazine that took over The Alternate Source Programmer's Journal subscriptions) that entitle club members to receive twelve months of Computer User for only $17.95 ($21.00 for Canadian subscriptions). Payment can be made by check (to Computer User, not to me!), or by VISA, Mastercard, or American Express card. It is definitely worth the money! If you'd like a card (or more information), contact me (or send a 20 cent U.S. postage stamp, or 20 cents U.S. or a Canadian quarter if you want me to mail you a card). Note: I have no way of knowing who "might" have become an MUI member had the club continued, so the cards will be made available on a "first come, first served" basis.

I'd like to close my opening remarks with an appeal to Ashton-Tate to continue their support of the TRS-80 version of dBaseII (their relational database management system). The TRS-80 needs this kind of software support if it is to survive. Charley Butler says that The Alternate Source would even be willing to assist Ashton-Tate in any way possible in their continuing support of the TRS-80 market, just to keep this fine product available to TRS-80 users. Also, for owners of TAS' EDM program, a macro is available that will generate dBaseII compatible source files (if you're interested, contact Charley and request more information).

THE EXTERMINATOR: Longtime readers of this newsletter will realize that this is the column where we vanquish the BUGS and other errata that has appeared in previous issues of NORTHERN BYTES.

However, before we cover previous issues of this newsletter, I'd like to point out a few errors in the first printing of my new book, TRS-80 ROM ROUTINES DOCUMENTED. First of all, a bogus routine somehow slipped in - it's the one at 0FD9H (entitled "PREPARE INTEGER FOR DISPLAY"), and it's falsely documented about halfway down page 35. Actually, there is a routine at 0FD9H and it is used for the stated purpose, but what the book doesn't explain is that you have to do quite a bit of work setting up registers and memory locations properly before calling the routine, or it won't work properly. It's totally unnecessary to do that though, since you can use the routine at 0FBDH (or 0FBEH, if you want special formatting), which will accomplish the same thing. Those routines are documented just prior to the errant routine (on pages 34 and 35).

Unfortunately, the description of the routine at 0FBDH on page 34 isn't entirely correct, either. See where it says, "... with the result string address (4130H) returned in the HL register pair."? Well, first of all, the address isn't necessarily 4130H, it might be a bit above that. And second, what HL really points to is a leading space character just prior to the actual ASCII representation of the number, so if you don't want a leading space character, just INC HL before you call whatever print routine you're using. When we do the second printing of the book, I'll completely remove the paragraph describing 0FD9H, and will change the paragraph describing 0FBDH to read something like this:

0FBDH PREPARE NUMBER IN ACCUM FOR DISPLAY (NUMERIC EDIT)
Non-formatted numeric edit routine. Converts number in ACCUM to display format. On exit, ASCII string will be stored in buffer located at 4130H - 4149H, and HL register will point to the last leading space (that is, the space just before the first non-space character of the created string). When positive integers (such as BASIC line numbers) are converted, the area from 4130H to 4136H is used, and the string is stored right-justified with leading spaces in locations 4130H to 4135H (4130H always contains a space character), while 4136H always contains a zero byte to terminate the string. Other number types may be converted and stored differently within the buffer.

The "(4130H)" bug also slipped into the description of the routine at 0FDEH, which also begins on page 34. The last sentence of the description of that routine (on page 35) will be changed to read, "On exit, HL points to the leading space character just before the start of the resulting string, and if number was single- or double-precision, DE points to end of resulting string plus one."

My apologies for these errors - I must have been half asleep when I wrote those two pages. I hope they haven't caused anyone any great difficulty.

There are a couple of other minor errors that need correction. On page 113 (yes, Charley, page 113 again!), in the first paragraph (following the Assembly Language code at the top of the page), there is a reference in the first sentence to "...two types could have been saved...". That should have been two BYTES, not types (I guess it was just a type-o). Finally, on page 15, the seventh line down from the top of the page reads in part as follows: "in range 0 to 2FH, then PUSH AF. Finally, jump to 150H...". The phrase "then PUSH AF" is wrong IN THIS LINE ONLY. Please line out or otherwise delete this one occurrence of the phrase "then PUSH AF", if you happen to have a copy of the first edition of my book. Thanks to Dave McGlumphy for catching that error. All three of these errors will be corrected in future printings. In addition, future printings will contain a four page "Hexadecimal Address Cross-Reference", which lists just about every hexadecimal address in ROM or low RAM that is mentioned

anywhere in the book, along with the page number(s) on which the address appears (an invaluable aid in locating all of the information pertaining to a given ROM routine and/or memory address).

If you have purchased a copy of the first printing of my book and would like a copy of the cross-reference (or if you are a perfectionist and would like "good" copies of the pages that contain errors), please send a large (9"x12" or thereabouts) envelope, onto which you written your name and address and affixed 37 cents postage (or 71 cents if you want me to throw in a piece of stiff cardboard to keep the post office from mangling the sheets). Also, please include "proof of purchase" (a copy of your receipt, or if you've misplaced it, a copy of your cancelled check will do - as a last resort, you could remove the final page of the book (pages 121-122) and send it, since that page will be replaced in the upgrade anyway). Finally, PLEASE specify how your copy of the book is punched (three hole or spiral). Don't forget any of the three items mentioned (large SASE, proof of purchase, and information as to how your copy of the book is bound) - if you omit one of the first two, you won't receive ANY reply, and if you forget the binding info, you'll get unpunched pages! And, please be patient, I do get "snowed under" with mail at times!

And speaking of Dave McGlumphy (I did, a couple of paragraphs back), here's information on a few bugs in a program that he originally wrote, and that I proceeded to "improve"...

COMPDIR 2.0 BUGS by Jack Decker - Some TRS-80 user group newsletters and Bulletin Board Systems have recently carried my rewrite of Dave McGlumphy's COMPDIR program, known as COMPDIR 2.0 (the program listing appeared in NORTHERN BYTES Volume 4 Number 7, on page 5). The listing as published contained four bugs that need to be fixed, as follows:

1) The program as published will sometimes terminate with an "Out of String Space" error when both disks have near the maximum 222 directory entries possible under NEWDOS/80. Even if this error doesn't occur, the user could experience string compression ("garbage collection") delays as the filenames are being printed.

2) Eight-character filenames with a one- or two-character extension are not listed properly (for example, a file with the filespec FILENAME/A would be printed by COMPDIR as FILENAMEA/A).

3) When a drive number is specified in an output filespec an "Extra Ignored" message is returned and the drive number is not recognized.

4) The program will not accept lowercase input ("y" or "n") in response to the "INCLUDE INVISIBLE FILES?" and "INCLUDE SYSTEM FILES?" questions.

To fix these bugs, add line 45 as shown below, and make the following changes to lines 1, 10, 40, 70, 130, and 170 of COMPDIR 2.0:

```
1 REM  COMPDIR 2.2 ..... (change version number to 2.2)

10 CLS: CLEAR 6000: ..... (change argument of CLEAR to 6000)

40 IF B>1 LINE INPUT "OUTPUT FILESPEC? ";X$: ..... (change INPUT to LINE INPUT, and add question mark and space characters to prompt string)

45      L=PEEK(16598): M=PEEK(16599): IF (L+M*256)-(PEEK(16544)+PEEK(16545)*256)<75 THEN M=FRE(X$): GOTO45
(add the above line to the program)

70 S=INSTR(P$(J)," "): IF S=0 OR S>9 LET S=9 ..... (add OR S>9)

130 POKE 16598,L: POKE 16599,M: NEXT J ..... (add these POKES to start of line)

170 X$=INKEY$: IF X$<>"N" AND X$<>"n" AND X$<>"Y" AND X$<>"y" THEN 170 ELSE PRINT X$: IF X$="N" OR X$="n" THEN ..... (add tests for lowercase response)
```

Note that only the portions of the lines that need to be changed are shown - don't delete the remainder of the line just because it isn't shown above!

The purpose of the new line 45 and the added code in line 130 is to eliminate "garbage collection" delays during the filename printout phase of the program. Since all strings created in the printout loop (lines 60 - 130) are temporary (that is, we don't need them once they have been printed to video, printer, and/or disk file), we can force BASIC to reclaim and reuse the same string space each time the loop is executed by simply resetting the "free string space" pointer at 40D6H-40D7H at the end of each pass through the loop. In line 45 we test to see that there is enough free string space left (the loop creates up to 74 bytes worth of new strings, so I test for 75 to be on the safe side). If enough string space is not available, we use the FRE(X$) function to force a garbage collection (this will rarely if ever need to be done). In any event, when we exit line 45 variables L and M will contain the LSB and MSB of the "free string space" pointer (BEFORE we create any "temporary" strings), which we then POKE back into 40D6H and 40D7H at the end of each loop (AFTER the "temporary" strings have been created). Maybe you haven't understood a word of this, but take it from me, it eliminates "garbage collection" delays nicely.

For a somewhat more complete explanation of how memory locations 40D6H and 40D7H can be manipulated to avoid "garbage collection" delays, see page 69 in the book "TRS-80 ROM ROUTINES DOCUMENTED" by yours truly. No, this didn't start out to be a plug for my book, but then who am I to let such an opportunity pass?

QUESTIONS AND ANSWERS ABOUT "NORTHERN BYTES" - The purpose of this column is to answer some of the most-frequently-asked questions about NORTHERN BYTES:

Q. What is NORTHERN BYTES, and why did I receive a copy in the mail?

A. NORTHERN BYTES was originally the club newsletter of Microcomputer Users International, a "general interest" computer club that served microcomputer users in the Sault Ste. Marie, Michigan U.S.A./Ontario Canada area. However, MUI has all but disbanded (only the Apple Computer user group is still active, and they are hoping to publish their own newsletter in the near future). During the entire period that NORTHERN BYTES has been published, it has had only one editor (Jack Decker). Since there is no longer an active TRS-80 user group in the Sault Ste. Marie area (except for an independent group of Color Computer users), and since I wanted to remain in contact with the various clubs and user groups around the country, and since The Alternate Source was willing to "sponsor" the publication in the absence of an active user group to support it, I decided to continue publishing NORTHERN BYTES on an irregular basis. The format has been changed somewhat, mostly in that it is now a TRS-80 only publication.

You got this issue in for one of several reasons: 1) You are a club or user group that is on our newsletter exchange list, 2) You are a former subscriber to OPINION-80, a now-defunct publication of The Alternate Source, and are receiving NORTHERN BYTES in fulfillment of your OPINION-80 subscription, 3) You have ordered from The Alternate Source, and requested that a free copy of NORTHERN BYTES be included with your order, 4) You have sent sufficent postage (usually 20 or 37 cents U.S., depending on whether there are more than twelve pages in the issue) and a self-addressed peel-off mailing label, or a self-addressed stamped envelope with 37 cents postage affixed, and requested a free copy of NORTHERN BYTES, 5) You have paid $2.00 to The Alternate Source to remain on their flyer mailing list (U.S., Canada or Mexico only, $4.00 in U.S. funds elsewhere), and are receiving NORTHERN BYTES as a free bonus. 6) You just got lucky, and received a promotional copy of NORTHERN BYTES.

Q. Can I subscribe to NORTHERN BYTES?

A. No, beacuse NORTHERN BYTES is an irregular publication. This means that while we intend to publish future issues, we don't PROMISE it. Nor do we have any sort of "timetable" for publishing new issues, so we can't say when any future issues might appear. We realize that many folks might not want to bother having to send postage or a SASE each time a new issue appears, but on the other hand, we don't want to take subscription money for a publication and then not deliver. Right now, the best way to assure that you receive future issues of NORTHERN BYTES is to remain on The Alternate Source's active mailing list, either by placing an order or by sending $2.00 to remain on the list (which automatically gets you the next issue of NORTHERN BYTES, although it would be a good idea to mention the fact that you already have this issue, so you don't wind up with a duplicate). You will then at least be notified when future issues of NORTHERN BYTES are available.

The only other way to get NORTHERN BYTES is to use one of the methods mentioned in the previous answer. That is, you can request a free copy of the current issue with any order from TAS, or you can send a self-addressed peel-off mailing label with sufficent postage (usually 20 cents in U.S. funds for a 12-page issue, or 37 cents for an issue with more than 12 pages) or a self-addressed stamped envelope with 37 cents postage. The advantage to the latter methods is that you get First Class delivery instead of bulk rate.

Canadians - If you wish to request a single issue and don't have U.S. stamps or coins handy, we'll be happy to take one or two Canadian quarters instead (send two for an issue of more than twelve pages).

If you live in a country other than the U.S.A., Canada or Mexico, you may send $4.00 in U.S. funds and get on the TAS flyer mailing list (and automatically receive the next issue of NORTHERN BYTES). Note that we cannot use postage stamps from countries other than the U.S.A.! If you belong to a local TRS-80 user group, you might consider getting them to exchange newsletters with us, but if you're not part of a user group (or just want your own private copy of NORTHERN BYTES) and you do not wish to place an order, send the $4.00 to get on the mailing list and we'll let you know what to do to get future issues. For your information, each copy of NORTHERN BYTES mailed overseas by surface mail costs 23 cents (for a 12 page issue) or 37 cents (for more than 12 pages), while airmail costs 50, 60, or 70 cents (depending on distance from the U.S.A.) for a 12 page issue, or 76 cents, 96 cents, or $1.16 for an issue of more than 12 pages.

Does it sound like we don't have all the bugs out of our distribution system yet? Well, let me put it this way - your suggestions at this point could influence our future methods of distributing NORTHERN BYTES. We reserve the right to start charging a fixed amount for NORTHERN BYTES at some future date (and to perhaps start offering subscriptions, if we decide to publish regularly), but at present the above information applies.

One final comment - one of the big reasons that we want to keep NORTHERN BYTES on a "non-commercial" basis is that many of the computer clubs and user groups with which we exchange newsletters offer reprint privileges for non-commercial use only, so to go commercial could potentially cut us off from that resource. I don't know if selling individual issues or subscriptions can be considered "commercial", if the price covers our production costs and postage only (I don't think so - any comments from the user groups?). Just in case you thought otherwise, let me assure you that at present NORTHERN BYTES is not a profit-making venture! However, at this point in time we can choose what direction to take with future issues of NORTHERN BYTES, and are probably as open to your comments and suggestions now as we will ever be.

Q. Are back issues of NORTHERN BYTES available?

A. Only limited copies of Volume 5, Number 1 (our previous issue, 12 pages in length) are still available. No previous copies are available. We may at some time start reprint the best of the TRS-80 articles from our early issues, if there would be any demand for it.

Q. I wrote you a letter and didn't get an answer. Why?

A. Most likely, you live in the U.S. or Canada and didn't enclose a self-addressed, stamped envelope (or postcard). We get LOTS of mail, and when someone writes and doesn't enclose the SASE, that letter goes on the bottom of my priority list. If you don't have a 20 cent U.S. stamp handy, then at least send 20 cents in coin (a Canadian quarter will also do) and a self addressed envelope, and I'll put on the stamp.

The other reason may be that you asked a question that requires either a lengthy explanation, information I don't have readily available, or for some other reason may require a great deal of time to adequately answer. The moral here is, try to keep your questions short and simple, or at least word them in such a way that if I can't help you, I can print your question in NORTHERN BYTES and try and get an answer for you that way.

One final point. Don't be surprised if you get your original letter back, with an answer written next to your question or on the back of the page. It may not be neat, but it's an answer and takes a whole lot less time than writing a formal letter.

Q. I'm a former OPINION-80 subscriber. What's in NORTHERN BYTES for me?

A. Well, for one thing, plenty of opinion! I'm a fairly opinionated person, as you may already know. However, you are always welcome to send in your opinions and "guest editorials" for possible inclusion in NORTHERN BYTES. And, of course, we hope you find the other TRS-80 related articles of interest as well.

Q. Do you plan to support all Models of the TRS-80?

A. We do NOT plan to support the Models II/12/16 (there is a National TRS-80 Model II Users Group at P.O. Box 234, Ada, Michigan 49301). We definitely DO plan to support the Models I/III/4 (except that for the present, we do NOT plan to support the use of CP/M on the Model 4). The amount of support we offer for Model 100 and Color Computer users will depend on what kind of material we receive for publication (in other words, we WILL support the Model 100 and Color Computers IF we can get enough good material to print). One thing Color Computer users should be aware of is that we are not interested in supporting "users" (folks that only want to use their computer, without learning anything about it) or game-players - we want to help folks learn how their computer works, and how to program it. If all you want to know is how to get a higher score in a certain game, you won't find much of interest here!

Q. Can I submit articles to NORTHERN BYTES? And, if you accept them, how much do you pay?

A. Yes, submitted articles are always welcome, and especially so if they are provided on machine-readable media (Model I/III/4 format preferred - we'll return your disk or cassette if you so request). Now, the bad news! We don't normally pay for submitted articles, EXCEPT that if we use your article, you will automatically be placed on the NORTHERN BYTES mailing list for an indefinite length of time (we'll let you know). This only applies to articles submitted directly to NORTHERN BYTES, not to articles reprinted from exchange newsletters, UNLESS either the editor of the newsletter or the author of the article we have reprinted will take the time to write us and advise us of the author's current address. Note that we assume that all programs submitted to NORTHERN BYTES may be placed in the public domain, so that they may be reprinted in other TRS-80 user group newsletters.

Once in a blue moon, The Alternate Source may actually "purchase" a program with the intent of publishing it in NORTHERN BYTES and/or placing it in the public domain. This is only done with programs that have exceptional value, but for one reason or another would probably not be a commercial success. Payments are generally small, or sometimes an allowance toward the purchase of TAS products may be made. This is not the normal method of acquiring material for NORTHERN BYTES, but we mention it because it may happen. If you think you've seen a blue moon, and want to discuss getting payment for an article, you'll have to discuss the matter with Charley Butler at TAS. Anything sent directly to NORTHERN BYTES is assumed to be a contribution!

Since NORTHERN BYTES is a newsletter rather than a magazine, we prefer the shorter, more concise programs and articles. We also like to receive the short, one or two paragraph "tips and techniques" that can be very valuable to TRS-80 users, but are just too short to build a full-fledged magazine article around. But, we'll consider anything you want to send us, so don't hold back just because you think we wouldn't be interested (after all, at the rates we don't pay, we can't be too picky!).

Q. What's the best way to send an article or program to you.

A. You can do it any of four ways. First, you can send us a diskette. We can read almost any normal TRS-80 Model I/III/4 format diskette, but PLEASE tell us how you formatted the diskette (with which operating system, using what density and so forth). Second, you can send us a 500 baud cassette (we haven't forgotten that some folks don't have disk systems yet, but your editor has a Model I at home, so please make life a bit easier for us and don't send a 1500 baud cassette). Third, you may transmit your article or program through MCIMAIL (to user name JDECKER), which will cost you as little as one dollar - but please leave your normal return address because if we need to write you, we may use the U.S. Mail (see article elsewhere for more information on MCIMAIL, and how to contact us through the MCIMAIL electronic mail system). Fourth, you can try to call the editor and upload your material over the phone line - but please don't do this unless you're willing to call back if I'm not home, or to be called back collect. As a general rule, I do NOT return long distance calls unless the caller specifically states that I can call collect.

Q. Is there any information that you'd particularly like to receive.

A. Yes - anything that will help TRS-80 users to make better use of the capabilities of their computer. But, right at

present, we're thinking about doing a special issue of NORTHERN BYTES that will contain all known patches and zaps to TRS-80 software, especially TRS-80 operating systems and the like. So, if you've developed patches or zaps that fix bugs in or enhance the capabilities of various software or operating systems far just make them easier to use), send them in. Even if you're not the original author, send them in anyway (please credit the original author if you know who he is, though). The only thing I'm not really interested in is the official zaps and patches - for example, we don't want to reprint any of Apparat's 90 zaps to Newdos/80 version 2, but if you know of a zap that Apparat hasn't published, send it to NORTHERN BYTES, Same principle applies for other operating systems and software packages.

Q. Will you accept outside advertising?

A. Yes, but only *a* limited amount, and even then only under certain circumstances. For non-commercial "unclassified" advertising, the cost is 10 cents per word per issue, but your name, address, and telephone number are free (within reason - if your address is extra long, we might edit it a bit). Payment must be made in advance, or you may charge it to your VISA or MASTERCARD. One exception: TRS-80 User Groups may place a FREE ad to promote the user group or its activities (but not to sell products). User Group ads may contain up to 25 words (not counting name, address, or phone numbers) and will be run once, but you may re-submit the same ad as many times as you like. In the near future, we may publish a directory of TRS-80 User Groups that exchange newsletters with us,

As for display advertising, the price will depend on the circulation of the issue of NORTHERN BYTES in which you wish to advertise, and certain other factors. For information on display advertising, please contact Charley Butler at The Alternate Source (telephone (517) 482-8270), Note that running a display ad in NORTHERN BYTES may be the perfect way to "test market" a TRS-80 related product or service, without spending a huge amount of money on a teeny-tiny ad in one of the "big" magazines!

Q, Last issue you mentioned starting a public domain software library. I sent you a disk full of public domain software, and haven't received it back yet. Why not?

A. Before I explain, let me apologize to those who sent disks, I DO intend to return them, with a number of good public domain programs recorded on them, as soon as I can find the time to do so. Now let me explain the problem, and hope that I don't offend anyone in the process.

What I asked for was "GOOD public domain software programs." What I got was, for the most part, quantity instead of quality. That's not to say that none of the programs were any good - indeed, there were a few excellent programs that I had not seen before. Trouble was, most of you felt compelled to "stuff" the disk (and about half of you sent double-sided disks, and used both sides!).

What I had in mind was for you to pick out one or two (or maybe a very few) really excellent public domain programs, document them (if they weren't already documented in one way or another) so that I'd know what they are and what they do, and send them to me. I did not, and do not, intend to wade through several disks full of programs that are not documented, and that in many cases are programs that originally came from a magazine (in most cases, those are NOT public domain programs anyway), or that have appeared on every BBS in the country (I already have most of those). I wish I had the time to examine each and every program submitted, but I don't. And, the last thing I want to do is start a public domain library that contains low-quality programs.

If you folks that did send disks would care to write and tell me which programs on your disk are the ones that you consider to be really excellent - the ones you find yourself using frequently, for example - it would help me considerably. And, for any of you folks that might want to contribute to the public domain software library, please, please, PLEASE don't feel compelled to fill the disk! Please make sure that the programs you do send are top-quality, at least reasonably well documented (either within the program itself, or preferably within a separate text file -note I'm NOT asking for lengthy documentation if the program is easy enough to use that a small amount of documentation will suffice), and that they are really public domain - especially that they are not reprinted from one of the "big" magazines. And one final note - there are already several CP/M public domain libraries around the country, so we're not starting another one, at least not at the present time. If you want to send CP/M programs for use on the Models I, III, or 4 anyway, go ahead, but please be aware that we may not use them for quite some time.

