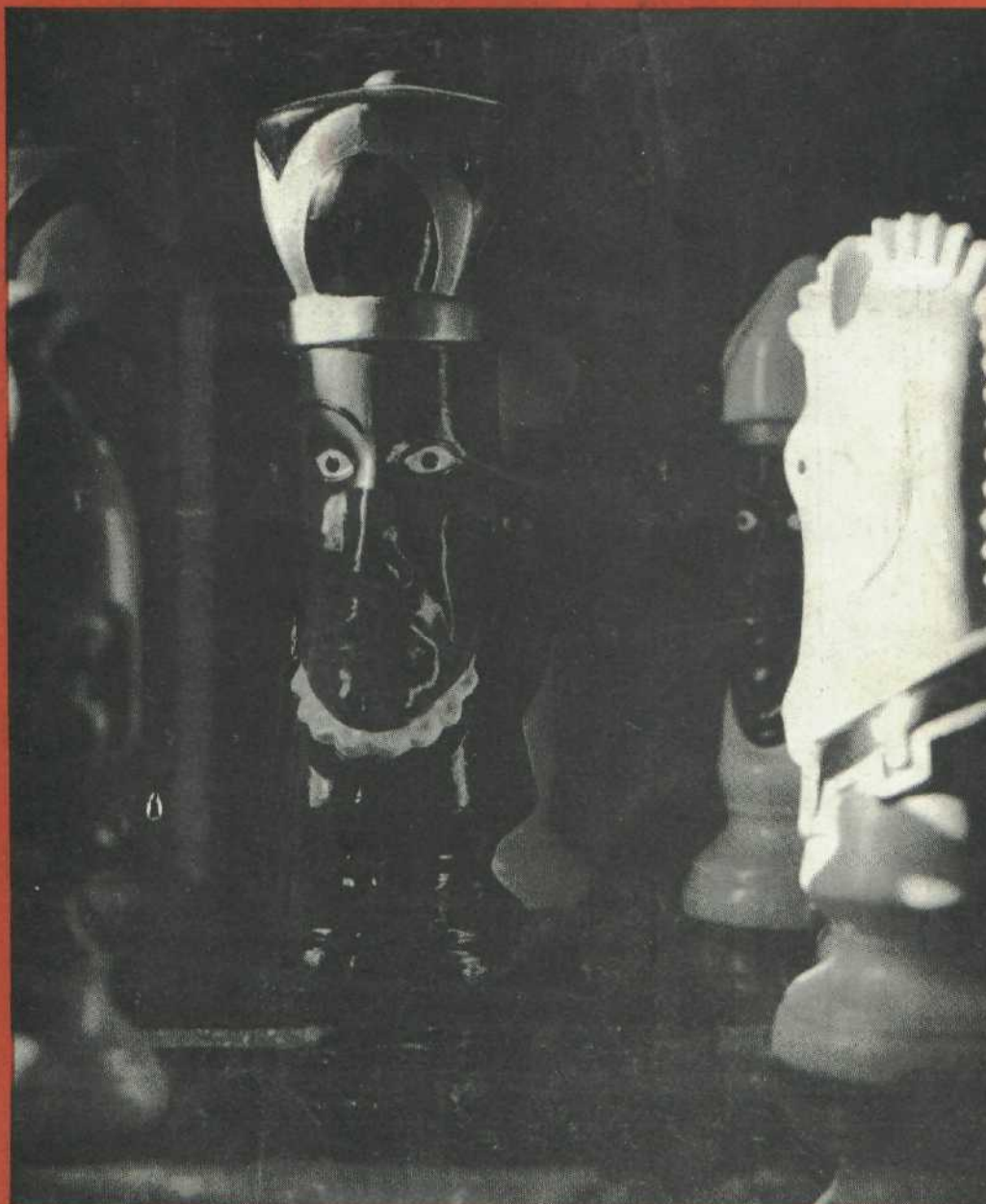


# THE S-EIGHTY™



New Products ..... 8

Review

Omikron Modification  
by John R. Marler ..... 15

Devices  
by Lance Micklus ..... 18

Choosing the Right  
Language

by Lance Micklus ..... 22

Pirates

by Lance Micklus ..... 26

VOL. 1, NO. 3, MARCH 1980



# PURE & SIMPLE

I come to you not as a prophet.....

HARVARD PENNINGTON....WOULD YOU BELIEVE A SECOND BOOK IN THE OFFING! This one will be called "**Basic Faster and Better**" and will have to be just that to outsell his first book called "**TRS-80 Disks and Other Mysteries**". Knowing Harvard as I do...the second will outsell the first....and the third will outsell the second. I'm going to ask him if I may borrow his Midas Pen for a month.

Here we go into another spring of shows for the micro community. I would like to hear from other past and present exhibitors regarding your feelings on the expense, time and manpower involved in these ventures. How about three major shows annually... One in California, one in mid America and one in either Boston or Atlanta...it's food for thought.

In the aftermath of the World Power incident, I would like periodically to advise you on some things that you should check carefully when reading an ad. If you see any or most of the following in the ad, then CONSUMER BEWARE!!

- (1) An address listing only a PO Box
- (2) No telephone number
- (3) Requests for front money....No COD shipments
- (4) No description of a complete package
- (5) Guarantees with no terms listed

Play it safe!

We at SoftSide Publications are looking forward to the 27th of February when we will be opening our retail computer store called "Computer Haven". "Computer Haven" will be the first privately owned computer store in Southern New Hampshire and will be a showplace for the TRS-80\* and its peripherals. I cordially extend an invitation to each and every one of you. We will begin with a press briefing and a tour of the total facility at 10 a.m. and then officially open the store. I imagine that there will be a bargain or two or mark the occasion. Hope to see you there.