Once again, my apologies to those who sent disks. Please be patient for a little while longer. Thanks!

Q. Won't these silly "Questions and Answers" ever end, so that we can read something really interesting?

A. Funny you should ask...

FIXING A "FINGERPRINTED" DISK - Have you ever accidently touched the shiny part of one of your diskettes, then found that you got parity errors when you tried to read it? Ever had one of your children pick up the disk while touching the diskette surface? That disk may not be damaged beyond repair!

The best fix is to get your backup disk and make a new copy of the disk in question. You say you hadn't made a backup of that valuable program you were working on when the accident happened? Well, there's still hope.

First of all, backup as much of the damaged disk as you can, just in case you manage to cause more problems than you cure, At least you'll be able to recover the data that wasn't damaged. You may need to use a utility such as TRAKCESS or SUPER-UTILITY to do this, although some DOSes will allow you to skip bad sectors and copy the remaining (good) sectors to another disk. One other point - if your directory track hasn't been damaged (you can still do a DIR of the affected diskette without getting an error message), use the "one file at a time" copy function of your DOS (the CBF function of NEWDOS/80's COPY command, the "wildmask copy" function of DOSPLUS 3.5, the VFU/CMD program of MULTIDOS, etc.) rather than the entire-disk BACKUP command, since this will ignore any bad sectors that are not currently used by any file on the disk.

Now that you've saved as much of the data as you can, hold the disk at an angle to a light source (so that you can see the fingerprints on the disk surface), then insert two fingers through the center hole of the disk and slowly rotate the disk inside its jacket, until the "fingerprinted" area is visible through the cutout in the disk jacket. Remember that TRS-80 compatible drives read and write on the back side of the diskette only (unless you have double-sided drives), so you should NOT be looking at the side of the disk, that has the label (or that would have the label if you weren't buying those cheap unbranded diskettes!).

When you can see the fingerprints on the diskette surface, you can attempt to remove them. For this purpose, we are going to use a miracle cleaning solvent that is available in almost every home. This solvent is gentle enough so that it won't damage the oxide surface of the disk, yet strong enough to remove the fingerprints (we hope). This super solvent is commonly known as water! That's right, plain old tap water (if you happen to have distilled water, that's probably better yet, hut not absolutely necessary). The other thing you will need is a sterile (or at least clean! cotton ball. It would probably help matters if you would wash your hands (with soap and water) before starting this process, just in case you accidently touch the disk surface again!

Put a couple of drops of water (no more than that, please!) on the cotton ball, and GENTLY rub the moist surface over the "fingerprinted" area until the discoloration of the body oil left behind by the fingerprint is no longer visible. You may have to rotate the disk slightly in its jacket to remove all of the body oil. Try not to leave cotton fiber residue on the disk surface (if you notice any, just try to work it back off the disk with the rest of the cotton ball). Once the body oil has been removed, you can use the dry part of the cotton ball to remove any excess water. Then manually rotate the disk in its jacket once or twice (using a couple of fingers through the center hole, please!) so that any remaining cotton fibers will be trapped by the cleaning surface inside the disk jacket.

Now try reading the diskette on your TRS-80. If you are lucky, parity errors will have disappeared. Just to be on the safe side, backup the diskette immediately (once again, try a "file by file" copy if you have problems with a full disk backup). If you are like me and are too cheap to throw the newly repaired disk away, at least re-format it and then copy the files back onto it from your backup diskette - if you have any problems while formatting the diskette or copying the files back, you should probably consider yourself lucky to have recovered your data, and use that disk for a frisbee or something.

PAINT - Here's yet another that's supposed to match Tandy silver-grey - it's General Motors "Silver Sand" code 15L, also sold as DupliColor DS-GM326, I don't guarantee a perfect match, I just print the info as received!

CONVERT YOUR CENTRONICS 737 PRINTER TO A CENTRONICS 739 - I'll bet that headline made a few of you sit up and take notice! Not so long ago, prior to the introduction of the Epson, NEC, and C.Itoh printers, the Centronics 737 was considered THE printer to get if you wanted a quality dot-matrix printer. Even today, in my opinion the proportional font of the Centronics 737 looks as good as, or better than any other font on any dot-matrix printer (with the exception of the new, expensive printers that have 18 or more wires in the print head, as opposed to the usual nine as found in the Centronics 737). Radio Shack sold the 737 under the designation "Line Printer IV".

However, the 737 lacked one thing - the capability to do graphics. Centronics then introduced the 739, which was fully compatible with the 737, but which also had the capability to do graphics. What they never told us was that at least some 737s can be converted to 739s! I found out about it from Joachim Kelterbaum, who resides in Essen, West Germany! I do not know if it can be done with all 737's nor do I know if it can be done with the LP IV, nor will I assume any responsibility if you try this and wreck your printer. All I can tell you is that it worked in my 737. If you try it, however, you're on your own. Understood? O.K.

This modification is known to work with 737s that have the inscription "ASSY. NO. 63680147" located near the power LED (below the power transistors) on the main printed circuit board. If you have a different assembly number, this modification may or may not work.

The first thing you will need is the Read-Only Memory for the 739. This designation for this ROM on the printed circuit board is ME4. How do you get that? Well, you may be able to get Centronics (or an authorized Centronics service center) to sell you one. Or, if you know of someone that has a Centronics 739, you may be able to get a copy of the ROM in their machine. The ROM itself is actually a standard 2716 EPROM (450 ns), so if you have access to an EPROM burner you can make your own. If you don't have an actual 739 ROM to copy, however, you'll have to try to get at least a hex dump of one, and use that to burn your EPROM. On the other hand, I suspect that some Centronics service centers may have extras to sell, though I haven't tried it. If you find a ready source, let me know and I'll let the rest of us know!

Let's assume you've got the EPROM. Here are the instructions for the conversion, more or less as I got them from Joachim:

1) Locate socket ME4 at top left hand side (parts side) of the printed circuit board. If only the mounting holes are there, solder a 24-pin IC socket into place and mount the 739 ROM (2716 EPROM mentioned above) into place (pin 1 goes to the top left).

2) Mount a 20-pin IC socket into place adjacent to the ME4 ROM, at the location designated ME5 on the printed circuit board. Plug a 74LS273 (octal D flip-flop) IC into this socket (pin 1 goes to the top left).

3) Mount R54, a 1K Ohm 1/4 Watt resistor, into place just above ME5 on the printed circuit board.

4) Unsolder capacitor or jumper (?) between C14 and position of R14 (located on left center portion of board, to the left of the ME8 and just above ME3, in a row of several resistors and capacitors). Insert this part into place of R49 (this is used to disable internal ROM for processor 8049 and enable EPROM).

5) Follow the selftest procedure at end of the 739 manual (page 5-4, also described below).

6) You may also hardware select a form length of 11 inches or 12 inches. 11 inches is the default, for 12 inches cut out R21 (located to the right of ME8).

As I mentioned, the above instructions were what I got from Joachim. However, my experience was a bit different. Before I undertook the conversion, I decided to look inside my Centronics 737, to determine what was involved in such a project. I was quite surprised to discover that all of the parts mentioned were already installed (although a 74L9273PC IC was used at ME5, instead of the 74LS273 specified above. The 74L9273PC was soldered directly into place - no socket was used). One possible reason for this is that at one time I was having problems with frequent breakdowns of my printer, so the Centronics repair center had replaced the main printed circuit board as part of their repair. Perhaps they used a 739 circuit board - I'm not sure. I thought that maybe I already had a 739 and didn't know it, so I tried a few 739-specific commands. They didn't work.

Next, I tried removing the 2716 EPROM that was already installed at ME4, and installing the replacement EPROM. I wasn't sure about the instructions for C14 and R49 - in my printer, both C14 and R49 appeared to be already installed, and between them there appears to be a space where some component should go, but nothing is installed there. R49 is a a 10K resistor (brown, black, orange, gold color bands). I decided to leave R49 alone for the time being.

After installing the EPROM, I tried running the self-test described at the end of the 739 manual. The instructions for the self-test are as follows:

"This printer contains a self-test function that checks all electromechanical functions and virtually all electronic circuits. To perform the self-test, refer to Figure 2-7 for the location of Switchpack S5 and perform the following:

1. Front panel switch to LOCAL and Power switch to OFF.

2. Remove top cover and carefully set all S5 switches to ON, using an eraser at the end of a pencil.

3. Install roll paper or fanfold paper (a single sheet will be ejected by the TOF).

4. Replace cover, plug power in and turn power ON.

5. Place front panel switch in ONLINE position. The printer will perform a TOF (Top of Form) followed by a printing of "rolling ASCII" characters. The first 96 characters will be underlined...

"Self test is terminated by placing the front panel switch in the LOCAL position, turning power OFF and setting Switchpack S5 to enable the desired character set (see Figure 2-7)."

There was only one minor problem - I couldn't find "Switchpack S5"! Even "Figure 2-7" didn't help (it showed the location in a way that could have put "Switchpack S5" just about anywhere on the left side of the printed circuit board). At this point I began to wonder if I had a different main printed circuit board than the ones used in West Germany, but since the assembly number was the same, I decided that wasn't the problem. I decided to try the "smoke test" - that is, I just connected everything up and applied power to see if it would work.

To my surprise, I was able to print the 739 graphics characters. To my disappointment, the printer would no longer do an automatic line feed after a carriage return, which meant that it kept overprinting the same line. Back to the instructions, which advised me that switch #4 of "Switchpack S5" had to be turned on to enable auto line feed. Switchpack S5 again! I still couldn't find it. After looking at the schematic diagram, the printed circuit manual, and the main printed circuit board for a while, I figured out where Switchpack S5 was supposed to go (it's supposed to be located between ME8 and the sensing switch that is activated when the printhead is at the "home" position, just above the legend "SWP1" (!) on the printed circuit board). It wasn't there.

Switchpack S5 controls both the country character set selection and auto line feed, as follows:

	1	2	3	4
U.S.A.	OFF	OFF	OFF	-
France	ON	OFF	OFF	-
U.K.	OFF	ON	OFF	-
Germany	ON	ON	OFF	-
Italy	OFF	OFF	ON	-
Sweden/Finland	ON	OFF	ON	-
Self Test	ON	ON	ON	-
Auto Line Feed	-	-	-	ON

Since I of course wanted to use the USA character set, it appeared that only switch #4 needed to be on, and since I had not had to fully disassemble my printer so far (and did not have a switchpack handy), I decided to just solder a short piece of wire across the switch #4 position (the position nearest the bottom of the board). That worked, and apparently I now have the equivalent of a 739 printer! Of course, you may wish to solder a four-position DIP switch into place. If you do, be sure that switch #1 is nearest the top of the printed circuit board.

Once you've made the conversion, the graphics function is entered by the sequence ESC%0 (27, 37, 48 decimal) and is terminated by any proper escape sequence, such as any of the sequences used to select a font (standard, condensed, or proportional). Graphics are formed by a six-pin vertical pattern, using one-half line spacing so that a solid graphic pattern is possible due to the dot overlap. After 594 bytes of graphic data (8 inches, 203 mm) are received by the printer, a one-half line feed and a carriage return (if Auto Line Feed is enabled) are automatically generated. Printing speed for graphics is five inches per second, and dot density is 75 dots per inch horizontal, and 72 dots per inch vertical.

While in the graphics mode, ASCII characters in the range 32 decimal through 95 decimal are valid. 32 decimal fires none of the six pins, while 95 decimal fires all of them. In between, the six-pin vertical pattern is bit-mapped, so that the top pin is fired if bit 0 is set, the next pin down if bit 1 is set, and so on until we get to the bottom pin, which is fired if bit 5 is set (I am using Z-80 bit notation and numbering the bits zero through five, Centronics numbers them one through six in their literature). Then you must add the offset of 32 decimal (20H) to get the correct code to send to the printer. This may sound a bit strange, but the effect is that almost all of the graphics character codes can be generated by printing characters that are readily available from the keyboard (numeric and punctuation characters, plus the uppercase alphabet). The only characters in this range that might be difficult to obtain from the keyboard are in the range 91 decimal through 95 decimal (these are printed as arrow characters and the underline character on the Model I video display, and as the brackets, backslash, tilde, and underline characters on the Model III video display).

The form feed function is also enabled by the conversion. Receipt of a Form Feed code (ASCII 12, which CANNOT be sent directly to the printer when the standard TRS-80 ROM printer driver is used) causes the printer to advance the paper one form length (66 lines or 72 lines, depending on whether resistor R21 has been removed).

The only problem that I have noticed since the conversion is that the printer sometimes has difficulty when a "CHR$(14)" "underline off" character is the last character of a BASIC "PRINT" statement. For some reason, the NEXT line to be printed is sometimes printed starting at the horizontal position on the line where the previous line ended (rather than at the start of the line). I never noticed this effect with the 737 EPROM, although I simply may have never tried that particular type of BASIC statement before (this happened in a program I wrote AFTER I installed the 739 EPROM). This may have even been an effect of the Disk Operating System, since I was using a DOS that I don't usually use. In any event, printing an additional space character after the CHR$(14) seems to cure the problem. I have NOT noticed any problem in running my word-processor software, which makes me further suspect that either the problem was there all along and I didn't notice it, or it was a problem with the DOS.

If you are successful in making this conversion, you'll be able to draw pictures with your 737 (now 739), just like those that have the newer and fancier printers. It would also be possible for you to design your own "fancy" character sets. If you've written any software for use with the 739 that you'd like to share with the rest of us, send it in to NORTHERN BYTES, and I'll see if I can duplicate your results!

CORPORATE DUM-DUM AWARD DEPT. - I don't usually go around taking jabs at Tandy Corporation. I like my TRS-80, and all in all, I think Tandy has a good product line. Unfortunately, Tandy has this nasty habit of occasionally going out and shooting themselves in the foot. I think maybe they've done it again.

O.K., here's the deal. I am looking at a Radio Shack computer catalogue - specifically, "1984 RSC-10 TRS-80 CATALOGUE" (as inscripted on the lower right-hand corner of the front cover). Now, I open it up to page two, and see the ad for the new Model 4P "transportable" computer. Among the features listed are, and I quote: '9" High-Resolution 80 x 24 Green Display'. Yes, that's right, I did say GREEN. And, in fact, the photo in the ad shows a nice, green display.

Wait a minute, you say - You've got the same catalog, and it doesn't say anything about a green display, nor does the photo show a green CRT! Well, Sherlock, you may have noticed that I said I had the RSC-10 "catalogue" - which indicates that it's not from the U.S.A., since we got lazy and stopped using those last two letters a long time ago. Actually, the catalogue I have is Copyright 1983 Radio Shack, Barrie, Ontario L4M 4W5.

It is a real education in promotion to compare the U.S. and Canadian versions of this catalog(ue). Many pages are almost exactly the same in the two pieces of literature. What's really interesting is that many of the photographs used are EXACTLY the same, EXCEPT that the characters on the CRT display are green in the Canadian catalogue, and white in the U.S.A. catalog. I counted 19 photos (give or take a couple) that had been "retouched" in this way (and the characters shown in green were almost always more readable, although that could have been attributed to normal differences in print quality, I suppose). Anyway, my point is that Tandy seems to take great pains to conceal the fact that Canadian units have green screens, and U.S. units don't. This is true with both the new Model 4P, and the regular Model 4.

When this happened with the Model 4, I theorized that maybe Tandy had a bunch of old Model III black-and-white CRTs that they wanted to use up before they introduced the green screen on the U.S. models. But, they've sold a lot of Model 4s, and still no green screen. More to the point, they've NEVER used a nine-inch CRT in any of their previous computers, so why stick us with a black-and-white CRT now? I don't know of any computer user that wouldn't prefer green to white on their display. The reason I say that maybe Tandy shot themselves in the foot is that the display is a very visible part of the computer, and the fact that it's black-and-white on the U.S. model could well influence a potential purchaser to buy something else (I sure as heck wouldn't want to stare at a black-and-white screen if I didn't have to!).

There is one consolation for U.S. buyers. You can buy a Model 4P here in the U.S.A. for $1799, remove and replace the CRT (should run around $100, once one of the aftermarket CRT suppliers gets into the act), and you'll have your Model 4P with green screen for around $1900. In Canada, you'll pay $2699 Canadian for the same unit (that's about $2200 U.S. at current exchange rates). A similar price differential exists with the regular Model 4 - the basic 16K unit lists for $999 in the U.S. catalog, and $1399 in the Canadian catalogue (that's around $1150 in U.S. dollars). But, the point is that it would have probably only cost Tandy $10 or $20 more to put a green CRT in the U.S. model - an extra amount I'm sure most buyers would be more than willing to pay. Besides, they'd save money by not having to have all those catalog(ue) photos retouched!

It's probably a good thing for somebody that I'm not a major stockholder in Tandy Corporation, because if I were, I think I'd try to find out whose bright idea it was to put the black-and-white CRT in the Models 4 and 4P (especially the 4P!), and make a motion that that person be invited to seek employment elsewhere - like at the South Pole during the winter (when the sun rarely shines), so he can see how dull black and white can get to be after a while!

NEWDOS/80 HINT - Have you ever tried to read the back side of a disk that was formatted in someone's "flippy" drive, on your plain-vanilla single-sided drive? Yes, it can be done! All you have to do is physically turn the disk over, so that the back side is in contact with your drive's read/write head. Well, that's ALMOST all you have to do. The problem is that some DOSes, before they actually try to read a disk, will first try to verify that a disk is really in the drive. "Gee, Uncle Jack, how do they do that?", I hear you ask. Well, they check for the presence of the index hole spinning past the disk. Now, in case you didn't know it, the drive really only needs to "see" the disk index hole during formatting - the rest of the time, it uses the formatted pattern on the disk, and ignores the index hole - unless, of course, the DOS decides to make sure it's there, to see if there really is a disk in the drive.

As you've probably guessed, when you turn a "flippy" disk upside down so that you can read the backside in a regular drive, the index hole in the disk jacket isn't in the correct position anymore. You COULD get out your trusty paper punch and punch a second index hole, but it's not your disk, right? There IS an easier way, if you're using NEWDOS/80. Boot the DOS, then use DEBUG, a machine language monitor, or a BASIC POKE statement to change the byte at 47E1H in the Model I memory, or 4787H in the Model III memory, from B7H to AFH. This will cause the DOS to ignore the fact that the index hole isn't there (in fact, it won't even be able to tell that a drive door has been left open - which is why I suggest you make this change temporarily in memory, and not permanent in the NEWDOS/80 SYS0/SYS file itself). This will at least allow you to copy the files you need from the "flippy" disk to one of your own disks. NOTE: This works only for disks made with "flippy" drives, NOT for disks created on a true double-sided drive. Also, I do not recommend leaving this change in place any longer than necessary, because there may be side effects I haven't discovered. Copy the disk to one of your normal disks, then reboot the system to restore normal DOS operation, and you shouldn't have any problems.

TRSDOS 6.0 HINT (UNVERIFIED) - We're told that "LSIDOS" is the password used with TRSDOS 6.0, but that it doesn't allow the copying of /SYS files under certain conditions. Has anyone else had this problem, or found a solution for it?

SETDATE by Jack Decker - This program is copyrighted, but may be reprinted for non-commercial purposes or placed in your computer club's software library. Just don't sell it or include in in a program package you're selling, and I won't object. As far as I am aware, it will work properly under any Model I or III Disk Operating System.

Computer users are basically lazy - that's why we use computers! This program eliminates that most tedious of all tasks on the computer - setting the date! Yes, you'd be amazed the lengths that computer users will go through to avoid having to set the date at power-up (including patching the DOS so it won't ask the dreaded question: DATE?)

Now you can have your date without typing it in, and without buying one of those battery-backup clock modules (which, by the way, is a hardware project I'd like to publish in these pages, if anyone would care to write it up and send it in!). This program always "remembers" the date when you last powered-up your system. If you power up a second time on the same day, you need only hit <ENTER> to tell SETDATE that the date hasn't changed. If you've been away from your computer for a day or two (who among us has the will power to stay away longer?), simply hit the up-arrow or right-arrow keys to advance the date. This is made easy by the fact that the day of the week is displayed - most people can remember that today is Tuesday, but it's a real mental effort to remember whether today is the seventh or the eighth of the month. By printing the day of the week along with the date, the chances of mental overload are reduced significantly.

Other keys do other things, but they are explained on the screen, so you don't have to remember them. Further documentation on the program is found in the remark statements of the source code, so I won't explain further here, except to say that the program is self-modifying, and rewrites the first sector of itself onto the disk before exiting (this is how the program "knows" what the date was the last time it was executed). Note that the program violates what is thought to be a prime rule of programming - that is, "Always close any open files before you exit the program." Normally that's a good rule to follow, but doing so in this program would truncate the file and destroy the program, so don't try it here!

Normally you'd use the DOS "AUTO" command to run SETDATE each time the disk is re-booted (see the comments in the source code for more information, and your DOS manual for particulars on the AUTO command of your DOS), but the problem remains: How do you get the DOS to NOT ask the date before it gets around to running SETDATE? Here's the procedure for most of the popular TRS-80 Disk Operating Systems:

DOSPLUS 3.5 - Use the DOS "SYSTEM" command to disable the date prompt (you may wish to disable the time prompt as well). A command of the form "SYSTEM (DATE=N)" would suffice, to also disable the time prompt, use "SYSTEM (DATE=N,TIME=N)".

LDOS 5.x - Use the DOS "SYSTEM" command to disable the date prompt (you may wish to disable the time prompt as well). A command of the form "SYSTEM (DATE=OFF)" would suffice, to also disable the time prompt, use "SYSTEM (DATE=OFF,TIME=OFF)". (Note: I do not own a copy of LDOS, don't much care for LDOS anyway, and the only documentation that I have for LDOS is their "Quick Reference Card". So, if this doesn't work, check your LDOS manual).

MULTIDOS - Use the exclamation mark specifier prior to the SETDATE filename of the AUTO command (for example, "AUTO !SETDATE"). I suggest that the pound sign also be used, to inhibit displaying the AUTO command ("AUTO !#SETDATE").

NEWDOS/80 - Use the DOS "SYSTEM" command to set SYSTEM options AY and AZ to "N" (a typical command line would be "SYSTEM 0 AY=N AZ=N").

TRSDOS 1.3 - Sorry, nothing as simple as a SYSTEM command setting here. According to John Hallgren, who is probably as much of an expert on TRSDOS (in its various forms) as anyone, the patch to skip the date/time prompt on power-up is as follows: PATCH *0 'ADD-ABE',FIND=06,CH= 17).