Now that we have two issues distributed, and you are reading the third, you should see the direction we plan to take editorially. The last thing I want for **The S-EIGHTY** is to have it associated with the "Games" magazines. Game programs are big sellers, and most likely will be for some time, but I feel that "**The S-EIGHTY**" readers is more interested in programs that will be of some value to him or her professionally or useful in some other application. We are looking for articles on business applications

and utilities. Articles not only keep the readers interested, but also provide a better market for the advertisers. If we can keep the readers and advertisers happy, then **The S-EIGHTY** will be around for many months to come.

Keep the press releases coming...it's the best publicity you can get to introduce a new product, and you can't argue with the price.

It's interesting to sit back and try to evaluate the growth and impact of a single industry on us as consumers. I was sitting here writing a press release for a newspaper feature and suddenly realized that a most accurate measurement was the growth of our own business here in Milford. **The S-EIGHTY** is an offspring of SoftSide Publications, which is an division of Robitaille & Sons Enterprises, Inc. Two years ago we were two people working from the basement of a local house, now we are 50 strong, working from two large buildings, have added two new business enterprises in HardSide and the Ramworks to cover hardware and software requests, and have added two new national publications in **SoftSide - Apple Edition** and **The S-EIGHTY**. Incredible!....not really, as we've been trying like crazy to grow gracefully, but the industry has placed demand on our products to the point where growth is a daily affair and we temper it as best we can. Considering the fact that the "Micro" industry is still in relative infancy...try and imagine what things will be like two years hence. Any projections?

Creative Computing Editor John Craig has departed from the New Jersey based publication to take over as the editor of Intelligent Machines Journal, now Info World, in California. Computer World Magazine bought the IMJ, offered Craig the job and he took it. A definite gain for Info World. Best of luck, John, and keep us posted.

I had dinner recently with Dick Govatski, Advertising Director with Popular Electronics Magazine, and he provided me with some most interesting information that I would like to share with you.

Popular Electronics polled 2,000 of their readers randomly with reference to microcomputers, and their response plus a few extra tidbits should make believers out of everyone. Their prediction of microcomputers sold during 1979 was 300,000 (which we feel is a little high) for a sales figure of \$350,000,000.00. Half of those polled either owned

or had access to a microcomputer. The people surveyed had an average investment of \$2,600 in hardware and software. 75% of the microcomputer owners reported an average expenditure of \$1,800.00 in the past year. 30% are planning to buy a micro, and spend \$1,300.00 not including the cost of software and peripherals.

It paints a very bright picture for those of us who have diversified into all of the elements of the industry to keep pace with the demand. Someday this is all going to end . . . but it won't be in our professional lifetime. I have a 6 year old daughter who is developing into a computer addict, and I wouldn't be surprised if when she is ready to purchase a system of her own, the TRS-80 will be selling for \$200.00 and be the size of some of the electronic games of today.

Advertising is always a subject of interest to a magazine editor, as without it he doesn't have a publication!...keep those ads coming folks!! I was doing some radio commercials for a new undertaking here at SOFTSIDE and I was amazed at how fast the material drew results. Radio ads give a touch of "Locality" to your product, and really bring out the interested customer. Try radio in your local ad campaigns and see how it works for you. We're in the process of preparing a television commercial as well to see what reaction we get from the three state region surrounding New Hampshire. Don't be frightened away by rumors of high prices...I was pleasantly surprised at the price structure. We won't have to sell many computers to have the ad pay for itself. Speaking of ads paying for themselves...a good formula to use in measuring results is not necessarily in the number of bingo cards received, but if your \$\$\$\$ sales are 10 times what you spent for the advertisement, then your campaign was successful. Our rate structure is so low in **The S-EIGHTY** that if you sell one business system, then your ad has paid for itself 10 times over....and if an ad in **The S-EIGHTY** doesn't move more than one computer, I'll hang up my pen and go back to being a policeman. You never would have guessed from the picture!

As you may have noticed, the name of our magazine has been changed slightly since last we were with you. We are now known as **The S-EIGHTY**, but nothing else will change with the exception of our name. It's really a treat when you get a call from the Secretary of State and he says "You can't use that name anymore!".....but being a former bureaucrat, the matter was quickly resolved. Onward and upward.

Effective with the April issue, the advertising rates in **The S-EIGHTY** will be increased ever so slightly to keep pace with the rising costs of paper and printing. Full pages will increase by \$50.00, half pages by \$30.00 and quarters by \$18.00

The new structure will look like this:

	1X	3X	6X	12X
Full	\$450	\$425	\$400	\$375
Half	\$270	\$240	\$225	\$210
Quarter	\$162	\$144	\$135	\$126

Current term advertisers will not be affected by the changes. As unpleasant as rate hikes can be, they are indeed necessary so that we can continue to bring **The S-EIGHTY** to you each month...it also gives us more latitude to bring additional services and specials your way.

#### Catalog Rates:

Catalog pages are always of great benefit to the advertiser, and the rates are always easier on the cash flow. **The S-EIGHTY** is featuring a special catalog rate of the page price for the first page, with additional pages being reduced by 10%. A maximum of 8 pages would be comfortable for us. You could run 8 pages for a total of \$2,178.50, and your catalog entry would be listed on the front of the issue....that's 8 pages at a rate of \$272.30 per page (with the 15% included). Not bad partner!

#### ABOUT THE COVER

We thought that you might like this photograph, which was taken of chess figures from an antique chess set, using natural light. We will be using many different styles of photographs on our covers...come computer related, and others just pleasing to the eye.