A couple of closing comments: First, this program will only be accurate during the 20th and 21st centuries. Please don't write for patches in February, 2100, since I may not be able to accomodate you. Second, I should mention that Dave McGlumphy more or less inspired this program - he wrote a program called ETDT, which had similar intent (to set the date without having to type it in), but worked differently. I decided to try and write something that was a little easier to use, and after about three days worth of hacking around, I came up with this program. And Charley wonders how come I can never get anything accomplished.

If anyone ever forms a chapter of "Hackers Anonymous" up here, I'm going to have to sign up. Just the other night I got the D.T.s - couldn't find my Editor-Assembler program. I saw pink TRS-80's dancing around my living room. I knew I was in real trouble when I spotted a green Apple, though... well, enough of this silliness, here's the program:

```
                00100 ;xxxxx  SETDATE/ASM  -  Copyright 1983 by Jack Decker
                00110
                00120 ;This program eliminates the need to type in the date
                00130 ;manually each time you power-up your computer (assuming
                00140 ;your DOS can be forced to bypass the "DATE" prompt when
                00150 ;it is booted up).
                00160
                00170 ;This program MUST be assembled using the filename
                00180 ;"SETDATE/CMD".  Normally, it is used by placing the
                00190 ;filename as part of a DOS "AUTO" command.  You may AUTO
                00200 ;a second program along with SETDATE by placing filename
                00210 ;of the second program in same AUTO command line,
                00220 ;immediately following the SETDATE command.  For example:
                00230
                00240 ;               SETDATE PROGRAM2
                00250
                00260 ;would first execute SETDATE, then PROGRAM2.
                00270
                00280 ;If there appears to be a valid date already stored in
                00290 ;memory (as might happen if the DOS is re-booted without
                00300 ;first turning off the power to the system), SETDATE will
                00310 ;simply clear the screen and exit.  However, you can
                00320 ;force SETDATE to execute by placing a space and
                00330 ;exclamation point immediately following the filename:
                00340
                00350 ;               SETDATE !
                00360 ;       or      SETDATE ! PROGRAM2
                00370
                00380 ;When SETDATE executes, it will display the date used the
                00390 ;last time SETDATE was executed.  You may advance or
                00400 ;backspace the date using the arrow keys.  When the
                00410 ;correct date is displayed, simply press the ENTER key,
                00420 ;which will store the date in memory and also within the
                00430 ;SETDATE program itself, for use the next time SETDATE
                00440 ;is executed.
                00450
                00460 ;Questions or comments MUST be accompanied by a self-
                00470 ;addressed stamped envelope if you live in the U.S.A. or
                00480 ;Canada (Canadian postage is O.K.) and wish a reply.
                00490
                00500 ;               Jack Decker
                00510 ;               1804 West 18th Street   Lot # 155
                00520 ;               Sault Ste. Marie, Michigan  49783
                00530
                00540
6000            00550         ORG     6000H           ;MUST end with 00H
                00560
                00570 ;String and date storage area used by program
                00580
6000 2D60       00590 TABLE   DEFW    SUN             ;Table of string pointers
6002 3360       00600         DEFW    MON             ;  point to strings
6004 3960       00610         DEFW    TUE             ;  containing days of
6006 4060       00620         DEFW    WED             ;  the week
6008 4960       00630         DEFW    THU
600A 5160       00640         DEFW    FRI
600C 5760       00650         DEFW    SAT
600E 5F60       00660         DEFW    JAN             ;Table of string pointers
6010 6660       00670         DEFW    FEB             ;  point to strings
6012 6E60       00680         DEFW    MAR             ;  containing months of
6014 7360       00690         DEFW    APR             ;  the year
6016 7860       00700         DEFW    MAY
6018 7B60       00710         DEFW    JUN
601A 7F60       00720         DEFW    JUL
601C 8360       00730         DEFW    AUG
601E 8960       00740         DEFW    SEP
6020 9260       00750         DEFW    OCT
6022 9960       00760         DEFW    NOV
6024 A160       00770         DEFW    DEC
6026 FE         00780 MARKER  DEFB    0FEH            ;Start date storage area
6027 53         00790 DATSTR  DEFB    83D             ;Year storage
6028 01         00800         DEFB    1D              ;Day storage
6029 0B         00810         DEFB    11D             ;Month storage
```

```
602A 02       00820          DEFB    2D              ;Day of week storage
602B 6C07     00830 CNTURY   DEFW    1900D           ;Century storage
602D 53       00840 SUN      DEFM    'Sunda'         ;Strings containing days
     75 6E 64 61
6032 F9       00850          DEFB    'y'+80H         ;  of week
6033 4D       00860 MON      DEFM    'Monda'
     6F 6E 64 61
6038 F9       00870          DEFB    'y'+80H
6039 54       00880 TUE      DEFM    'Tuesda'
     75 65 73 64 61
603F F9       00890          DEFB    'y'+80H
6040 57       00900 WED      DEFM    'Wednesda'
     65 64 6E 65 73 64 61
6048 F9       00910          DEFB    'y'+80H
6049 54       00920 THU      DEFM    'Thursda'
     68 75 72 73 64 61
6050 F9       00930          DEFB    'y'+80H
6051 46       00940 FRI      DEFM    'Frida'
     72 69 64 61
6056 F9       00950          DEFB    'y'+80H
6057 53       00960 SAT      DEFM    'Saturda'
     61 74 75 72 64 61
605E F9       00970          DEFB    'y'+80H
605F 4A       00980 JAN      DEFM    'Januar'        ;Strings containing
     61 6E 75 61 72
6065 F9       00990          DEFB    'y'+80H         ;  months of year
6066 46       01000 FEB      DEFM    'Februar'
     65 62 72 75 61 72
606D F9       01010          DEFB    'y'+80H
606E 4D       01020 MAR      DEFM    'Marc'
     61 72 63
6072 E8       01030          DEFB    'h'+80H
6073 41       01040 APR      DEFM    'Apri'
     70 72 69
6077 EC       01050          DEFB    'l'+80H
6078 4D       01060 MAY      DEFM    'Ma'
     61
607A F9       01070          DEFB    'y'+80H
607B 4A       01080 JUN      DEFM    'Jun'
     75 6E
607E E5       01090          DEFB    'e'+80H
607F 4A       01100 JUL      DEFM    'Jul'
     75 6C
6082 F9       01110          DEFB    'y'+80H
6083 41       01120 AUG      DEFM    'Augus'
     75 67 75 73
6088 F4       01130          DEFB    't'+80H
6089 53       01140 SEP      DEFM    'Septembe'
     65 70 74 65 6D 62 65
6091 F2       01150          DEFB    'r'+80H
6092 4F       01160 OCT      DEFM    'Octobe'
     63 74 6F 62 65
6098 F2       01170          DEFB    'r'+80H
6099 4E       01180 NOV      DEFM    'Novembe'
     6F 76 65 6D 62 65
60A0 F2       01190          DEFB    'r'+80H
60A1 44       01200 DEC      DEFM    'Decembe'
     65 63 65 6D 62 65
60A8 F2       01210          DEFB    'r'+80H
60A9 1C       01220 MSG      DEFB    1CH             ;1CH will home cursor
60AA 1F       01230          DEFB    1FH             ;1FH will clear screen
60AB 1A       01240          DEFB    1AH             ;1AH moves down one line
60AC 1A       01250          DEFB    1AH
60AD 50       01260          DEFM    'Please set correct date:'
     6C 65 61 73 65 20 73 65 74 20 63 6F 72 72 65 63
     74 20 64 61 74 65 3A
60C5 0D       01270          DEFB    0DH
60C6 1A       01280          DEFB    1AH
60C7 1A       01290          DEFB    1AH
60C8 1A       01300          DEFB    1AH
60C9 50       01310          DEFM    'Please press one of the following keys:'
     6C 65 61 73 65 20 70 72 65 73 73 20 6F 6E 65 20
     6F 66 20 74 68 65 20 66 6F 6C 6C 6F 77 69 6E 67
     20 6B 65 79 73 3A
60F0 0D       01320          DEFB    0DH
60F1 3C       01330          DEFM    '<ENTER> if date is correct'
     45 4E 54 45 52 3E 20 69 66 20 64 61 74 65 20 69
     73 20 63 6F 72 72 65 63 74
610B 0D       01340          DEFB    0DH
610C 3C       01350          DEFM    '<BREAK> or <CLEAR> to exit without setting date'
     42 52 45 41 4B 3E 20 6F 72 20 3C 43 4C 45 41 52
     3E 20 74 6F 20 65 78 69 74 20 77 69 74 68 6F 75
     74 20 73 65 74 74 69 6E 67 20 64 61 74 65
613B 0D       01360          DEFB    0DH
613C 3C       01370          DEFM    '<up-arrow> or <right-arrow> to advance date'
     75 70 2D 61 72 72 6F 77 3E 20 6F 72 20 3C 72 69
     67 68 74 2D 61 72 72 6F 77 3E 20 74 6F 20 61 64
     76 61 6E 63 65 20 64 61 74 65
6167 0D       01380          DEFB    0DH
6168 3C       01390          DEFM    '<down-arrow> or <left-arrow> to backup date'
     64 6F 77 6E 2D 61 72 72 6F 77 3E 20 6F 72 20 3C
     6C 65 66 74 2D 61 72 72 6F 77 3E 20 74 6F 20 62
     61 63 6B 75 70 20 64 61 74 65
6193 0D       01400          DEFB    0DH
6194 3C       01410          DEFM    '<SHIFT> plus arrow key to change date at high speed'
     53 48 49 46 54 3E 20 70 6C 75 73 20 61 72 72 6F
     77 20 6B 65 79 20 74 6F 20 63 68 61 6E 67 65 20
     64 61 74 65 20 61 74 20 68 69 67 68 20 73 70 65
     65 64
61C7 1C       01420          DEFB    1CH             ;1CH will home cursor
61C8 1A       01430          DEFB    1AH             ;1AH moves down one line
61C9 1A       01440          DEFB    1AH
61CA 1A       01450          DEFB    1AH
61CB 9A       01460          DEFB    1AH+80H
61CC 1E       01470 ENDYR    DEFB    1EH             ;1EH clears to line end
61CD 9D       01480          DEFB    1DH+80H         ;1DH moves to line start
              01490
              01500 ;Start of actual machine-language program
              01510
61CE 7E       01520 START    LD      A,(HL)          ;Get argument (!) if any
61CF FE21     01530          CP      '!'             ;Is it exclamation point
61D1 F5       01540          PUSH    AF              ;Save result (Z flag)
61D2 CC781D   01550          CALL    Z,1D78H         ;Bump HL past excl. point
61D5 F1       01560          POP     AF              ;Restore Z flag
61D6 E5       01570          PUSH    HL              ;Save input buffer pointr
61D7 2817     01580          JR      Z,USEPGM        ;Skip date test if ! used
61D9 CD1A63   01590          CALL    GETBFR          ;Get loc memory date stor
61DC 1A       01600          LD      A,(DE)          ;Get year from memory
61DD FE64     01610          CP      100D            ;Is it in range 00 - 99?
61DF 300F     01620          JR      NC,USEPGM       ;Go if invalid year
61E1 13       01630          INC     DE              ;Point to day in memory
61E2 1A       01640          LD      A,(DE)          ;Get day from memory
61E3 3D       01650          DEC     A               ;Adjust valid to 0 - 30
61E4 FE1F     01660          CP      31D             ;Is day 1 - 31 in memory?
61E6 3008     01670          JR      NC,USEPGM       ;Go if invalid month
61E8 13       01680          INC     DE              ;Point to month in memory
61E9 1A       01690          LD      A,(DE)          ;Get month from memory
61EA 3D       01700          DEC     A               ;Adjust valid to 0 - 11
61EB FE0C     01710          CP      12D             ;Is month 1 - 12 in mem?
61ED DA7662   01720          JP      C,EXIT2         ;Go if memory date valid
61F0 21A960   01730 USEPGM   LD      HL,MSG          ;Point to message
61F3 CDF862   01740          CALL    DSPMSG          ;Display message
61F6 012A60   01750 RESTRT   LD      BC,DATSTR+3     ;Point to day of wk byte
61F9 0A       01760          LD      A,(BC)          ;Get day of week (0 - 6)
61FA CDF062   01770          CALL    PRTSTR          ;Print day of week string
61FD CD0F63   01780          CALL    PRTCOM          ;Print comma and space
6200 0B       01790          DEC     BC              ;Point to month byte
6201 0A       01800          LD      A,(BC)          ;Get month (1 - 12)
6202 C606     01810          ADD     A,6             ;Offset for string table
6204 CDF062   01820          CALL    PRTSTR          ;Print month string
6207 CD1463   01830          CALL    PRTSPC          ;Print space character
620A 0B       01840          DEC     BC              ;Point to day byte
620B 0A       01850          LD      A,(BC)          ;Get day (1 - 31)
620C 6F       01860          LD      L,A             ;Put day in L
620D 2600     01870          LD      H,0             ;HL = day
620F C5       01880          PUSH    BC              ;Save date storage pntr
6210 CD0663   01890          CALL    PRTNUM          ;Print day
6213 C1       01900          POP     BC              ;Restore date storage ptr
6214 CD0F63   01910          CALL    PRTCOM          ;Print comma and space
6217 0B       01920          DEC     BC              ;Point to year byte
6218 0A       01930          LD      A,(BC)          ;Get year (0 - 99)
6219 4F       01940          LD      C,A             ;Put year in C
621A 0600     01950          LD      B,0             ;BC = last 2 digits year
621C 2A2B60   01960          LD      HL,(CNTURY)     ;Get century offset
621F 09       01970          ADD     HL,BC           ;HL = Year (all 4 digits)
6220 CD0663   01980          CALL    PRTNUM          ;Print year
6223 21CC61   01990          LD      HL,ENDYR        ;Point to ctrl chr string
6226 CDF862   02000          CALL    DSPMSG          ;Output it to video
6229 3A4038   02010 GETKEY   LD      A,(3840H)       ;Get BREAK/CLEAR row
```

```
622C E606    02020          AND    6                ;Mask out other keys
622E 2043    02030          JR     NZ,EXIT          ;Exit if BREAK/CLEAR
6230 CD2B00  02040          CALL   2BH              ;Get keystroke if any
6233 28F4    02050          JR     Z,GETKEY         ;If no key was pressed
6235 FE0D    02060          CP     0DH              ;Was it the ENTER key?
 237 2047    02070          JR     NZ,NOTCR         ;Go if not ENTER
6239 212760  02080          LD     HL,DATSTR        ;Point to program date
623C E5      02090          PUSH   HL               ;Save program date pntr
623D CD1A63  02100          CALL   GETBFR           ;Find memory date storage
6240 010300  02110          LD     BC,3             ;Number of bytes to move
6243 EDB0    02120          LDIR                    ;Move from program to mem
6245 0600    02130          LD     B,0              ;Logical Record Lngth=256
6247 114263  02140          LD     DE,FCB           ;File Control Block pntr
624A 216263  02150          LD     HL,FILBUF        ;File I/O Buffer pntr
624D CD2444  02160          CALL   4424H            ;DOS OPEN routine
6250 C28944  02170 ERREXT   JP     NZ,4489H         ;If error use ERROR rtne
6253 CD3644  02180          CALL   4436H            ;READ sector 0 to buffer
6256 20F8    02190          JR     NZ,ERREXT        ;Go if error
6258 E1      02200          POP    HL               ;Get program date pointer
6259 D5      02210          PUSH   DE               ;Save FCB pointer
625A 118C63  02220          LD     DE,MARKER-TABLE+4+FILBUF
             02230 ;Above instruction points DE to first location in sector
             02240 ;zero of disk file that can possibly contain MARKER byte
625D 1A      02250 FNDDAT   LD     A,(DE)           ;Get byte frm disk sector
625E 13      02260          INC    DE               ;Bump pointer to next loc
625F FEFE    02270          CP     0FEH             ;MARKER byte found?
6261 20FA    02280          JR     NZ,FNDDAT        ;Try again if not
6263 010400  02290          LD     BC,4             ;Move 4 bytes frm program
6266 EDB0    02300          LDIR                    ;  storage to disk sector
6268 D1      02310          POP    DE               ;Restore FCB pointer
6269 CD3F44  02320          CALL   443FH            ;Reset to sector zero
626C 20E2    02330          JR     NZ,ERREXT        ;Go if error
626E CD3944  02340          CALL   4439H            ;Write sector back to dsk
6271 20DD    02350          JR     NZ,ERREXT        ;Go if error
             02360 ;Do NOT attempt to CLOSE file after above procedure!!
             02370 ;It is not necessary, and will truncate file if you do!
6273 CDC901  02380 EXIT     CALL   1C9H             ;Clear screen
6276 E1      02390 EXIT2    POP    HL               ;Restore input buffer ptr
6277 3E0D    02400          LD     A,0DH            ;Check for <CR> character
 279 BE      02410          CP     (HL)             ;  (end of command strng)
627A CA2D40  02420          JP     Z,402DH          ;DOS READY if no more cmd
627D C30544  02430          JP     4405H            ;Execute next DOS command
6280 FE5B    02440 NOTCR    CP     5BH              ;Was keystroke up-arrow?
6282 2812    02450          JR     Z,ARROW          ;Go if up-arrow
6284 47      02460          LD     B,A              ;Save keystroke in B
6285 F610    02470          OR     10H              ;Make shifted=unshifted
6287 FE18    02480          CP     18H              ;Invalid keystroke if
6289 389E    02490          JR     C,GETKEY         ;  less than ASCII 18H
628B FE1C    02500          CP     1CH              ;Keystroke valid if less
628D 309A    02510          JR     NC,GETKEY        ;  than ASCII 1CH
628F A8      02520          XOR    B                ;Check for shifted char
6290 2003    02530          JR     NZ,UNSHFT        ;If not shifted char
6292 323C40  02540          LD     (403CH),A        ;Zero arrow row storage
6295 78      02550 UNSHFT   LD     A,B              ;Restore original char
6296 21F661  02560 ARROW    LD     HL,RESTRT        ;RET addr for following
6299 E5      02570          PUSH   HL               ;Save it on stack
629A 212A60  02580          LD     HL,DATSTR+3      ;Point to day-of-week
629D 0F      02590          RRCA                    ;Shift bit 0 into carry
629E 3030    02600          JR     NC,BACK          ;If key was back/down arr
62A0 34      02610 ADV      INC    (HL)             ;Increment day of week
62A1 7E      02620          LD     A,(HL)           ;Get day of week
62A2 FE07    02630          CP     7                ;Is it greater than 6?
62A4 3802    02640          JR     C,SETDAY         ;Go if valid day
62A6 3600    02650          LD     (HL),0           ;Else reset to Sunday
62A8 2B      02660 SETDAY   DEC    HL               ;Point to month storage
62A9 7E      02670          LD     A,(HL)           ;Get month
62AA CD2663  02680          CALL   MAXDAY           ;Get number days in month
62AD 2B      02690          DEC    HL               ;Point to day storage
62AE 7E      02700          LD     A,(HL)           ;Get current day
62AF B9      02710          CP     C                ;Compare with maximum
62B0 34      02720          INC    (HL)             ;Advance day of month
62B1 D8      02730          RET    C                ;Finished if valid day
62B2 3601    02740          LD     (HL),1           ;Else first of new month
62B4 23      02750          INC    HL               ;Point to month storage
62B5 7E      02760          LD     A,(HL)           ;Get current month
62B6 FE0C    02770          CP     12D              ;See if it's December
62B8 34      02780          INC    (HL)             ;Advance month count
62B9 D8      02790          RET    C                ;Finished if valid month
62BA 3601    02800          LD     (HL),1           ;Else January of new year
62BC 2B      02810          DEC    HL               ;Bump pointer back down
62BD 2B      02820          DEC    HL               ;  to year storage
62BE 7E      02830          LD     A,(HL)           ;Get current year
62BF FE63    02840          CP     99D              ;See if last of century
62C1 34      02850          INC    (HL)             ;Advance year count
62C2 D8      02860          RET    C                ;Finished if valid year
62C3 3600    02870          LD     (HL),0           ;Else reset year count
62C5 116400  02880          LD     DE,100D          ;Add 100 years to century
62C8 2A2B60  02890 CHGCEN   LD     HL,(CNTURY)      ;Get current century
62CB 19      02900          ADD    HL,DE            ;Adjust century offset
62CC 222B60  02910          LD     (CNTURY),HL      ;  and re-save it
62CF C9      02920          RET                     ;Finished for sure
62D0 35      02930 BACK     DEC    (HL)             ;Decrement day of week
62D1 F2D662  02940          JP     P,SETDA2         ;Go if valid day
62D4 3606    02950          LD     (HL),6           ;Else reset to Saturday
62D6 2B      02960 SETDA2   DEC    HL               ;Point to month storage
62D7 7E      02970          LD     A,(HL)           ;Get month
62D8 2B      02980          DEC    HL               ;Point to day storage
62D9 35      02990          DEC    (HL)             ;Decrement day of month
62DA C0      03000          RET    NZ               ;Finished if valid day
62DB 3D      03010          DEC    A                ;Else A=# of previous mth
62DC CD2663  03020          CALL   MAXDAY           ;Get # days previous mnth
62DF 71      03030          LD     (HL),C           ;Day=last day prev. month
62E0 23      03040          INC    HL               ;Point to month storage
62E1 35      03050          DEC    (HL)             ;Decrement month count
62E2 C0      03060          RET    NZ               ;Finished if valid month
62E3 360C    03070          LD     (HL),12D         ;Else December prev. year
62E5 2B      03080          DEC    HL               ;Bump pointer back down
62E6 2B      03090          DEC    HL               ;  to year storage
62E7 35      03100          DEC    (HL)             ;Decrement year count
62E8 F0      03110          RET    P                ;Finished if valid year
62E9 3663    03120          LD     (HL),99D         ;Else last yr prev cntury
62EB 119CFF  03130          LD     DE,-100D         ;Subtract 100 frm century
62EE 1808    03140          JR     CHGCEN           ;Adjust century & finish
62F0 87      03150 PRTSTR   RLCA                    ;A=AX2 (2 byte pointers)
62F1 6F      03160          LD     L,A              ;L=LSB string table addr
62F2 2660    03170          LD     H,TABLE/-8       ;H=MSB string table addr
62F4 7E      03180          LD     A,(HL)           ;A=LSB actual string addr
62F5 23      03190          INC    HL               ;Point to MSB string addr
62F6 66      03200          LD     H,(HL)           ;H=MSB actual string addr
62F7 6F      03210          LD     L,A              ;HL=string location addr
62F8 7E      03220 DSPMSG   LD     A,(HL)           ;Get byte to display
62F9 B7      03230          OR     A                ;See if zero terminator
62FA C8      03240          RET    Z                ;Finished if zero byte
62FB F5      03250          PUSH   AF               ;Save sign flag status
62FC E67F    03260          AND    7FH              ;Mask off bit 7
62FE CD3300  03270          CALL   33H              ;Display it on video
6301 F1      03280          POP    AF               ;Restore sign flag
6302 F8      03290          RET    M                ;Finished if bit 7 set
6303 23      03300          INC    HL               ;Advance string pointer
6304 18F2    03310          JR     DSPMSG           ;Go print next byte
6306 CD9A0A  03320 PRTNUM   CALL   0A9AH            ;Number in HL to ACCUM
6309 CD8D0F  03330          CALL   0F8DH            ;Convert # to string
630C 23      03340          INC    HL               ;Skip leading space char
630D 18E9    03350          JR     DSPMSG           ;Display converted number
630F 3E2C    03360 PRTCOM   LD     A,','            ;Comma character in A
6311 CD3300  03370          CALL   33H              ;Display it on video
6314 3E20    03380 PRTSPC   LD     A,' '            ;Space character in A
6316 CD3300  03390          CALL   33H              ;Display it on video
6319 C9      03400          RET                     ;Back to calling routine
631A 114440  03410 GETBFR   LD     DE,4044H         ;DE=Mod I date storage
631D 3A5400  03420          LD     A,(54H)          ;Check which model TRS-80
6320 3D      03430          DEC    A                ;A will be 0 on Model I
6321 C8      03440          RET    Z                ;Return if Model I
6322 111A42  03450          LD     DE,421AH         ;DE=Mod III date storage
6325 C9      03460          RET                     ;Pointing to Mod III date
6326 0E1E    03470 MAXDAY   LD     C,30D            ;C=# days in some months
6328 FE04    03480          CP     4                ;Is month April?
632A C8      03490          RET    Z                ;Return if April
632B FE06    03500          CP     6                ;Is month June?
632D C8      03510          RET    Z                ;Return if June
632E FE09    03520          CP     9                ;Is month September?
6330 C8      03530          RET    Z                ;Return if September
6331 FE0B    03540          CP     11D              ;Is month November?
6333 C8      03550          RET    Z                ;Return if November
6334 0C      03560          INC    C                ;C=31 (# days most mnths)
6335 FE02    03570          CP     2                ;Is month February?
6337 C0      03580          RET    NZ               ;Return if 31 day month
6338 0E1C    03590          LD     C,28D            ;C=# days in February
633A 3A2760  03600          LD     A,(DATSTR)       ;Get current year
633D E603    03610          AND    3                ;Year divisible by 4?
```

```
633F C0        03620          RET     NZ              ;If not leap year
6340 0C        03630          INC     C               ;C=29 ($ days leap Feb.)
6341 C9        03640          RET                     ;Finished
6342 53        03650 FCB      DEFM    'SETDATE/CMD'   ;File Control Block area
     45 54 44 41 54 45 2F 43 4D 44
634D 03        03660          DEFB    3               ; with program filename
0014           03670          DEFS    20D             ; (total 32 bytes)
0100           03680 FILBUF   DEFS    100H            ;File I/O buffer area
61CE           03690          END     START
00000 TOTAL ERRORS

ADV     62A0   APR     6073   ARROW   6296   AUG     6083   BACK    62D0
CHGCEN  62C8   CNTURY  602B   DATSTR  6027   DEC     60A1   DSPMSG  62F8
ENDYR   61CC   ERREXT  6250   EXIT    6273   EXIT2   6276   FCB     6342
FEB     6066   FILBUF  6362   FNDDAT  625D   FRI     6051   GETBFR  631A
GETKEY  6229   JAN     605F   JUL     607F   JUN     607B   MAR     606E
MARKER  6026   MAXDAY  6326   MAY     6078   MON     6033   MSG     60A9
NOTCR   6280   NOV     6099   OCT     6092   PRTCOM  630F   PRTNUM  6306
PRTSPC  6314   PRTSTR  62F0   RESTRT  61F6   SAT     6057   SEP     6089
SETDA2  62D6   SETDAY  62A8   START   61CE   SUN     602D   TABLE   6000
THU     6049   TUE     6039   UNSHFT  6295   USEPGM  61F0   WED     6040
```

MODEL III PRINTER DRIVER BUG - Here's a newly discovered one (at least to me). You won't even find this bug mentioned in my book, since I just discovered it myself. Try this line on your TRS-80 Model III:

```
A$=STRING$(128,"*");LPRINTA$A$:LPRINT"NEXT LINE"
```

Ignore for a moment the fact that your printer only prints 80 or 132 characters per line, and did an automatic wrap-around (maybe more than one). The point is, after it printed A$ twice (a total of 256 characters, which you may recognize as a somewhat significant number), did it do a carriage return before it printed "NEXT LINE"?

Or did it only do a linefeed?

If you only got a linefeed (your printer did not return to the START of the next line before it began printing), you found the BUG (if you didn't, it may be because your Disk Operating System uses a substitute printer driver)! Try adding one character to the first LPRINT statement (try LPRINTAA"*") and you'll notice the difference!

Wait a minute, if the average printer only prints 132 characters maximum, why would you want to send it a line of 256 characters? Well, you may only be able to PRINT 80 or 132 characters on a line. But, if you are embedding (non-printing) printer control codes within a string (or otherwise doing some fancy string manipulation), you just might hit the magic number of 256.

Strangely enough, this problem exists because of a fix to a bug that existed in the Model I. On the Mod I, if you typed LPRINT:LPRINT:LPRINT you would NOT get three carriage returns as you might expect, at least with some printers. The reason is that the logic in some printers refuses to recognize a carriage return (ASCII 0DH) character if nothing else has been printed on a line. The Model III attempts to correct this fault (which could really be considered a printer design fault) by changing a carriage return character to a linefeed if (and only if) no other characters have already been printed on the current line. The way it tells whether any characters have already been printed on the current line is to check a one-byte counter located at 402AH in the printer Device Control Block. Note that only one byte is alloted for the counter, thus once 255 characters have been printed, the next character to be printed (the 256th) "rolls over" the counter byte from 0FFH to zero. In other words, if zero characters OR ANY MULTIPLE OF 256 CHARACTERS have been printed on the current line, the printer driver converts the carriage return character to a linefeed.

If you have a program that LPRINTs strings that could possibly be exactly 256 bytes long, there is a quick (but not elegant) solution: Make sure that the byte at 402AH is never a zero byte when the carriage return character is sent to the printer driver. Returning to the example line at the start of this article, we could change LPRINTAA to LPRINTAA" " (if the combination AA was always 256 characters long) or even to LPRINTAA;:POKE16426,1:LPRINT (16426 decimal=402AH).

A proper fix would involve rewriting the printer driver, so that printer control sequences would not be counted in the total character count for the line, but since each make of printer uses different control codes (even different models of Radio Shack printers use different control sequences), it would be impossible to put one printer driver in ROM that would work correctly with all printers. The next best fix would be to prevent the byte at 402A from ever incrementing past 255. If you are patching the Model III ROM (as in the Model 4, where ROM can be copied into RAM and then worked on), you may be interested to know that a machine language INC (IX+5) instruction is found at 041FH. If this were replaced by a CALL to a subroutine elsewhere in memory (plus a NOP space filler), we could use the following patch subroutine to fix the problem:

```
INC (IX+5)      ;Replace patched instruction
RET NZ          ;If INC did not make number zero
DEC (IX+5)      ;Set back to 0FFH (255 decimal)
RET
```

Note that while the above patch would fix the 256 character linefeed problem, it would NOT fix the problem of printer control sequences being counted as normal characters. A patch to accomplish that would have to be customized to recognize valid printer control codes as used by the printer currently connected to the system.

PATCHES FOR TRSDOS 2.7DD by Tom Price - 10/24/82 (downloaded from Compuserve):

This article provides some corrective patches to TRSDOS 2.7DD, the double density operating system provided with the Radio Shack Double Density Conversion Kit. All of the patches can be applied with the PATCH command.

(1) These patches correct two errors in the PATCH command itself and this one MUST be applied first to allow proper operation of PATCH when patching 2.7DD system files.

```
PATCH *9 (R=6,B=235,F=07,C=00)
PATCH *9 (R=6,B=242,F=13,C=12)
```

(2) These patches correct a memory conflict between two of the system overlays during certain file I/O sequences. In addition, the 4430H call to the program loader was not preserving the IX register and this is also corrected.

```
PATCH *17 (R=1,B=4,F=321F45,C=C33451)
PATCH *17 (R=2,B=52,F=E5E5E5E5E5,C=321F45DDE5)
PATCH *17 (R=2,B=57,F=E5E5E5E5E5,C=CD0750DDE1)
PATCH *17
        (R=2,B=62,F=E5E5E5E5E5E5E5E5E5,C=F53E11321F45F1C9)
```

(3) These patches correct a problem with the $RAMDIR call which was causing the system to hang when retrieving single directory entries if the accessed directory slot was empty.

```
PATCH *14 (R=1,B=242,F=C0CB76,C=C3E54F)
PATCH *14
   (R=2,B=229,F=E5E5E5E5E5E5E5E5E5,C=C07EE6F0FE10C3F54E)
```

(4) This patch fixes a problem with BACKUP which prevented a valid backup of any diskette which contained any files with more than four extents.

```
PATCH *22 (R=2,B=196,F=80,C=90)
```

(5) This OPTIONAL patch removes the restriction in DEBUG which prevents accessing of any memory below 5400H.

```
PATCH *5 (R=1,B=207,F=54,C=00)
```

MODEL III TRSDOS 1.3 BUG by Morris "Mojo" Jones [71555,271] (downloaded from Compuserve):

If you use Model III TRSDOS 1.3 for assembly language file processing, you should be aware of the following bug. On an OPEN call to TRSDOS, the specification requires that register B contain the Logical Record Length to use. If the LRL specified in B is different from the LRL used to write the file, then subsequent reads will not work correctly. The OPEN call does not place the LRL from B in the File Control Block, but instead uses the LRL from the directory entry for the file. In order to do logical record processing correctly, you must copy the correct Logical Record Length to the File Control Block after the OPEN. Here is an example that avoids the bug:

```
LD DE,FCB       ;Point to file control block
LD HL,DSKBUF    ;and disk buffer
LD B,LRL        ;and my logical record length
PUSH DE         ;Save FCB address
PUSH BC         ;Save LRL
CALL $OPEN      ;Call TRSDOS for open
POP BC          ;Restore LRL
POP IX          ;Move FCB address to index reg
LD (IX+9),B     ;Copy correct LRL to FCB
```

Open calls should be done in this manner in any code that you may expect to run under TRSDOS 1.3 on the Model III. Happy programming!

THE FIRST CHECKSUM - This article appeared several months ago in Northern Bytes, and as this is the time of the year that people of the Christian and Jewish faiths celebrate significant religious holidays, and is the last month of the "Year of the Bible" here in the U.S.A., I felt that it bears repeating (especially since we are now have a much larger readership than when it first appeared). The following text is excerpted from a publication entitled The Holy Bible - Wholly True, which is authored by Winkie Pratney, a native of New Zealand. If you'd like a free copy of the of the complete publication you may write to Last Days Ministries, Box 40, Lindale, Texas 75771, and request a copy of EG#1. Here's the excerpted text from the publication:

In 1882, a young immigrant from Russia was just graduating from Harvard. He had a wonderful conversion to Christ, after traveling as an agnostic who often lectured on atheism! A brilliant mathematician, multi-lingual scholar and literary figure [A personal friend of men like William James, Pres. Eliot of Harvard, Ralph Waldo Emerson, Louisa May Alcott's father and other well-known scholars of that day ("Shorter Works" p. 11)], Ivan Panin began to study the Scriptures as a Christian. Knowing Hebrew, Aramaic and Greek, he began to read them in the original languages. Now both Hebrew and Greek are unique in that they do not have a number system. In other words, they do not use special symbols for their numbers (like our Arabic numerals 1, 2, 3, etc.) but use instead the letters of their alphabets to represent numbers. Each letter also has a numeric value [the publication contains a chart to illustrate this -editor].

Aware of the numerical values of the Greek and Hebrew alphabets, Panin experimented one day by replacing the letters with their corresponding numbers in Scripture. Here is Panin ... a mathematical genius, a Hebrew and Greek scholar, and he loves playing with numbers. Suddenly, his trained mind saw a mathematical pattern! As he studied more intensely, his excitement grew. A few short hours of work had him utterly amazed. The verses he had studied bore unmistakable evidence of an elaborate mathematical pattern, far beyond random chance, or human ability to construct. This discovery marked the turning point of his career, and from that time until his death in 1942, he devoted his entire life to the study of Bible numerics.

He showed, first of all, that the Bible, in its original language, is a skillfully designed product of a mathematical mastermind - far beyond any human possibility of deliberate structuring. He later supplied a representative of the Nobel Research Foundation with over 43,000 sheets of his studies [remember, this was before computers! -editor] accompanied by his statement that this was his evidence that the Bible was the Word of God. Their reply was, "As fas as our investigation has proceeded... we find the evidence overwhelmingly in favor of such a statement." He then issued a challenge throughout leading newspapers of the world to offer a "natural explanation" or refute the facts; not a single person was able to do so.

Panin found that patterns of prime numbers, such as 11, 13, 17, and 23, but especially 7, were found in great clusters. He would add up the sum of all numerical values for different words, sentences, paragraphs, passages, and whole books, and he found the same patterns in each of these forms! He found that the number of words in a vocabulary divides by 7. The number of proper names, both male and female divides by 7. The number of words that begin with a vowel divides by 7, likewise the number of words that begin with a consonant. The number of letters in a vocabulary divides by 7, and of those letters, those that are vowels and those that are consonants also divide by 7. Words that occured more than once divide by 7, and also words that appeared only once! The number of nouns is divisible by 7, also the words that are not. Even the number of words beginning with each letter of the alphabet! And on and on... Panin only stayed on a passage long enough to confirm beyond reasonable doubt the statistical evidence for supernatural design. But he stated that the longer you stayed on one, it would continue to yield further and further evidences of pattern within patterns until the mind reeled!!

Here is an example from the Old Testament. Just the very first sentence of the Bible - "In the beginning God created the heavens and the earth" (Genesis 1:1). That's the way it comes out in English, in the Hebrew it is exactly 7 words. The 7 words have exactly 28 (4x7) letters. There are three nouns (God, heavens, and earth). Taking the letters of these, substituting their number equivalents and adding them up, you get a combined total of 777 (111x7)! There is one Hebrew verb - created. Its total numerical value is 203 (29x7). The first three words contain the subject with exactly 14 (2x7) letters, likewise the other four are the object with exactly 14 letters. The Hebrew words for the two objects (heaven and earth) each have 7 letters. The value for the first, middle, and last letters in the sentence is 133 (19x7). The numeric value of the first and last letters of all words is 1393 (199x7); the value of the first and last letters of the first and last words of the verse is 497 (71x7). The value of the first and last letters of each of the words between is 896 (128x7). And so on, and so on ... in this verse alone there are 30 different features of 7. I have listed only 11 of them! The chance of this happening accidently is 1 in 33,000,000,000,000 (33 trillion).

And now an example from the New Testament! Matthew chapter 1, verses 1 through 11. The vocabulary has 49 words (7x7). 28 words begin with a vowel (4x7), the remaining 21 with a consonant (3x7). 7 end with a vowel, 42 with a consonant (6x7). The 49 words have 266 letters (38x7). Out of the 266 letters, 140 are vowels (20x7), 126 are consonants (18x7). Also, of these 49 words, 14 occur only once (2x7), 35 occur more than once (5x7), 42 (6x7) are nouns, 7 are not. The remaining common nouns have exactly 49 letters (7x7). Male names occur in all 56 times (8x7). The names of only 3 women appear in the passage, and the Greek letters of their names add up to 14 exactly!

Panin said it would have taken Matthew several months, working 8 hours a day to construct the genealogy [this passage is part of the genealogy of Jesus -editor], even if it were possible. But the names were chosen BEFORE MATTHEW WAS BORN!

The whole Bible is like this. I am just taking one small chunk of it and doing it in detail. Every paragraph, passage and book in the Bible can be shown to be constructed in the same marvelous way. What kind of fantastic collaboration between the disciples could have produced this structure without computers? How could mere fishermen and tax-collectors produce this kind of incredible structuring and design? What is crazy, is that Mark is a Roman, Luke a Greek, and Matthew a Jew, but they all wrote with the same pattern. Each one wrote with their own unique flavor. Mark's style is different, but the pattern is the same right through! So who wrote it? One Mind, one Author - one God - many different writers, but one Writer. Can you imagine what kind of Mind would do this and not even care if you ever found out?! What I want you to see is how smart God is! These are not just words, it's an incredible mathematical pattern. It dances with its own poetry in mathematics. A computer would go into raptures over this! It's like a building where every piece joins perfectly into each other. And what is wild, is you can't pull even one word out, without damaging the whole pattern. So the Bible carries within itself, a self-checking self-verifying protection factor. If a person comes along and says I don't like this one, the whole pattern falls apart. This cannot be found in any other religious "holy" book in the world.

[One closing editorial comment - since the Bible appears to have this "checksum" built in, it would seemingly be possible to apply this test for mathematical patterns to any "disputed" portion of scripture (there are a very few verses in our present-day Bibles that some theologians seem to think were added "after the fact" - by applying this test it should be possible to tell whether or not they were in fact part of the original scriptures). It would be interesting to program a computer to search for all such patterns in the entire Bible, and find out how many there actually are (and what are the chances that they would occur randomly). "Inspiration Of The Scriptures Scientifically Demonstrated" and other works of Ivan Panin are available from Box 206, Waubashene, Ontario L0K 2C0 (write for list and prices).]

REACH OUT, REACH OUT AND CRUSH SOMEONE... Here is a quote from an AT&T pamphlet, which was recently mailed to business customers:

"In 1984, 'free' directory assistance service for interstate calls-now subsidized through AT&T Long Distance and AT&T WATS- will end. Starting 1/1/84, there will be a charge for interstate directory assistance. This charge will ensure the recovery of the costs of providing this service from customers who actually use it frequently, as well as those who use it infrequently, if at all. The filed charge for an interstate directory assistance call is seventy-five cents.

"There will be one additional change in the pricing structure. Customers will be entitled to one free interstate directory assistance call per billing period for each customer main-billed account when at least one AT&T Long Distance call (including calling card or 3rd number calls) or AT&T WATS call is also placed during the same billing period."

Two comments. First of all, if you want to scream at someone at AT&T, you can call them toll free at (800) 222-0400.

Second, I think that there are some valid reasons for objecting to this charge. As it stands right now, AT&T has a monopoly on long distance directory assistance. Local telephone companies update their directory assistance listings on a daily basis, but only provide them to AT&T. If they could be forced to provide these listings in a timely manner to anyone requesting it, it's conceivable that companies such as MCI, SPRINT, etc. could offer competing directory assistance at a lower price. It's even possible that DA could be put online, so that personal and business computer users with telecommunications capability could access Ma Bell's database, thus eliminating the need for operator intervention (and presumably the need for the charge).

It is a phone company requirement that we use the proper number when we call someone else. This was not always the case. When the telephone network was converted from manual ("Hello, Central?") to dial switching, I'm sure that one of the phone company promises was that numbers would be easily obtainable, through the phone book for local numbers and through directory assistance for new or out-of-town numbers. Since the phone company requires their use, I believe that the phone user should not have to pay extra to get them. Next they'll want to charge you $19.95 per telephone book!!

(Joe Werner of the Mid-Michigan Microcomputer Group apparently has had some thoughts along these same lines. Writing in ENERGY, the Mid-Michigan Computer Consortium Magazine, Joe made the following observation: "Other traditional practices may change, too. For example, when you make a long-distance call, you tie up a long-distance line even if the telephone you're calling is busy. But when the long-distance and local telephone services are no longer provided by the same company, why shouldn't AT&T charge you for using its facilities to find out that someone is on the phone on the other end, or even that no one is home? (This is strictly my speculation. AT&T has said nothing about such charges. But are they unreasonable?)" While Joe makes a valid point, there is one BIG difference in that if AT&T begins charging for busy signals or no answers, there's a good chance that one or more of AT&T's competitors would NOT choose to do so - in other words, the fear of losing business to the competition will probably keep AT&T from doing it. But, as I've pointed out already, AT&T has a de facto monopoly on the directory assistance business, so they have no fear of charging for that. Remember that next time AT&T tries to tell you what a bad thing competition is!)

Note that they are generous enough to give you ONE free call per month, IF you make an AT&T long distance call. The way I read it, even if you have a business with five lines and make $1,000 in AT&T long distance calls per month, you'd only get ONE free call since there would only be one main account for the five lines.

If you object to this charge, write the Federal Communications Commission and let them know about it, in no uncertain terms. I will offer the following comments as to how I feel about it, which might give you some ideas for your letter:

I do not believe this charge should be allowed, at least until such time as local telephone companies are required to provide their number changes (that is, information on new numbers installed and on changed and disconnected numbers) to all interested parties on the same basis that they provide them to AT&T (that is, in the same timely manner and with the same charge that they charge AT&T for the service, if any). You say the local telephone companies don't charge AT&T to provide them with this information? Why not?? AT&T is going to use it to make money!

If the FCC is still determined to let AT&T charge for directory assistance, then I think that the minimum call allowance should be upped to a more reasonable amount of calls, say 20 per month as is currently in effect within your local area cr n Michigan. Also, I believe that there should und o circumstances be a charge when AT&T is unable to provide a requested number, either because it isn't in the directory, is unlisted, or whatever. Also I do not think there should be a charge when a new listing is requested (a number recently installed or changed), since directory assistance may be the only way to obtain such numbers.

Furthermore, if this charge is allowed, I think that local phone companies should be required to provide a new phone number at no charge up to one year in advance of an anticipated move to a new location (assuming that the previous phone number cannot be reused). This new number should be listed in the directory at no charge, and if a customer dials it prior to the time that the move is made, the automated recording should respond with something like "The number you have dialed, xxx-xxxx, is not yet in service. The number now in effect is xxx-xxxx.") Or perhaps an "anticipated activation date" could be included in the book listing. The point of this would be to allow listing in the phone book and more importantly, on personal and company stationary well in advance of the anticipated move. In addition, I suggest that phone companies should be required to place a recording giving the new number to callers to the previous number for at least one year after the move has been made. This should be provided even if the customer moves away from his previous local telephone service area. And, out-of-town telephone directories should be provided FREE to anyone that requests them (as is already the case within your own area code in Michigan).