(Cover photograph taken by Elaine Cheever)

**FORTH** is an advanced language and system for advanced programmers. **MMSFORTH** is a professional version tailored to the Radio Shack TRS-80 Model I and supported by MMS.

Interpreter, Compiler, Assembler, Full-Screen Editor, Structured & Modular Programming, Expandable Instruction Set w/Source Code, Graphics, Strings, Arrays, Double-Precision, Very Fast and Compact, Virtual Memory Includes 5 Demo Programs w/Source Code

MMSFORTH System Diskette (16K, 1 drive).....	\$64.95
MMSFORTH System Cassette (16K, Level II).....	\$44.95
microFORTH PRIMER (required for above).....	\$15.00
THE DATAHANDLER in MMSFORTH (32K, 1 drive) — an exceptionally fast, flexible and easy to use Data Base Management System.....	\$49.95
P I M S MANUAL (required for DATAHANDLER).....	\$ 9.95
FLOATING POINT MATH and FULL Z80 ASSEMBLER.....	\$29.95
Diskette (16K, 1 drive).....	\$25.00
Other FORTH literature:.....	\$19.95
USING FORTH — beyond microFORTH PRIMER.....	\$ 6.95
URTH TUTORIAL MANUAL — very readable introduction.....	
CALTECH FORTH MANUAL — good on FORTH internal structure.....	
Shipping is \$2 plus \$1 per extra book. Mass. orders add 5% tax.	

# mmsFORTH

**MILLER MICROCOMPUTER SERVICES**  
 61 Lake Shore Road, Natick MA 01760 (617)-653-6136  
 Send SASE for free information

# REVIEW

## Mapping the TRS-80 into the Big Time

### Omikron Modification Review

by John R. Marler

The time for all TRS-80 MODEL 1 owners (who feel rejected from Radio Shack's apparent overlooking of our needs for more disk storage) to be in the big time with 8" drives has finally arrived! **OMIKRON Systems** of Berkeley, California has developed a modification system for the Model I that allows the use of normal CP/M. Nearly everyone knows about CP/M and that it is an (almost) universal operating system for Z80 and 8080 microprocessor systems. One of the key features of CP/M is the fact that it is "debugged" and is operating in many types and brands of computers throughout the world and the amount of software written for CP/M system operation would probably fill many catalogs. CP/M version 2.0 is fast becoming the most acceptable operating system for the MODEL II. Two newsletters I subscribe to that are dedicated to the TRS-80 have published the "rumor" that Radio Shack will soon announce CP/M instead of TRSDOS for the MODEL II. The running of CP/M really protects the investment one can develop in software due to the upward compatibility of CP/M to other computers.

A system consists of **Mapper I** and **Mapper II** and can be purchased with/without 8" Shugart disk drives (two drives will give MODEL I the same storage as the MODEL II with more than \$1,000.00 savings!). The **Hardware** consists of the electronic boards called **MAPPER I** to utilize the Random Access Memory as **low memory** and not located above the ROM where any machine activity begins under normal TRS-80 operation. This permits the use of "normal" CP/M to be used and makes the world of "real" CP/M available for us Model I owners for the first time. **Mapper II** consists of a board that, coupled with the disk controller chip, provides true data separation and eliminates the cause for most disk I/O errors — that of poorly separated data reading from the mini disk. **Mapper I** also provides a way of using the graphics capability of the TRS-80 and adds cursor control in a manner similar to the **Soroc 120 CRT Terminal** which uses a very simple and easy to understand X, Y addressing of the cursor.

Installing the system is easy for even a novice. The instructions are easy to understand and written in plain English! First, open the keyboard casing and locate the Z80 chip. Pull this chip out and **MAPPER I**'s board is inserted into the now vacant socket. The Z80 is now inserted into a similar socket on **MAPPER I**. Two "pincher" connectors then "wire" the board with surprisingly strong connection. No soldering!!! Pull the connector wire through the opening where the interface cable is connected, close the keyboard case and **MAPPER I** is ready.

**MAPPER II** is installed in the expansion interface. When the expansion interface is opened, locate the **DISK CONTROLLER** chip; remove the chip; insert **MAPPER II**; re-insert the disk controller chip on **MAPPER II**; two "pinch" connections; pull the end of the connector through the opening where the interface cable is connected, close the expansion interface, connect the **MAPPER** connection wire, connect the interface cable and you are ready to connect the disk drives.

**OMIKRON** designates which drives are to be Mini drives and which drives are to be 8" Drives. Mini drives are not required for this modification, a system of four 8" drives can be ordered. **OMIKRON** sells the drives or will sell the **MAPPER** boards only. I purchased the **MAPPER** boards and TWO 8" **SHUGART** drives in a two drive enclosure. The enclosure allows for the mini drives to be placed on top of the 8" drives. This system costs \$1795.00, but is worth far more in convenience. Connecting the drives is as simple as making the connections.

Now, the modified MODEL I TRS-80 is ready to run. On power-up the screen displays:

**OMIKRON**

C = CP/M

T = TRS80.

Insert the CP/M disk provided by **OMIKRON** in drive A, depress the letter C and CP/M boots up with the sign-on:

**OMIKRON CP/M VER 1.4**

**48K MEMORY**

**A>**

This indicates that a 48K version of CP/M is up and running and waiting for the next command.

An immediate backup of the CP/M should be made.