Finally, if the local telephone company makes a mistake in the phone book listing, there should be no charge to obtain that number from directory assistance. And if the directory assistance operator makes an error when giving out a listing, either by reporting that there is no number for a given party when such a number in fact exists, or by giving out the wrong number, then I think that they should be required to at least issue a cre r the call, and preferably should be assessed damages in the m of a double or triple refund for the call.

The bottom line is that I think that (1) There should NOT be a charge for long distance directory assistance at this time, (2) That if the charge is allowed anyway, it should be reduced to a lower cost and more "free" calls should be allowed, (3) That persons and companies other than AT&T should have equal access to local telephone company number information on a timely basis, (4) AT&T should not be permitted to charge when a listing is not given, or is unavailable from any other source (as in the case of new numbers, etc.), (5) Credit should be given for the call when the directory assistance operator makes an error, and (6) Local telephone companies should issue new numbers well in advance of an anticipated move, and provide recordings where necessary to avoid having to call directory assistance to obtain changed numbers, even if the changed number is served by another telephone company, or is out of the local calling area.

If you agree with any of this, feel free to use any portion of this text as part of a letter that you can send to the Federal Communications Commission, and possibly to your state's public utilities commission as well. Let's fight these encroachments upon our ability to communicate freely!!

And one last point ... perhaps now is the time to call directory assistance and get all the phone numbers you think you might ever need. Also, it might be a good idea to start putting your phone number on ALL of your correspondence, so the louses won't be as easily able to charge someone to get YOUR number if they are able to make this stick. Finally, if you make more than $25 in long distance calls per month, think about subscribing to an alternate long-distance service, such as one of the services offered by GTE SPRINT (for information dial (800) 521-4949), ITT LONGER DISTANCE (for information dial (800) 221-7267), MCI TELECOMMUNICATIONS (for information dial (800) 482-1[illegible]), WESTERN UNION METROFONE (for information dial 562-0240), or your local WATS line reseller (see your yellow pages under Telephone Companies or similar listings). Note that the (800) numbers shown above may not be good throughout the United States - if you reach a recording, check your phone book for a number that services your area.

ELECTRONIC MAIL COMES OF AGE - Perhaps you have seen the advertisements on television for MCI Mail. MCI Mail is the new electronic mail service from MCI, the alternate long distance telephone company that has AT&T so worried. Apparently, having lusted AT&T's monopoly on long distance phone service, they are now going to move in on the post office's territory and start transmitting mail electronically.

The nice thing about the MCI Mail system is that it seems to have been designed with the personal computer user in mind. The system has been made easy to use, and so far seems to be quite cost effective. However, I don't think that MCI has promoted it in such a way that personal computer users realize the advantages to signing up for such a service. I think that all personal computer users that have telecommunications capability should register with MCI Mail, even if you don't intend to use it right away. It doesn't cost anything to register, and registering makes you "known" to the MCI Mail system, so that anyone can send you an "instant" letter. The "instant" letter is a computer-to-computer letter, and costs only $1.00 for the first "electronic ounce" (7500 characters).

Registering for MCI Mail is simple. Within the U.S.A., you may call (800) MCI-2255 (in Washington, D.C. call 833-8484) and talk to someone at MCI Mail. However, you have a computer, right? So, why not register electronically? To do so, set up your terminal program for 8 bits, no parity, 1 stop bit, and either 110, 300, or 1200 baud depending on your equipment. Then call MCI Mail at (800) 323-7751. When you hear the computer answer the phone and send carrier tone, connect your MODEM and press <ENTER> (or <RETURN>, if you're not using a TRS-80). Press it once if you're using 1200 baud, twice if using 110 or 300 baud. When asked for a user name, enter "REGISTER". When asked for a password, enter "REGISTER" again (the password will not be echoed back to your display). Then simply answer the questions as they are displayed on your computer.

You can register for either a personal or company account. The main difference is that if you register for a company account, you can obtain separate user I.D.s and accounts for each of your employees, yet have them all billed in one monthly statement. A personal account is for one user only, and is billed to that user personally. In either case, registration is FREE - you don't pay anything until you actually send a letter.

You will be given the option of selecting a user name other than the one that would normally be used for you. For example, if your name is John Smith, your user name would be JSMITH (your first initial plus your last name). You might be tempted to use something else. Don't. The reason is that if anyone wants to send you "instant" mail, they will usually try to send it using your first initial and last name. If you're registered some other way, there is no way for anyone to get your correct user name unless you have previously told them what it is. If you have a company and want a user name that is distinctive to your company, set up a separate User I.D. with that user name (remember, additional accounts don't cost anything until you actually send mail from them). But make sure that your personal account uses your first initial and last name, so that others can send you mail.

What if there's already a John Smith on the system, I hear you ask? That's okay - user names are not unique. What will happen is that when someone tries to leave electronic mail for JSMITH, a list of all possible "J. Smiths" (names and addresses) will be displayed, and the sender will be asked to select which is the proper addressee. Each user also receives a seven digit MCI Mail ID that IS unique, and can be used instead of the user name to specify a particular individual without having to pick out the proper person from a list of those with the same first initial and last name.

Once you have registered, there will be about a two-week wait until you receive a welcome package from MCI Mail. This will contain information such as your user name, password, customer number, MCI Mail ID, and customer service numbers for MCI Mail. It also contains a "Welcome Kit and Service Guide", a "MCI Mail Service Guide", a "Basic User's Guide", a "Basic Quick Reference Card", and information on advanced services available through the MCI Mail system. It also contains information on the Dow Jones News/Retrieval service, which is also available (charged according to number of minutes used and time of day) through the MCI Mail service.

One thing that you might expect is that MCI Mail would be accessed through the MCI long distance telephone network. If you thought that, you'd be wrong. At the time of this writing, only 33 major cities have access to MCI Mail through a local, seven-digit telephone number, and it is not the same number used for MCI long distance phone service. However, outlying areas can use a toll-free telephone number to access MCI Mail - the number is (800) 323-7751, and it should work from anywhere within the continental United States. Once you are on MCI Mail, you can type HELP PHONES to find out if a local number is available to serve your area.

Once you have registered with MCI Mail, there are four kinds of letters you can send:

1) An "instant" letter. This is a "computer-to-computer" letter. You leave it on the system, and it is immediately available to the addressee the next time he or she signs onto the MCI Mail system. Cost is $1.00 for the first "electronic ounce" (7500 characters).

2) A standard MCI letter. This is used to send mail to folks that aren't registered with MCI Mail. The letter is sent electronically to an MCI Mail postal center near the recipient, where it is laser-printed on high-quality white bond paper, placed in an orange MCI Mail envelope, and deposited with the U.S. Mail, which hopefully will get it to the addressee by the next day. Cost is $2.00 for the first "electronic ounce" (three laser-printed pages), $1.00 for each additional.

3) MCI Mail Overnight Letter. This is hand delivered by noon of the following business day, in most areas of the country (except for areas that are really back in the boondocks - like Sault Ste. Marie, so don't bother trying to send me an overnight letter!). Cost is $6.00 for first "electronic ounce" (three laser-printed pages), $1.00 for each additional.

4) MCI Mail Four-Hour Letter. Available to major cities only, this is hand delivered within four hours of being sent. Cost is $25.00 for first "electronic ounce" (three laser-printed pages), still only $1.00 for each additional "ounce".

The "instant" letter service is probably the one that most personal computer users would prefer to use. Keep in mind that any ASCII text can be sent in an MCI Mail letter. That includes BASIC programs saved in ASCII format, word-processor ASCII text files, Editor-Assembler source code programs that have had the tab characters converted to spaces, or machine language programs that have been converted to ASCII format using the HEX/CMD program included with MODEM80, or a similar program. Utility programs that convert various types of files to ASCII are available in the download sections of many of the Bulletin Board Systems around the country. When you have converted your file to ASCII format (if necessary), you can compute how much it will cost to transmit by figuring $1.00 per 7500 characters (go ahead, write yourself a short utility program to count characters in a file!).

You may say, well, why should I use MCI Mail when I already have CompuServe or The Source? Simple. You may save money. With MCI Mail, your "letter" isn't actually "sent" until you tell the system to send it - and in the meantime, it doesn't cost you a dime. How many times have you tried to upload to CompuServe and had to wait for the computer, struggle with the FILGE editor, re-transmit a file that was garbled in transmission, and generally frog around with the system, while time (and money) was ticking away? That won't happen with MCI Mail, because you're only charged for what you actually send. If someone in your house picks up an extension phone while you're in the process of uploading a file, simply abandon the upload, discard the "draft" copy of your "letter", and start over.

I mentioned the FILGE editor of CompuServe. If you hate it, you'll love the CREATE command of MCI Mail. Simply type CREATE, specify whom the letter is to go to, and begin typing your letter (or uploading it, if you've already prepared it offline). After you've typed (or uploaded) the last line, type a slash ("/") on a line by itself. You can then edit the draft, send it, or abandon it. User-friendly menus guide you through every step. If you enter the CREATE function and discover that the addressee isn't "known" to the system, you have the option of sending a laser-printed letter by entering the appropriate address information, or aborting the letter by entering a slash ("/") at the "TO:", "CC:", or "SUBJECT:" prompts. So, if you're not sure whether someone is registered with MCI Mail, you can still attempt to send them an "instant" letter, and just abort the attempt if you find that they aren't an MCI Mail user (and you don't want to send them a laser-printed letter).

The only caution that must be observed when using CREATE is to properly terminate each line with a carriage return (the ENTER key on the TRS-80). MCI Mail will let you continue to type forever without inserting the <C/R>, but if you don't insert the carriage returns, one "forced" <C/R> will be sent automatically

after each 256 characters when retransmitting your letter - even if the break occurs in the middle of a word! If you are sending a laser-printed letter to someone, your lines should be as close to 80 characters in length as possible (but must NOT be longer than 80 characters) for best appearance of the final copy. TRS-80 Model I and III users that must contend with a 64 column display may find it more convenient to use a word-processing program to create the body of their document, then save it in "final copy" format (using 80 characters per line maximum) to an ASCII text file, and then upload the text file to MCI Mail.

Uploading text from a disk file to MCI Mail is made easy by the fact that the CREATE function supports XON/XOFF protocol. This means that if you are using a terminal program such as MODEM80 to upload a text file, the terminal program will automatically stop and wait whenever the MCI Mail text buffer is full. This in effect means that you can transmit text at full speed, with no lost characters. Note, however, that XON/XOFF protocol is NOT an error detection or correction protocol - it is simply a way of automatically getting the sending computer to stop and wait when MCI Mail's text buffer is full. It would be nice if MCI Mail had some way of supporting the CP/M XMODEM protocol for error detection and correction (as well as allowing files with control characters to be sent), but MCI Mail is primarily a mail service, and was not really designed for the transfer of computer programs.

Another side effect of MCI Mail being designed for letters rather than sending programs is that when your recipient gets a program or text file you send, he will probably have to load it into a word-processing program and remove extraneous system prompts, carriage returns, and other text that is not part of the program. This can be minimized, however, if he will (1) Use the ACCOUNT command of MCI Mail to temporarily set the terminal line length to the longest possible setting (132), to help reduce the number of program lines that will be broken up by extra carriage returns, and (2) Use the PRINT command, rather than the READ command, to PRINT the program without stopping every few lines to give the operator a chance to read it (and leaving user prompts in the middle of the program listing). The program can then be downloaded to a disk file, and any text placed before or after the actual program can be eliminated by using a word-processing program (or other utility program that lets you selectively kill lines in a downloaded file - this may also be available on your local Bulletin Board System).

MCI Mail has some nice features that make life easy for the user. All "mail" is associated with a "desk". Incoming mail can be either in your "inbox" (mail you haven't read yet) or on your "desk" (for mail you've already read, but may wish to read again - as, for example, if you tried to download the "mail" and were for some reason unsuccessful on the first try). Outgoing mail can be either in the form of a "draft" (the letter you're currently creating, or a letter you've created but haven't sent yet - you can only have one "draft" in the works at a time), or can be in your "outbox" (this is mail you've already sent, but may wish to review). Mail in your "inbox" is never automatically cleared - it stays there until you read it. Mail in any other location is automatically cleared after being there for 24 hours (presumably, that's when the electronic "janitor" comes in and cleans your desk).

A "draft" copy you're working on is not lost if you are accidentally disconnected from the system. Simply sign back on within 24 hours, and you can use the editor to finish creating your letter. No more sinking feeling after you've just typed in five pages of text, and the dog knocks the extension phone off the hook and breaks the connection!

Other nice features include the ability to send "carbon copies" of a letter to as many recipients as you wish. And, if you use MCI Mail to send hardcopy letters, you can for a small fee register your personal or company letterhead and/or your signature with MCI Mail, and these can then be laser-printed on the letters you send, in effect meaning that you can electronically send letters with your signature and company letterhead. You can even register multiple letterheads or signatures if you are an "advanced" user. Once you've registered them, there's never an extra charge to actually use them in a letter you're sending.

It is possible to get confirmation that a letter you've sent has been delivered (or actually read, in the case of an "instant" letter), by using the RECEIPT option. The confirmation message appears in your "inbox". There is no extra charge for this service!

What is so great about this service is that there are no minimum charges, no monthly fees, no storage fees, no connect-time charges, and no charge to read mail sent to you. The only thing you pay for is mail that you actually send to others (and for Dow Jones News/Retrieval, should you choose to access that service).

If you are willing to pay $10 per month, you can become "advanced" MCI Mail user. There are several potent... advantages to becoming an advanced user, including:

1) You can use "Command:" prompts instead of menus. As an advanced user, you get the control and flexibility to enter commands and options when you want them, the way you want them.

2) You can address a number of people at once with a mailing list. Create a mailing list with everyone's name on it -- and send your message TO: the listname. A single entry -- but everyone on the list gets the message. A list can have any number of electronic and postal addresses alike.

3) You can avoid the multiple listing for similar names; create a personal "address book". If you always have to select your Joe Smith from the display of other Joe Smiths on the system -- create a mailing list just for Joe's name, and use it every time you need to address him.

4) It's possible for an advanced user to forward a message to someone else. When you want to pass along some information you've received, you can forward a copy of the message, including a "cover letter" from you with your own comments about what you're forwarding.

5) You can choose a different "style" of correspondence. In addition to the business-letter style on MCI Mail, you can opt to send a more casual MEMO, where the paper copy looks just like the electronic copy, with the complete envelope including all TO and CC recipient names.

6) It's possible to use another letterhead and/or another signature. An advanced user can register additional signature and letterhead graphics. Add the personal yet professional touch to your MEMOs and LETTERs using a logo and signature that reflect a formal or informal style.

7) You get a bigger Mailbox, with longer message retention. Advanced service allows up to 250 kilobytes of storage, as well as 5 days' storage for your messages and DRAFTS. If you need m storage, MCI Mail will increase your Mailbox by another 2... kilobytes at $10/month; there's no limit to the size Mailbox you may have.

[I cannot tell a lie, I downloaded the info on MCI Mail's advanced user features, and worked the text into this article using my word processor. Computers tend to make one a bit lazy.]

Additional services are also available for business users, such as a volume mail service that can be used to reach many people fast, and "reverse charge mailboxes". A reverse charge mailbox is the MCI Mail equivalent of "business reply" envelope (or a toll-free telephone number), in that whenever an "instant" letter is sent to the User I.D. associated with the reverse charge mailbox, the recipient pays the "postage" for the letter instead of the sender. Some companies now pay close to $2.00 per incoming call to "toll-free" telephone answering services (these answering services will take orders for your company on their (800) service INWATS telephone lines, but will usually take only a limited amount of information per call and will charge extra if additional information must be obtained from the caller), but if customers could be encouraged to order electronically through the MCI Mail system, the same type of "toll-free" service could be provided for only $1.00 per "instant" letter (of 7500 characters or less) received. Of course, most companies would want to use their reverse charge mailbox for orders only (or perhaps for communication with "preferred" clients or customers, or with their distributors), and have another User I.D. (that is NOT a reverse charge mailbox) for receipt of "regular" mail.

Starting in early 1984, MCI Mail goes international. MCI Mail users will be able to send and receive MCI Mail messages to and from all Telex addresses worldwide. Every MCI Mail subscriber will have a Telex number. Telex messages sent to this number will be placed in your MCI Mailbox along with your "instant" letters. When you send MCI Mail messages, you will be able to include Telex addresses just as you may now enter pos addresses. These messages will be delivered by M... International (MCII) through the Telex networks.

Once MCI Mail Telex Service becomes available, your Telex number would be 650 followed by your MCI Mail ID number. For example, my MCI Mail ID number is 102-7413, so my Telex number will be 6501027413. There is no charge to receive Telex messages, just as there is no charge to receive an "instant"

letter. If you choose to send a Telex message, it will be sent at "competitive Telex rates."

Any complaints about MCI Mail? Only a few minor problems. For one thing, it's sometimes just a bit difficult to find the information you need in the "Basic User's Guide". For example, if you wanted to know how to change your terminal line length setting, you might not know that you have to look under "Changing Your MCI Mail Account Information". An index in the back of the book would be nice.

Also, when you send a letter via regular mail and find it necessary to also send money, you can enclose a check or money order, and you can't do that with MCI Mail. You can, of course, use a credit card when you order from a business (if you even have a credit card, and I for one don't believe that everyone should be required to have a credit card to do business!). It would be nice if you could somehow register a check form (with your own personal bank account number encoded on it) that could be electronically laser-printed and sent (by U.S. mail) when you need to send money. A function of the "instant" letter could confirm that a check had been printed and was in the mail, so that "instant" letter recipients would know that they would receive the check no later than the next day in most cases. When used with other types of MCI Mail, the check could be enclosed with the MCI Mail letter!

But for the present, I see MCI Mail as a service that has great potential for personal computer users. We have registered and are on the system, so if you want to submit a program or text file to Northern Bytes, send it directly to me (user name: JDECKER, MCI Mail ID: 102-7413). You can also use MCI Mail to place a Visa or Mastercard order with The Alternate Source (user name: TAS, MCI Mail ID: 109-7407), or to send a letter specifically to Charley Butler at TAS (user name: CBUTLER, MCI Mail ID: 113-3644). If you find any new or unique uses for this system, let us know and we'll let others know.

THE ALTERNATE SOURCE ANNOUNCES "TOLL FREE" ORDERING for MCI Mail users. Here's how it works: When you wish to place a C.O.D., Visa or Mastercard order for $19.90 or more, send us an MCI Mail "instant" letter. Our user name is TAS and our MCI Mail ID number is 109-7407. Be sure to specify that you're ordering C.O.D, or if you're using a credit card tell us which one you're using and the card number and expiration date. For each item ordered, specify which version you need, if applicable (Model I or Model III, Cassette or Disk version, etc.). Also don't forget to include your full address, including a street address for U.P.S. delivery! We normally charge $3.00 for shipping and handling of your order, but when you place a C.O.D. or credit card order of $19.90 or more via MCI Mail, we will reduce the shipping and handling charge to $2.00 (plus C.O.D. fee if applicable). The dollar you save will cover the cost of your "instant" letter, and you can place your order any time of day or night. NOTE: This method may also be used to place smaller orders (or to send other types of mail to TAS), but the $1.00 credit will be given only when your total order exceeds $19.90, exclusive of the shipping and handling charge.

UNKILLING FILES IN TRSDOS 1.3 by Dale McClure - When you kill a file you only alter the directory entry, the Hash Index Table (HIT), and the Granule Allocation Table (GAT). The file is still on the disk, and if you haven't written to the disk (too much) you can recover most or all of the file.

Most DOS's kill the directory entry by zeroing one byte. TRSDOS 1.3, in Tandy's infinite wisdom, zeroes all 48 directory entry bytes. Fortunately, it does so by zeroing the first byte and then using the LDIR function with a byte count of 47 to move that first zero into the other directory bytes.

We can correct this procedure by changing the 47 to 01, thus only the first two bytes will be zeroed. Use your disk zap utility to change byte B9, sector 14, track 16 of a TRSDOS 1.3 disk from 2FH (47) to 01.

To unkill a file just zap a 10H into the first byte of the directory entry (the second byte isn't used and can be ignored). If you are using Super Utility Plus, use the repair section to repair the HIT and GAT and you are through. If you don't have SU+, use the techniques in Pennington's "TRS-80 Disk & Other Mysteries" to recover the HIT and GAT entries.

RETRIEVING KILLED NEWDOS/80 FILES by Tim Porreca is reprinted from TBUG (newsletter of the TRS-80 Baltimore Users Group) and revised and edited by Jack Decker and Dave McGlumphy to add some additional information, and to make it more universally applicable to the many different disk formats that NEWDOS/80 is capable of producing. This article appeared in the June, 1983 issue of Northern Bytes, but I thought I'd run it again (with corrections included) for the benefit of the wider audience we now have!

One good thing about computers is that everything is so well documented. The problem is, can you understand the documentation? I have always been plagued with questions from computer owners who have the information they need but can't understand it. A good example of this is the NEWDOS/80 manual. The wording in this manual can be very difficult and technical for a beginner. I found one book, TRS-80 ASSEMBLY-LANGUAGE PROGRAMMING, to be geared in the direction that the person reading it was already a master assembly-language programmer.

In this article I will try to explain an easy way that one can recover a killed file using Superzap on NEWDOS/80 version 2.0. When one first enters Superzap one must first use the DNTH command (Display Name/Type Hashcode). You will then enter the filename and type of file (i.e. for LMOFFSET/CMD, LMOFFSET is the filename and CMD is the file type). Make note of the hex hashcode that is displayed. Now type DFS (Display File's Sector). Answer the FILESPEC? question with DIR/SYSn (replace n with the drive number that contains the disk with the killed file), then answer the RELATIVE-SECTOR-WITHIN-FILE #? question with "1". This screen is the HIT sector for the directory. It can be recognized by the FRS 1 on the bottom left. Now press the ; sign on the keyboard once and the drive will start and the screen will change, this should be remembered as number 1. If you do not see the filename of the file you want to recover along the right (ASCII) side of the screen, press the ; sign again and count 1 more for each screen until the program you want to retrieve is seen on the right side of the screen. When you reach this screen, write down your current count (the number of times you had to press the ; sign to get to this screen). Now, type the letters M O D 0 0. A large cursor should appear at the upper left of the screen. Use the arrow keys to move this cursor to the beginning of the line that your file is on. For any /CMD, /BAS, /TXT or other "user" file type the numbers 1 and 0 followed by <ENTER>. For a system (SYSnn/SYS) file type 5F and for BASIC/CMD type 1E. After <ENTER> is pressed, press "Y" to modify the disk sector. Now make note of the horizontal line on which this correction was made (i.e. E0, 20, 60, etc. - this is shown in the vertical column of two-digit hex numbers near the left-hand side of the screen). Return now to the HIT sector (type "K" and answer the RELATIVE-SECTOR-WITHIN-FILE #? question with "1").