**OMIKRON** provides several utilities that are very useful and easy to implement at no additional cost. These are:

1. **SETUP.COM** This allows for custom selection of the following options: A). Selects if a deleted character is to be echoed to the screen. B). Selects automatic line feeds with carriage returns. C). Provides a lower case driver for a modification from a reprint of an article in **COMPUTRONICS** and **80-US Newsletter** (which cost less than \$20.00). D). Implements the graphic characters of the TRS-80 in the same manner as **LEVEL II**. E). Utilizes the form feed character in the same manner as **TRS-80 DISK BASIC** allows. This is done by keeping an internal counter of lines used and sending out lines feeds to equal the value for the size page being used.

2. **SERIAL.COM** This program reads the switch setting of the RS232 board and sets all printer output to the RS232 board for serial printers. Pretty nice feature.

3. **MEMTEST** This is a most extensive memory test. The minimum time (unless a defective chip is discovered) is 15 minutes for each bank of 16K. It tests the relationship and interaction of the memory chips.

I now have my **OMIKRON** modified TRS-80 MODEL I working and working very well. I was told that a system could take 6 weeks from receipt of order but I got delivery in three. Installation was easy, and **OMIKRON**'s utilization of the graphics of the TRS-80 and the addressable cursor shows that **OMIKRON**'s heart is in the right place. With the **OMIKRON** System the MODEL I owner has the best of BOTH worlds - **FULL CP/M capability AND Graphics**. Any system this capable would normally be very costly, a computer that provides the use of three different operating systems **AND** using **BOTH** mini and 8 inch drives **AT THE SAME TIME!**

**APPARAT, INC** of Colorado, (NEWDOS+) have advertised that they have **SUPERDOS** to allow **TRSDOS** compatible operation with this disk configuration.

There's the Great news that yes, Radio Shack Mod I owners, there is a way to get into the Big Time - With **OMIKRON**'s **MAPPER I** And **MAPPER II**.



by Lance Micklus

Now that's power. It means that you, as a machine language programmer, do not have to worry about DRIVERS, whether you are dealing with a printer or a file. You treat them all the same way, as DEVICES, and let the operating system do the work. The DCB and the DRIVER can be anywhere in memory.

This is the secret to making KVP work. KVP is nothing more than three DRIVERS - keyboard, video, and printer. All KVP does is change the address of the drivers in the \*KI, \*DO, and \*PR DCB's.

Here's something else to think about. In BASIC, did you ever stop to think that PRINT, LPRINT, and PRINT# (for file I/O) all have the word PRINT in them? BASIC handles all of these statements exactly the same way. The only difference is in which DCB it uses to output the bytes.

And how about this? Take RSM2D and look at Microsoft's M80 assembler program. Please notice that nowhere does it call addresses X'4436', X'4439', and X'443C'. In other words, it appears, at first, that it never reads or writes to files. **BUT IT DOES!** You will find calls to X'1B' and X'13'. That's right. It does not input or output to files. It inputs and outputs to DEVICES. **AND A FILE CAN BE TREATED AS IF IT WERE A DEVICE.** Therefore, input and output can be processed by anything - **ANYTHING!**

Now, use RSM2D and follow a CALL 33H. First, it loads DE with the address of the \*DO DCB. Next, it jumps to address X'1B'. At this location, the DE register is assumed to have the address of the DCB. It now sets up the registers to indicate an output function.

We could put the \*DO DCB any place in memory, in which case we bypass the first step by loading DE with the address of our newly moved \*DO DCB. Instead, we just call X'1B' directly. The only advantage in keeping the \*DO DCB in its present location is it saves the trouble of loading register pair DE with the DCB address.

What does all of this mean to the end user who is running his payroll? It means that a summary report is sent, not to the line printer, but to a DEVICE. Thus, the user, when he runs the program, can decide where he physically wants the output to go. In a typical situation, a payroll program makes only one summary report. Suppose, on some special occasion, you need five reports. Then, from the DEVICE you normally send the output to, ROUTE the output to a file. After running your payroll program, RESET the DEVICE to CLOSE the file. Now, PRINT the file five times.

Here's something else you can do. You can make your own DEVICES up. A case in point was THE MEAN CHECKERS MACHINE. The source code file for all of the machine language subroutines is 32 granules. This part of the program was written for the M80 Microsoft Assembler. To get a listing from the assembler, you would normally specify a /LST file. After assembly, you would just print the /LST file.

There were two problems with this. First, the /LST file produced is more than 67 granules. The second problem is the way the listing appeared on my printer. I kept my listings bound in report folders. Unfortunately, the binding blocked the leftmost edge of the paper from view, so I couldn't always see all of the information on the page. What I needed was the ability to tab each line 10 spaces before anything was printed on a page.

The solution to the first problem was very simple using DOS 3.0. Rather than specifying a /LST file, I specified a DEVICE, which was \*PR. That made it possible to print this rather large listing with no problem at all.

The solution to the second problem was a little more work, but was worth the trouble. Because the problem of the left hand margin comes up so often, I simply created my own new DEVICE which I called \*LM. The letters are my initials. If your name is George Blank, then I guess you can change the DEVICE to \*GB. Why not?

The assembly listing at the end of this article shows you how I wrote the DRIVER program from my new \*LM DEVICE. It was written for the Microsoft M80 assembler, but I don't think you'll have any trouble modifying it for EDTASM. If you decide to put this little program together for yourself, remember one thing. Since this is a DRIVER, the COMMAND file you create must have the file extension /DVR and not /CMD. What I did was create this program under file name of SPACE/CMD and then renamed the file SPACE/DVR. Don't let the extension throw you. DRIVERS are nothing more than command files, they just use a different file extension.