Type M O D 0 0 and move the the cursor to the leftmost side of the screen and then to the line on which the correction was just made in the sector that contained the filename (check the two-digit hex number near the leftmost side of the screen to make sure). Now recall the count of the number of times you had to press the ; sign to get to the screen containing your filename, subtract one, multiply by two, and press the right arrow key that many times. After this is completed type in the hashcode that was previously recorded. Now hit <ENTER> and respond "Y". Type E X I T and your program should be there.

Your program may appear to be restored at this point, but there is still one problem - the GAT sector hasn't been fixed and writing another file to that diskette may overlay the data sectors. You can prove that by using a blank diskette, creating a file, KILLing it, fixing it by using the above method, writing another file to the same diskette and running DIRCHECK. The simplest solution is to copy the "unKILLed" file to another diskette, KILL the "unKILLed" file, and copy the copied file back again. Whew!

In order for this to work one must understand a few basic things. First, when a program is killed it is not deleted from the disk, but a code is removed so the directory will skip over it. With these instructions one can rebuild the code. Second, if another program has been saved on the disk since the file was killed, then there is a very good possibility that the program wrote over the previous one, in which case it will be impossible to recover the original program. One should try this process on a spare disk before using it on the real thing, for if something is done wrong then it may destroy some data if not the entire disk. I would like to thank Ray and Glenn Kreisel for their help in doing this process.

Happy Computering!

TRSDIR - A program that permits users of NEWDOS/80 version 2.0 to read and display the directory of Model III TRSDOS disks (One or more disk drives are required). The original version of this program was written by Tony Domigan, P.O. Box 150, Thomastown, Victoria 3074, Australia. The version of this program presented here has been extensively modified (but not entirely rewritten) by Jack Decker, in order to eliminate some shortcomings of the original version, including making it compatible with both Models I and III, and allowing a 40-track TRSDOS disk directory to be read on an 80-track drive (skipped track operation). The following text was originally written by Mr. Domigan, but has been changed or added to where necessary, in order to reflect the current operation of the program as published herein:

I prefer to use NEWDOS/80 (v2.0) almost exclusively on my Model III, however, as TRSDOS remains the most common medium of program transfer, many of my programs and program packages still reside in the TRSDOS environment. NEWDOS/80 allows good access to TRSDOS disks via the Copy command, Debug's load sector option and Superzap's display disk sector option, but it is not possible to 'DIR' the TRSDOS disk. This is not a catastrophe but it is certainly a nuisance when you wish to quickly scan the TRSDOS disk for a particular file or for free space. My program TRSDIR is designed to overcome this inconvenience by providing a TRSDOS 'DIR' from NEWDOS/80.

TRSDIR will display the disk name, the number of free granules and the active filenames with their extensions. This directory will be different in screen format from the NEWDOS/80 'DIR', partly because I wished to simplify the program task, but also to fit the entire directory contents in one video screen. TRSDOS has 16 pages (sectors) of directory entries with each page holding 5 FPDE's giving it a total of 80 possible FPDE's. NEWDOS/80, on the other hand, has 8 pages of directory entries with 8 FPDE's per page giving a total of 64 possible FPDE's (if the standard Model III format is used). TRSDIR will display 5 TRSDOS filespecs across each video line. In the event that the target TRSDOS disk contains more than 75 active files, the SCROLL routine will scroll the video up one line so that all directory entries are visible on one page. As a consequence, however, the header data will be lost.

TRSDIR will work on any drive you nominate. For single drive users, TRSDIR will prompt for the TRSDOS disk to be mounted on drive zero. If you have specified drive zero in error, you can abort TRSDIR at this point (and return to NEWDOS/80 READY) by pressing the BREAK key (assuming that it has not been disabled through use of SYSTEM option AG).

The program begins by checking to see whether it is running on a Model I or a Model III. It was originally written for the Model III, so if it is running on a Model I, some memory references and PDRIVE table values need to be changed. Once that is taken care of, a check is made to see if a drivespec was specified in the TRSDIR command line. The format of the command line is 'TRSDIR 0', 'TRSDIR :2' etc (the ':' delimeter is optional). The default drivespec is drive zero. If the letter 'S' is placed in the command line (after the drivespec, if any - example: 'TRSDIR 2 S'), the program assumes that a 40-track TRSDOS disk is being read on an 80-track drive, and the "skip track" option is invoked.

As TRSDOS does not use relative byte numbering for its tracks and sectors, PDRIVE 4 (TRSDOS) must be substituted for the destination PDRIVE. The PDRIVE information is stored at relative sector 2 in the NEWDOS/80 disk and portions of this data are passed to reserved RAM locations 4291H-42B8H (4371H-4398H in the Model I) when NEWDOS/80 is 'booted in'. The current PDRIVE information in memory is stored in PDVSTO by the INITZ routine and to simplify decision making within the program, the TRSDIR PSWAP routine replaces all PDRIVES with PDRIVE 4.

The first sector of the directory, the GAT sector, is then read into the program's general I/O buffer at 5200-52FFH by the SREAD routine.

The GAT sector is decoded into the disk name, disk date, and the number of free granules. The lower 6 bits of each allocation byte in the granule allocation table are decoded as allocated (=set) or free (=reset). A value of 3FH therefore represents all granules as allocated for that track (see "Reconstructing Model III TRSDOS Directory Entries" by Charley Butler, Issue 12, The Alternate Source).

The HIT sector is bypassed and each subsequent FPDE page is read into the input buffer one at a time for decoding. Each FPDE position of each page is checked to see if the first byte is allocated (not zero), and if active the filespec is printed.

There are 5 filespec print positions across the screen, which corresponds to the number of FPDE positions within a directory page. Where an FPDE position is inactive, the screen print position is not incremented. If all directory pages are full then there will be 16 lines of display used, causing the header data to be lost as the screen scrolls up.

When the directory display is complete, a call is made to 49H so that the directory is not overwritten by the 'NEWDOS/80 READY' prompt before you have a chance to view it. When you are ready to continue, entering 'R' will repeat the original TRSDIR command, and hitting any other key will exit to DOS. PDVSTO is then moved to 4291H (4371H in the Model I), restoring the PDRIVES to their original settings. On exit the cursor is positioned to the third last video line as the best compromise to prevent overwriting of the main text area at the top half of the screen.

TRSDIR is not foolproof. If the drive selected does not contain a Model III TRSDOS disk, you can expect garbage to be displayed instead of meaningful directory data. Single drive users should replace the NEWDOS/80 disk in the drive prior to exiting the TRSDIR program (by pressing a key other than 'R' to repeat the TRSDIR command).

TRSDIR could be enhanced by adding code to include the 'A' option to decode each FPDE's length in grans, number of records etc. I did not think that a directory print routine was necessary as ctrl-* (or the JKL function of NEWDOS/80, if enabled) works satisfactorily within the call to 49H in QLOOP.

(NOTE: If you do enhance this program, why not share you efforts with our readers? Send us your enhancements, and we'll print them in a future issue of Northern Bytes! -editor)

```
                  ;**************************************************
                  ;**                     TRSDIR                   **
                  ;** Model III TRSDOS Directory from NEWDOS 80 V2.0 **
                  ;**   Original version copyright 1983 T. Domigan  **
                  ;**               P.O. Box  150                   **
                  ;**           Thomastown, Victoria, 3074          **
                  ;**                 Australia.                    **
                  ;**  This version has been modified by Jack Decker **
                  ;**  to run on Model I or Model III, and to allow **
                  ;**  40 track disks to be read on 80 track drives. **
                  ;**************************************************
                  ;
                  ;
5300                      ORG     5300H           ;Anywhere above 5300H
                  ;
                  ;       ** Test for Model (I or III) **
                  ;
5300 3A5400       START   LD      A,(54H)         ;Get byte from ROM
5303 3D                   DEC     A               ;Determine if Mod 1 or 3
5304 2023                 JR      NZ,DRVSPC       ;Go if Model III
                  ;
                  ;       ** Modify Program for Model I **
                  ;
5306 E5                   PUSH    HL              ;Save input buffer pntr
5307 215305               LD      HL,0553H        ;Mod I scrn scroll rtne
530A 22A554               LD      (GOSCRL+1),HL   ;Store it in program
530D 217143               LD      HL,4371H        ;Mod I PDRIVE table loc
5310 223B54               LD      (PDPTR1+1),HL   ;Store it in program
5313 229253               LD      (INITZ+1),HL
5316 229F53               LD      (PDPTR2+1),HL
5319 213046               LD      HL,4630H        ;Mod I sector read rtne
531C 22DE54               LD      (RSEC+1),HL     ;Store it in program
531F AF                   XOR     A               ;A=0
5320 324A55               LD      (TBL2),A        ;Store in PDRIVE4 table
5323 3E33                 LD      A,33H
5325 324655               LD      (TBL1),A        ;Also in PDRIVE4 table
5328 E1                   POP     HL              ;Restore buffer pointer
                  ;
                  ;       ** Check if Drivespec Given **
                  ;
5329 E5           DRVSPC  PUSH    HL              ;For possible user restrt
532A 7E                   LD      A,(HL)          ;Get char after filespec
532B FE0D                 CP      0DH             ;Drive no. specified?
532D 2836                 JR      Z,DZERO         ;Drive 0 assumed if not
532F E6DF                 AND     0DFH            ;Lowercase = uppercase
5331 FE53                 CP      'S              ;Skip track parameter?
```

```
5333 2815    00560          JR      Z,SKPTRK        ;If skipping tracks
5335 7E      00570          LD      A,(HL)          ;Restore original char
5336 FE3A    00580          CP      3AH             ;Is char a colon?
5338 2002    00590          JR      NZ,NOTCOL       ;If not a colon
533A 23      00600          INC     HL              ;Bump buffer pointer
533B 7E      00610          LD      A,(HL)          ;Get char after colon
533C FE34    00620 NOTCOL   CP      34H             ;Drive No. too high ?
533E 3805    00630          JR      C,NOERR         ;Go if not too high
             00640 ;
             00650 ;        xx Error Exit for Invalid Drivespec xx
             00660 ;
5340 3E20    00670 DRVERR   LD      A,20H           ;Illegal/missing drv code
5342 C30944  00680 ERREXT   JP      4409H           ;Abort with error message
             00690 ;
             00700 ;        xx Check for Drive 0 or Invalid Drivespec xx
             00710 ;
5345 FE30    00720 NOERR    CP      30H             ;Drive 0 ?
5347 38F7    00730          JR      C,DRVERR        ;Error if invalid char
             00740 ;
             00750 ;        xx Check for "S" (Skip Track Argument) xx
             00760 ;
5349 23      00770          INC     HL              ;Bump buffer pointer
534A F5      00780 SKPTRK   PUSH    AF              ;Save drive no., Z flag
534B 7E      00790 LPBACK   LD      A,(HL)          ;Get char at HL
534C FE0D    00800          CP      0DH             ;Was it ENTER?
534E 2812    00810          JR      Z,ENTFND        ;Go if ENTER found
5350 23      00820          INC     HL              ;Bump buffer pointer
5351 FE20    00830          CP      20H             ;Was it a space char?
5353 28F6    00840          JR      Z,LPBACK        ;Try next char if so
5355 E6DF    00850          AND     0DFH            ;Lowercase = uppercase
5357 FE53    00860          CP      'S'             ;Skip track parameter?
5359 3E34    00870          LD      A,34H           ;Error message number
535B 20E5    00880          JR      NZ,ERREXT       ;If not skip trk flag
535D 3E15    00890          LD      A,15H           ;Byte to change
535F 324B55  00900          LD      (T&L3),A        ;Store in PDRIVE4 table
5362 F1      00910 ENTFND   POP     AF              ;Restore drive no./Z flag
5363 2019    00920          JR      NZ,POSTD        ;Go if not drive 0
             00930 ;
             00940 ;        xx Drive Zero Routine xx
             00950 ;
5365 CDC901  00960 DZERO    CALL    01C9H           ;Clear screen
5368 21E054  00970          LD      HL,ZERMSG       ;Point to Drive 0 msg
536B CD6744  00980          CALL    4467H           ;Display msg
536E CD4900  00990 GETKEY   CALL    49H             ;Wait til key pressed
5371 3D      01000          DEC     A               ;Was BREAK key pressed?
5372 2004    01010          JR      NZ,NOABRT       ;Go if no abort
5374 3E39    01020          LD      A,39H           ;Operator abort err code
5376 18CA    01030          JR      ERREXT          ;Display msg and exit
5378 FE0C    01040 NOABRT   CP      0CH             ;Was ENTER pressed?
537A 20F2    01050          JR      NZ,GETKEY       ;Try again if not
537C 3E30    01060          LD      A,30H           ;Drive No. in A reg
             01070 ;
             01080 ;        xx Post the Drivespec xx
             01090 ;
537E 321255  01100 POSTD    LD      (MSG1+6),A      ;Store Ascii Drive No.
5381 D630    01110          SUB     30H             ;Calc. binary drive no.
             01120 ;
             01130 ;        xx Select Drive xx
             01140 ;
5383 CD5E44  01150          CALL    445EH           ;Select & rotate drive
5386 20BA    01160          JR      NZ,ERREXT       ;Err if disk not mounted
5388 CDC901  01170          CALL    01C9H           ;Clear screen
538B 210C55  01180          LD      HL,MSG1         ;Pt to Title
538E CD6744  01190          CALL    4467H           ;Display Title
             01200 ;
             01210 ;xxxxxxxx Swap Mem/TRSDOS PDRIVES xxxxxxxxxxxxxxxxxxxxxxxx
             01220 ;
             01230 ;        xx Store Original PDRIVES xx
             01240 ;
5391 219142  01250 INITZ    LD      HL,4291H        ;Pt to PDRIVES in mem
5394 114E55  01260          LD      DE,PDVSTO       ;Pt to storage area
5397 012800  01270          LD      BC,28H          ;Length of PDRIVE
539A EDB0    01280          LDIR                    ;Xfer it
             01290 ;
             01300 ;        xx Replace each PDRIVE with PDRIVE 4 xx
             01310 ;
539C 0604    01320 PSWAP    LD      B,04H           ;4 drives to PDRIVE
539E 119142  01330 PDPTR2   LD      DE,4291H        ;start of PDRIVE area
53A1 C5      01340 GFLOOP   PUSH    BC              ;Save Drive No.
53A2 214455  01350          LD      HL,PDRIV4       ;TRSDOS PDRIVE 4
53A5 010A00  01360          LD      BC,0AH          ;Length of PDRIVE 4
53A8 EDB0    01370          LDIR                    ;XFER IT
53AA C1      01380          POP     BC              ;Restore drive number
53AB 10F4    01390          DJNZ    GFLOOP          ;Loop for 4 drives
             01400 ;
             01410 ;xxxxxxxx Read a Sector xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
             01420 ;
53AD CDD654  01430          CALL    SREAD           ;Read sector to buffer
             01440 ;
             01450 ;xxxxxxxx Decode and Display GAT Sector xxxxxxxxxxxxxxxx
             01460 ;
             01470 ;        xx Disk Name xx
             01480 ;
53B0 21D052  01490          LD      HL,52D0H        ;Pt to Disk Name
53B3 11153C  01500          LD      DE,3C15H        ;Pt to Screen
53B6 010800  01510          LD      BC,8            ;8 characters in name
53B9 EDB0    01520          LDIR                    ;Xfer name to screen
             01530 ;
             01540 ;        xx Disk Date xx
             01550 ;
53BB 111F3C  01560          LD      DE,3C1FH        ;Pt to screen
53BE 010800  01570          LD      BC,8            ;8 characters in date
53C1 EDB0    01580          LDIR                    ;Xfer to screen
             01590 ;
             01600 ;        xx Free Granules xx
             01610 ;
53C3 110052  01620          LD      DE,5200H        ;Pt to G.A.T.
53C6 210000  01630          LD      HL,0000H        ;Zero Free Grans counter
53C9 0E28    01640          LD      C,40            ;40 tracks assumed
53CB 1A      01650 GLOOP    LD      A,(DE)          ;Get allocation byte
53CC 0606    01660          LD      B,6             ;6 bits to check
53CE 0F      01670 ILOOP    RRCA                    ;Is granule allocted
53CF 3801    01680          JR      C,NOADD         ;Bump count if not
53D1 2C      01690          INC     L               ;Bump free gran counter
53D2 10FA    01700 NOADD    DJNZ    ILOOP           ;Loop for 6 bits
53D4 13      01710 NEXT     INC     DE              ;Bump memory pointer
53D5 0D      01720          DEC     C               ;Decrement track count
53D6 20F3    01730          JR      NZ,GLOOP        ;Loop for 40 tracks
53D8 CD9A0A  01740          CALL    0A9AH           ;No. free grans to ACCUM
53DB CDBD0F  01750          CALL    0FBDH           ;Change to numeric string
53DE 2E30    01760          LD      L,30H           ;HL=4130H = string start
53E0 11293C  01770          LD      DE,3C29H        ;Pt to screen
53E3 010600  01780          LD      BC,6            ;6 figures maximum
53E6 EDB0    01790          LDIR                    ;Xfer free grans
53E8 21313C  01800          LD      HL,3C31H        ;Pt to screen
53EB 222040  01810          LD      (4020H),HL      ;Load cursor with address
53EE 212155  01820          LD      HL,MSG2         ;Pt to free granules msg
53F1 CD6744  01830          CALL    4467H           ;Display msg
             01840 ;
             01850 ;xxxxxxxx Update Sector Number xxxxxxxxxxxxxxxxxxxxxxxxxx
             01860 ;
53F4 AF      01870          XOR     A               ;Zero DIR page count
53F5 214255  01880 UPSECT   LD      HL,SECTOR       ;Relative sector number
53F8 34      01890          INC     (HL)            ;Bump sector number
             01900 ;
             01910 ;xxxxx Loop to Read, Decode & Print each FPDE page xxxxxx
             01920 ;
53F9 3C      01930          INC     A               ;Increment DIR page
53FA FE11    01940          CP      17              ;All read yet ?
53FC 2839    01950          JR      Z,QLOOP         ;Exit if so
53FE F5      01960          PUSH    AF              ;Store DIR page number
53FF 3D      01970          DEC     A               ;Is it HIT sector
5400 2001    01980          JR      NZ,NONO         ;Jump if not
5402 34      01990          INC     (HL)            ;Increment past HIT sect.
5403 CDD654  02000 NONO     CALL    SREAD           ;Read sector to buffer
5406 FD2A4055 02010         LD      IY,(HZPT)       ;Pt to print @ posn
             02020 ;
             02030 ;xxxxxxxx Decode FPDE Page xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
             02040 ;
540A DD210052 02050         LD      IX,5200H        ;Pt to input buffer
540E 210552  02060          LD      HL,5205H        ;Pt to filespec posn
5411 223C55  02070          LD      (SCE),HL        ;Store filespec pointer
5414 0605    02080          LD      B,05H           ;Filespecs per DIR page
5416 C5      02090 DILOOP   PUSH    BC              ;Save FPDE counter
5417 DD7E00  02100          LD      A,(IX+0)        ;Get status byte
541A B7      02110          OR      A               ;Is it an active FPDE
541B 280E    02120          JR      Z,SKIP2         ;Skip if FPDE inactive
541D CD6154  02130          CALL    DISP            ;Go to display routine
5420 013000  02140 SKIP1    LD      BC,0030H        ;Increment to next Fspec
5423 DD09    02150          ADD     IX,BC           ;Update FPDE pointer
```

```
5425 C1       02160          POP     BC              ;Restore FPDE counter
5426 10EE     02170          DJNZ    DILOOP          ;Loop till page done
5428 F1       02180          POP     AF              ;Restore DIR page number
5429 18CA     02190          JR      UPSECT          ;Update sector pointer
542B 2A3C55   02200 SKIP2    LD      HL,(SCE)        ;Pt to filespec
542E 013000   02210          LD      BC,0030H        ;Increment to next Fspec
5431 09       02220          ADD     HL,BC           ;Bump Fspec pointer
5432 223C55   02230          LD      (SCE),HL        ;Store pointer
5435 18E9     02240          JR      SKIP1
              02250 ;
              02260 ;******** Replace PDRIVES and Exit to DOS **********
              02270 ;
5437 214E55   02280 QLOOP    LD      HL,PDVSTO       ;Pt to orig. PDRIVES
543A 119142   02290 PDPTR1   LD      DE,4291H        ;Pt to PDRIVE area in RAM
543D 012800   02300          LD      BC,28H          ;Length of PDRIVE
5440 EDB0     02310          LDIR                    ;Replace orig. PDRIVES
5442 21403F   02320          LD      HL,3F40H        ;Pt to 3rd bottom line
5445 222040   02330          LD      (4020H),HL      ;Store as cursor posn
5448 CD4900   02340          CALL    49H             ;Hit a key to continue
544B E6DF     02350          AND     0DFH            ;Lowercase=uppercase
544D FE52     02360          CP      'R'             ;Repeat ?
544F C22D40   02370          JP      NZ,402DH        ;Exit to DOS if no repeat
5452 213055   02380          LD      HL,ITLVAL       ;HL=reinitialization tbl
5455 113A55   02390          LD      DE,PCOUNT       ;DE=area to reinitialize
5458 010C00   02400          LD      BC,12           ;No. bytes to move
545B EDB0     02410          LDIR                    ;Re-initialize program
545D E1       02420          POP     HL              ;Restore input buffer ptr
545E C32953   02430          JP      DRVSPC
              02440 ;
              02450 ;******* Display an Active Filespec **********************
              02460 ;
5461 2A3C55   02470 DISP     LD      HL,(SCE)        ;Pt to filespec
5464 CDA754   02480          CALL    SCPOS           ;Screen posn check
5467 0608     02490          LD      B,8             ;8 characters in Fspec
5469 7E       02500 DELOOP   LD      A,(HL)          ;Get ASCII character
546A FE20     02510          CP      20H             ;Is it a space ?
546C 2802     02520          JR      Z,PASS          ;Skip if so
546E 12       02530          LD      (DE),A          ;Display if not a space
546F 13       02540          INC     DE              ;Bump screen pointer
5470 23       02550 PASS     INC     HL              ;Bump pointer
5471 10F6     02560          DJNZ    DELOOP          ;Loop to end of filename
5473 7E       02570          LD      A,(HL)          ;Get next character
5474 FE20     02580          CP      20H             ;Is it a space ?
5476 2804     02590          JR      Z,MISS          ;No extent if so
5478 3E2F     02600          LD      A,'/'           ;Add delimeter
547A 12       02610          LD      (DE),A          ;Display a slash
547B 13       02620          INC     DE              ;Bump screen pointer
547C 010300   02630 MISS     LD      BC,3            ;Length of extent
547F EDB0     02640          LDIR                    ;Display extent
5481 3A3A55   02650          LD      A,(PCOUNT)      ;Get program counter
5484 3C       02660          INC     A               ;Increment program number
5485 323A55   02670          LD      (PCOUNT),A      ;Store counter
5488 FE4B     02680          CP      4BH             ;75 programs displayed ?
548A 280B     02690          JR      Z,SCROLL        ;Scroll screen up if so
548C FE50     02700          CP      50H             ;Last FPDE has files ?
548E C8       02710          RET     Z               ;Modify print if so
548F 012500   02720          LD      BC,25H          ;Screen increment
5492 09       02730          ADD     HL,BC           ;Calc. new mem pointer
5493 223C55   02740          LD      (SCE),HL        ;Store mem pointer
5496 C9       02750          RET
5497 21C03F   02760 SCROLL   LD      HL,3FC0H        ;Last line of display
549A 223E55   02770          LD      (VIDEO),HL      ;Store it
549D 224055   02780          LD      (HZPT),HL       ;Redefine screen pointer
54A0 AF       02790          XOR     A               ;Zero posn in line
54A1 323B55   02800          LD      (SPOINT),A      ;Store as first posn
54A4 C38705   02810 GOSCRL   JP      0587H           ;Scroll up
54A7 3A3B55   02820 SCPOS    LD      A,(SPOINT)      ;Print posn
54AA FD2A4055 02830          LD      IY,(HZPT)       ;Get print @ posn
54AE B7       02840          OR      A               ;First file position ?
54AF 2809     02850          JR      Z,GETOUT        ;Exit if so
54B1 FE05     02860          CP      05H             ;Last file position ?
54B3 2811     02870          JR      Z,CR            ;Cr if so
54B5 010D00   02880          LD      BC,13           ;Increment
54B8 FD09     02890          ADD     IY,BC           ;Calc. new pointer
54BA 3C       02900 GETOUT   INC     A               ;Bump screen/file posn
54BB 323B55   02910          LD      (SPOINT),A      ;Store file position
54BE FD224055 02920          LD      (HZPT),IY       ;Store print @ posn
54C2 FDE5     02930          PUSH    IY              ;Move print @ posn
54C4 D1       02940          POP     DE              ;to DE register
54C5 C9       02950          RET

54C6 014000   02960 CR       LD      BC,40H          ;1 line of video
54C9 FD2A3E55 02970          LD      IY,(VIDEO)      ;Get current line
54CD FD09     02980          ADD     IY,BC           ;Calc. new video line
54CF FD223E55 02990          LD      (VIDEO),IY      ;Store new line reference
54D3 AF       03000          XOR     A               ;First file position
54D4 18E4     03010          JR      GETOUT
              03020 ;
              03030 ;******** Read a Sector **********************************
              03040 ;
54D6 210052   03050 SREAD    LD      HL,5200H        ;Input buffer
54D9 ED5B4255 03060          LD      DE,(SECTOR)     ;Sector to read
54DD C3DB45   03070 RSEC     JP      45DBH           ;Read sector
              03080 ;
              03090 ;******** Strings to Display to Screen *******************
              03100 ;
54E0 0A       03110 ZERMSG   DEFB    0AH
54E1 50       03120          DEFM    'Press ENTER when TRSDOS disk is in Drive 0'
    72 65 73 73 20 45 4E 54 45 52 20 77 68 65 6E 20
    54 52 53 44 4F 53 20 64 69 73 6B 20 69 73 20 69
    6E 20 44 72 69 76 65 20 30
550B 0D       03130          DEFB    0DH
550C 44       03140 MSG1     DEFM    'Drive 0  TRSDOS --> '
    72 69 76 65 20 30 20 20 54 52 53 44 4F 53 20 2D
    2D 3E 20
5520 03       03150          DEFB    03H
5521 46       03160 MSG2     DEFM    'Free Granules'
    72 65 65 20 47 72 61 6E 75 6C 65 73
552E 0A       03170          DEFB    0AH
552F 0D       03180          DEFB    0DH
              03190 ;
              03200 ;***** Initialization Values for Storage Locations ******
              03210 ;
5530 0000     03220 ITLVAL   DEFW    0000H
5532 0552     03230          DEFW    5205H           ;Filespec pointer
5534 403C     03240          DEFW    3C40H           ;Screen pointer
5536 403C     03250          DEFW    3C40H           ;Screen pointer
5538 3201     03260          DEFW    0132H
              03270 ;
              03280 ;******** Constants, Counters and Storage ***************
              03290 ;
553A 00       03300 PCOUNT   NOP
553B 00       03310 SPOINT   NOP
553C 0552     03320 SCE      DEFW    5205H           ;Filespec pointer
553E 403C     03330 VIDEO    DEFW    3C40H           ;Screen pointer
5540 403C     03340 HZPT     DEFW    3C40H           ;Screen pointer
5542 3201     03350 SECTOR   DEFW    0132H
5544 1128     03360 PDRIV4   DEFW    2811H           ;PDRIVE 4 = TRSDOS
5546 27       03370 TBL1     DEFB    27H
5547 28       03380          DEFB    28H
5548 1206     03390          DEFW    0612H
554A FE       03400 TBL2     DEFB    0FEH
554B 11       03410 TBL3     DEFB    11H
554C 1102     03420          DEFW    0211H
0028          03430 PDVSTO   DEFS    28H             ;Store PDRIVE here
5300          03440          END     START
00000 TOTAL ERRORS
```

CR	54C6	DELOOP	5469	DILOOP	5416	DISP	5461	DRVERR	5340
DRVSPC	5329	DZERO	5365	ENTFND	5362	ERREXT	5342	GETKEY	536E
GETOUT	54BA	GOSCRL	54A4	HZPT	5540	ILOOP	53CE	INITZ	5391
ITLVAL	5530	LPBACK	534B	MISS	547C	MSG1	550C	MSG2	5521
NEXT	53D4	NOABRT	5378	NOADD	53D2	NOERR	5345	NONO	5403
NOTCOL	533C	QLOOP	53CB	PASS	5470	PCOUNT	553A	PDPTR1	543A
PDPTR2	539E	PDRIV4	5544	PDVSTO	554E	POSTD	537E	PSWAP	539C
QFLOOP	53A1	QLOOP	5437	RSEC	54DD	SCE	553C	SCPOS	54A7
SCROLL	5497	SECTOR	5542	SKIP1	5420	SKIP2	542B	SKPTRK	534A
SPOINT	553B	SREAD	54D6	START	5300	TBL1	5546	TBL2	554A
TBL3	554B	UPSECT	53F5	VIDEO	553E	ZERMSG	54E0		

UN-DISK DEPARTMENT - I recently received a letter from Winfield Smith of Chicago, Illinois, who advises me that the single sheet carrier for tractor printers that I mentioned in the last issue of Northern Bytes has an inexpensive competitor, that's made right here in the Great Lakes State. It's "The Original PAPER PORTER", available from Beeline Services, Otsego, Michigan 49078. Winfield says that they sell for $3.25, or at least they did a year ago when he bought several by mail. He adds in his letter that "PAPER PORTER is a plastic sheet that fits 9 1/2" sprockets and takes anything up to a legal-sized sheet. The pocket at the top is 1", which may mask the top of some things like checks, although I should think one could out part of the pocket to accommodate the space normally used for the date. Like the product you mentioned, it isn't the sort of thing you would want to use for volume printing, but it does let you do the occasional sheet on your old letterhead, etc."

Winfield also says that he is a devotee of Un-Disk operation. He owns two Exatron Stringy-Floppy units (he has the second one wired so that he can use it either as a stand-alone with his second Model I, or as Drive 1 with his main Model I. He's particularly interested in "things that work well in Un-Disk systems, whether or not they're equipped with an ESF", and figures that there must be lots of others that are also interested in Un-Disk programming. He asks if I have any suggestions on how to bring them together. Actually, I don't, unless you want to do what Winfield has done and drop me a line. In the meantime, if you're also an "Un-Disk" devotee, you can contact Winfield Smith by writing him at 5825 South Blackstone Avenue, Chicago, Illinois 60637, or phone him at (312) 684-3609

For the benefit of Winfield and the rest of you ESF users, the following article may prove to be of interest to you!

PATCH FOR STRINGY FLOPPY TO TASMON by Dan Poorman

Between disk and cassette I/O lies the Exatron Stringy Floppy. The ESF or stringy, as it is known, is a high speed tape system that uses tiny wafers to read and write data and programs.

Although the ESF was inexpensive, highly reliable, and easy to use, it suffers from an identity crisis. Very few people know about it. The situation probably won't get any better with Exatron no longer selling the hardware for the TRS-80. But a new company called A & J Micro Drive Sales and Service will be selling wafers and drives now, so maybe all is not lost. The future of the TRS-80 stringy is apparently up in the air - if there is a future.

But those of us who have stringies love them. Still, there is a problem. What with most of the more experienced programmers opting for the higher priced disk, stringy I/O isn't considered. TASMON is a perfect example. It's a fantastic monitor utility distributed by The Alternate Source that includes among it many features reading and writing of both disk and cassette. Yet, up until now, stringy users were left out.

ESFPAT will allow TASMON version 2.12 to read and write stringies. It is for a Model I tape system with at least 32K of memory. Actually, with the amount of memory space TASMON needs, I doubt if anyone uses it on a 16K machine. But the ESFPAT program can be modified to fit into a smaller machine by changing its origin point. As listed, it is set higher than necessary to resolve a conflict with yet another TASMON patch I use.

Type in the accompanying program using either TASMON's (M) command to input the object code, or type the source code into an Editor-Assembler to make an object tape.

Load TASMON from tape being sure that it resides at 6000H. If TASMON is someplace different, use its relocate command (X) to move it to 6000H for the patch. After being patched, TASMON may be moved wherever you would like.

If you have used an Editor-Assembler to assemble the ESFPAT source code, load TASMON and without executing it, load the ESFPAT program from tape. When the SYSTEM prompt again shows, type "/" and ENTER. That causes the patch to execute. If you have used TASMON to input just the object code, you will have to tell TASMON to (G)o to 8175H to activate the patch.

To save your patched version of TASMON, press (W) but instead of selecting the (T) for tape as you normally would, press (S) and a "#" symbol will appear. It is asking you for the ESF file number. Input the number and then the hex starting address, hex ending address, and hex execute point. For patched TASMON as file 1, you would input 1 6000 8174 6000. The ESF will start automatically and return you to the TASMON right arrow prompt. ESFPAT does not check or report stringy I/O errors, if any. After a write, control is returned to TASMON.

Reading a stringy is not quite so easy due to the propensity of the ESF software to want to autostart a program. If you want to load a program using TASMON and have it execute, fine. Simply type the command (L) and instead of your usual (T) selection for tape, type (S) for stringy. Again, a "#" will appear indicating that you should input the file number of the file you want. A file number is mandatory.

If you want the program to auto-execute, just sit back and watch it load. But, if you want it to load into memory without executing, hold down the SHIFT key until the wafer stops. You will see the familiar decimal addresses listing starting address, program length, and execute or auto-start point. The normal "FD Error" will be reported and the BASIC "READY" prompt will be there. You (actually, your TRS-80) are in BASIC. To get back to TASMON, type the command (CMD). You will see the right arrow appear for the TASMON command mode prompt and TASMON is awaiting your next move.

If you are using disk, you'll have to know where TASMON resides in memory and jump back to it using the SYSTEM, "/", and decimal starting address sequence. You'll also have to modify the ESFPAT program by zeroing memory locations 80DCH through 80E9H by changing them to NOP's. This eliminates the patch to the disk BASIC "CMD" that non-disk users need to get back into TASMON.

TASMON, by the way, loads in 11 seconds from stringy as opposed to 2 minutes and 37 seconds from tape.

ESFPAT will not allow writing of Editor-Assembler source files from TASMON - only memory dumps of programs. It also won't recognize offsets from the actual memory location written on a wafer, and there is no check to be sure that you are not overwriting TASMON itself. Be sure TASMON is in some out of the way spot in high memory. I keep mine at D000H which seems to be a good location that does not conflict with other high memory utilities and yet is out of the way of most programs.

The (W) and (L) commands of TASMON function normally when you are writing to tape or disk.

```
                00000 ;              xxxxx  ESFPAT  xxxxx
                00010 ;
                00020 ; WRITE AND READ STRINGY FOR TASMON 2.12
                00030 ; TASMON WRITTEN BY BRUCE HANSON
                00040 ; DISTRIBUTED BY THE ALTERNATE SOURCE
                00050 ; STRINGY PATCH WRITTEN BY DAN POORMAN
                00060 ;       JUNE 18, 1983   1600 HRS
                00070 ;
                00080 ;
                00090 ;
6000            00100 TASMON  EQU   6000H   ;TASMON COLD START
3000            00110 BOT     EQU   3000H   ;FIND BEGINNING OF TAPE
300C            00120 WDAT    EQU   300CH   ;WRITE IT TO ESF
300F            00130 BFILE   EQU   300FH   ;MOVE TO BEGIN OF FILE
300C            00140 WDAT    EQU   300CH   ;WRITE IT
60EE            00150 BREAK   EQU   60EEH   ;BREAK WHEN TASMON @6000H
717A            00160 SPC     EQU   717AH   ;WRITE SPACE TO VIDEO
6543            00170 GETHEX  EQU   6543H   ;FOUR DIGIT VALUE RETS HL
6586            00180 HEXINP  EQU   6586H   ;INPUT SINGLE HEX
695C            00190 WL      EQU   695CH   ;WRT CHR AND SPC
79BA            00200 WR      EQU   79BAH   ;WRT A TO VIDEO
6539            00210 CRRET   EQU   6539H   ;CARRIAGE RETURN
7E48            00220 LO      EQU   7E48H   ;END OF PROGRAM
7E4A            00230 HIGH    EQU   7E4AH   ;BEG OF PRG
7E4C            00240 TADD    EQU   7E4CH   ;TRSFR ADDR
76DC            00250 GETPAR  EQU   76DCH   ;TAPE PARAMETERS
6531            00260 DISHEX  EQU   6531H   ;MAKE IT BINARY
7413            00270 KY1     EQU   7413H   ;VECTOR FROM LOAD
74E6            00280 KY2     EQU   74E6H   ;VECTOR FROM WRITE
60E3            00290 KYINP   EQU   60E3H   ;GET A CHARACTER
4016            00300 KBDCB   EQU   4016H   ;KBD DEVICE CONTROL BLOCK
4173            00310 CMD     EQU   4173H   ;DOS LINK FOR CMD
69AB            00320 CLS     EQU   69ABH   ;TASMON CLS
                00330 ;
                00340 ;
                00350 ;
80D4            00360         ORG   80D4H
                00370 ;
0002            00380 LENHEX  DEFS  2       ;LENGTH OF FILE STORAGE
0002            00390 FILSAV  DEFS  2       ;FILE NUMBER IN BINARY
                00400 ;
80D8 1829       00410         JR    WPTCH
80DA 1812       00420         JR    LPTCH
                00430 ;
```

```
                00440 ;       xx MAKE CMD JUMP TO TASMON AFTER COLD START xx
                00450 ;
80DC C3EE60     00460 BREAK1  JP      BREAK   ;TASMON BREAK FOR CMD
                00470 ;
80DF 21DC80     00480 CMDLNK  LD      HL,BREAK1   ;TASMON BREAK
80E2 117341     00490         LD      DE,CMD      ;DOS VECTOR FOR CMD
80E5 010300     00500         LD      BC,03
80E8 EDB0       00510         LDIR
80EA CDAB69     00520         CALL    CLS         ;TASMON CLEAR SCREEN
80ED C9         00530         RET
                00540 ;
                00550 ;       xx  PATCH FOR LOAD xx
                00560 ;
80EE CDE360     00570 LPTCH   CALL    KYINP   ;GET A KEY
80F1 FE53       00580         CP      'S'     ; IS IT S
80F3 CA5681     00590         JP      Z,STRNGY ;IF SO, DO STRINGY
80F6 FE54       00600         CP      'T'     ;IF T THEN BACK TO TASMON
80F8 2806       00610         JR      Z,GBK
80FA FE44       00620         CP      'D'     ;IF D THEN BACK TO TASMON
80FC 2802       00630         JR      Z,GBK
80FE 18EE       00640         JR      LPTCH   ;NO VALID CMD SO GET ANOTHER
8100 C31674     00650 GBK     JP      KY1+3   ; GO BACK TO TASMON ROUTINE
                00660 ;
                00670 ;       xx  PATCH FOR WRITE xx
                00680 ;
8103 CDE360     00690 WPTCH   CALL    KYINP   ;GET KEY
8106 FE53       00700         CP      'S'     ;IS IT S
8108 280D       00710         JR      Z,WRITE ;YES, WRITE A STRINGY
810A FE54       00720         CP      'T'
810C 2806       00730         JR      Z,WGBK  ;GO BACK TO TASMON
810E FE44       00740         CP      'D'
8110 2802       00750         JR      Z,WGBK
8112 18EF       00760         JR      WPTCH   ;NO VALID CMD SO GET ANOTHER
8114 C3E974     00770 WGBK    JP      KY2+3   ;BACK INTO TASMON
                00780 ;
                00790 ;       xx WRITE ROUTINE xx
                00800 ;
8117 CDBA79     00810 WRITE   CALL    WR      ;PUT W ON SCREEN
811A 3E23       00820         LD      A,'#'   ;WRITE A FILE SYMBOL
811C CD5C69     00830         CALL    WL      ;WRITE TO SCREEN
811F CD8665     00840 GETIT   CALL    HEXINP  ;GET A HEX NUMBER
8122 2802       00850         JR      Z,FNUM  ;IF IT IS CONTINUE
8124 18F9       00860         JR      GETIT   ;NO HEX SO GET IT AGAIN
8126 F5         00870 FNUM    PUSH    AF
8127 CDBA79     00880         CALL    WR      ;WRITE ASCII  FILE NUM TO SCREEN
812A F1         00890         POP     AF
812B D630       00900         SUB     30H     ;MAKE IT A FILE NUMBER
812D 32D680     00910         LD      (FILSAV),A      ;STORE FILE NUMBER
8130 CDDC76     00920         CALL    GETPAR  ;GET ESF PARAMATERS
8133 CD3965     00930         CALL    CRRET   ;MOVE CURSOR DOWN ONE
8136 ED5B4A7E   00940         LD      DE,(HIGH)       ;STORE PARAMETERS
813A 2A487E     00950         LD      HL,(LO)
813D ED52       00960         SBC     HL,DE   ;SUBTRACT THE TWO ANS IN HL
813F 22D480     00970         LD      (LENHEX),HL     ;STORE LENGTH
8142 2A4A7E     00980         LD      HL,(HIGH)       ;START OF PRGRM
8145 ED4BD480   00990         LD      BC,(LENHEX)     ;LENGTH OF PRGRM
8149 ED5B4C7E   01000         LD      DE,(TADD)       ;TRANSFER ADDR
814D 3AD680     01010         LD      A,(FILSAV)      ;FILE NUMBER
8150 CD0C30     01020         CALL    WDAT            ;WRITE THE STRINGY
8153 C3EE60     01030         JP      60EEH           ;TASMON BRK
                01040 ;
                01050 ;       xx   READ ROUTINE  xx
                01060 ;
8156 CDBA79     01070 STRNGY  CALL    WR
8159 3E23       01080         LD      A,'#'           ;FILE SYMBOL
815B CD5C69     01090         CALL    WL              ;PUT IT ONTO SCREEN
815E CD8665     01100 HEXAGN  CALL    HEXINP          ;GET A HEX FILE NUM
8161 2802       01110         JR      Z,FNUM1         ;IF HEX GO ON
8163 18F9       01120         JR      HEXAGN
8165 F5         01130 FNUM1   PUSH    AF
8166 CDBA79     01140         CALL    WR              ;PUT  FILE NUM ON SCREEN
8169 F1         01150         POP     AF
816A D630       01160         SUB     30H             ;MAKE IT BINARY
816C CD0F30     01170         CALL    BFILE           ;POSITION BEGIN OF FILE
816F CD5531     01180         CALL    3155H           ;@LOAD
8172 C3EE60     01190         JP      BREAK           ;GOTO TASMON
                01200 ;
                01210 ;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                01220 ;x                                                              x
                01230 ;x      ONCE ESFPAT IS RUN FOLLOWING CODE NOT                   x
                01240 ;x      NEEDED AND IS NOT SAVED ONTO THE FINAL                  x
                01250 ;x      WAFER CONTAINING TASMON ESF                             x
                01260 ;x                                                              x
                01270 ;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                01280 ;
                01290 ;       xx PATCHES TO TASMON @ 6000H xx
                01300 ;
8175 219981     01310 INIT    LD      HL,WPTCH1       ;CODE FOR PATCH
8178 11E674     01320         LD      DE,74E6H        ;TASMON WRITE
817B 010300     01330         LD      BC,03           ;THREE BYTES
817E EDB0       01340         LDIR                    ;DOIT
8180 219C81     01350         LD      HL,LPTCH1       ;TASMON READ
8183 111374     01360         LD      DE,7413H        ;TASMON READ
8186 010300     01370         LD      BC,03           ;THREE BYTES WORTH
8189 EDB0       01380         LDIR
818B 219F81     01390         LD      HL,CMD1 ;PATCH CMD CALL
818E 110160     01400         LD      DE,6001H ;TASMON START + 1
8191 010300     01410         LD      BC,03
8194 EDB0       01420         LDIR
8196 C36066     01430         JP      TASMON  ;TASMON COLD START
                01440 ;
8199 C30880     01450 WPTCH1  JP      80D8H   ;WRITE LINK
819C C3DA80     01460 LPTCH1  JP      80DAH   ;READ LINK
819F CDDF80     01470 CMD1    CALL    CMDLNK  ;MAKD CMD JP TO STRINGY
                01480 ;
8175            01490         END     INIT    ;INITALIZE
00000 TOTAL ERRORS
```

```
BFILE   300F    BOT     3000    BREAK   60EE    BREAK1  80DC    CLS     69AB
CMD     4173    CMD1    819F    CMDLNK  80DF    CRRET   6539    DISHEX  6531
FILSAV  80D6    FNUM    8126    FNUM1   8165    GBK     8100    GETHEX  6543
GETIT   811F    GETPAR  76DC    HEXAGN  815E    HEXINP  6586    HIGH    7E4A
INIT    8175    KBDCB   4016    KY1     7413    KY2     74E6    KYINP   60E3
LENHEX  80D4    LO      7E48    LPTCH   80EE    LPTCH1  819C    SPC     717A
STRNGY  8156    TADD    7E4C    TASMON  6000    WDAT    300C    WGBK    8114
WL      695C    WPTCH   8103    WPTCH1  8199    WR      79BA    WRITE   8117
```

MODEL 4 MISCELLANEOUS is excerpted from the Marin County TRS-80 Users Group (MCTUG) newsletter:

To set formfeeds on the Model 4 in TRSDOS 6.x use the following:

SET *FF FORMS/FLT <ENTER>
FILTER *PR *FF <ENTER>
FORMS (LINES=56, PAGE=66) <ENTER>

You naturally can set the number of lines and page size to what you require.