To create the DEVICES \*LM, you type the following line from VTOS 3.0's prompt of "DOS READY":

```
SET *LM SPACE 10
```

What happens is the following: First, the SET command will load the file SPACE/DVR. Then, it will locate a place in memory where the DCB is to be built. Control will then be turned over to my DRIVER as follows. HL will point to the next ASC character on the input line. In this case, it will be pointing to the "1" in "10". DE will point to the address in memory where I should build the DCB for my DRIVER. My program looks at the number 10, or whatever number was typed in, which is the number of spaces to TAB at the beginning of every line. This number is put into my DRIVER routine. The DRIVER routine is then loaded into high memory, and HIGH\$ is reset to protect my DRIVER's memory from use. My program also builds the DCB, specifying that this DEVICE is output only, with the second and third bytes of the DCB being the address of my DRIVER.

Be sure you're clear on this. The program I made is **NOT** the DRIVER. Rather, it is a program which **BUILDS** a DRIVER. The name of the DEVICE is defined by the command you give DOS 3.0 SET \*LM means you want to name your DEVICE \*LM. If you type \*GB, then your DEVICE will be named \*GB. So, anybody can use my routine and call it anything they want.

Once control returns back to DOS 3.0, the DEVICE command should now show my new \*LM DEVICE in the list.

I'd like to make a subtle, but important point about my DRIVER. It outputs to the \*PR DEVICE. You should not think of it as outputting to the printer. It might very well be that the \*PR DEVICE is not a printer. So, what the \*LM DEVICE really does is output to the \*PR DEVICE anything it receives. Anytime a RETURN or FORM FEED character is sent, it will add a specified number of spaces after those characters.

Now, my problem is solved. I have the M80 Assembler list to DEVICE \*LM, but leave DEVICE \*PR tied to my printer. M80 then lists on my printer, and every line is moved to the right 10 spaces. When I put the listing in my report folder, a large left hand margin is under the binding, and I can read all of the listing.

Here's another interesting idea. How about building a device which encrypts any data sent to it, and another device which decrypts. This would give you a system where you could read and write to a disk file in scrambled form, yet your BASIC program, communicated through your DEVICES, always is seeing real information (i.e. decrypted).

# CHOOSING THE RIGHT LANGUAGE

by Lance Micklus

Which computer language is the best language? The answer is, "None of them." In any specific situation, one language may be better than another. The proper choice can make the difference between a well written program and a poor one. There are times when even Level I BASIC is the best language to choose.

One very good reason for selecting BASIC as the language to write in is because it's the only language you know. There's nothing wrong with that, but it might be much better if you had a selection greater than one.

There are several popular languages you can run on the TRS-80. These are BASIC, PASCAL, FORTRAN, and machine language. Under TRS-80 CPM, you can also get COBOL. We will not include COBOL in this discussion, because so few people have it. And then there is MMSFORTH, which does not have the wide acceptance the others have, either. Although that may change, we will also leave FORTH out of our discussion.

PASCAL appears to be a big up-and-coming language, but the TRS-80 version, distributed by FMG, is a disappointment. It has single precision math, and is thus worthless for any serious work with numbers. Second, PASCAL, like other fringe languages, lacks the good I/O handling that BASIC, FORTRAN, and COBOL have. What many people are calling PASCAL, is PASCAL plus the I/O handlers of another language, often FORTRAN. That gives you a combination of PASCAL with FORTRAN format statements. The third problem with TRS-80 PASCAL is that it is still under development. It costs you \$150 to buy a prototype. I still haven't gotten my instruction manual and one of the books I ordered with my PASCAL package. So, for the time being, we will have to forget about PASCAL.

That narrows the logical choices for the TRS-80 programmer to BASIC, FORTRAN, and machine language. These are all good choices. Any can be run on a TRS-80 without special operating systems which must be purchased separately. And all can be loaded and run on either tape or disk systems. Thus, most people could put programs written in any of these languages to work immediately.

Many TRS-80 programmers have not had experience with FORTRAN. And that's a shame. FORTRAN is an old language, but it's still nice to work in.

FORTRAN's advantages are numerous: It's fast, the coding technique is protected, it has powerful library functions, it is an excellent number cruncher, its functions are complete subroutines instead of single lines, and it interfaces easily with machine language subroutines. (The last feature eliminates loading machine language programs, entering BASIC, and try-

ing to tie into the machine language programs with USR functions, especially if you're passing several arguments to and from the machine language routines.)

FORTRAN does have its bad points. Unlike BASIC, FORTRAN programs must be written with a text editor program. The code must then be saved on disk. Now the FORTRAN compiler must be called to create a /REL file. If syntax errors are found, you must load the source code back into the text editor, and repeat the process. After eliminating the syntax errors, you must then LINK the /REL file to create machine language code which can be executed.

Debugging your FORTRAN program can be a difficult, time-consuming task. There are no line numbers in the machine language code, so it's difficult to figure out where in memory anything is. There's no way to start and stop the program at will, as there is in BASIC, just by pressing the (BREAK) key, so it's hard to figure out what your variables are doing.

Some of the large timesharing computers have special debug options in their FORTRAN programs which help overcome these problems. But the TRS-80 FORTRAN is minimal, so you're left to your own devices to find problems. Unfortunately, in order to keep the FORTRAN compiler and run time package small, Microsoft had to leave out a number of error checking features which normally would identify problems for you.

A case in point is THE MEAN CHECKERS MACHINE which developed an interesting problem when the forced jump option was added. After coding in all of the additional instructions, and taking care of syntax errors, I finally got a good compile and link. I loaded TMCM into memory. When the checker board appeared on the screen, I entered my move. It was rejected. No move I could make was considered legal by the computer. So I typed UMOV, which lets the computer take its turn first at the beginning of a game. In less than a second, the computer declared itself the winner even though not a single piece had been moved. It prompted me to hit the (ENTER) key and it came back all sad because it lost the game. But wait a minute, nobody had even made a single move!

It took me eight hours to find the problem. One of the arguments being passed to a subroutine was the constant 1. This gets passed to another subroutine, which passes it to a third subroutine. The third routine changes the constant 1 to a 2, but later puts it back to a 1. That is a no-no. It does prove, indeed, what Murphy's Law says about computer programs. "ALL CONSTANTS ARE VARIABLE, AND ALL VARIABLES ARE CONSTANT."

O.K. I can take a hint. I changed the argument to a variable whose value was 1. The variable worked like a constant and the problem was solved.

The worst problem was a subroutine called TITLE. TITLE was written in machine language. All it does is display THE MEAN CHECKERS MACHINE title and copyright on the screen for 8 seconds, and then returns to the calling FORTRAN program. No arguments are passed. TITLE is a very simple little program. Yet, it blew up THE MEAN CHECKERS MACHINE. Why?

The problem had nothing in the world to do with the TITLE subroutine. One of my variables was never set to zero. It turned out that whenever I loaded TMCM, that memory location just happened to be zero'd. Luck! But when I added CALL TITLE to the program, it shifted that variable's location three bytes higher, and it wasn't zero there any more.

Now that we know what a terror FORTRAN can be, let's look at BASIC. We can code and run immediately. This makes it a lot easier to work on and debug a program. We can easily set break points with the STOP command and check variable. In BASIC, all variables are variables, all constants are constant, and all variables start out being zero.

But BASIC does have its weak points. The biggest problem with BASIC is the fact that it's slow.

Before writing THE MEAN CHECKERS MACHINE in FORTRAN, I had a working version written in BASIC. The BASIC version took 10 seconds to generate a playing board, while the FORTRAN version takes about a quarter of a second. Playing at an IQ of 2, the BASIC version took five minutes to find a move. The FORTRAN version takes 40 seconds and does 50% more work.

The final alternative to the TRS-80 programmer is to write in machine language, such as the Radio Shack Editor Assembler and the Microsoft M80 assembler and L80 linker.

There are two major advantages with machine language. One is fast execution speed, and the other is that the memory requirements are usually smaller. No compiler can write code as well as the shrewd and cunning programmer. In addition, the code is kept rather secret because you can't list it very easily. Your source code can be loaded with tons of remarks, and is self-documenting, yet the remarks never appear in the finished product because remarks generate no code in memory.

In machine language you are actually controlling the machine. In BASIC, you are spending most of your time solving a problem.

EDTASM is the Radio Shack Editor/Assembler which is also available on disk from Apparat. It is an in-memory program, which means the maximum size of the program you're building is limited by the amount of memory in your system. Being both editor and assembler in one, you can code and then compile without reloading. This makes errors easy to fix. Once the syntax errors are fixed, you can assemble the machine language code, save your source code, and run. Because it is an absolute assembler, you know where everything in memory is from the listings you can get on your printer. This makes debugging a lot easier.

The major drawback to EDTASM is that it will not assemble large programs. ST-80D is a prime example. It was written with an earlier version of EDTASM which was modified to run under TRSDOS. Although it has fewer features than the Apparat one, it does have more memory to work with. Even with a 48K system, ST-80D completely fills the memory of the machine. The Apparat EDTASM will not assemble ST-80D because it runs out of memory. This means that little more can be added to ST-80D because there is just no more room to assemble it.

Microsoft's M80 assembler is disk-oriented. It is limited not by memory, but how much room you have on your disk drives. Just on that basis alone, it will assemble a much larger program than EDTASM. However, by writing several machine language programs and linking them together, huge machine language programs can be assembled. This is, in fact, how Level II BASIC, which is 12K, was assembled.

A second powerful feature of M80 is conditional assembly. This lets you write code which can be assembled or not, at will. Using this technique, you can write two routines, one for a Level II machine, and

another for a disk TRS-80, and include both in the same code. By changing one true or false condition, you can assemble either version.

But M80 does have its drawbacks. Like FORTRAN, you have to use an editor to code, an assembler to get a /REL file, and then the linker to get machine language code. In other words, there are a lot more steps.

If you're lucky enough to know three computer languages, how do you decide which language to use? The best way I can answer that is to give you some examples, based on my own experience.

STAR TREK III was written in BASIC for a very good reason. At the time I started to write the program, I still had a Level I machine, my EDTASM was on back order, and FORTRAN was still in the works. If I had to do it over again today, I would code it a little differently, but I would still use BASIC.

BASIC executes the program fast enough; time is not a major problem. BASIC has all the computing power needed to make the program run. Of all of the languages, BASIC is the easiest to write code for, and debug. And, from a marketing standpoint, people prefer games written in BASIC rather than some other language which they can't understand.

THE MEAN CHECKERS MACHINE was written in FORTRAN mostly to get an increase in speed. Like BASIC, FORTRAN can handle arrays very nicely. On the other hand, FORTRAN is much easier to code than machine language, especially for a program which is so complex. Believe me, it isn't easy to code a 13.5 K machine language program.

KVP was written in machine language to get control of the machine. There are things you can do in machine language that you can't do any other way. Also, KVP can be in the machine running while the user runs his own program. Thus, there's no problem having the user integrate his program with the KVP program.

MASTERMIND II is an example of a program written in the wrong language. There's nothing wrong with it. It plays an almost unbeatable game, and the display is nice. But, it would best be written in FORTRAN which would still give it the speed, but make the coding job much easier. You have to remember that in machine language, you have to do all of the work. There's no such thing as a PRINT command in machine language. You've got to run the printer yourself.

True, you can call a subroutine in the ROM and save yourself a little effort. But, it's not nearly as easy as an LPRINT statement in BASIC. In the MASTERMIND program, much of the code is spent on simple things like displaying information on the screen. So, another reason for picking one language over another is the fact that a particular language may save you a lot of work.

It all comes down to this. No one language is going to be right all the time. FORTRAN works best for some applications, and is useless in others. The thing to do is to keep an open mind on the subject. When you start out to write a program, consider each of the languages you know. Weigh the pros and cons of each language, then pick the language best suited to the task. The result will be a better program and less work.



# PIRATES

by Lance Micklus

For quite a long time, I have wanted to write an article on the subject of pirates. But, for one reason or another, I never did. Much to my surprise, little seems to have been written on this subject. So, I'm going to bite the bullet, and speak out, quite frankly, on this subject.

I guess living in Burlington, Vermont, does have a few disadvantages. One of which is that I am somewhat isolated from the larger areas of the country. We're doing pretty good just to have a computer club in Burlington representing all types of systems. Yet, many of you have so many computer people living near you, you can have the luxury of a club of just TRS-80 users.

Still, I hear many things...many stories. I find some of the stories hard to believe, yet, because I hear them so often, I must assume that they are true. I have heard of TRS-80 computer clubs who swap software. One club I heard about in New Jersey is connected with a college. At the January 1980 meeting, one member brought in a copy of Microsoft's Disk Adventure game. How he got a copy of the program is unknown, because the product had not yet been released for sale. This person, very proudly announced that he, because of his superior knowledge, had found a way to make a back up copy of the disk. He had with him, several back up copies, which he was willing to fence for \$15 each, provided at least four members would buy these pirate copies to cover his expenses. There were plenty of takers.

A certain gentleman, who had come to the meeting for the first time, spoke up. The man, who later relayed this whole story to me, pointed out that what the members were doing was not ethical, if not out right illegal. The gentleman was boo'd down. He was told all about Supreme Court rulings which made this type of thing O.K., and how expensive computer software was.

After thinking about the incident, the gentleman who stood up to defend the pirating of Microsoft's Adventure program, called me for some technical information. But, our conversation soon spread to this computer club. A few weeks later, the gentleman again called me back. We chatted a little more about this computer club and he said he was planning on returning to the next meeting to see if he could talk some sense into these people. I wish him luck.

It seems that some people are offended by the idea that somebody else is making money. They seem to think that they have it so bad. Yet, not a single one of them would be willing to pay \$600 for phone services a month out of their own pocket. I doubt that any of them would work more than 70 hours a week, with no over time, for only \$4.00 per hour. Not to mention the fact that they might not get paid on time, or might not get paid at all. Worse yet, they also have to hock everything they own just to keep their job. I mean really, wouldn't you have to be crazy to take a job like that. Well, I guess, I'm crazy.

What was funny about this one computer club is that they're college people. You'd think they'd

realize that people like myself have to make money so we can live. If a guy goes and trades a Star Trek III for a Scott Adam's Adventure, then how are Scott and I going to make any money?

It seems these people don't see it that way. They look at all the money they spent buying their TRS-80 Level II with 16K. And then, they found out that the machine was stupid. Now they've got to spend more money to buy software so it does something, and they feel ripped off. All right. I can understand that. But I didn't sell them the computer. Why are they ripping me off? If they feel ripped off, then at least rip Radio Shack off. That might make a little sense. But me, why?

Of course the software people are at the disadvantage. If a guy only has \$150 to spend, the argument goes, then that's all he is going to spend. No sale was lost because he wouldn't have bought the software anyway. That's a valid argument if your name is TSE, or Computer City. But not if you are Lance Micklus. Whether the guy buys a modem at \$150, or an ST80-III, TSE gets all the guy's got. If he wants both, he can buy the modem and steal the software. TSE still gets all the money the guy is going to spend. But, the company that makes modems is going to make a lot of sales and I will make none. You can not photocopy or make a back up copy of a modem. You can with software. So, in the pinch, the software guy always loses.

A few nights ago, Randy Cook and I had a rather interesting discussion about this.

It turns out that at one time, Randy Cook had an interest in magic. Also, at one time, I was very active as a magician. Both Randy and I found that our experience with magicians involved one very interesting fact. The method of performing a feat of magic is a secret. If everybody knew how to make the lady float in the air, it wouldn't be a mystery any more. Of course, to be a magician, you must learn what these secrets are. Long ago, magicians developed a set of ethics. To really become involved with magic, you will probably join a magic club. The first thing you will learn are the rules of ethics. Magicians, it seems, understand that there are people whose ability to make a living is based on the fact that people don't know how to make a lady float in the air. Magicians respect that fact. They also know that only by seeing to it that all magicians follow good ethics, can the art of performing magic survive.

This doesn't seem to happen with computer clubs. Many do not try to instill good ethical character into their members. Indeed, some clubs, like this one club in New Jersey, are more interested in seeing how much they can get away with.

In the 1960's, everything that was wrong, that could not be blamed on somebody else, was blamed on society. Remember the song, "Society's Child". Today, we blame all of our troubles on the establishment. Another thing I found that Randy Cook and I have in common is the fact that we appear to many people as being establishment. Fortunately, my articles have helped to humanize me. But there are still people who think that there is no such person as Lance Micklus. Randy Cook, getting less exposure than I do, tends to suffer worse. Not too many people know anything about Randy Cook. Perhaps some software pirating is not based so much on financial considerations as it is trying to rip off the establishment in an act of retaliation.

One of the very funny things about Randy Cook is that he has a reputation for being so secretive. Actually, you might be surprised. Randy really enjoys helping people learn about computers. In fact, so do I. That's why I enjoy writing articles. Randy is really proud of what he's done and he really wishes he could find a way to share what he's learned with people just like yourself, no matter who you are. But, if he let the cat out of the bag, then he would probably would find himself ripped off. It takes a lot of money and equipment to write a disk operating system. If everybody starts ripping him off, Randy would be unable to continue his work. In a way, Randy's DOS is almost like another child to him. Letting too many secrets out might almost be like putting his child in danger of being killed. Rather than do that, he'd rather protect that child, no matter what. I can understand that because I've felt that way, myself, about some of my own programs. After a while, the program almost becomes part of you, and you begin to become very protective of it. Considering that Randy's DOS is the result of 13 years of work, you can imagine how much a part of Randy that program has become.

In fact, you could almost say that TRSDOS and VTOS were NOT written by Randy Cook. They ARE Randy Cook. They are as much a part of Randy Cook as his arm or his leg. Perhaps, even more a part of him than his wife and two children.

What can be done to stop pirating? Several things can be done. Here's what they are:

First, software people can use various little tricks to bury into their code little things that later, could be used to prove that it is their code. The most amazing example of this can be found on any copy of TRSDOS or NEWDOS — but not VTOS. Read this first — then do. From DOS, type the following command: BOOT/SYS0.WHO. As soon as you hit the (ENTER) key and your disk drive starts to run, press and hold down both the 2 and the 6 keys. It wasn't until TRSDOS 2.3 that Radio Shack found it, and changed it. It's interesting that Tandy Corp. and Randy Cook both contain 10 letters.

But this type of thing is only good for use with a big law suit where the damages run into at least six figures. The fact of the matter is, the guy who, all by himself, swaps or even sells software probably will get away with it. He's just too small a fish to worry about. That's where you, the reader come in.

First, never accept free software unless it is in the public domain, or you and the other guy are the original authors. If you do receive software as a gift, don't accept it unless it is an original copy.

Second, do not associate yourself with computer clubs like this one in New Jersey. For one thing, such clubs are almost never incorporated. As such, if the club ever was sued, you, a member of the club could also be sued. In fact, it is even possible that you could personally be sued for something the club did, EVEN THOUGH YOU WEREN'T EVEN THERE.

Third, encourage your computer club to always observe good rules of ethics. Then, try to teach those ethics to the newcomers who are just starting out. If our computer club is like the one in New Jersey, then start your own computer club. Most people really want to do what's right. And certainly, part of the pirating problem is that some people don't know what is right and what is wrong. Help them.

The rules of ethics shouldn't be too hard to understand. A computer program represents the way some people make their living. They are entitled to protect their livelihood. A software author is entitled to a royalty for each computer which runs his code. That doesn't mean that if you own two TRS-80's, you need to buy two Star Trek III games. Unless you're some kind of a nut, you're really only going to be playing it on one computer, even if it's one of the two computers that happens to be free at the moment. But, programs like KVP or ST80 are the kinds of things that you buy to use on two computers. Ethically, then the author is entitled to receive a royalty for two computers, not one.

Nobody really knows how much pirating is really going on. My guess would be that for each copy of Star Trek III sold, anywhere from 2 to 5 pirate copies are made. Electric Pencil, I suspect, is pirated once for each copy sold. I'm sure games suffer the most from pirates. Far more than practical and utility programs like Pencil and ST80.

Am I serious? You bet. Haven't you noticed, the last game I released was in September 1979. It was The Mean Checkers Machine. Although TCM has been upgraded several times (we are now at version 1.3), it is the last game I've worked on. No new games are being worked on now. Instead, I find I must devote most of my time to the higher ticket items which do not suffer as much from pirating.

Still, I think there's hope that some how, this problem can be solved. But the only person who can really solve it is you.

#### DATA BASE MANAGER IDM-IV \$69

You can use it to maintain a data base & produce reports without any programming. Define file parameters & report formats on-line. Features key random access, fast multi-key sort, field arith., label generator, audit log. MOD-II version with more than 50 enhancements \$199.

#### ACCOUNTS RECEIVABLE ACCT-III \$69

One or more drives. Order entry calculates sales tax, shipping, amount for multiple items. Credit checking, aging, sales analysis, invoices, statements and reports. As opposed to most other A/R, ours can be used by doctors, store managers, etc. MOD-II version \$149.

#### WORD PROCESSOR 16K \$39, 32K \$49, MOD-II \$49

First word processor specifically designed for the TRS-80 that uses disk storage for text. Written in BASIC. No special hardware and text limit. Use for letters, manuals & reports. 32K version features upper/lower case without hardware change and multiple input text files.

#### MAILING LIST advanced MAIL-V \$59

Fast sort by any field. Multiple labels and reports. 4-digit selection code, new zip code ext., screen input, live keyboard, powerful report writer. MOD-II \$99.

#### INVENTORY INV-V \$99

9-digit alphanumeric key for fast key random access. Reports include order info, performance summary, etc. Calculate E.O.Q. Powerful report writer. MOD-II \$149.

All programs are on-line, interactive, random access, virtually bug free, documented and delivered on disks. MOD-I requires 32K DOS. We challenge all software vendors to offer low cost manuals so you can compare and avoid those high-priced undocumented, 'on-memory' programs. Send \$5 for a MOD-I manual and \$10 for MOD-II.

MOD-II programs are extensively modified, guaranteed to run with 1 year newsletter & updates. 10% off for ordering more than 1 MOD-II programs.

#### MICRO ARCHITECT

96 Dothan St., Arlington, MA 02174

READER SERVICE: U