MBASIC on the Model 4 has four undocumented functions: MOD, XOR, EQV, and IMP.

You may experience difficulty with Model III to 4 conversions because of syntax errors generated by the use of the semicolon after PRINT USING. The TRSDOS 6 manual on page 2-150 (2-141 in later manuals) shows an example:

20 PRINT USING "!";A$;

This and all other uses of the semicolon after PRINT USING fails to work. Changing to a comma allows the line to execute, but of course advances the cursor.

TRSDOS 6.x comes with a CLICK/FLT that gives a simulated keyclick when a key is pressed. To make it sound a little more like a click and also to make it a little easier to hear over the real keyboard typing, use the following patch:

PATCH CLICK/FLT.FILTER (D00,7E=04 42:F00,7E=05 00)

The hex code at 05 controls the length and the code at 00 controls the pitch if you would like to experiment.

The following patch by Roy Soltoff should cure the problem of crashes during long sessions with COMM in TRSDOS 6.x:

```
. Patch to 6.0 COMM/CMD to possibly correct "crashes"
. Developed by Roy Soltoff 11/12/83
. (not in conjunction with LSI)
. PATCH COMM/CMD.UTILITY using ...
D05,0D=DD 34
F05,0D=18  35
D05,26=ED  5F F3 F5 CD 18 35 E1 F5 CB 55 28 01 FB F1 C9
F05,26=00  DD E5 E1 AF ED 52 11 39 3E 19 01 04 00 11 58
```

LINE EDIT FUNCTION & DEF FN by Bill Brown is reprinted from ENERGY (the Michigan Computer Consortium Magazine):

The following is a short demonstration program for a DISK BASIC function that can be used as a line editor within a program like the MESSGEN program offered in the Connection-80 bulletin board download section or in the Connection-80 message section software itself. It operates on a string of characters to be edited by replacing a portion of it with another string and returning the corrected string.

The function has three arguments:

A$ -- is the string to be edited.
B$ -- is the sub-portion of A$ that is to be "corrected".
C$ -- is the replacement for B$.

```
10 CLEAR 1000
20 DEF FN RP$(A$,B$,C$) =
     LEFT$(A$,INSTR(A$,B$)-1*(-(INSTR(A$,B$)<>0))) +
     LEFT$(C$,LEN(C$)*(-(INSTR(A$,B$)<>0ANDB$<>""))) +
     RIGHT$(A$,(LEN(A$)-INSTR(A$,B$) -
     LEN(B$)+1)*(-(INSTR(A$,B$)<>0))) +
     LEFT$(A$,LEN(A$)*(-(INSTR(A$,B$)=0)))
30 A$="THIS IS THE BADD LINE.":B$="BADD":C$="GOOD"
40 A$=FN RP$(A$,B$,C$): PRINT A$
```

In an actual program, input would be taken from the keyboard for B$ and C$, then the function called. Note the variable names in the definition of the function (A$, B$, C$) do not have to be the same as those used in the call to the function. That is, lines 30 and 40 above could be changed to:

```
30 X$="THIS IS THE BADD LINE.":Y$="BADD":Z$="GOOD"
40 X$=FN RP$(X$,Y$,Z$): PRINT Z$
```

and the function call would work fine on those variables.

All the complicated Boolean operations in the function definition are for purposes of "error checking". As the function is set up, it assumes that A$ has a string value of length greater than zero. If B$ cannot be found in A$ or if B$ is null, the original string will be returned. C$ can be input as a null string, in order to delete B$ from A$.

An explanation of parts of the function:

If we were not doing error checking, that is, were sure that B$ would always be found in A$, the function definition could read:

```
20 DEF FN RP$(A$,B$,C$) = LEFT$(A$,INTR(A$,B$)-1) + C$ +
     RIGHT$(A$,(LEN(A$)-INSTR(A$,B$)-LEN(B$)+1)
```

The LEFT$ function gets the "good stuff" to the left of where B$ is found. C$ adds in the replacement. The RIGHT$ function gets the "good stuff" to the right of where B$ is found. When added all together, the function returns the corrected line.

In the actual function definition above, the "good stuff to the left" only gets added in when B$ has been found:

```
LEFT$(A$,INSTR(A$,B$)-1*(-(INSTR(A,B$)<>0)))+
```

Note that the Boolean operation "INSTR(A$,B$)<>0" will equal zero (false) when B$ is not found and -1 (true) when it is found. Negating this value will give zero or one. When that is multiplied times the starting position of B$ in A$ minus one (INSTR(A$,B$)-1), which gives the number of characters to take from the left of A$ to get the "good stuff", it will cause either that many characters to be taken (true) or no characters to be taken (false). It took me a long time to figure out how to get this right, so don't be discouraged if it doesn't jump right out at you on the first reading. The logic of the use of the Boolean opertions in the remaining portions of the function is pretty much the same.

C$ only gets added in when B$ has been found AND B$ is not null:

```
LEFT$(C$,LEN(C$)*(-(INSTR(A$,B$)<>0ANDB$<>""))+
```

The "good stuff" on the right gets added in only when B$ has been found:

```
RIGHT$(A$,(LEN(A$)-INSTR(A$,B$)-
LEN(B$)+1)*(-(INSTR(A$,B$)<>0)))+
```

Note that "(LEN(A$)-INSTR(A$,B$)-LEN(B$)+1)" is all required to get the number of characters of "good stuff on the right".

The actual function definition has a fourth element:

```
LEFT$(A$,LEN(A$)*(-(INSTR(A$,B$)=0)))
```

which causes the original value of A$ to be used when B$ is not found in A$.

VISICALC FOR THE MODEL I AND MODEL III is reprinted from DIME:

Version VC-150Y0-T83 for the Model III, sometimes referred to as the enhanced version because a few additional boolean algebraic expressions are included, may be run on the Model I under NEWDOS/80 Version 2.

The zaps shown below allow the TRSDOS 1.2 Model III VisiCalc Version VC-150Y0-T83 to operate with the Model I NEWDOS/80 Version 2.

The directory search capability of VisiCalc is disabled as NEWDOS/80 does not have the RAMDIR facility that Model III TRSDOS does. However, it is possible to use MINI-DOS to search the directory by (1) typing in the /SL command, (2) pressing DFG to enter MINI-DOS, (3) perform the MINI-DOS functions, (4) clear the display, (5) exit from MINI-DOS, (6) back in VisiCalc, press three or more CLEARs to clear the command state, (7) executing one of the /T commands to restore the VisiCalc display.

```
VC/CMD,86,CA change  E5 CD 90 42 FD
                  to E5 3E 08 B7 FD
VC/CMD,00,1E change  2A 11 44 7D
                  to 2A 49 40 7D
VC/CMD,00,AA change  2A 11 44 11
                  to 2A 49 40 11
VC/CMD,84,CB change  C9 32 03 42 CD
                  to C9 00 00 00 CD
VC/CMD,83,9D change
F5 C5 D5 E5 CD 8D 02 28 0A CD 2D A4 E1 D1 C1 F1 to
F5 3A 40 38 E6 04 CC 35 A4 C4 2D A4 28 0B 00 F1
VC/CMD,84,C2 change  E5 CD 5A 00
                  to E5 00 00 00
VC/CMD,85,05 change  E5 CD 55 00
                  to E5 00 00 00
VC/CMD,84,B8 change  FF 32 FA 41 DD
                  to FF 00 00 00 DD
```

Note: ZAP 011, issued by Apparat 08/04/81 is mandatory to run VisiCalc under NEWDOS/80, for both Version 1.20Z for the Model I and the above Version for the Model III.

ASSISTANCE WANTED - Can you help any of these folks? If so, write to them directly:

Ida Ruth Davis, 3245 North Gypsy, Tucson, Arizona 85705 needs help with the Radio Shack Digitizer. Write or call collect to (602) 888-0467 (ask for Ida Ruth) if you can help.

Paul C. Hession, 3843 Timothy Trail, Lafayette, Indiana 47905 is interested in receiving public domain programs. His main interest is in the field of education, specifically industrial arts, so he is especially interested in obtaining TRS-80 programs that are appropriate for those areas.

Tim Donnelly, President of the Sudbury and District Colour Computer Club, P.O. Box 2163, Station A, Sudbury, Ontario P3A 5J3, CANADA would like to contact and exchange public domain software with other CoCo user groups.

Steve Hartmann, 1608-2 Aurelius, Holt, Michigan 48842 has one Model I CRT and one Model III CRT he'd like to sell for $25 each or best offer. Both CRTs are black-and-white.

Does anyone know if a Radio-Immuno Assay (RIA) program exists? (Say WHAT???) If so, Sherry Huey would like to know about it. You can reach her at (517) 353-5282 (or 337-1434). If you don't want to call, drop a line to The Alternate Source, 704 North Pennsylvania, Lansing, Michigan 48906 and we'll pass the info along to Sherry.

Do you read the Dutch language? Would you be interested in occasionally helping translate TRS-80 related articles from Dutch to English, for publication in NORTHERN BYTES? The articles in question appear in the publication of a TRS-80 user group in the Netherlands. What we need is someone that can tell us what subject matter is covered in the various articles, and then if we decide we'd like to take a closer look at one, translate it into

English for us. This could be accomplished by typing it into a word processor, saving it onto disk and sending the disk to us, or by dictating the translation onto a cassette tape (we would supply the disk or tape and return postage, of course). The pay is lousy, but at least you'll be made famous (imagine actually having your name in Bytes - NORTHERN BYTES, that is)! Please don't reply unless you think you would be willing to do it, without having to be prodded along to get it done!

UPPERCASE FUNCTION FOR MODELS I & III TRS-80 by Jack Decker (this article originally appeared in the April, 1981 issue of NORTHERN BYTES):

The routine below adds a new function to BASIC by way of the USR function. Any syntax that would be legal for other string functions will work, such as Z$=USR(Z$), X$=USR(Y$+" "+Z$), PRINT USR(Q$), IF USR(B$)>USR(C$)..., etc. The result of using this function is that all lowercase characters (from ASCII 96 decimal to ASCII 127 decimal) are converted to their uppercase equivalents. All other characters in the string (including graphics and space compression characters) are unaffected. Note that the original string is unchanged unless a statement such as X$=USR(X$) is used.

The BASIC program shown below is for Level II or Disk BASIC. Enter only the lines shown below to start with and RUN - lines 20 through 40 pack the machine language routine into line 10 (A$) and then delete themselves. You may then write the remainder of your program around line 10. You may include additional statements in line 10, but once lines 20 through 40 have been executed and deleted you cannot EDIT line 10. You may EDIT other lines, or add, delete, or renumber lines as you wish. Both variables in line 10 (A$ and X) may be reused elsewhere in the program to save memory.

This routines uses two ROM calls: 0AF4H (which gives a Type Mismatch error if a string was not used as an argument), and 2A68H (part of the LEFT$ - RIGHT$ - MID$ functions, it's being used here to create a copy of the argument string, so that the original string will be left unchanged). For the benefit of those that may wish to experiment with the routine at 2A68H, the B register contains the maximum string length on entry (either the value in the B register or the actual length of the string, whichever is shorter, will be used). B is set to 0FFH here, the maximum string length. The C register indicates the number of characters at the beginning of the string that should be ignored, it's set to zero here because we want to copy the entire string. Setting the C register to any value higher than zero could produce undesired effects unless you are careful - for example, if you have a ten character string and the C register contains 5, the routine would ignore the first five characters of the string, and copy the ten characters following. In this case, you'd get the last five characters of the string, plus the whatever five bytes are located just beyond the string in memory. The exception to this would be if the B register were set to a low enough value (5 or less in this case) to prevent picking up any "garbage" characters. On exit from the routine, the VARPTR of the newly-created string is stored in the arithmetic buffer at 4121H-4122H. These memory locations and the ROM calls mentioned above are the same on the Model I and the Model III, so this routine should run on either Model. Here's the routine:

```
1 'Lowercase to uppercase conversion using USR function
2 'by Jack Decker
10 A$="12345678901234567890123456789012": X=VARPTR(A$)+1:
X=X+(X>32767)*65536: IF PEEK(16809)=201 THEN POKE
16526,PEEK(X): POKE 16527,PEEK(X+1) ELSE
X=PEEK(X)+PEEK(X+1)*256: DEFUSR=X+(X>32767)*65536
20 IF PEEK(16809)=201 THEN X=PEEK(X)+PEEK(X+1)*256
30 X=X+(X>32767)*65536: FOR Y=X TO X+31: READ Z: POKE Y,Z:
NEXT: DELETE20-40
40 DATA 205, 244, 10, 6, 255, 79, 235, 205, 104, 42, 42, 33, 65, 70,
35, 94, 35, 86, 26, 238, 128, 254, 224, 56, 3, 238, 160, 18, 19, 16,
243, 201
```

This is the assembly language source code for the machine language object code stored in the above DATA statement. Note that it does not contain any position-dependent instructions (such as JP, CALL, etc., except where the CALLs are to ROM), so it can be located anywhere in user memory.

```
0000 CDF40A   00100        CALL   0AF4H      ;TM Error if not string
0003 06FF     00110        LD     B,0FFH     ;B=Maximum string length
0005 4F       00120        LD     C,A        ;C=0 (no offset frm strt)
0006 EB       00130        EX     DE,HL      ;Put string pointer in HL
0007 CD682A   00140        CALL   2A68H      ;Create new string
000A 2A2141   00150        LD     HL,(4121H) ;Get VARPTR of new string
000D 46       00160        LD     B,(HL)     ;B=string length
000E 23       00170        INC    HL         ;Point to new string addr
000F 5E       00180        LD     E,(HL)     ;DE=new string address
0010 23       00190        INC    HL
0011 56       00200        LD     D,(HL)
0012 1A       00210 LOOP   LD     A,(DE)     ;Get character of string
0013 EE80     00220        XOR    80H        ;Lwrcase will be E0H-FFH
0015 FEE0     00230        CP     0E0H       ;Check for lowercase char
0017 3803     00240        JR     C,NOTLWR   ;Go if not lowercase
0019 EEA0     00250        XOR    0A0H       ;Make normal uprcase char
001B 12       00260        LD     (DE),A     ;Save back to string
001C 13       00270 NOTLWR INC    DE         ;Bump to next character
001D 10F3     00280        DJNZ   LOOP       ;If more chars in string
001F C9       00290        RET               ;Finished
0000          00300        END
00000 TOTAL ERRORS

LOOP   0012   NOTLWR 001C
```

HELP FOR A TIRED THUMB by Dave Hanaburgh is reprinted from Byte Babble:

It has happened to me several times and I know that it has happened to others. The play button on the CTR-80 won't stay down. The first time that this frustrating event happened, I had had my machine only six weeks. I was darned if I was going to sit there and hold down the play button while programs loaded. I returned the recorder to Radio Shack and received a brand new replacement. They told me that the repair was an involved deal.

I had the new recorder about three months when the same thing happened. By this time, the warranty had expired and I didn't have the coin for a new recorder. Determined to fix it, I opened the recorder case and noticed that the play button held a spring loaded bar up with a camming action. The cam flat had softened from the lubricant used and now did not have enough of a shoulder to hold the bar up. It was a simple matter to cut a new flat with a knife.

For nearly two years, I have continuously cut new flats in the play button every three or four months. However, the last time, the machine went into fast forward when the play button was pressed. The cam had become too thin, causing the push rod to miss its position.

I wasn't to be defeated. I had nothing to lose except some thumb exercise by taking the recorder apart to see what was going on. Upon doing this, I noticed that all of the keys, except the eject key, were identical. I decided that a switch was needed. By removing the clip near the eject key, I could slide out the bar that held the keys in place. I then proceeded to pull out all of the keys except the stop key, which is held in by a spring. That was a mistake. Once the keys are removed, the actuating mechanics come back and fall out of their alignment grooves. The correct procedure is to remove only the play button and one other, such as the stop key, which has no actuating mechanism. When the play key is removed, hold the white plastic bar in and install a pencil to hold it. Push the stop key back and up to get at the spring. Remove the spring and put it on the former play key. Return the stop key to position. Now remove all of the lubricant from the play key and with a cotton swab, wipe the area that it works in free of the lubricant. I believe that wear on the cam is less detrimental than the lubricant. Install the key in position, replace the bar and reassemble the recorder.

Wallah! It works like new. Note that if the actuating mechanics fall out of position, you will probably have to remove the PC board to get them back. There are three screws on the bottom that hold it in as well as the side panel where the jacks are mounted.

NEW SOFTWARE!

Bridge Academy

BRIDGE ACADEMY (TM) is a series of three instructional packages teaching the card game Bridge. Books I, II, and III introduce the bidding and playing of the average hand. More than 90 percent of all opening bids are covered in these books.

Bridge is played by four people. This puts the beginner in an awkward position: First, it may be embarassing to play with three seasoned players when one makes so many elementary mistakes; second, the novice makes a mistake, he is corrected and taught a new rule - unfortunately, a similar situation may not come up again for weeks and by that time the rule has been forgotten.

BRIDGE ACADEMY (TM) overcomes these problems. Every new rule comes with many exercises. You practice the new rule until you feel that you have mastered it.

BRIDGE ACADEMY (TM) was designed for the beginner who wants to learn the elements of the game and for the seasoned player who wants to brush up on some rules.

From the very beginning you can play bridge with the most patient partner you can imagine: YOUR COMPUTER. You can bid and play the same hands over and over again, ask for the same type of hand, or ask for new hands.

BOOK I teaches you the fundamental rules of bridge, and one notrump bidding and playing. BOOK II introduces the bidding and playing of one suit. Two responses to such an opening bid (the two-over-one and the jump shift) are practiced in BOOK III, along with some more pointers on playing.

Each BOOK consists of three programs, called Chapter 1, Chapter 2, and Chapter 3 (these are all hybrid programs: a basic program and a machine language program); a Cue Card, and Instructions for the User.

The programs are designed to run on the TRS-80 Model I, Model III(4) (32K disk).

Those who order BEFORE December 15th can receive all three books for just $39.95. After this date, they are available for only $19.95 each.

Macro Typing

MACRO TYPING is a unique program that will help you become a better typist! Currently available for the Models I, III, and 4 (in the native 4 mode), MACRO TYPING can accurately track your typing speed to over 100 words per minute. A special feature allows you to generate "random sentences" for practice drills. An editor is included that will allow you to construct specific drills and exercises. The Model 4 provides both audible and visual feedback when you type a character wrong. This feature requires an auxillarily amplifier on the Models I and III. A special "MACRO MODE" prints the letters to be typed in large graphic letters, perfect for persons with impaired vision. After some practice, you can take 1, 3 and 5 minute timed-writings, just like those in formal typing classes, and find out how fast you REALLY are!

All three versions of MACRO TYPING are included with the package. Support is also included for Model I systems without lower case. MACRO TYPING is only $29.95 complete with instructions.

Please send me the following:

- ☐ BRIDGE ACADEMY BOOK I $19.95
- ☐ BRIDGE ACADEMY BOOK II $19.95
- ☐ BRIDGE ACADEMY BOOK III $19.95

- ☐ ALL THREE FOR JUST $39.95 (Before December 15th)

- ☐ MACRO TYPING $29.95

NAME:
ADDRESS:
ADDRESS:
CITY: STATE: ZIP:
COUNTRY:
Visa/MasterCard #:
Expiration Date:
Michigan residents, please add 4% sales tax.
North America: Please add $3 for shipping and handling.
Foreign: Surface, add 10%; Air, add 20%.

STRUCTURED PROGRAMMING!

TRSDOS and MSDOS users, you don't need a Macintosh and a $250 BASIC package to do structured programming! The Alternate BASIC (ABASIC) eliminates line numbers, but that's only the beginning. ABASIC is an enhancement to your current Microsoft BASIC.

Use your word processor to create libraries of relocatable BASIC code that can be used and reused by many different programs. These routines can be merged with your programs, simply by using a "COPY" command. All ABASIC subroutines will have a label. You simply use one line in your BASIC COPY LABLE from "filename" and the code will be included in the translation to Microsoft BASIC. The library of subroutines with ABASIC include a complete HELP system.

ABASIC code can be written with any word processor. With ABASIC, you can use "structured constructs". What this actually means is that we're adding new BASIC commands to your existing BASIC. Some examples are:

IF ENDIF -- With this command, you can start an IF statement, and follow it with as many lines as is necessary for this condition, until an ENDIF is encountered. Your IF statements are no longer limited to one 255 character line.

DO WHILE and DO UNTIL -- The difference between these two structures is that if a condition doesn't exist, it will never be executed with a DO/WHILE loop; it will be executed until it does exist with a DO/UNTIL loop. Examples are provided in the 100 page ABASIC manual.

SELECT/CASE -- With this powerful command, ABASIC can selective use and bypass portions of BASIC code. Programming a large set of conditional situations has never been easier.

There's lots more. ABASIC includes a set of utility programs written in ABASIC and their Microsoft translation that you can use immediately. These include CREF, LVAR, XVAR and XLATE. Full descriptions are provided in the documentation.

The Model 4 and MSDOS versions of ABASIC are compiled. The Model I III versions are in BASIC. ABASIC is "written in itself" and can translate itself. The Alternate BASIC, complete with manual, lots of sample code, help files, utilities and extensive documentation is only $69.95. There are lower priced packages, but we urge you to compare, feature for feature. To our knowledge, you won't find more comprehensive support for structured BASIC programming for microcomputers at a better price. Please specify which machine you are running, whether it is MSDOS or TRSDOS and the operating system you generally use. Orders for The Alternate BASIC can be placed with The Alternate Source, 704 North Pennsylvania Avenue, Lansing, Michigan, 48906 or by phoning (517) 482-8270.

NORTHERN BYTES

c o Jack Decker
1804 West 18th Street
Lot # 155
Sault Ste. Marie, Michigan 49783
MCI Mail Address: 102-7413
Telex: 6501027413
(Answerback: 6501027413 MCI)

Bulk Mail
U.S. Postage Paid
Permit #815
Lansing, MI

POSTMASTER: If undeliverable return to:

The Alternate Source, 704 North Pennsylvania Avenue, Lansing, Michigan 48906

To: