July 1981
US $2.95

80 microcomputing

the magazine for TRS-80* users

Assault on BASIC

The continuing search for better ways to communicate with your computer.

Inside:

80 examines the business microcomputer and what it can do for you.

0 74470 65947 07

TRS-80* COMPUTING EDITION

©1981 Percom Data Co., Inc. The Percom Peripheral 35 cents

Percom's DOUBLER II® tolerates wide variations in media, drives

GARLAND, TEXAS — May 22, 1981 — Harold Mauch, president of Percom Data Company, announced here today that an improved version of the Company's innovative DOUBLER™ adapter, a double-density plug-in module for TRS-80* Model I computers, is now available.

Reflecting design refinements based on both theoretical analyses and field testing, the DOUBLER II™, so named, permits even greater tolerance in variations among media and drives than the previous design.

Like the original DOUBLER, the DOUBLER II plugs into the drive controller IC socket of a TRS-80 Model I Expansion Interface and permits a user to run either single- or double-density diskettes on a Model I.

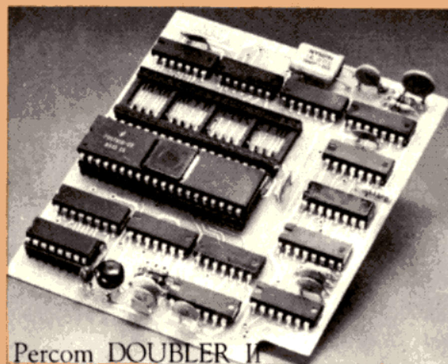
With a DOUBLER II installed, over four times more formatted data — as much as 364 Kbytes — can be stored on one side of a five-inch diskette than can be stored using a standard Tandy Model I drive system.

Moreover, a DOUBLER II equips a Model I with the hardware required to run Model III diskettes.

(*Ed. Note:* See "OS-80™: Bridging the TRS-80* software compatibility gap" elsewhere on this page.)

The critical clock-data separation circuitry of the DOUBLER II is a proprietary design called a ROM-programmed digital phase-lock loop data separator.

According to Mauch, this design is more tolerant of differences from diskette to diskette and drive to drive, and also provides immunity to performance degradation caused by circuit component aging.



Percom DOUBLER II

Mauch said "A DOUBLER II will operate just as reliably two years after it is installed as it will two days after installation."

The digital phase-lock loop also eliminates the need for trimmer adjustments typical of analog phase-lock loop circuits.

"You plug in a Percom DOUBLER II and then forget it," he said.

The DOUBLER II also features a refined Write Precompensation circuit that more effectively minimizes the phenomena of bit- and peak-shifting, a reliability-impairing characteristic of magnetic data recording.

The DOUBLER II, which is fully software compatible with the previous DOUBLER, is supplied with DBLDOS™, a TRSDOS*-compatible disk operating system.

The DOUBLER II sells for $2[illegible]5, including the DBLDOS diskette. Now $169.95!

Owners of original DOUBLERs may purchase a DOUBLER II upgrade kit, without the disk controller IC, for $30.00. Proof of purchase of an original DOUBLER is required, and each DOUBLER owner may purchase only one DOUBLER II at the $30.00 price.

The Percom DOUBLER II is available from authorized Percom retailers, or may be ordered direct from the factory. The factory toll-free order number is 1-800-527-1592.

Ed. note: Opening the TRS-80 Expansion Interface may void the Tandy limited 90-day warranty.

Circuit misapplication causes diskette read, format problems.
High resolution key to reliable data separation

GARLAND, TEXAS — The Percom SEPARATOR® does very well for the Radio Shack TRS-80* Model I computer what the Tandy disk controller does poorly at best: *reliably* separates clock and data signals during disk-read operations.

Unreliable data-clock separation causes format verification failures and repeated read retries.

CRC ERROR–TRACK LOCKED OUT

The problem is most severe on high-number (high-density) inner file tracks.

As reported earlier, the clock-data separation problem was traced by Percom to misapplication of the internal separator of the 1771 drive controller IC used in the Model I.

The Percom Separator substitutes a *high-resolution* digital data separator circuit, one which operates at 16 megahertz, for the low-resolution one-megahertz circuit of the Tandy design.

Separator circuits that operate at lower frequencies — for example, two- or four-megahertz — were found by Percom to provide only marginally improved performance over the original Tandy circuit.

The Percom solution is a simple adapter that plugs into the drive controller of the Expansion Interface (EI).

Not a kit — some vendors supply an untested separator kit of resistors, ICs and other paraphernalia that may be installed by modifying the computer — the Percom SEPARATOR is a fully assembled, fully tested plug-in module.

Installation involves merely plugging the SEPARATOR into the Model I EI disk controller chip socket, and plugging the controller chip into a socket on the SEPARATOR.

The SEPARATOR, which sells for only $29.95, may be purchased from authorized Percom retailers or ordered directly from the factory. The factory toll-free order number is 1-800-527-1592.

Ed. note: Opening the TRS-80 Expansion Interface may void the Tandy limited 90-day warranty.

All that glitters is not gold
OS-80™ Bridging the TRS-80* software compatibility gap

Compatibility between TRS-80* Model I diskettes and the new Model III is about as genuine as a gold-plated lead Krugerrand.

True, Model I TRSDOS* diskettes can be *read* on a Model III. But first they must be converted and re-recorded for Model III operation.

And you *cannot write* to a Model I TRSDOS* diskette. Not with a Model III. You cannot add a file. Delete a file. Or in any way modify a Model I TRSDOS diskette with a Model III computer.

Furthermore, your converted TRSDOS diskettes cannot be converted *back* for Model I operation.

TRSDOS is a one-way street. And there's no retreating. A point to consider before switching the company's payroll to your new Model III.

Real software compatibility should allow the *direct, immediate* interchangeability of Model I and Model III diskettes. No read-only limitations, no conversion/re-recording steps and no chance to be left high and dry with Model III diskettes that can't be run on a Model I.

What's the answer? The answer is Percom's OS-80™ family of TRS-80 disk operating systems.

OS-80 programs allow *direct, immediate* interchangeability of Model I and Model III diskettes.

You can run Model I single-density diskettes on a Model III; install Percom's plug-in DOUBLER™ adapter in your Model I, and you can run double-density Model III diskettes on a Model I.

There's no conversion, no re-recording.

Slip an OS-80 diskette out of your Model I and insert it directly in a Model III.

And vice-versa.

Just have the correct OS-80 disk operating system — OS-80, OS-80D or OS-80/III — in each computer.

Moreover, with OS-80 systems, you can add, delete, and update files. You can read *and write* diskettes regardless of the system of origin.

OS-80 is the original Percom TRS-80 DOS for BASIC programmers.

Even OS-80 utilities are written in BASIC.

OS-80 is the Percom system about which a user wrote, in Creative Computing magazine, ". . . the best $30.00 you will ever spend."†

Requiring only seven Kbytes of memory, OS-80 disk operating systems reside completely in RAM. There's no need to dedicate a drive exclusively for a system diskette.

And, unlike TRSDOS, you *can* work at the track sector level, defining and controlling data formats — in BASIC — to create simple or complex data structures that execute more quickly than TRSDOS files.

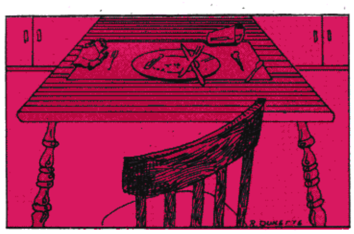
The Percom OS-80 DOS supports single-density operation of the Model I computer — price is $29.95; the OS-80D supports double-density operation of Model I computers equipped with a DOUBLER or DOUBLER II; and, OS-80/III — for the Model III of course — supports both single- and double-density operation. OS-80D and OS-80/III each sell for $49.95.

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE. PRICES DO NOT INCLUDE HANDLING AND SHIPPING.

PERCOM DATA COMPANY, INC. 211 N. Kirby Street Garland, Texas 75042 (214) 272-3421 ✓258

™Trademark of Percom Data Company, Inc. *TRS-80 and TRSDOS are trademarks of Tandy Corporation which has no relationship to Percom Data Company. †Creative Computing Magazine, June, 1980, page 26.

80 Contents



COMING NEXT MONTH

The August issue of *80* is our annual games issue. Wile away the summer with this *80* games sampler. Our roving journalist/gamesman, Bert Latamore, shares some industry plans with you. In case the games prove too much for you (or your rpg character), there's a program that will help you generate a custom-designed will.

PUBLISHER
Wayne Green

EXECUTIVE VICE PRESIDENT
Sherry Smythe

CORPORATE CONTROLLER
Charles Garniss, Jr.

ASSOCIATE PUBLISHER
Edward Ferman

ASSISTANT PUBLISHER
Jeff DeTray

ADVERTISING MANAGER
Kevin Rushalko

CIRCULATION MANAGER
(603) 924-7296
Debra Boudrieau

BULK SALES MANAGER
Ginny Boudrieau

ASSISTANT TO PRESIDENT
Matt Smith

ADVERTISING SALES
(603) 924-7138
Penny Brooks
John Gancarz

ABC *Paid Audited Circulation*

80 Microcomputing (ISSN 0199-6789) is published monthly by 1001001 Inc., 80 Pine St., Peterborough NH 03458. Phone: 603-924-3873. Second class postage paid at Peterborough, NH, and additional mailing offices. Subscription rates in U.S. are $18 for one year and $45 for three years. In Canada, $20—one year only, U.S. funds. Canadian distributor: Micro Distributing, 409 Queen St. West, Toronto, Ontario, Canada M5V 2A5. Foreign subscriptions (surface mail), $28—one year only, U.S. funds. Foreign subscriptions (air mail), $60—one year only, U.S. funds. In Europe contact Monika Nedela, Markstr. 3, D-7778 Markdorf, W. Germany. In South Africa contact *80 Microcomputing*, P.O. Box 782815, Sandton, South Africa 2146. Australian Distributor: Electronic Concepts, Attention: Rudi Hoess, 55 Clarence Street, Sidney 2000, Australia. All U.S. subscription correspondence should be addressed to *80 Microcomputing*, Subscription Department, P.O. Box 981, Farmingdale, NY 11737. Please include your address label with any correspondence. Postmaster: Send form 3579 to *80 Microcomputing*, Subscription Services, P.O. Box 981, Farmingdale, NY 11737.

PUBLISHER/EDITOR
Wayne Green

MANAGING EDITOR
Michael Comendul

TECHNICAL CONSULTANT
Jake Commander

PRODUCTION EDITOR
Susan Gross

NEWS EDITOR
Bert Latamore

REVIEW EDITOR
Pamela Petrakos

TECHNICAL EDITORS
Chris Brown
G. Michael Vose

ASSISTANT EDITORS
Debra Marshall
Michael Nadeau
Betty Thayer

LAYOUT EDITORS
Joan Ahern
Bob Dukette
Sharon Phinney

TECHNICAL CONTRIBUTING EDITOR
Dennis Kitsz

EDITORIAL ASSISTANT
Janet Fiderio

EDITORIAL ADMINISTRATION
Pat Graham
Nancy Noyd

DESIGN ASSOCIATE
Diana Shonk

PRODUCTION MANAGER
Nancy Salmon

ASST. PRODUCTION MANAGER
Michael Murphy

AD GRAPHICS MANAGER
Robert Drew

AD COORDINATOR
Sue Symonds

ADVERTISING PRODUCTION
Steve Baldwin, Bruce Hedin, Jane Preston

PRODUCTION DEPT.
Fiona Davies, Dianne Riston, Linda Drew, Kenneth Jackson, Ross Kenyon, Theresa Ostebo, Deborah Stone, Thomas Villeneuve, Gary Graham, Donna Wohlfarth, Sandra Dukette

PHOTOGRAPHY
William Heydolph, Terrie Anderson, Bill Suttenfield, Paul Babich

TYPESETTING
Barbara Latti, Sara Bedell, Michele Desrocher, Luann Keddy, Mary Kinzel, Kelly Smith, Karen Stewart

Cover by J. P. Morris

80 REMARKS

by Wayne Green

"We need a flexible language mated to that hardware. . . without (it), software development moves very slowly."

Where *Are* The Business Programs?

I'm constantly running into people who have written a program to help with some business chore, but who have never thought about brushing it up for publication. Yet, it could be exactly what newcomers to computers are looking for. Useful business programs don't end with accounting and word processing. These are important programs, but in the long run every business needs specific applications software. Schools need programs to keep track of students, grades and schedules. Cities and towns need software to keep track of residents, voting lists, property evaluations, taxes and licenses—hunting, fishing and driving.

With this huge need for software, why are there so few programs available for business use in computer stores? The answer is complex.

Three fundamental items are needed before we can produce sophisticated business software. First we need a reasonably good hardware system, sold in enough quantity to support software development, and which will be around for a while. We have that in the TRS-80.

Next we need a flexible language mated to that hardware: we have this in Microsoft BASIC. Without a high level language, software development moves very slowly.

Finally, we need a good disk operating system. We have had our best DOS systems for the TRS-80 for only about a year. I'm talking particularly of NEWDOS+ and DOSPLUS. Thus, in practical terms, programmers have had the tools they need for only about a year. Since complicated programs can take weeks or even months to write, and far longer to debug, is it any wonder we are just now beginning to see some really first rate business programs emerging?

Further, there is the major problem of Radio Shack being uncooperative with outside suppliers of products for the TRS-80. If you try to deal with Radio Shack you need a barrage of lawyers and an extremely sharp business manager. And, even if you've written a good program, it can take them years to make a decision.

If you decide to go to an independent publisher, you still have to watch your step; too few have a reputation for being honest with programmers. Fewer still have the distribution facilities necessary to see that your program is advertised properly and distributed to enough computer stores to earn you significant royalties.

One answer is the large-scale program publishing house...such as Instant Software. Yes, I'm biased—Instant Software is a division of Wayne Green, Inc. Keep that in mind as you read the following.

The Marketplace

A couple of years ago I sensed the importance of business programs as a way to help computer stores sell systems. Yet, the Instant Software group has not been able to publish anywhere near the number of business programs they would like to. However, the number of good business programs submitted has been increasing in recent weeks.

"With this huge need for software, why are there so few programs available. . . ?"

Now, to put in a word for Instant Software. It does take quite a while to get a program published there, but this is important to you. Customers and dealers alike have been ripped off by poor quality software; Instant Software takes the time to carefully evaluate every submission. From over 10,000 programs submitted for publication during the last three years, only about 1,000 have been okayed for production.

If you don't have the experience in some business field to write the needed programs, get together with someone who does. Not only will specialized applications programs sell well, in many cases they can bring about an equipment sale. I believe that the more good programs we have, the more computers will be sold.

For this reason it's unfortunate that Radio Shack has been so reluctant to cooperate with program publishers. Instant Software has been trying to distribute programs through Radio Shack stores for over two years without success. I feel that they have everything to win and little to lose by stocking more programs in their stores. Radio Shack's philosophy is to sell only high volume programs. That merchandising philosophy prohibits their even considering handling slower selling but specialized programs. Pity. . . and an opportunity for some other firm to meet the needs of the market.

Instant Software has had considerable success reaching the Radio Shack franchise stores and these store owners tell us repeatedly that our programs are a key to many of their sales.

Instant Software plans to maintain a library of several thousand programs. For now, most will be sold through computer stores, but with technology evolving the time may come when it will no longer be necessary for a store to carry an inventory of programs. We may eventually be able to load them via telephone or even by satellite.

By the way, Instant Software's looking for more sales reps for several areas of the country, to handle software, books, magazines, and even some advertising sales. If you have sales experience and are interested in working with a fast growing firm, let me know.

What To Do

After you've written your business program, debug it. Then, do the best you can writing documentation. Put as much of the documentation as possible into the program, so people like me who hate to read instructions will be able to blunder through it with ease. Then, submit it to a top-notch software publisher of your choice. ■

INSIDE 80

by Ed Juge, director of
computer merchandising, Tandy Radio Shack

"We're aware of the difficulty some of you have had in getting through. . . we've recently expanded the staff. . ."

This month completes my first full year of Inside 80. It's been fun, and I'm thankful every month hasn't been quite like this one. I'm writing this in early May, a time when bad weather is traditional in Texas. Anyway, I had left this writing for the weekend. Friday night high winds, rain, and hail hit the Fort Worth/Dallas area. Our electricity went off at 11 p.m. Friday, and remained off for just over 20 hours. Needless to say, Scripsit and I didn't get together.

Finally, I was able to get started. I cranked out Inside 80, and was about to repaginate and print it, so I could get on with some other homework. During repagination, I got a disk I/O error, and was never able to recover any of the text. Fortunately, my Scripsit disks are all backed up periodically, so that one file was all that was lost.

Pocket Computer News

This month, we're introducing an exciting add-on for the TRS-80 Pocket Computer—a printer with built-in cassette interface! It uses plain paper one and three fourths inches wide, and a cartridge ribbon. It prints 16 characters per line, at 60 lines per minute, and it responds to Print and List commands, when the print switch is on. Power comes from built-in rechargeable batteries, with a charger included.

". . . in early May. . . bad weather is traditional in Texas. . . . high winds, rain and hail hit. . ."

You'll get about 8,000 lines of printing on a single battery charge! Also, so you won't have to carry so many pieces along with you, the printer also contains a cassette interface. It's the most often requested item to go with the Pocket Computer, and it's available now for only $149.

You're probably thinking that if you knew about the printer while the Pocket Computer sale was on in May, you'd have bought one. Well, don't worry. We've announced a new low everyday price on the pocket computer, $229. Now you can enjoy a complete system at a good price.

During the coming months, I'll have some news for Color Computer and Model II owners. At this point, I hesitate to hint what it might be, because of the chastising mail I get talking about products before you can actually buy them.

I would like to encourage your comments on what subjects you'd like me to cover in this column. Would you like to hear about new products, in-depth product information, bugs and fixes, insight into our thinking and actions, or. . . .? Let me hear from you!

Computer Services (Hot Line) Expanded

Routine questions are being answered by our Computer Services group. Frankly, the merchandising team is too limited to be able to reply to those kind of questions and still bring you products.

We're aware of the difficulty some of you have had in getting through to our Computer Services group. In an effort to relieve the problem, we've recently expanded the staff from 32 to about 45 people. Since the first of the year, we've also added another 15 WATS lines, bringing the total to 41.

There have also been many requests for specific telephone numbers for business software questions. They are:

Model I/III
Business Software. 1-800-433-5641
In Texas. 1-800-772-5973
Model II
Business Software. 1-800-433-5640
In Texas. 1-800-772-5972
All other calls. 1-800-433-1679
In Texas. 1-800-772-5914

The new Business Software numbers are set up to be answered directly by a service representative, not by a receptionist. Those folks are specialists, and probably won't be able to answer any questions not related to their specific responsibilities, so don't just keep trying numbers until you get an answer. Call the proper number.

To get information to you quicker, Computer Services asks that you have the following information on hand when you place the call:

1) Your name
2) Your phone number
3) What TRS-80 computer system you're using, and the number of disk drives, etc.
4) The name and catalog number of the software package you're using and the version number of the software if you know it.
5) What error codes you've received.
6) How the error occurred, and the function you were executing when it occurred.
7) Information about any patches or program corrections you may have made.

If you have this information for us the whole system should function much more effectively. I'd also like to recommend that when the phone is answered, you ask who you're talking to. Write the name down, so you can refer back to the same person later, if you need to.

Software Progress

You might be interested to know that during March and April we began shipping 16 new software packages, and released 11 more, most of which will be shipped in May. (Color Computer ROM packs take much longer, of course). Among them were a couple of educational packages for the Models I and III; our long-waited Medical Office System for 48K, four-drive Mod I/III; our first three-disk Model II accounting package including Accounts Receivable, and the Model II Mailing List II which interacts with Profile to produce personalized form letters. We've also shipped a program called Reformatter, which allows the Model II with at least two drives and 64K RAM to transfer files be-

Continued to p. 36

80 INPUT

"If you set a transistor radio near the keyboard, an astonishing range of sound effects may be heard. . . from train whistles to birdcalls."

Paper Tiger Graphics

Bob Boothe's "Advanced Graphics Techniques—Part 2" (May, 1981) is a quality article that makes *80 Microcomputing* many times worth its cover price.

A few simple changes to the assembly language program make it compatible with the IDS Paper Tiger. Since the Paper Tiger looks only for the six least significant bits (0—5), each byte of the storage area should never contain a value higher than 63. This means more memory is required and more lines are needed to get a complete printout.

What young Mr. Boothe has given us are true hi-res vector graphics. Many plotter routines may be simply converted to the indicated format. The printouts obtained are something that will show those skeptics just what your TRS-80 can do!

One note: If you set a transistor radio near the keyboard, an astonishing range of sound effects may be heard as the program runs—everything from train whistles to bird calls!

I anxiously await Part 3 of the series.

Dan Rollins
Azusa, CA
Micronet: 70250,631

Computing Misfit

Though "A Field Guide to Computerists" (May, 1981) was a funny and interesting article, I feel it brought up one problem about the image of a kid in computers. I am 14, and have been programming for a little more than two years. Besides myself, nobody in my family knows anything about computers, so most of what I know came from experimenting and many accidental and non-accidental discoveries. In my two years, I have learned BASIC and assembly language. I recently sold programs to *CLOAD* magazine. I've mastered S-80 sound. I can make sound effects superior to all the sound I've heard in the past. I'd discovered text strings long before the many articles about them came out. I know a hundred POKEs and other tricks on my S-80 that have it doing some interesting things. I've even found a high resolution graphics trick for a 384 × 192 display, through software and *no* hardware. Some call me a Whiz Kid on computers.

Only one problem: I don't wear plaid pants and shirt, as "A Field Guide To Computerists" suggests. I wear Levi's and an OP shirt. I don't have a "plastic shirt pocket protector containing an inordinate amount of pencils." And my TI-30 calculator stays home, rather than with me in a leather calculator pouch. I quote from the article, "Don't try to find the whiz kid at sporting events, or at discos; try the local computer store or library instead." Well, I happen to love watching basketball and football, and even more, I like to play them. I play tennis all the time. True, I won't be found at a disco, because you know what they say, "Disco is dead, but rock is rolling." Rock is much better than disco. On occasion, I do stop by a computer store to buy something, but you'd be wasting your time looking for me *there*, because I'm usually not. I hate reading, except of course reading *80 Microcomputing*.

So I don't fit the image of a Whiz Kid. And I'm sure that there are other kids who don't, but are quick on the computer. By this stereotype, we can't be Whiz Kids. Do I have to dress differently, act differently, and think differently before I can become a Whiz Kid?

Ron Goodman
North Hollywood, CA

Micro 'Master Teachers'

I read with interest *80 Remarks* in the May edition of *80 Microcomputing*. As an educator and computer hobbyist I agree that the microcomputer may be one of the most important innovations happening in today's education environment. However, before the microcomputer is accepted by the better teachers and administrators, I see a great need for the makers of the programs to involve the "master teachers" in subject/content aid. I see too many good programs (from a computer program sense) with poor educational methodology. As the computer industry starts making programs for the more subjective curriculum (history, political science, language arts), you must begin to recruit some non-computer "master teachers" to aid the cause!!

Brian James, Media Specialist
Winston Churchill High School
Eugene, OR

Descending Sort Program

Once again Mr. Barden has come to my rescue with his assembly language bubble sort in the April, 1981, issue. Please keep up the good work.

Some readers might be interested to know that changing one particular byte of the sort program can cause it to sort in a descending rather than an ascending order.

To implement the change, replace location 7F52 with F2; originally it was FA.

Peter Gibbs
University of the West Indies
Bridgetown, Barbados

Erase First

I purchased a Percom TFD-100 disk drive for my TRS-80 in July, 1980. One of the reasons I purchased this particular drive was because of the claim that the drive would read and write to both sides of a diskette, thus increasing potential storage to 204K for a diskette.

I have never had any problems with the drive since it arrived, except for one small glitch: I couldn't get the drive to write to Side B of a diskette! I tried nearly everything I could think of to get that drive to write to Side B, but it totally refused. Finally, after fiddling with this for quite awhile, I simply put the matter out of my mind, figuring it would come to me someday.

One afternoon I committed the cardinal disk drive sin: I shut off the drive with a disk in it. I zipped up to Radio Shack and purchased a bulk eraser, came back home and erased the disk, front and back. I put

Continued to page 14

80 AID

Address Aid

With reference to the inquiry form A. E. Kazee in your May issue, about CRTs made by Clinton Manufacturing: The company is Clinton Electronics Corp., 6701 Clinton Rd., Rockford, IL 61111, telephone (815) 633-1444. They have a sales office in Vista, CA, at 1764 Kent Place, telephone (714) 758-3160.

A 12-inch Clinton CRT designed for a Motorola monitor is available, new with documentation, for $38.97 from Technical Electronics, P.O. Box 2361, Woburn, MA 01888, telephone (617) 935-7328. This is Clinton Part No. CE394—M12P39TE15, with 90° deflection and a 7-pin octal base. I do not know if this is a replacement for the Radio Shack CRT, nor the color of the phosphor.

G. F. McClure
1730 Shiloh Lane
Winter Park, FL 32789

Low-cost Component Interface

I just discovered Mr. Mike Bloom's letter in your February issue. I am faced with the very same problem: I use a Model I, Level II, 48K, and a MIN printer. I used it for some time in 110 bauds, then discovered how to get something reliable at 300 bauds, but it is still very slow.

Where and how could I buy the two-component interface for less than $75?

Marie-Claude Weber
23 Rue P. Brossolette
F 93500
Pantin, France

Eliminate Memory Waste

Like Joe Brandiner (April 1981), I use a BASIC software driver to operate my printer. However, I rarely have any reason to print lowercase material other than from Scripsit. I did notice that the program bombed the one time I tried, but I didn't make the connection until I read Joe's letter.

My printer is a Dynatyper from Rochester Data. It is a little slow, but it gives excellent copy on an electric office typewriter. It uses BASIC to POKE a driver program into high memory when I want to use the LLIST or LPRINT commands. Since my system has 48K, I reserve memory at 65280 before POKEing the 256-byte program.

Sure enough, ULCBAS destroys the driver program. After rebooting, I tried answering the memory size question with 28672, which is the starting address for ULCBAS. Everything worked fine, but this approach is grossly wasteful of memory.

With the data from page 4 of the ULCBAS booklet, I rebooted the system and answered the memory size question with 64500 (an easy number to remember). From available data, it appeared that I needed only 815 bytes. By changing one number in my driver program, I relocated it between 64511 and 64743, leaving 729 bytes for ULCBAS. This is still wasteful of memory, but it made changing the driver program easier: 150 P = 4•(P + 16) – 1 to P = 4•(P + 16) – 4.

Finally, I used a PEEK/LPRINT program to obtain a copy of the combined programs. The ULCBAS program was relocated just below the protected memory. The driver program was easily identified just above protected memory, followed by a new program that has little resemblance to the original ULCBAS program. I'm not sure I understand what happened, but it works, and I hope it will solve Joe's problem.

Now maybe someone can solve my problem. I have some machine language programs, including In-Memory Information, that give a "printer not ready" message, since I am using a non-standard printer connected to the TRS-80 bus. I would like to know how to defeat these printer signals or alter the programs.

Scott Smith
2919 26th Ave. West
Seattle, WA 98199

Tab Solution

I have noted many people trying to find solutions to the LPRINT TAB(63 +) problem. The solution that I received from Fort Worth over two years ago (and forgotten by them) is the following:

```
STRING$(T – PEEK(16539),32)
T = tab setting
Sample: LPRINTSTRING$(40 – PEEK(16539),32)
"40";STRING$(80 – PEEK)1653),32)"100"
```

All items tabbed as above will appear in the same column. I even used it to tab to 162 with an IBM Selectric. I use this for all my programs and have had no trouble at all.

Lou Wiener
20776 W. Plum Cyn. Road
Saugus, CA 91350

English Aid

I am interested in software applications in Life Insurance, Training Management Games and Computer Assisted Training. I would appreciate any information anyone has on programs, research, or contacts from the United States with similar interests.

A. E. Sheil
Refuge Assurance
Oxford St.
Manchester, M60 7HA
England

Hi-Res Graphics Games

I'm looking for hi-res graphical games for the TRS-80 Level II 16K. They could be run on the following products: Percom's Electric Crayon, Programma's 80-Grafix Board, and E/RAM.

Percom's Electric Crayon gives you hi-res graphics using a 256 × 192 pixel

Continued to page 18

the disk back in the drive, reformatted it, and, purely by chance, pulled it out, flipped it over, and tried to format Side B. Imagine my utter astonishment when it worked!

I immediately took a blank, never-before-used disk and erased both sides and formatted it. Again, it worked. Since that day, I have always first erased the new diskette, and I have never had an ounce of trouble reading or writing to both Side A and Side B.

I'm happy to report that I'm finally getting my 204K's worth of storage.

Michael K. Salsgiver
Portland, OR

RSM-2 for Model III

This short note may be of interest to new Model III owners who have Model I experience. I, as many others, have found that Small System Software's RSM-2 monitor is a very valuable tool for debugging and otherwise studying software problems. Small System Software is working on a version for the Model III, but the release date is unknown. However, a cassette tape version of RSM-2, or RSM-2D, will load and execute many of RSM's functions in a Model III. Since the Model III is a port-based system, whereas the Model I is memory mapped for everything but cassette, the printer and disk operations have been changed. Also, the difference in baud rates means that RSM-2 cannot *read or write* the 500-baud tapes on the Model III, and likewise that the Model III System command cannot read tapes written by RSM-2 on a Model III.

However, there is a simple way to alleviate this problem using RSM-2 itself. All you need to do is change four memory locations using the RSM-2 edit command:

Address	Change From	To
6CBD	81	92
6CC7	0F	11
6CCF	0F	11
6CD7	60	6C

These addresses are for a 16K version of RSM-2. For a 32K version the addresses are 4000 (hex) larger, that is, they begin with A, and in a 48K system they begin with E. The last three digits are the same for all three sets of numbers. These values reflect the faster clock rate in the Model III. With the memory changed, you can now use RSM-2's P command to write a system tape that will save the changes. You will find that you will have to start and stop the cassette yourself, as that is also done differently by the Model III. Nevertheless, this simple change will let you have most of the features of RSM for a Model III until Small System Software releases the complete package.

Maynard B. Neher
Columbus, OH

Software Incompatibility

After many, many moons of waiting, including numerous periods of total frustration, I finally received our first piece of Radio Shack software specifically written for the new Model III: Scripsit (cassette version).

Anyone who has tried knows that the older Model I version (regardless of what RS advertising or storepersons state) will load but not work . . . and we spent considerable time and long-distance phone calls, not to mention repeated trips to no less than three RS Computer Centers, trying to separate the "fly-specks from the pepper", so we were understandably quite elated when our tape finally arrived from Fort Worth. (We finally went direct to solve our problem . . . and I must admit, the Computer Services people were great!!!)

Much to our delight, the program is a gem! Everything works! We were, however, suprised to find the cassette manufactured to load at the low 500 baud speed instead of the newer 1500 baud capability inherent in the Model III. And in investigating why, I was advised by Radio Shack at Fort Worth that they: "*Do not plan to support the 1500 baud cassette rate with any of the RS software created for the Model III.*"

Is Radio Shack telling me that the single-most valuable difference (for us non-diskers) between the Model I and III is not going to be utilized? I'd sure like Mr. Juge or one of his people to address this small question. One could almost start believing this is RS's not-too-subtle way of getting me to buy a disk!!!

Larry M. Mohr
Design Systems
Kankakee, IL

Radio Shack Replies

80 Microcomputing *has forwarded your recent letter to me, regarding your complaint on our Model III software. I'm sorry you had the problem you had, and I'm pleased that Computer Services was able to help. I'll be happy to address your question.*

First of all, let me assure you that we do intend to support the 1500 baud Model III tape format. Your tape was created before we had generated the 1500 baud version of the in-house software we use to verify cassettes. Rather than take a chance, we produced them at 500 baud.

Our November 1980 newsletter dealt very thoroughly with Model I/III program compatibility, detailing which programs will and which won't work, as does a sheet shipped with each Model III (which I must assume you didn't receive). We specified those requiring modification, and those (including Scripsit) for which a Model III specific version would be required. I believe some 20 percent of Model I software turned out to be not compatible or need modification or revision.

We regret the incompatibilities that were necessary between Models I and III. Our hope had been total compatibility, but we weren't designing a "warmed-over" Model I, we were trying for an enhanced and improved computer. Some concessions were unavoidable. We've also invested considerable time and expense upgrading existing warehouse stocks of Model I programs as Model III compatible versions have become available.

Ed Juge, Director
Computer Merchandising
Tandy-Radio Shack
Fort Worth, TX

Disk BASIC Adaptation

I just finished the article "A Turn of the Screw" in the April 1981 *80 Microcomputing.*

After typing in and attempting to run program seven on page 123, I found that it would not work with TRSDOS BASIC and a 48K machine.

After some debugging I added an additional line to the program (245 DEFUSR = &HFE53) so it would work under Disk BASIC.

The hex address in the above might have to be changed for different size systems. It's used on my 48K machine.

The program listed in the magazine works fine with Level II BASIC.

Richard P. Stiles
Windsor, CT

Know-It-All?

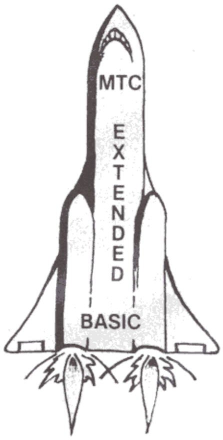
Alan Sehmer's VARDOC2 ("Know-It-All", May 1981) is a super program, albeit (as he admits) rather slow. I'm not sure whether I did something wrong, but I couldn't get it to work until I changed line

Continued on page 19

META TECHNOLOGIES

26111 Brush Avenue, Euclid Ohio 44132
CALL TOLL FREE 1-800-321-3552 TO ORDER
IN OHIO, call (216) 289-7500 (COLLECT)

TAKE YOUR MODEL II TO NEW HEIGHTS



MTC is proud to announce MTC EXTENDED BASIC for the Model II, by R. Ryen. Features include "fixes" to existing BASIC, multi-line functions, extending an existing sequential file, PEEK, POKE, greatly enhanced screen control and expanded editing capabilities. The contents of variables are NOT CHANGED when editing, deleting, inserting or merging lines, allowing continued program execution! All this and much more.

MTC EXTENDED BASIC................ $ 99.95

80 – 20 = 129.95

THE ORIGINAL NEWDOS/80

WITH MTC QUE CARD

★ SPECIAL PRICE ★

SAVE $20 **$129.95**

Apparat's long-awaited successor to NEWDOS+ is here! This is not an enhanced version of NEWDOS, but a completely new product. Simplified DOS commands can be instantly executed from BASIC, even within a program, without disturbing the resident code. System options, such as password protection, number and type of disk drives, BREAK key enable/disable and lowercase modification recognition, can be quickly and easily changed. Five new random-access file types allow record lengths of up to 4096 bytes, and no FIELDing! A powerful CHAIN facility allows keyboard INPUTs to be read from a disk file. An improved RENUMBER facility permits groups of statements to be relocated within program code. Diskettes may even be designated as RUN-ONLY! Features all NEWDOS+ utilities (SUPERZAP 3.0, etc.) and much more! One MTC technical staff member said having NEWDOS/80 is "better than sex" (you'll have to judge for yourself!). Includes 180-page instruction manual and MTC QUE card.

NEWDOS/80 # SPECIAL # $129.95
MTC QUE Card only .... $ 7.50

CALL REGARDING OUR NEWDOS+ UPGRADE PRICING

MODEL III VERSION

Has all the features of the Model I version plus enhancements. Allows any mix of single- or dual-sided 40- or 80-track disk drives. Most BASIC and many machine code programs written for the Model I will run without modification. Includes a utility for converting Model I single density to Model III double density.

NEWDOS/80 for Model III $149.95

CALL FOR AVAILABILITY

MICROPARAPHERNALIA

DISKETTES (box of ten)

5¼" PLAIN JANE™ $21.95
5¼" PLAIN JANE™ *Gold* $25.95
5¼" DATALIFE™ MD 525-01 .. $26.95
8" DATALIFE™ FD34-8000 .... $43.95

NEWDOS by APPARAT

NEWDOS/80 by Apparat $149.95
NEWDOS+ to
NEWDOS/80 UPGRADE ..... CALL
NEWDOS+ with ALL UTILITIES
35-track $69.95
40-track $79.95

BOOKS

TRS-80™ DISK
AND OTHER MYSTERIES .. $19.95
MICROSOFT™ BASIC DECODED $29.95
1001 THINGS TO DO WITH YOUR
PERSONAL COMPUTER .... $ 7.95

Michael Shrayer's

ELECTRIC PENCIL VERSION II

for
Model I and Model III

An expanded version of the critically acclaimed original word processing system! Includes all features of Version I plus many new extensions. Runs under most disk operating systems, has improved video text handling, loads any ASCII file for editing (including BASIC files), single sheet mode for printing on letterhead and more! Simple to use, features 2-key commands. An incredible package at an incredible price!

SPECIFY MODEL I OR III
Disk Version $79.95
Tape Version $69.95

CALL FOR AVAILABILITY

Complete for Model I with all utilities
Plus exclusive MTC QUE card!

NEWDOS +

$69.95 by Apparat

Includes REF, RENUM, SUPERZAP, EDITOR/ASSEM., DISASSEM., DIRCHECK, and more! This is the original NEWDOS with all of Apparat's utility programs. Includes exclusive MTC QUE (Quick User Education) card.

40-Track Version $79.95
MTC QUE Card only .... $ 1.50

MORE PRODUCTS

TRS-80 is a trademark of the Radio Shack Division of Tandy Corporation. DATALIFE is a trademark of VERBATIM. PLAIN JANE, AIDS-I, AIDS-III, CALCS-III, CALCS-IV, MERGE-III are trademarks of MTC.

MOST ORDERS SHIPPED WITHIN ONE BUSINESS DAY

Products damaged in transit will be exchanged.

PRICES IN EFFECT July 1, 1981 THRU July 31, 1981

Prices, Specifications, and Offerings subject to change without notice.
8107

WE ACCEPT
- VISA
- MASTER CHARGE
- CHECKS
- MONEY ORDERS
- C.O.D.

- Add $3.00 for shipping & handling
- $3.00 EXTRA for C.O.D.
- Ohio residents add 6½% sales tax.

80 DEBUg

Babydub Flub

It was bound to happen. Total incompatibility, and my mailbox shows it. Babydub, March, 1981, just won't work on machines which have the Radio Shack cassette modification installed, or which are running under just about any DOS. These improved tape loaders are more stern with their timing than the original CLOAD routine. The timing after the synchronization byte is just a few microseconds too fast for it to catch, so a delay must be inserted in the program. Okay, so make the five changes below:

```
LINE 10:  CHANGE 20701 TO 20711.
LINE 90:  CHANGE THE 8TH DATA ITEM FROM
          208 TO 218.
LINE 100: CHANGE THE 2ND DATA ITEM FROM
          179 TO 189.
LINE 170: CHANGE THE 7TH DATA ITEM FROM
          193 TO 203.
LINE 200: AFTER THE 3RD DATA ITEM (WHICH IS
          79), ADD THESE TEN ITEMS:
          245,197,1,32,0,205,96,0,193,241
```

If you're entering the hexadecimal version, change byte 4301 from B3 to BD; byte 4376 from C1 to CB; and starting at 43A3, insert these ten bytes: F5 C5 01 20 00 CD 60 00 C1 F1. Move the remainder of the program along ten places.

Also, high-memory freaks should note that you can't relocate this program to high memory without crashing it. Why? Because it reads the tape information into memory starting at 7FFF and filling memory backwards. If you relocate it to high memory, it will wipe itself out within seconds.

Dennis Kitsz
Roxbury, VT 05669

Disk Error

There are several potential problems with the program listed in my article "Sans Disks" (April 1981).

1. When using the Select If, Change If, or Delete If commands, don't use spaces just before or after the =, >, <, or <> symbols. That will cause errors. For example:

SELECT IF NAME = JONES	Is wrong.
SELECT IF NAME=JONES	Is correct.

2. When records are being entered, there is no indication when memory is full. To correct this add the following lines:

```
305 IF(D2+(F2-1))<32767 THEN 310
308 D2=D2-.1:IF PEEK(D2)<>2 THEN 308 ELSE
    POKE D2+1,5:PRINT "MEMORY IS FULL":
    GOTO 360
```

With 16K memory use 32767 in line 305. With 32K, use −16385 and with 48K, use −1.

3. The program was written only for 16K memory. For 32K or 48K make the changes to the lines indicated in Listing 1.

Stewart F. Hunter
15510 Murray Hill
Detroit, MI 48227

Wherzit Fixit

I am the proud owner of a TRS-80 Model III. Since I'm also a novice to microcomputing, I have difficulty spotting bugs in my programs. Having entered the "Wherzit" program on page 252 of the April issue, I discovered, much to my dismay, that not only was line 83 missing entirely, but also that the delete (/) function was incomplete as printed.

Any time a record is deleted from storage, it is duplicated at the end of the file and the first record disappears. Line 69 is the offending line in the program. To remedy the situation, I have added this statement at the beginning of line 69: R$(N) = R$(1).

It sets the first record in the file equal to the Nth record, thereby protecting it from oblivion (when the rest of the line alters the file record sequence).

The duplicate of the deleted record is not saved to tape and is over-written by any subsequent record additions. Line 83, as I constructed it, reads: 83 N = N + S:GOTO 63.

Another typo is the omission of a semicolon at the end of line 64 before A$(8).

Colin Alexander
120-28th St.
San Francisco, CA 94131

Continued to page 20

```
94   DATA0,58,224,116,254,1,202,10,117,33,1,128,205,132,2,126,35,205,100,2,254,4,194,239,116,33,
     232,131,126,35,205,100,2,254,5,194,252,116,205,248,1,201,205,147,2,33,1,128,205,53,2,119,
     35,254,4,194,16,117,33,232,131,205,53,2,119,35,254,5,194
127  D1 = -32767
235  D1 = -32767:D2 = -31767
330  IF PEEK(D1) = 4 THEN D1 = -32767:POKE D2,2:D2 = D2 + 1:PRINT:GOTO 240
500  D1 = -32767:D2 = -31767
640  D1 = -32767:CT = 0
660  D2 = -31768
705  DST = -31767
805  D1 = -32767
900  D1 = -32767:FORI = -31767TO -16386:IFPEEK(I)<>5NEXT:ELSE905
910  X = 29610:Y = INT(X/256):Z = X - (Y*256):POKE16526,Z:POKE16527,Y:HH$ = MID$(CO$,9,244):
     D1 = -31767:CT = 0:GOSUB9800
925  RP = -31767:S = 0
9102 D5 = D1:D1 = -32767:GOSUB9200:GOSUB3000:MS$ = F1$:GOSUB9600:MS$ = "=":
     GOSUB9600
9800 C = 0:FI(0) = 0:FORI = -32767TO -31768:IFPEEK(I) = 4THEN9810
9810 S = C/20:D3 = -32767:FORI = 1TOS:D3 = D3 + 16
9903 IFCT = 1THEND9 = D2:II = 1:GOTO9906
9906 IFD9<0 THEN D9 = D9 + 65536
```

Listing 1. Sans Disks Corrections

65095 to read:

```
65095 . . . ELSE IF C$<A$(J) THEN 65125 (not C$>A$(J)).
```

Also, the readout was in reverse alphabetical order until I changed line 65160 to read:

```
65160 FOR R=A1 TO 1 STEP-1: LE= . . . etc.
```

A change in 65085 will tell whether arrays are single or multi-dimensioned:

```
65085 IF T=40 AND (PEEK(F+2)=44 OR PEEK(F+3)
=44 OR PEEK(F+4)=44) AND PEEK(F+2)<>41 AND
PEEK(F+3)<>41 THEN C$=C$+"(#,#)" ELSE IF
T=40 THEN C$=C$+"(#)"
```

Richard R. Losch
Salem, MA

Small Business Needs

Your publication regularly asks for articles by businessmen on their use of small computers. I know next to nothing about them, but I do have a complete Radio Shack Model II System installed in my small business and thought that a letter would suffice as input from a typical user.

We are a small manufacturer of photograhic equipment, with about 20 employees and sales of a little under $1 million.

Our system includes the 64K computer, an expansion drive and the Line Printer III. We use all standard Radio Shack software including general ledger, payroll, accounts receivable, accounts payable, mailing list and profile.

After using this system for just about a year, we feel quite comfortable with it and accept the fact that off-the-shelf will never be more than 80 percent of what you might want. However, that is an acceptable compromise. Our problems have generally centered around repeated disk failures which were finally solved by switching to Verbatim disks. In our application at least, these seem to be the most dependable.

While those who write the programs are undoubtedly much smarter than I am, they seem to know little about the routine needs of business. For example, programs such as accounts receivable and accounts payable simply must have a mailing list option where you can easily print out customer names and addresses on standard mailing labels. Every business has to send out price announcements, special promotions, notification of vacation closings, etc. It is frustrating to have to type out all of the customer or vendor names and addresses when these are already in the computer. In short, those suppliers who intend to provide accounts receivable and accounts payable programs for the small businessman must incorporate an address label option with

80 AID

screen with the TRS-80 Level II 16K. It also includes eight different colors. It costs $249.95.

Programma's 80-Grafix Board gives you hi-res graphics using a 384 × 192 pixel screen; it's in black and white. I don't recommend this product because it goes inside the keyboard, and that would cancel Radio Shack's guarantee.

E/RAM gives you the same graphics as the Electric Crayon, but no color. I think it is priced too high for its performance. It costs $349.95.

I have been reading your magazine for about four months now and haven't seen any programs except for the hi-res add-ons. It really is a shame that the TRS-80 can have better graphics than the Apple II+, or almost as good, and there aren't any programs out for them. I want to see that changed. By having hi-res programs for the TRS-80 there will be more people buying the TRS-80, this magazine, the products above, and software in hi-res.

The TRS-80 may have the most games out on the market, but it's time for a change in TRS-80 gaming. TRS-80 owners could stand high and proud knowing they have the best home computer on the market. The United States needs a home computer like this to keep us ahead of the Japanese market. The TRS-80 can be that computer and more with your help.

Sean Hockabout
210 Ironwood Road
Alameda, CA 94501

Sean tells us he's 13 years old.-Eds.

Computerese

I would appreciate any information concerning a computer program reported in *Computer Power For The Small Business*, by Charles J. Sippl and Fred Dahl and published by Prentiss-Hall, page 156, developed by Mr. Ashok Nagrani and called "SPREG."

Quote from the book:

> You might become so proficient at programming that you will come up with a program as clever and as daring as that of Mr. Ashok Nagrani who invented SPREG. Using Altair's Extended BASIC version 3.2, he composed a program that would spout "Computerese," a language very similar to that used by politicians, diplomats, and double talkers who find themselves pressed for an honest answer. Drawing on a data base of nouns, verbs, adjectives, and other parts of speech, the program structures them in a way that is grammatically correct (well, almost) but essentially meaningless. The beauty of the program is that it can generate up to a trillion such meaningless sentences without ever repeating itself.

If you have any information concerning the above we would appreciate your notifying us.

Sam W. Allred, Administrator
Capitol Home Health Agency
307-A Clinton Blvd.
Clinton, MS 39056

On First Command

After having read so much about KBEEPFIX I was delighted to find the disk version featured in the March issue. Since the article stated this particular version was identical to the original, save for a couple of minor code changes, I found it easy enough to modify so that it should be the same as the original. I deleted the first five bytes, changed the jump in 7F95 to jump to 1A19 (BASIC's starting point) and changed the instruction in location 7F91 to 7F, as I have a 16K system.

When I loaded the T-BUG-created system tape and tried it out, I found I got all the benefits claimed. However, I have to issue a New or Clear as my first command, or I will get an OM error on entering any other command as my first command. My question is whether this is normal, or a problem with my RAM or program. Although the problem is minor and easily circumvented, it is still an irritation. Any help you can give me would be appreciated.

Lastly, I would like to see more articles on assembly programming, especially articles dealing with uses of ROM routines.

Bernard F. Gaffney Jr.
524 Riley St.
Lansing, MI 48910

NEVER UNDER$OLD.

Thats right. If you can find a lower price in this magazine WE WILL BEAT IT! Period.*

SPECIALS

SPECIAL #1 . . . . $44.50
TRS-80 Disk and Other Mysteries Book, Box of Verbatim Diskettes, and a Plastic Library Case.

SPECIAL #3 . . . . . . $149
Apparat NEWDOS/80, Box of Verbatim Diskettes, and a Plastic Library Case.

SPECIAL #4 . . . . . . $190
Microsoft Basic Compiler, Box of Verbatim Diskettes, and a Plastic Library Case.

SPECIAL #5 . . . . $49.95
Microsoft Basic Decoded and Other Mysteries, Box of Verbatim Diskettes, and a Plastic Library Case.

PRINTERS

EPSON MX-80 . . . . CALL
MX-80 Graphics ROM . CALL
NEW EPSON MX-80 TFCALL
Okidata Microline 80 . . $415
Okidata Microline 82 . . $625
Okidata Microline 83 . . $899
IDS Paper Tiger 460G$1050
Anadex DP-9000,1 . . $1195
Anadex DP-9500,1 . . $1295
Centronics 737 . . . . . . $689
NEC Spinwriter . . . . . . $CALL
NEC w/ Sellum Option . $CALL
Starwriter 25 cps . . . $1440
Starwriter 45 cps . . . $1770
Malibu 165 . . . . $1995
MPI 88G $495

TRS-80 HARDWARE

TRS-80's . . . . $CALL
Micropolis 77 track . . . $399
Percom Doubler $199
Percom Separator $27
Orchestra 80 . . . . $79
Shugart 40 tr. $299
Pertec 40 tr. $299
MPI 40 tr. $299
16K Memory Kit . . . $19.95
Cat Modem $145
D-Cat Modem $155
Lexicon Modem $125
Verbatim 525-01 . . $26.50
Verbatim 8" DD $44.00
Isolator ISO-2 . . . . . . $49.95

NEW! Double Density Special

A Percom Doubler, Apparat's NEWDOS/80, Double Zap II (to convert the NEWDOS/80 to single and double density), a box of Verbatim Datalife diskettes, and a Plastic Library Case. A $436 Value...for

$359

Send For Our New Free Catalog!

*Include $2.00 shipping and handling on all orders. Visa and Master Card accepted. Never undersold offer applies only to items currently deliverable from other retail vendors at advertised price. All never undersold offers good as supply lasts. Please add $2.00 for all COD orders. Please call for items not listed. We gladly answer any questions on all of our hardware, software, and supply needs. Quanitity discounts available. School purchase orders accepted. Please remember to figure competitors shipping and handling charges when arriving at never undersold price.

POWERFUL SOFTWARE

Racet's New DISCAT $45
This New machine language disk catalog program will help you keep track of your programs in a neat, orderly library. It can catalog up to 1900 programs in each of 9 catagories for a total of 17000 program location records.

Snapp Spooler $90
Fully automatic print spooler for the model I. Increases your computers efficiency by freeing the computer while printing.

TRACKSESS $24
This is the complete disk zap utility, capable of copying standard and non standard formatting. The dynamic editing allows you to change any byte in any sector with amazing ease

SCRINPUT $29
Format your screen displays and input routines with this machine language subroutine. Gives your program a more professional look.

WORD PROCESSING

SPECIAL DELIVERY $118
The best in mailing list programs for the TRS-80 Models I & III. Special Delivery is easy to use & is written in fast machine language. Special Delivery allows you to merge your mailing list file with text files written with the Electric Pencil® or Scripsit® to create personalized form letters.

XTRA SPECIAL DELIVERY $189
Does everything special delivery does and more. Extra allows you to include special keyed messages in your text file, and includes Mailsort which allows sorting on multiple fields.

PROSOFT $79.95
'PROSOFT' Word processing software is written for the TRS-80 and the Centronics 737. Takes full advantage of all the capabilities available on the 737 including true proportional spacing. Comes complete with a 110 page users manual.

213-883-8594

ALPHA BYTE STORES

-69

5115 Douglas Fir Road
Suite B
Calabasas, California
91302

their software.

Another thing that is annoying, although less so, is that the basic menu formats and operating procedures will differ substantially for different programs from the same supplier (Radio Shack). On one program you type in the password and you see it on the screen. On another program you type in the password and all you get is a series of number signs so that someone viewing the screen can't see the password. It would be a lot easier if the programs were consistent so that an operator wouldn't have to follow different procedures for different programs. Similarly, on some programs you use the Break key and on other programs this would court disaster. Again, if these programs were from different manufacturers it would be understandable, but coming from the same source they should be similar.

Finally, when printing out several reports, an option that lets you initialize the accounts receivable program and indicate which type of paper or form should be used would be a nice touch. If you could put in a 10 or 12-letter identifier next to each report on the menu, the operator would immediately know whether to use three-part paper, invoices, labels, etc.

Steven Hess
President, Saunders Photo/Graphic
Rochester, NY

Backup Change

I found the article "Backup/Display" by Craig Lindley in the May 1980 issue very useful, but had some difficulty with the printout part, because I have the official Radio Shack lowercase modification.

When the transfer is made from the video memory, I lose all the alphabetic characters. After some experimenting, I found that the alphabetic characters are stored on the screen in the range 00-1F hex, so the trick is to add 40H to all characters below 20H as they are brought from the screen.

The following changes to the code in the article should correct things:

1. Change line 1400 to read:

```
1400 LDATA   LD    A,(HL)
```

2. Add lines 1402-1408 as follows:

```
1402         CP    20H
1404         JR    NC,GT20H
1406         ADD   A,40H
1408 GT20H   LD    C,A
```

After this change, I had no further problems, and I am now busy making backup copies of all my programs.

George Rogers
St. Laurent du Var, France

80 DEBUg

Cursor Correction

In my article "Block That Cursor" (April, 1981), it seems that the Tab function doesn't work with the block cursor running.

Fortunately, the fix is as easy as swapping a couple numbers around in the data statement. The first five numbers of line 50 should read as follows:

```
50 DATA 205,88,4,245,197—all the rest are the same.
```

An updated listing to the program with the corrections made follows.

Ron Balewski
412 E. Ridge St.
Nanticoke, PA 18634

```
1 REM ***** BLOCK CURSOR PATCH
2 REM BY RONALD A. BALEWSKI
10 FORK=32635TO32654
20 READX
30 POKEK,X
40 NEXTK
50 DATA205,88,4,245,197,237,75,32,64,10,
   254,95,32,3,62,143,2,193,241,201
60 POKE16414,123
70 POKE16415,127
```

Block That Cursor Correction

Ledger Tape

A number of readers have experienced trouble making the initial run of my program, "The General Ledger" (page 222, May, 1981). The program is designed to run with a data tape, but if a blank tape is used as a data tape "FD Error in 3000" will be displayed.

To make a suitable data tape with zero balances in the various accounts the following procedure may be used:

1. Enter the program.
2. Place a blank cassette in the recorder and set it to record.
3. Key in: RUN 620, and Enter.
4. Press Enter again. The recorder will then record the various accounts with zero balances. This tape may then be used as a data tape to run the program initially.

If the user wishes to start entering account data without making a zero balance data tape, he can load the program and enter: RUN 220. This data can then be recorded after a trial balance of the new data has been run.

Also note that a number has been left off in line 1340 of the published program. The end of the line should read: If X = 1 GOTO 130.

R. L. Conhaim
15506 Kiamichi Road
Apt. 1
Apple Valley, CA 92307

Car Error

One of your readers has found an error in my article, "The Auto Mentor," in the May 1981 issue. The error is in lines 1220 and 1230. The correct lines are:

```
1220 X1 = DR - (X(21) /12/ 100) :'EFFECTIVE RATE
     FOR OLD CAR FUEL
1230 X2 = DR - (X(22) /12/ 100) :'EFFECTIVE RATE
     FOR NEW CAR FUEL
```

The time to replace the old car in the example with these changes is 49 months.

Leslie E. Sparks
1014 Evergreen Drive
Durham, NC 27712

Once Again

The "80 Input" department of your April issue published a comment from me with a fix for the chi square program of the Radio Shack Advanced Statistical Analysis package. Unfortunately, some of those pesky parentheses in the fix were not typeset accurately, so the correction won't work.

The proper version of the fix for the last statement in line 280 is: CS = CS + (ABS(O(I,J) – E(I,J)) – CC)↑2/E(I,J).

Alfred L. Brophy, Ph.D.
Director, Guidance Exchange
421 Mackenzie Drive
West Chester, PA 19380

THE ALPHA I/O SYSTEM

A COMPLETE FAILURE?

It happened 3 years ago, when our President made a decision. At the time we specialized in custom analog and digital circuit design. The decision was to attempt to develop a line of standard interface hardware for the emerging microcomputers. At the time (1977) we had to decide which of the new machines could become the "industry standard" of the low cost micros.

Despite a few aggravating but minor deficiencies, the TRS-80 seemed tohave the most chance of success and it had the best price/performance ratio. Also, with some imagination, their large sales organization could become the largest service network in the world, a reassuring throught for the many novices in this new field.

It became clear that the TRS-80 could be used (with our then hypothetical system) to solve problems in many fields where computers were not yet used, mostly because of their high cost.

The IDEA was simple! ALPHA PRODUCT would supply the missing link between the TRS-80 and the "outside world". (more about this "outside world" later).

DANGER! If Radio-Shack entered the same market, we probably would not have survived, but the expectation was that they would be too busy developing their basic line (drives, printers, modem etc.). Thanks to our more specialized products, we would not be competing with them.

BAD START! We began with a failure. Our first product was supposed to be a simple, low cost, general purpose device. It would allow the TRS-80 to accept inputs other than the keyboard. Many kinds of external devices (the "outside world" mentioned before) like photocells, sensors, thermostats, switches, contacts, etc., could be connected easily. In addition, there were two relays to control (on or off) external loads such as motors, lamps, appliances, heaters, etc., etc.. In other words, it would allow the computer to interact or interface with external devices. We called it the INTERFACER 2. What a mistake! It sounded too much like "expansion interface". Many enthusiastic TRS-80 users called thinking that our "INTERFACER 2" was a low cost Expansion Interface (at $85 that would have been a real bargain!). We wanted to change the confusing name. That meant reprinting the manual, changing the ad, scrapping the flyers, discarding the silk screened cases. Well, "INTERFACER 2" it would stay.

TROUBLE! We also found that the majority of TRS-80 users were AFRAID of the hardware. They could be very comfortable with fancy programming but thought you had to be a computer specialist or technically inclined to put the INTERFACER 2 to work. In truth, some IMAGINATION and a SCREWDRIVER is all you really need. Anyone able to wire a switch could use this device.

WORSE! There was also the fear of plugging a "foreign device" into the precious computer. This notion has all but disappeared as there are now so many quality products designed for the TRS-80 that plugging in a non Radio-Shack device has become common.

Our ad in Creative Computing (80-Microcomputing did not yet exist) hardly paid for itself. We had a decision to make. Were we wrong or just too early? Our first INTERFACER 2 was sold to someone who wanted to, and succeeded in, controlling his fancy model railroad with his TRS-80. Interesting, but what made us stick with the concept was that some of our INTERFACERS began finding use in applications with fascinating possibilities. Space is lacking to describe them, but the most exciting was the successful use of the system in assisting a handicapped young boy. We were pleased to hear of such a meaningful application.

Three years later, as you can see in our ads, The INTERFACER 2 is alive and well. The price went up a bit, and despite the introduction of the more powerful INTERFACER 80, the sales have been steady.

Then came the least understood product! the ANALOG 80. This $139, nicely designed module is an Analog to Digital converter with 8 input channels. Used with your TRS-80, it provides a powerful "data acquisition system". This jargon simply means that you can monitor, measure and record 8 independant varying voltages. Very few people realized its real power. Such a system would have cost over ten thousand dollars just a few years ago.

The possibilities in scientific and engineering environments are endless. This system could replace chart recorders, digital data recorders, programmable calculators, data analyzers and many other specialized and expensive pieces of equipment. Furthermore, up to 8 ANALOG 80's could be used simultaneously for a total of 64 channels of analog input! They simply plug into the TRS-80 using our "X" series of bus extenders (EXPANDABUS).

Our next product was to be a second generation, Input/Output interface, with more flexibility than the INTERFACER 2. Careful design and refinement yielded the INTERFACER 80, the most powerful real world interface on the market today. It has 8 inputs, each optically-isolated and 8 outputs, each with a relay contact. The INTERFACER 80 is fully compatible with our ANALOG 80, allowing these to be used together in order to create systems that control external devices based on "sensed" input under control of the TRS-80.

A FAILURE! in spite of our extensive advertising, very few are aware of the existence of the powerful ALPHA I/O SYSTEM.

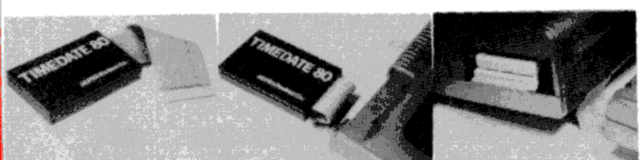
THE FACTS ARE:

- The ALPHA SYSTEM/TRS-80 combination forms an incredibly versatile and powerful tool for acquisition/processing/control.
- In spite of its moderate cost, the system is sophisticated and reliable.
- The entire system can be easily programmed in BASIC using INP(X) and OUT X,Y commands.
- The modular approach and our EXPANDABUS allow for instant expansion as requirements demand.

The following pages contain more information about the devices mentioned here. We invite you to call or write to discuss your particular application.

TIMEDATE 80

WHY LOSE PRECIOUS TIME ?



Neat, Compact Design 3 Years Battery Life | Slips Inside E/I (Y Option Shown) | Real Time Without Expansion Interface

- Complete, self contained "true" real time clock/calendar, TIMEDATE 80 continues to keep accurate time and date when the computer is turned off or experiences a power failure.
- **TIMEDATE 80** only needs to be set once, and it's two replaceable "AAA" batteries (not included) keep TIMEDATE 80 running in excess of 3 years. Costly Ni-Cad batteries and charging circuits are eliminated.
- The instant power is applied to the TRS-80, TIMEDATE 80 provides MO/DATE/YR, DAY of WEEK, HR:MIN:SEC and AM/PM information with quartz accuracy.
- **TIMEDATE 80** replaces the computer's internal clock. Extremely useful for automatic operation of remote systems with no operator in attendance. If the power fails and then is restored, only TIMEDATE 80 will update the system with current TIME and DATE information, an impossibility with the computer's internal clock.
- **TIMEDATE 80** is quartz crystal based with INTELLIGENT CALENDAR, including provisions for leap year! TIME display may be by 12 hour AM/PM or by 24 hour military and Eruopean format.
- **TIMEDATE 80** plugs directly into the rear of the TRS-80 keyboard and gives the "TIME$" function even without an Expansion Interface. For those with a disk system, it plugs into the left side panel of the Expansion Interface. An optional "Y" connector can provide for further expansion.
- **TIMEDATE 80's** small size keeps the computer table uncluttered. If you have an Expansion Interface, TIMEDATE 80 literally "DISAPPEARS" by slipping into the empty space in the bottom of the interface.
- Two sets of software, on cassette, come with TIMEDATE 80—"TIMESET" and "TIME$". "TIMESET" is a step by step set of simple instructions for setting TIMEDATE 80, "TIME$" is a set of poke routines which patch DOS and Level II TIME$ to read TIMEDATE 80 and is easily incorporated into any user software. "TIME$" will always print the time and date when LISTING a program—great for keeping track of revisions!
- Other valuable uses for TIMEDATE 80 are: accurate date and time information for business reports like payroll records, financial reports, etc., or to various I/O devices requiring 24 hour clock input, such as laboratory instrumentation, and to communication systems needing "Log In/Log Out" data (bulletin boards).
- **TIMEDATE 80**, fully assembled and tested, 90 day warranty, complete with instructions and software on cassette, $95.00. "Y" option, add $12.00

PRINTER STAND

$59.95

PRINTER CABLE EXTENDER

Adds 4 ft. to your existing cable, printer extension connects between Exp. Int. and your present printer cable........$27.50

✓210

ALPHA Product Co.

85-71 79 St. WOODHAVEN N.Y. 11421 ☎ Info and order: (212) 296•5916

80 REVIEWS

Edited by Pamela Petrakos

"The primary message is clear: to avoid time-consuming costly pitfalls when acquiring a first computer system."

So you are thinking about a Small Business Computer
Canning Publications
Vista, California
Softcover 100 pp.
$12.50

by David W. Smith

Here is a manual designed to help the person who is unfamiliar with computers learn how to select a small computer system successfully. This is a highly organized book and the reader should have no difficulty following and comprehending the material.

The book is written for the small business owner who possesses a limited knowledge of computers, but who has at least one problem or task he/she believes a computer can solve or perform. This manual presents full coverage of the basic information you need for making an intelligent decision on system acquisition, without getting caught up in inappropriate technical detail.

The primary message is clear: to avoid time-consuming and costly pitfalls when acquiring a first computer system. This requires that the business person do his/her homework. Before seeing a single task performed by the newly installed computer, you will need to assimilate a fair amount of information on the subject. This manual contains most of the information you will need to know, and lists references for all the rest.

Dispelling Myths

In preparing you for the learning task ahead, the authors enumerate many of the benefits to be gained by computerization. This is no pie-in-the-sky outlook, but rather a highly realistic analysis. Warnings and cautions are given to help dispel common myths concerning computers. For example, the authors point out that anticipated savings in payroll frequently do not occur at first in the small business just because a computer has been put into service. Such a business usually has too few employees for the computer to replace any one of them entirely. The savings in payroll is likely to occur, however, when business volume begins to grow, and the company finds it can do without proportional increases in staff. This is only one of the many illustrations given to help you develop a truer sense of the potential impact of the computer on your business.

The authors are thorough in their coverage of important basic information—how a computer works; what constitutes hardware and software; the differences among maxi, mini, and microcomputers and their manufacturers and sales procedures. A glossary of common computer terms is provided. There is also an excellent discussion on how to use a computer consultant effectively. The book includes many photographs of representative systems with captions describing the components, the capabilities and prices. An appendix lists leading suppliers of different sized computers.

All Facets of the Subject

All facets of the subject appear to have been examined. In laying out the procedure to help the business person assess his/her computer needs, the authors even introduce and discuss the possibility of avoiding an in-house system by farming out the tasks to a computer service instead. Heavily emphasized, however, is the goals-oriented approach to choosing a system, starting first with the software. The authors repeatedly suggest searching out existing combinations of hardware and software which are already performing the desired job, and in the same line of business.

Finally, the text closes on the suggestion that, no matter what reason has prompted a business person to investigate computerization, chances are very high that a successful acquisition will open up more avenues of use for the system. The recommendation is that future expenses can be reduced or eliminated by purchasing or leasing a somewhat larger system than is required for the immediate job at hand.

For organization and thoroughness this text can not be surpassed. It will lead you to a clear understanding of the elements involved in the wise selection of a computer system without overwhelming you with technological jargon. ■

Programming in BASIC for Personal Computers
David L. Heiserman
Prentice-Hall, Inc.
Englewood Cliffs, NJ
Softcover, 333 pp.
$7.95

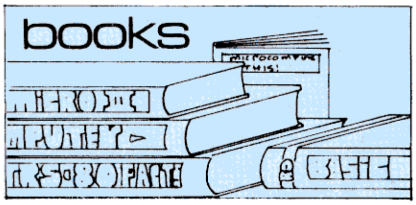
by Bryan Moran

If you just bought a microcomputer this book may be a worthwhile investment, but for anyone who has written a few BASIC programs it's not worth the price. There are no special techniques or innovative programming ideas introduced that are not covered in any elementary programming text.

Although the programs developed in this text are designed to run on any machine using Microsoft BASIC, the author had the TRS-80 user in mind.

Flogged Unmercifully

For the new TRS-80 owner, who has never programmed and needs to be led step by step through beginning BASIC statements, something is to be gained from studying this book. The reader is taken in a very deliberate fashion from powering up the computer through the common BASIC statements. Statements are illustrated by examples as they are introduced. Some, however, are flogged unmercifully. For example, in chapter four I counted nine illustrations of FOR... NEXT loops for timing delays, with little else of value in the examples. I considered



this an overkill.

I found it somewhat amusing that the author refused to refer to a BASIC interpreter. No reference was made to any program that allows the user's program to execute. Interpreters were referred to as "brands" of BASIC or even as (BASIC) "schemes."

Troubles

There are enough typographical errors scattered through the book to discourage the novice. However, most, if not all, should be picked up by the alert reader. On at least one flow chart arrows are mislabelled, but if the reader understands IF...THEN statements there should be no real problem.

More disturbing than typos is a sorting routine that works beautifully as long as at least one of the data is positive. The problem is in a segment where a search is made in an array for the largest element. Why the author chooses to set a temporary variable equal to zero instead of equal to the first entry of the array is beyond me.

An additional trouble spot is the author's inconsistency when referring to rows and columns in two dimensional arrays. In one instance, the first subscript refers to a row while in the next it refers to a column.

Baited Breath

After learning how to use data statements we are told in chapter 12 that better ways to input data will be presented in chapters 13 through 15. With baited breath we wait. What? There is no chapter 15. What have we missed? This is in keeping with a reference in chapter five made to a nonexistent figure.

A habit a programmer should develop early on is to write good, accurate, and meaningful documentation. The author seems to think that simply keeping listings of a program, as it is developed, constitutes documentation. There is no real emphasis on variable listings, descriptions of what variables represent, how program segments or subroutines work, either in the form of REMarks or with accompanying written text.

A sore point with a lot of computerists is the use of multiple-statement lines. The only advantage I see is memory conservation, which is rarely a problem for a beginning programmer. The author uses multiple-statement lines and encourages their use if the machine has a good editor. But he fails to point out difficulties with a line, such as:

```
120 IF A = 5 THEN 40: GOTO 10
```

which is taken from an example in the text. Under what condition is "GOTO 10" ever executed on the TRS-80? Never, but this fact is not pointed out.

In a text designed for a novice programmer it would seem that the concept of an algorithm should be introduced. The word is never used. This seems to be a disservice to the new programmer. What better time to introduce the idea?

In summary, I cannot recommend this book to anyone with even a moderate amount of programming experience. The book is inadequate in its coverage and suffers from poor editing. The novice could, however, learn enough from making corrections to programs to justify the expense. ■

TRS-80 Interfacing Book 2
Jonathan A. Titus,
Christopher A. Titus,
and David G. Larsen
Howard W. Sams
Indianapolis, IN
Softcover, 254 pp.
$9.95

by George D. Dooley

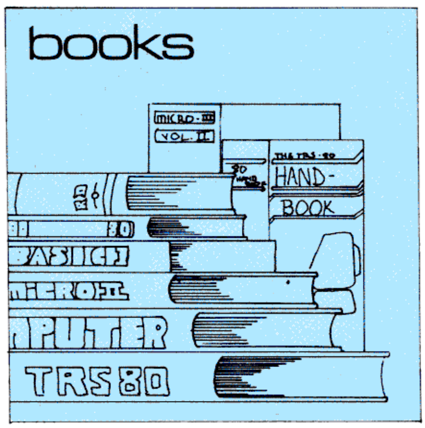
This book, written by the Blacksburg Group, is the second volume of a series on interfacing the TRS-80. The first volume dealt with the signals available on the TRS-80 and software commands used to control I/O devices. It also dealt with the construction of simple I/O ports, address decoders and interfacing A/D and D/A converters. The second volume deals with more advanced and sophisticated interfacing techniques, such as data acquisition, signal processing, remote control, interrupts, and using D/A converters for graphics.

The first chapter is dedicated to the interface circuitry required to use the microcomputer to drive devices such as lamps and ac motors. This chapter includes a thorough discussion of open-collecter type integrated circuits that can drive small loads. The rest of the chapter deals with controlling devices that are powered by the ac line. It covers the theory of the triac, the optical isolator (used to electrically isolate the microcomputer from the ac line) and the solid state relay. All discussions are clearly illustrated by ample use of diagrams and by design examples using common integrated circuits.

Chapter 2 explains how to use the TRS-80 to generate and measure a signal that represents a physical measurement. To generate an external voltage a D/A converter is required. This chapter covers the construction and design considerations of interfaces using eight and ten-bit D/A converters. The discussion of D/A conversion continues, carrying into their application in computer graphics. The graphics are generated on an X-Y plotter, printing out data or forming patterns such as the sine wave.

An A/D converter is used to measure an analog voltage and use that value in a computer program. The interfacing requirements for an A/D converter are more complex than for a D/A converter and this section details the control circuitry well. Two data acquisition design projects are described. One measures the intensity of a light bulb as a function of its distance from a photocell, and the other measures ambient temperature from a solid state temperature sensing element.

These data acquisition projects lead into a discussion of the more practical aspects of the subject. Anyone who has ever operated a television near TRS-80 is aware of the noise it generates. The noise can be



superimposed on the signal we are trying to measure and induce serious errors. The effect of the imposed noise can be countered by techniques such as data averaging and digital filtering. Examples of both techniques are given.

Serial communication and remote control are the subject of chapter four. In some applications it's necessary to control events or to measure signals at a location removed from the computer. The problems that arise from using long lengths of expensive multiconductor cable to connect to the data acquisition unit, and the degradation of analog signals (resulting from the extended distance), can be solved by using a serial data link to a remote I/O device. This project, along with a detailed study of a serial data transmission, can be used as a basis for designing a home solar control system.

GAME BREAKTHROUGH:

THE STICK 80

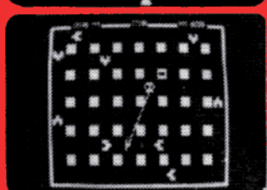
- ☑ FEATURES THE FAMOUS **ATARI JOYSTICK.**
- ☑ WORKS WITH ANY LEVEL 2 AND DISK SYSTEMS.
- ☑ PLUGS DIRECTLY INTO KB. OR E/I (EXPANSION PORT).
- ☑ INCLUDES SIMPLE, DETAILED INSTRUCTIONS TO MAKE JOYSTICK VERSIONS OF MOST ACTION GAMES.
- ☑ COMPATIBLE WITH **ANY** OTHER TRS-80 ACCESSORIES.
- ☑ FULLY ASSEMBLED AND TESTED. READY TO PLUG IN AND ENJOY.
- ☑ FUN TO EXPERIMENT WITH IN **BASIC:** SIMPLY USE INP(Ø).
- ☑ 90 DAY PART AND LABOR WARRANTY.

UNCONDITIONAL **MONEY BACK GUARANTEE**.Try your STICK-80 for 14 days. If for any reason you are not delighted with it, return it for a prompt and courteous refund (including shipping and handling). PRICE INCLUDES ATARI JOYSTICK + ALPHA INTERFACE + DETAILED INSTRUCTIONS + DEMO PROGRAM. ONLY $39.95

BIG FIVE SOFTWARE SUPER FAST MACHINE LANGUAGE ACTION GAMES

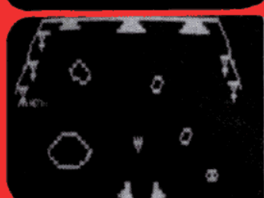
If you and your TRS-80 have longed for a fast-paced arcade-type game that is truly a challenge, then **SUPER NOVA** is what you've been waiting for. In this two player machine-language game, large asteroids float ominously around the screen. Suddenly your ship appears and you must destroy the asteroids before they destroy you! (But watch out because big asteroids break apart into little ones.) The controls that your ship will respond to are thrust, rotate, hyperspace, and fire. All right! You've done it! You've cleared away all the asteroids! But what is that saucer with the laser doing? Quick! You must destroy him fast because that guy's accurate! ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆ **SUPER NOVA®**

The sound of the klaxon is calling you! Cruel and crafty invaders have been spotted in battle formation warping toward Earth at an incredible speed. Suddenly, your ship materializes just below the huge flock of invaders. Quickly and skillfully you shift right and left as you carefully fire your lasers at them. But watch out! A few are breaking out of the convoy and flying straight at you! As the whine of their engines gets louder, you place your finger on the fire button knowing all too well that this shot must connect—or your mission will be permanently over! With sound effects! ◆◆◆◆◆◆◆◆◆◆◆◆ **GALAXY INVASION®**



Your TRS-80 screen has been transformed into a maze-like playfield for this game. As your ship appears on the bottom of the screen, eight alien ramships appear on the top. All of them are traveling at flank speed directly at you! Quickly and boldly you move toward them and fire missiles to destroy them. But the more aliens you destroy, the faster the remaining ones become. If you get too good you must endure the wrath of the keeper of the mazefield: the menacing "Flagship". You must destroy him fast because, as you will find out, that guy's accurate! With sound effects! ◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆ **ATTACK FORCE®**

With thousands of stars whizzing by you, your **SPACE DESTROYER** ship comes out of hyperspace directly under a convoy of aliens. Almost effortlessly, you skillfully destroy every last one. But before you can congratulate yourself, another set appears. These seem to be slightly more intelligent than the first set. Quickly you eliminate all of them, too. But your fuel supply is rapidly diminishing. You must still destroy two more sets before you can dock with your space station. All right! The space station is now on your scanners! Oh no! Intruders have overtaken the station! You must skillfully fire your neutron lasers to eliminate the intruders from the station before your engines run out of fuel and explode! With sound! ◆◆◆◆◆◆◆◆◆◆◆◆◆◆ **COSMIC FIGHTER®**



NEW

The second **Big Bang** has occurred and the galaxy is full of stray asteroids and meteors. As you look through your space port you see a belt of asteroids drifting across the screen blocking your path to the safety of the space station above. But be careful because meteor showers, exploding suns and invading aliens may strike your ship and send it hurtling back to ground level. How many times can you and your opponent maneuver through those obstacles before time runs out? With sound effects! ◆◆◆◆ **METEOR MISSION II®**

SPECIAL

INTRODUCTORY OFFER:

Order the STICK-80 with one or more games and choose one:

- **FREE ALPHA GREEN SCREEN**

OR

- **FREE MAGIC ARTIST PROGRAM**

IN ADDITION YOU MAY:

- ORDER 2 GAMES AND TAKE 10% OFF the games
- ORDER 3 GAMES AND TAKE 15% OFF the games

All games are written in machine language and supplied on cassette.

EACH GAME IS AVAILABLE IN 2 VERSIONS

A (SIDE 1: LEVEL II 16K FOR MODEL 1; SIDE 2: LEVEL II 16K FOR MODEL 3) $15.95

B (SIDE 1: DISK 32K FOR MODEL 1; SIDE 2: LEVEL II 16K FOR MODEL 1) THE DISK VERSION SAVES THE HIGH SCORES ON DISK $17.95

• ALL THESE NEW JOYSTICK GAMES WILL ALSO WORK USING THE KEYBOARD WITHOUT ANY MODIFICATIONS.

GOOD NEWS: If you already have a non-joystick version of these BIG FIVE SOFTWARE games, send the original tape(s) with your STICK-80 order + $3 per game. We will send you a new Joystick version of your game(s).

ALSO AVAILABLE **DOUBLE-STICK 80:** 2 ATARI JOYSTICKS + ALPHA DUAL INTERFACE + INSTRUCTIONS + DEMO PROGRAM: $59.95

ALPHA Product Co.

⌐551

ADD $2.50 PER ORDER FOR SHIPPING AND HANDLING
ALL ORDERS SHIPPED FIRST CLASS MAIL
WE ACCEPT VISA, MASTER CHARGE, CHECKS, M.O.
COD: ADD $2.00 EXTRA
QUANTITY DISCOUNTS AVAILABLE
N.Y. RESIDENTS ADD SALES TAX

85-71 79 St. WOODHAVEN N.Y. 11421 ☎ phone: (212) 296 5916

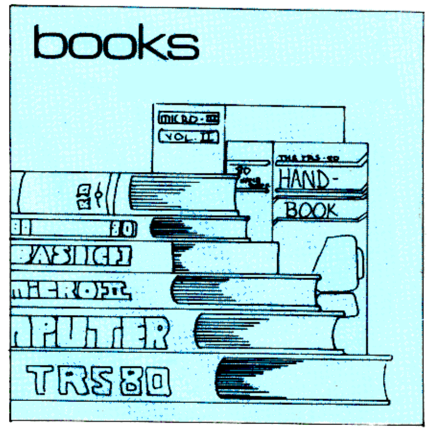
Interrupts are the subject of the final chapter. Since the operation of interrupts can be confusing to a novice, visual aids illustrate the basic concepts of how a computer acts when it has been interrupted. Luckily, the text is full of useful diagrams. It's difficult to discuss interrupts while limiting the discussion to only Level II BASIC. The authors have developed BASIC language programs that POKE machine language commands into memory and then execute the program by calling it from the BASIC program, to illustrate certain principles of interrupt operation.

Clear and Unpretentious

Apart from the subject matter itself, what I enjoyed most about this book is its style. The writing style, like other books in the Blacksburg series, is clear and unpretentious. Liberal use of diagrams, sample programs and tables make complicated concepts understandable. The authors describe useful interfacing projects that are general enough to be used for a wide variety of applications.

The only thing I didn't like about the book is the lack of documentation for some of the larger programs. Programs of only a few lines are not difficult to figure out, but a few well placed comments and a description of variables would sure help reader understanding of the larger programs.

Over all, my praises of this book far outnumber my criticisms. The authors have tackled a complex subject and have produced a book that is an excellent addition to the library of anyone who is interested in using the TRS-80 as a control system. ■



Pascal
David L. Heiserman
Tab Books, Inc.
Blue Ridge Summit, PA
Softcover, 350 pp.
$9.95

by William L. Colsher

Pascal has recently become the number one buzz word among programmers just about everywhere. It is called the hottest thing since 16K RAM, and it's also said to be the language of the future.

Unfortunately, most of us haven't been able to determine whether any of the stories are true: the cost of admission has been far too high. First, there is the 48K two disk machine you need to run it, and then the $150-plus price of a compiler. Pretty steep for an experiment!

Not long ago (1978) a couple of grad students at the University of Illinois developed a Pascal compiler called Tiny Pascal. Then in 1979, a company called Supersoft brought out a version of that compiler for the world's most popular micro—the TRS-80. Best of all, that compiler operates on a 16K tape-based system, the most popular TRS-80 configuration. Today that compiler is available from Radio Shack (and elsewhere) for only $19.95!

Programmer's Guide

Heiserman has written the definitive programmer's and user's guide to Tiny Pascal. Three hundred and fifty pages of examples and exercises take you from loading the tape to writing sophisticated programs, in 16 easy chapters.

The first two chapters of *Pascal* are essential introductory material. How to load the tape and use the various editing features are covered in enough detail to enable the first time user to sit down and get a program running within a few minutes. Saving a program on tape, compiling it (including the use of several compiler options and what to do about a number of errors) are all explained clearly and succinctly.

None of the pedantry that seems to infect many books on Pascal is apparent in this book. No "Integration by Simpson's Rule" here; just practical, hobbyist stuff like dice rolling and drawing pictures on the screen.

Chapter three introduces Pascal syntax diagrams. Learning to read these diagrams is critical to learning Pascal well. Once they are mastered you'll find it a simple matter to actually write programs in Pascal. After introducing the syntax diagrams, Heiserman dives into I/O with the Write statement. Chapter four goes into more detail on the many things that can be done using Write, including graphics and screen control. The idea of string and integer constants is also introduced through their use in the many examples of Write that are given.

Introducing Variables

The next two chapters, five and six, introduce variables and what to do with them. Special care is taken not to confuse BASIC programmers with the difference between an equal sign (=) and the Pascal assignment statement (:=).

IF...THEN...ELSE is covered in detail along with the Boolean operators AND, OR, and NOT. As is the case throughout this book, examples are clear and easy to understand; in virtually every case there are complete programs you can enter and check out for yourself. Finally, syntax diagrams for all parts of Tiny Pascal are presented.

Chapters seven and eight present the looping features of Tiny Pascal: REPEAT–UNTIL, WHILE–DO, and FOR–DO. GOTO is not included in the list; the designers of Tiny Pascal refused to make any concession leading to unstructured programming and design. Once more, the when and why of using these statements are developed with clear examples that are meant to be typed in and run, not just read.

Chapter nine presents "A Miscellaney of Pascal..."; a collection of most of the features you'll need when writing programs. The most important statement discussed here is the CASE–OF which is similar to BASIC's ON–GOTO. CASE–OF gives you the ability to do various things based on the value of some expression. PLOT and INKEY, for graphics and "on-the-fly" keyboard input, are also introduced.

Chapter 10 completes the introduction of Tiny Pascal features with procedures, functions and arrays. This chapter begins to use more sophisticated programs as examples. An animation of an inchworm is used as an example of the use of procedures with a parameter.

The next three chapters of *Pascal* consist of the practical uses of the language introduced in the first ten. Dice games, roulette, a slot machine, etc.

Chapter 14 is a good introduction to structured design and programming. Numerous examples show the development of several programs from idea to completion. A long program called Screwball Golf finishes off the chapter—15 pages of development and explanation!

Chapter 15 really puts it all together in a long game called Space Ranger. Detailed explanations are given of the techniques used, including the reasons for structuring the program as it is. This makes the chapter a long one, but the 25 pages have a lot of detail to cover, not to mention explaining how to play the game.

Putting Ideas Into Familiar Light

The final chapter of *Pascal* is one that I would have welcomed when learning the language: translating BASIC into Pascal. Each statement is covered in good detail, but because of the highly structured nature of Pascal and the unstructured nature of most BASIC programs, there is really no easy way to make direct translations. Nevertheless, the chapter is useful because it helps the beginner put things into a more familiar light.

The appendix contains tables of cursor control codes, TRS-80 graphics characters, and the Pascal syntax diagrams. For some reason, a list of errors has been omitted, though one is mentioned early in the book, a minor problem considering the overall excellence of the book.

Pascal is, as I said, an excellent book, especially for the beginner. For less than $30 (the cost of this book plus Tiny Pascal) anyone with a TRS-80 can begin to learn the techniques of structured programming. However, I would like to voice one complaint. Many of the longer programs have evidently been typeset directly from computer printouts, although most of the examples are not. This can occasionally cause some ludicrous errors, such as spelling tomb "toumb".

If you are interested in Pascal programming get this book, the Radio Shack tape and get to work. It'll be the best 30 bucks you've spent on your computer.■

Boss
Soft Sector Marketing Inc.
Garden City, MI
$29.95

by Bruce Douglass

Boss, a utility program for the TRS-80, has several features useful in writing and debugging BASIC programs. They are: improved trace capability, single stepping through BASIC programs, reviewing variables, stacking programs, setting breakpoints and complete relocatability of the utility itself. Each of these capabilities will be dealt with separately.

The first thing you should know about Boss is that it comes on cassette along with a separate lowercase driver (if you have the modification) and manual. The program may be loaded into memory, relocated and saved on tape or disk. It runs in Level II BASIC or Disk BASIC.

Generally, when using Boss, you don't need to worry about it interfering with other machine language programs you may have in memory. When it loads, it will prompt you for the lowest address you wish to protect, and will relocate itself beneath that address. It also gives you the proper response to the memory size question.

I use Boss with a full-screen text editor for BASIC called XBE. In my 48K disk system, XBE loads in from 60416 and up. Boss loads in underneath and informs me to set memory size to 57571.

The nice thing is that I can keep both in memory, and both are operable. The programs will run well except when I use the command to review my variables; somehow that seems to eat the XBE program above Boss. The @ key becomes the control key for Boss (you must use <Shift> 0 for the normal @, but <Shift> @ remains the same). By pressing @ and a character you may access various Boss functions. The manual refers to @ as <CON>, and I will use that notation here.

Trace Function

The first capability to be mentioned is the improved Trace functon. By pressing <CON> and 1 at the same time, you turn the Trace off. <CON> 2 turns the Trace on and <CON> sends the Trace to a printer.

If you have ever tried to use the Level II Trace function, you noticed that it is hard to follow and destroys the display. The Boss Trace sets up the Trace so that it is only displayed in four rows in the right-hand corner. A right arrow moves to the row containing the executing line number. If the line is multi-statement, the line number is displayed only once during the execution of that line.

Sending Trace to the printer results in the line numbers being printed out along a horizontal line. The printer will continue to print the Trace until the program ends, the Trace is redirected, or turned off.

Boss is one of the most powerful tools I can use. It allows me to single step through the routines in the program and review just what the variables are doing.

To single step through BASIC, the following functions are available:

- <CON> 4 to turn single step off.
- <CON> 5 to single step to the end of line.
- <CON> 6 to single step each instruction.
- <CON> 7 to single step with a variable time delay.

All may be used in conjuntion with Trace functions as well.

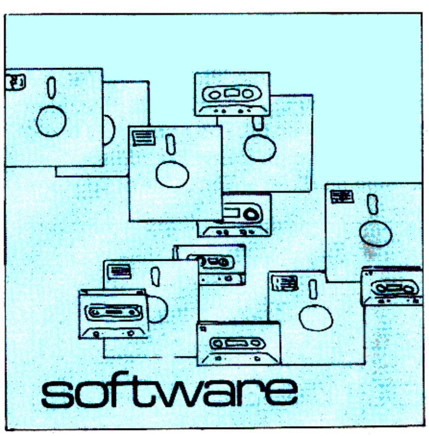
<CON> 5 single steps each line it executes. That is, when it begins to execute the next line, it will execute the entire line, and pause at the end of that line. Pressing <Space> will cause the next line to be executed.

<CON> 6 will pause at each statement terminator (":"). Otherwise it is similar to <CON> 5.

<CON> 7 will cause a time delay before executing the next line. The time delay default value is about one-fourth of a second. This single step mode may also be used in conjunction with single stepping by statement (rather than by line) and pressing "<CON>7 <CON>6." The time delay may be increased or decreased with <CON> <Up Arrow> or <CON> <Down Arrow>.

You may use as many breakpoints in the program as you like by inserting the line POKE 16667,5 wherever you need to break. You may include most of the Boss commands as POKE statements, so that your program will run normally until it reaches these commands, whereupon the Boss functions it calls for (for example, single stepping) will begin to execute. Thus you needn't worry about doing everything in Command mode.

The next useful capability is that of reviewing variables. <CON> N allows you to select the variables you wish to review, while <CON> 0 actually displays them. <CON> N may be invoked at any time. Upon the invocation of <CON> N, you will be asked for maximum variable length, followed by the variables to be reviewed. The number of variables you may review is inversely proportional to the maximum



variable name length. This is explained in the manual. <Break> terminates the function.

Any time during program execution, <CON> 0 may be invoked and the first variable will be displayed. From here, you may return to the program execution (where it left off) or select another variable to be displayed by pressing <BREAK> or C, respectively. Pressing any other key will cause the next variable (previously selected) to be displayed. As previously mentioned, use of this Boss function eats the program residing above it in high memory despite the claims by the manual that the other programs are protected. When I attempt to jump to the latter program with a SYSTEM ? and /60416, I get a reboot. I have taken to doing my text editing before using the <CON> N command.

In my opinion, the reviewing capability, together with Trace and Single Step make up the important functions of the program. You may stack BASIC programs up in memory and PUSH and POP them around as you like. You are, of course, limited by available RAM. In a disk system this is of little consequence. When you download a BASIC program from memory, it will destroy the current one in memory, so you also may switch the curren tone with the one in high memory. Memory size is adjusted automatically, so that the programs won't get eaten by BASIC variables and stack useage.

You do have the capability of appending programs, either the current program with the last saved program or the current program with the next-to-last saved program. Line sequence must be correct for the appending to work properly. That is, the saved program line numbers should all be higher than the current program line numbers. Again, with a disk system and the ability to merge programs this is not particularly useful, but for Level II users it is a very handy ability to have. Thus their favorite routines (such as matrix inversion, or line printer routines) may be stored on tape with high line numbers and appended to the current program. If you append several routines, make sure you do them in the right line number sequence!

DOS users should be aware that reboots are an inconvenience here, as BASIC * will not recover the utility.

For the sake of completeness. See Fig. 1 for a list of the commands available in Boss.

In conclusion, this utility is a powerful debugging tool. Of importance to Level II users is the ability to append programs, even though a Merge would be more useful. This program is very useful for anyone who would like to spend some time away from program debugging. ■

Command	Function
<CON> 1	Trace off
<CON> 2	Trace on (video)
<CON> 3	Trace on (printer)
<CON> 4	Single Step (SS) off
<CON> 5	SS to end of line
<CON> 6	SS instruction
<CON> 7	SS with time delay
<CON> up	slow execution (time delay)
<CON> down	speed execution (time delay)
<CON> N	select variables for review
<CON> O	review variables
<CON> -	Save BASIC program in high memory (push)
<CON> :	Recall last saved BASIC program from high memory
<CON> 8	Append last saved program to current program
<CON> 0 (#)	Recall next-to-last saved program

Fig. 1.

Discat
Model I 32K/48K
Myatt & Smith
Tustin, CA
$50

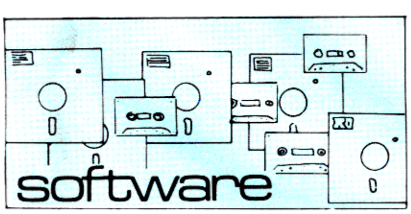
by Robert C. Daigh

When a little voice in your diskette storage box whispers "One of these days we've really got to get organized!", don't get upset—help is on the way.

If you are like most of us, you know you've saved a wonderful little program in there somewhere. And you need it right now. . . if only you could find it.

Bill Myatt of Myatt and Smith solved his frustrations, and ours too, with a machine language program named Discat. The Disk Catalog Index Program is for the TRS-80 Model I, with a Model III version soon to be released. The program is being marketed through Racet Computes.

Discat will load an index file of your programs in about twenty seconds, and then tell you where to find any one of some 17,000 different programs by displaying the disk number and indicating which side the program is on. (For a 32K system, reduce that number to around 7,200 programs.) The complete program listing for either side of any disk in your index can be displayed in answer to a Disk Number query.



The program automatically keeps track of the free space available and, on command, will display a listing of free grans on each disk in the index. You no longer need to search for free space to dump a program in memory.

For those of us who really mean it when we say *organized,* Discat permits not one, but nine different index files, each of which can hold 800 program locations. Now you can have separate index files for business, games, utilities, data base, and so on.

The program requires an expansion interface, a minimum of 32K of memory and at least one disk drive. Up to four drives are supported. For best utilization of the program capabilities, a parallel printer should be part of the system.

Getting Started

Detailed instructions describe each step involved in transferring Discat from its distribution diskette to merge with your DOS system diskette under TRSDOS, NEWDOS+, NEWDOS80 or VTOS. The machine language program is called from DOS in the usual manner. During the short initialization period the program will search all active disk drives, loading the names of up to nine index files.

Each segment of the program is menu-driven. The main menu has the following options:

- Display catalog of indexes
- Program / disk search menu
- Display free disk space
- Create (or update) index
- Display the current index
- Print the current index
- Sort the current index
- Editing menu
- Special utilities
- Save the current index to disk

Setting up your own catalog of indexes is very simple; the program does all the tedious work for you. Two optional systems are discussed. If, for example, you choose to arrange all your utility programs onto one group of diskettes, the documentation suggests that you assign a diskette number in the 500-599 series for utility. Then you have only to insert each disk into the drive you have selected as

Orange Micro

"THE COMPUTER PRINTER SPECIALISTS"

UP TO 25% DISCOUNTS! — SAME DAY SHIPMENT!

NEW CENTRONICS 739 (RADIO SHACK LINE PRINTER IV)

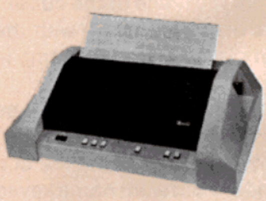
With Graphics and Word Processing Print Quality

• 18 x 9 dot matrix; suitable for word processing • Underlining • proportional spacing • right margin justification • serif typeface • 80/100 CPS • 9½" Pin Feed/Friction feed • Reverse Platen • 80/132 columns • Top of form

CENTRONICS 739-1 (Parallel)	(List $995)	$ Call
CENTRONICS 739-3 (Serial)	(List $1045)	$ Call

ANADEX

Dot Graphics, Wide Carriage

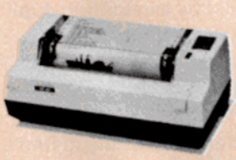


• 11 x 9 dot matrix; lower case descenders • Dot resolution graphics • Bi-directional, logic seeking • Up to 200 CPS • RS 232 Serial & Parallel • Forms control • X-ON/X-OFF • Up to 6 part copy.

ANADEX 9501	(List $1650)	$1350

AXIOM

Dot Graphics, Low Price



• Dot addressable graphics • 70 CPS • 12 characters per inch • 80 columns • adjustable tractors • 3 part copies • Manufactured by Sieko • Parallel, RS232 serial, IEEE-488 interfaces available.

AXIOM GP-80M	(List $399)	$349

IDS PAPER TIGERS

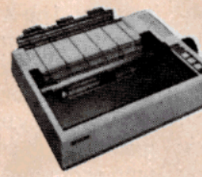
Dot Resolution Graphics, quality print, speed

• 7 wire printhead (445); 9 wire printhead (460) with lower case descenders • Over 150 CPS • bi-directional, logic seeking (460) • 8 character sizes; 80-132 columns • Adjustable tractors • High-resolution dot graphics • Proportional spacing & text justification (460).

IDS 445G 7 wire printhead, graphics	(List $895)	$ 750
IDS 460G 9 wire printhead, graphics	(List $1394)	$1150
IDS 560G 9 wire, wide carriage, graphics	(List $1794)	$1400

EPSON MX80/MX70

Low-Priced Professional Print Quality



• 9 x 9 dot matrix • Lower case descenders • 80 CPS • Bidirectional, Logic seeking • 40, 66, 80, 132 columns per line • 64 special graphic characters: TRS-80 Compatible • Forms handling • Multi-pass printing • Adjustable tractors

We also carry a full line of Epson Accessories.

EPSON MX80	(List $645)	$Call
EPSON MX 70 Dot graphics, 5 x 7 matrix	(List $450)	$Call
EPSON MX100 wide carriage	(List $995)	$Call
GRAPPLER™ Apple graphics parallel interface and cable		$ 165

MX80/70 FRICTION FEED KIT

Uses installable kit for single sheets.
Easy 15 minute installation. $ 75

ANACOM

Low Cost, High Speed, Wide Carriage

• 9 x 9 dot matrix • Lower case descenders • Wide carriage • Adjustable tractors to 16" • 150 CPS, Bidirectional, Logic Seeking

ANACOM 150	(List $1350)	$ Call

NEC SPINWRITER

High Speed Letter Quality

• 55 CPS • Typewriter quality • Bidirectional • Plotting • proportional spacing.

5510-5 RO, Serial, w/tractors	(List $2995)	$2675
5530-5 RO, Parallel, w/tractors	(List $2970)	$2650

TELEVIDEO CRT'S

AT DISCOUNT PRICES!

QUANTITY PRICING AVAILABLE

TVI 910
TVI 912C
TVI 920C
TVI 950

Please Call Toll Free Prices are too low to advertise

PRINTERS

MALIBU 165 wide carriage, graphics, letter quality	(List $2495)	$ 1975
QUME 5/45 typewriter quality	(List $2905)	$ 2559
VISTA-C. ITOH Starwriter, letter quality	(List $1895)	$ Call

INTERFACE EQUIPMENT

EPSON ACCESSORIES	$ Call
ORANGE INTERFACE for Apple II parallel interface board & cable	$ 110
MICROTRONICS Atari parallel interface	$ 69
TRS-80 CABLES to keyboard or Exp. interface	$ Call
NOVATION D-CAT direct connect modem	$ Call

CALL FOR FREE CATALOG

(800) 854-8275

CA, AK, HI (714) 630-3322

At Orange Micro, we try to fit the right printer to your application. Call our printer specialists for free consultation.

RETAIL PRINTER STORES:
3150 E. La Palma, #I, Anaheim, CA.
13604 Ventura Bl., Sherman Oaks, CA.
Store Hours: M-F 10-6 Sat. 10-4

✓296

Phone orders WELCOME; same day shipment. Free use of VISA & MASTERCARD. Personal checks require 2 weeks to clear. Manufacturer's warranty included on all equipment. Prices subject to revision.

Orange Micro, Inc.
3150 E. La Palma, Suite I
Anaheim, CA 92806

update drive. Respond to the query with the diskette number and F (front) or B (back) side and the diskette data is read automatically. The screen now displays disk number, side, free grans, and the listing of each program with name, extensions and size in grans. At this point you may choose (Y/N) to add this entire diskette directory to the current index.

The authors have thoughtfully provided a much simpler system for you programmers who are a little lazy. You do not have to reorganize your entire disk inventory just to fit the program. (Who really has time to cull all his utility programs onto separate disks?) As each disk directory is read and displayed, you may selectively add to the current index any program on the display. You are asked:

—> ADD DIRCHECK/CMD TO CURRENT INDEX (Y/N) ?

Your response is displayed on the screen and the next program on that diskette scrolls by in sequence. Now you can select all your utility programs from wherever they may appear on your diskettes and organize them into a separate index.

Once the diskette identification is assigned, it is written to the diskette and need not be entered again. It can, however, be changed under the editing submenu. When you later prepare an index, let's say for games, you will have to insert each disk again and select all the game programs as they appear for inclusion into the games index.

The user is warned that non-standard DOS diskettes such as Pascal, Forth, and CP/M must not be inserted into the update drive. The writing of diskette identification might cause irreparable damage to non-standard systems. Fear not that these disks must lie unwanted and unclassified in your collection. Discat allows manual entry of the diskette number and all its programs into whichever index you may choose.

This is all easier to do than it is to explain. On my first trial run I assembled a composite index of 529 programs from both sides of 32 diskettes. The time went by quickly and I enjoyed seeing all those program names I had forgotten I owned.

Using the Indexes

Nine separate indexes can be assembled and saved by the program. Now let's find out how to use an index.

From the main menu, one calls the sort sub-menu, then chooses to sort the current index either by program name or by disk number.

A short buzz sound from the expansion interface notified me that my 529 programs were now sorted; it took less than 14 seconds, after which I was returned to the main menu. Display of the current index can be either in streak or page mode. Streak provides a scrolling display that can be stopped and restarted by touching any key. The X key aborts the scroll and returns to the main menu. One may select to display only those programs with like extensions, for example /PCL or /CMD.

A hardcopy printout of the current file in memory can be set for either single column or double column format. The user is asked to set the number of lines per page; each page is numbered and identified by index name. The listing includes program name and number, diskette number and side, and the size of the program in grans. Each page lists 110 programs under the double column format.

Probably the most useful routine in Discat is the search program. The submenu lists:

1—Display Disk Directory
2—Program Search
Choice ?

If <1> is selected, the user is asked for diskette number and side, whereupon the chosen directory is displayed. Upon keying the <2> selection from menu, you are asked to enter program name and extension. Searching by program name does not require that the entire name be entered. Suppose you wanted to locate the program named Termites/BAS. You could shorten the program name to Termites without the extension and the screen would respond with:

Termites/BAS
Termites/CIM
Termites/CMD

provided, of course, that your index contained all three items.

The search could be broadened by asking for Term, in which case the screen will respond with all programs beginning with the key letters TERM----. The search key could be just T, which would display every program beginning with the letter T and your Termites/BAS would be there in proper alphabetical order.

A full editing menu allows deletion of a specific program or an entire diskfull of program names.

All sub-menus return the user to the main menu where the name of the current index is always displayed along with the total number of disks and of programs in that index.

The current index in memory may be saved to disk at any time. The program allows the index to be saved under the current name or under a different name keyed in by the user.

The program satisfies just about every need I could envision for a disk file organizer. It is easy to use, fast, adequately documented and capable of handling far more information than most TRS-80 users will probably need.

When I discovered how many diskettes I could recover just by eliminating duplication and recapturing unused space, the savings in diskettes nearly equalled the price of the program.

Best of all, though, I know what program material I have and where to find it in a hurry.

At last, I'm *organized*. ■

Silver-it
Fuller Software
Grand Prarie, TX
$5

by Paul R. Prescott

As with most TRS-80 users I was slowly being driven crazy by the unreliable contacts between the keyboard and the expansion interface. Although the well-known trick of cleaning the circuit board connections with an eraser worked, my '80 had reached the point where this was necessary before each session! Then a small Fuller Software ad caught my eye. They promised that a $5 investment in Silver-It would put an end to spontaneous reboots. Their response was quick, and the kit was received within one week.

Not For the Beginner

Imagine my surprise when I opened the package and found no instructions for use. A disclaimer was enclosed that warned the purchaser that silver soldering is not for the beginner and that computer circuits are delicate. To their credit, Fuller offers a full refund to anyone who does not feel up to the challenge, but I feel this warning should be included in the advertisement.

The kit consists of a small piece of solder wick and several inches of silver solder. Being the brave sort, with a lot of soldering experience, I pressed on. For those of you who might wish to give it a try, here's my approach.

First, you need a 25-watt soldering iron, absolute alcohol, and liquid rosin flux. Absolute alcohol is available at liquor stores under the name "Everclear," and is used to clean the contacts. Don't try to substitute with rubbing alcohol. The rosin flux is needed as the solder rod is flux free. Be certain not to use the acid flux that is generally used for soldering.

Place the keyboard face down on a clean cloth to avoid marring the keys. Remove the six screws from the bottom, keeping track of which screws belong in which holes, and place the base off to the side. Very gently lift the main circuit board; you will see spacers separating it from the keyboard circuit. Remove the spacers without flexing the cable connecting the two boards, then lift both boards out as a unit. Use the alcohol to clean both sides of the circuit board contacts and allow them to dry.

Radio Shack saved a few pennies by coating the contacts with regular solder instead of silver or gold; this solder must be removed. Heat each contact with the soldering iron while using the solder wick to remove the old solder. Be careful not to cause a solder bridge between contacts. Next, clean the contacts again and immediately coat the surface with the rosin flux.

Applying the silver solder is relatively simple. Start at the outer edge and heat the contacts with the iron. Touch the silver solder to the edge, at the same time start to move the iron and the solder towards the center of the board. Keep the solder just behind the iron in continuous contact with the board. When the inner edge is reached lift both off together. You should achieve an even, shiny coating without much trouble. After all the contacts on both sides have been soldered use the alcohol to remove any residual flux.

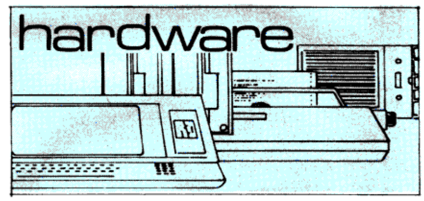
After checking your work for solder bridges or large lumps in the solder track, gently place the circuit boards back in the keyboard top. Very carefully reinstall the plastic separators on the posts and reassemble the base.

The interface procedure is exactly the same. I silver soldered all the edge connectors on the interface board in hopes of avoiding any future problems. The entire procedure took about two hours.

Practice First

Since reconnecting the computer six months ago, I have found no immediate problems. Everything still functions perfectly, no spontaneous reboots and no more eraser cleaning sessions!

This method, however, is not a cure to be applied by anyone not familiar with good soldering techniques or aware of the delicacy of computer circuits. With the keyboard wide open a stray static discharge could easily zap a chip. If you have soldered before but not along edge connectors, I suggest that you pick up a blank board with connectors at Radio Shack and practice first. There is more than enough silver solder to do this provided you don't go overboard. If you have the experience, join the fun! My TRS-80 now works as reliably as a new unit! ■



Speak-2-Me-2
Percom Data Co., Inc.
Garland, Texas
$69.95

by Edward Louis

This is a review of the Percom Speak-2-Me-2 interface hardware, and the accompanying manual and software, used in conjunction with a modified Texas Instrument Speak&Spell and a TRS-80 Model I computer. Although I used a disk system with TRSDOS 2.3., the Radio Shack Expansion Interface unit and a General Electric Terminet 300 printer, these additional peripherals are not necessary to make good use of the hardware.

A few years ago I was associated with a Navy project that made use of a Votrax speech synthesizer. Every time I walked into the lab and heard the droning voice giving voltage and power readings in what seemed to be a slightly Swedish accent, I had to pause for a moment to realize that this was a computer "talking" and not some poor lab assistant chained to a set of meters. Needless to say, I was intrigued by the prospect of having this kind of electronic wizardry to experiment with.

Then an advertisement for a Percom interface device to work with the TI Speak&Spell caught my eye. It was called Speak-2-Me-2 and it sounded like the answer.

I took the plunge and bought the Speak&Spell and ordered the Speak-2-Me-2 unit from Percom.

The Percom advertisement said that the device, when used in conjuction with a modified Speak&Spell, could be the voice of a computer through the use of a few BASIC program lines. It stated that some modification of the Speak&Spell was necessary and that an external power source was needed. It also mentioned that either an expansion interface or printer cable adaptor were required, and that an advanced speech driver and games disk was available.

When my package came, I examined the contents and found a small printed circuit board and a twenty-two page manual.

Glancing through the material I noticed a standard warranty, a release form if I were to choose to send the interface and the Speak&Spell back to Percom for modification (along with $25), and a number of warnings. There was also a separate envelope containing a ribbon cable wth connectors at both ends.

As well as this overall package was presented, there were two items that confused me from the start: First, I could not tell from the advertisement, the documentation or any marking on the hardware which cable assembly I had. In any case, it eventually worked with my expansion interface. Second, one of the separate sheets of warnings and cautions stated that under no circumstances would Percom undertake installation of the Speak-2-Me-2 interface in a previously tampered-with Speak&Spell. Within the manual, however, I found a statement that suggested that after one had tried everything and been unable to get the combination of devices to work, they could be sent back to Percom with an initial repair fee of $15 and Percom would attempt to correct the difficulty. In fact, the statement went on to say that they had never found a unit that couldn't be repaired satisfactorily.

Initially I was disappointed because there seemed to be a very small amount of hardware for $69. I eventually realized that more than half of what I paid for was the knowledge and facility to get to the heart of the Speak&Spell and make it work with my system.

The manual stated that if the modification was beyond the buyer's capability or if the hardware in the Speak&Spell was in any way different than that depicted in the manual, the whole package, including the Speak&Spell, could be returned to Percom along with $25 for them to do the modification. The other alternative was to send Percom's package back and get a full refund.

Documentation

In the main body of the manual, several detailed diagrams were found which showed a number of different views of the circuit board inside the Speak&Spell. The warning was repeated in the text stating that unless the circuit board of the

Speak&Spell being modified looked exactly like the diagrams, the modification should not be made.

An additional page in the manual described a new keying technique for the two connectors which connect the Speak-2-Me-2 interface to the Speak&Spell and to the computer. My unit did not employ the missing pin keying as described, but used red paint on the connector and on the board to indicate connector polarity. This worked well, along with the explicit diagrams showing proper ribbon cable dress.

I have to conclude that the manual is quite thorough. The mechanical and electrical detail was precise and understandable, there are some BASIC programs to run on the new system, and an additional section contains not only a machine language program and a verbal flow diagram, but also some explanation of how this device could be interfaced with other computers, (although none were specifically identified).

The software in the manual was limited to a driver program in the form of a BASIC data POKE, a short program producing a single sentence (rather humorous but I'll let Percom surprise you) and a simple game program with part of the interaction coming through the Speak&Spell.

These programs were easy to type in and use, as soon as I realized I was under TRSDOS BASIC and had to use DEFUSR instead of POKEing the starting address of the machine language portion into the locations. The only other problem I had with Percom's software was the format used. For some reason, the programmer decided to use multiple statement lines and to place the : separating them at the beginning of each new line. For example:

```
10 X = 1 TO 100
: A = PEEK(X)
: PRINT A,X
20 NEXT X
```

In my opinion, this is a difficult form to edit and doesn't add to clarity.

Modifying Speak & Spell

First, there was the mechanical problem of opening up the Speak&Spell box and getting access to the circuit board on both sides.

Next, the trick was to locate a pattern on the printed circuit board identical with that in the diagram. I found this more difficult, but I finally thought I had come close enough to do the deed. The modification required either cutting or unsoldering the integrated circuit leads or cutting the runs from those leads to the rest of the board. The manual suggested that cutting the runs would result in less risk to the chip in question. Fortunately, I followed their suggestion, as it turned out I made the wrong cuts on this first try.

After making these changes, it was a simple matter to solder the two jumper wires and the six ribbon cable connections as specified. Somehow, when I finished things didn't look exactly right, but I proceeded to insert the Speak-2-Me-2 in the battery compartment and connect the external power supply. Before connecting the finished product to the computer, I turned on the power and found that most of the Speak&Spell still worked normally, although a few of the letters would not respond audibly when pressed.

I proceeded to tie in the computer and then typed in the various programs described above and ended up spending an entire evening debugging programs and trying to figure out why there was no computer control.

The next evening, after having thought a bit about the problem, I decided I had been and should take the unit apart again to see if I could figure out where I went wrong.

Somewhere during this activity, I happened to turn to the end of the manual and found an addendum with a circuit diagram which exactly matched my unit.

I had to restore the original condition of the circuit board and start over. The difference was essentially a 180 degree reversal of the chip on the board. Within a few minutes I was ready to try the unit again.

I turned the power on before tying into the computer. This time nothing at all happened. This could be disaster . . . or maybe the computer had to be in the circuit to turn the unit on.

After connecting to the parallel printer connection on the expansion interface unit and going over all of my programs, I still could get nothing from the unit.

At this point, I decided to break down and haul my old Tektronix 514 oscilloscope up from the depths of the cellar and see where things were going awry. I did this, but in the process I had removed and replaced the connector to the expansion interface unit and—you guessed it—the Speak&Spell started to talk although it was mostly R2D2 type noise with only a couple recognizable words.

I had been using a home-brew 10-volt supply to run the unit after finding that a standard nine-volt battery did not have sufficient current capacity. The manual suggested using the Radio Shack PN 274-251 power adapter. I purchased one and found it to work very well, with more than enough current capacity.

At this point I had added a useful new peripheral to my system and I was ready to experiment. Percom offers a separate software package to give an expanded vocabulary of partial words, etc. For myself, I much prefer the challenge of experimenting on my own.■

The MicroConnection
The MicroPeripheral Corporation
Redmond, WA
$249

by Eric Keener

Dial-up systems for the computer hobbyist are relatively new. Such a system used to require an expansion interface, the optional RS-232 board, and then, an expensive acoustically coupled modem. This adds up to at least $650 just to get on-line.

The MicroPeripheral Corporation (formerly the Peripheral People) now has the MicroConnection. As a matter of fact, they have a whole line of MicroConnections, but, I will only deal with the version for the Model I TRS-80. This modem is FCC accepted, directly connects to the phone line, and does not require an expansion interface or any sort of RS-232 interface. For that matter, it has its own RS-232 port for driving a serial printer. Also, its RS-232 port provides it with the ability to operate as a stand-alone modem. That is, you can disconnect the MicroConnection from your TRS-80 and connect a standard RS-232 compatible serial terminal (set for 300 baud) to the serial port on the modem. This allows you to connect your terminal to the phone lines without your TRS-80.

The MicroConnection operates at 300 baud, but can be converted to run at 110 baud through a simple hardware modification that is described in the operator's manual. The word protocol is under software control, thus, you set the MicroConnection to operate with even, odd, or no parity, a 5, 6, 7, or 8-bit word, and 1, 1.5, or 2-stop bits. The magic behind this control is an 8251 USART that is used in the MicroConnection. The MicroConnection also provides an input and output to be used with your amateur radio equipment for ASCII Bell 103 standard teletype. So far, though, I have not tried this feature as I haven't been able to find any activity using the Bell 103 standard (200 cps shift).

The MicroConnection comes with a dumb terminal program, on cassette,

See List of Advertisers on page 306

80 MICROCOMPUTING ANNOUNCES...

. . . and saves you hours of typing and aggravation.

LOAD-80 is a monthly dump of the major program listings in 80 Microcomputing on cassette. Publisher Wayne Green tells you more . . .

"Frankly, after hundreds of hours of frustration, I seldom even try to keyboard a published program. Even if the magazine manages to get the program typeset correctly (which seems rare), I inevitably screw it up when I keyboard it. Who needs the aggravation?

"This is why I've started a new series of cassettes called **Load 80**. Each cassette will have program dumps of the listings in an issue of **80 Microcomputing**. These listings are direct from the authors and tested by the 80 staff. All but the very short program listings will be on these **Load 80** cassettes. Thus you will be able to save hours of inputting programs and even more of debugging your keyboarding errors.

"Though the authors of these programs will share the royalties from the sale of the cassettes, this will not preclude the better programs from being issued separately by Instant Software (with royalties) with full documentation and associated hoopla. The documentation for the **Load 80** programs will be entirely in **80 Microcomputing.**

"I originally was holding out for "Trash Dump" as a name for the cassettes, but cooler heads prevailed. If there turns out to be enough interest in **Load 80**, we'll set up a monthly subscription arrangement."

Wayne Green, Publisher

The Load-80 cassette is simply the program listings that appear in the articles in *80 Microcomputing*. It was created to save you the time involved in typing in the listings yourself. Successful loading of the programs depends on reading the documentation in the articles. If you have your current magazine at hand when you load the cassette, you should have no difficulty. If you still have problems, please return the tape for a replacement.

Send in the attached card and you will receive the cassette for the major programs in this issue

If the card is gone, photocopy the coupon.

Please note there is no warranty expressed or implied that this program is going to do anything other than save you typing.

☐ Yes . . . send my LOAD 80 Cassette for only $9.95

☐ Check Enclosed   Bill my ☐ AE ☐ MC ☐ VISA

NAME ____________________

ADDRESS ____________________

CITY __________ STATE ______ ZIP ________

CARD # ______________ EXPIRE DATE ______

SIGNATURE ____________ INTERBANK # ________

Please allow 4-6 weeks for delivery
80 Microcomputing Pine Street Peterborough NH 03458
Attn: Debra L. Boudrieau

8007

called S80. This program provides redefined keys to transmit the following symbols:

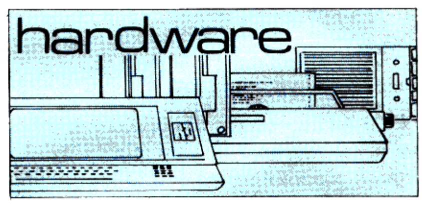
Up-arrow 1-Escape
Up-arrow 2-Left brace
Up-arrow 3-Right brace
Up-arrow 4-Vertical broken
Up-arrow 5-Wave
Up-arrow 6-Back slash
Up-arrow 7-Back apostrophe
Up-arrow 8-Left bracket
Up-arrow 9-Right bracket
Up-arrow 0-Null

By the way, the up-arrow is your control key so you can transmit any other control code. The manual provides the addresses to change, if you wish, to redefine the above special characters. Due to the uniqueness of the TRS-80 character generator, characters six, eight and nine will show up on your screen as different symbols. If you have the new character generator IC in your keyboard, characters five through nine will show up different on the screen.

S80 also provides for printing the screen on a parallel printer. The command mode is accessed using a shift up-arrow. Shift up-arrow P turns on the printer and shift up-arrow S turns it off. Also, shift up-arrow E returns you to Memory Size? and shift up-arrow I returns you to the initialization routine for setting half or full duplex.

I did have one problem which I attribute to my ignorance of word protocol. Regarding the changing of parity and word length, I tried changing the MicroConnection to transmit with even parity. Also, I attempted to transmit an eight-bit word. To those of you who don't know, the total of the parity bit plus the word bits can only equal eight. I tried transmitting nine. Needless to say, it didn't work. Once I discovered my error, changing the protocol was easy. POKEing a 122 into 17229 produces even parity seven word bits, and one stop bit. A 90 produces odd parity, seven word bits, and one stop bit. So far, though, I have not found a need for odd parity.

The MicroConnection provides an easy and efficient method of checking into the various bulletin boards (Forum-80, ABBS, etc.) as well as CompuServe and The Source. Also, the MicroPeripheral Corporation has a whole line of smart terminal programs as well as the other MicroConnections. It is well worth the investment to get into this exciting and interesting facet of the computer hobby.■



MM+ (Memory Expansion)
Exatron
Sunnyvale, CA
$399

by Harley Dyk

Owners of the 16K Level II TRS-80 Model 1 generally feel that they own a cost-effective computer. If you do a lot of programming you rapidly learn what the OM error means (programs always seem to grow and fill or exceed available memory). Most of you are not content with a cassette-based system and often turn to disks: Stringy Floppy, Beta-80, TC-8, etc. In either case, more memory and/or a floppy controller is often needed.

The MM+ (memory + interface) recently released by Exatron is a quality alternative to the Radio Shack expansion box and could cost you less depending on your situation.

Standard Features

The unit is designed to fit under the TRS-80 monitor and comes fully assembled. Standard features are: 32K of memory, built-in power supply, serial printer port (RS232-C), real time clock, light pen port, parallel printer port (Radio Shack/Centronics compatible), and a general parallel port (IBM Model 50 compatible). A floppy interface was not included as a standard feature since the unit was designed for TRS-80 owners who need more memory but do not own a floppy disk.

A floppy controller and an additional 32K will be the first options available on a second circuit board (which will fit in the present chassis). The present power supply runs at or under 50 percent capacity and will handle the additional load.

Stringy Floppy owners are being polled by Exatron as to their preference regarding options for the second board. Exatron hopes to offer the options in the order of preference indicated by the needs of its customers.

Options Under Consideration

Some of the options under consideration are: RS232-C serial I/O, hard disk controller, color graphics, communications modem, IBM Model 50 bidirectional interface (use typewriter keyboard), port FF audio output circuit (for sound effects), IEEE-488 bus interface, A/D and D/A interface, multi-port parallel I/O, and a TRIAC/SSR/OPTO-Isolator control interface.

Size	17" × 7" × 3"
State	Assembled only
Memory	32 K only
Floppy Interface	No (but an option soon)
Real Time Clock	Yes (can use with Level III Basic)
Serial I/O	Printer output only (300 & 600 baud)
Parallel Printer Port	Yes
Light Pen Port	Yes
Bus Extender	Yes
On-board power supply	Yes
Warranty	1 year/30 day money back
Dealers	None, mail-order or order through program chairman (selected Stringy Floppy owners)
Toll free number	Yes—800-538-8559
Contact	Exatron 181 Commercial St. Sunnyvale, CA 94086
Misc.	Guaranteed to run at 3.55 MHz, has memory bank select circuit so can add another 32 K, has on board memory-mapped address decoding

Fig. 1. Major Features of the MM+

Unique Features

The light pen port is a unique feature of the MM+. Most light pens can be used with a cassette recorder serving as an amplifier, however the port on the MM+ is more convenient and leaves the recorder free. The port was designed for the Photopoint light pen by MicroMatrix but should work with any light pen that is designed to connect to a cassette recorder.

Exatron offers a speed-up kit for the TRS-80 that allows you to run the computer with a 50 percent or 100 percent increase in speed. The MM+ is guaranteed to handle the 100 percent increase if your CPU board and memory will run at the 3.55 MHz frequency.

If you have no immediate need for a floppy interface and need more memory, this unit deserves your consideration. I have used my unit for several months with no problem, and have subjected it to overnight memory tests to confirm its quality. It works well with disk alternatives such as the Beta-80 and the Stringy Floppy. It should satisfy your memory requirements for the time and give you many additional, useful features.■

From p. 8

tween TRSDOS and 3741 single-density IBM-format diskettes. Conversion between EBCDIC and ASCII character sets is also done. So now, for $249, the Model II can become conversant with IBM at the diskette level. Catalog number is 26-4714.

Around August 1, we should also be shipping two new BiSync Communications packages for the Model II. One is a 3270 package, which allows communication with IBM Systems 360/370 and 30-Series CPU's, or any non-IBM devices equipped with BSC 3270. Catalog number is 26-4715, and the price is $995. The other BiSync package (3780) allows the Model II to function as a remote job entry terminal. You can select the use of IBM 2770/2780/3780/3741 protocols and communicate with IBM System 360/370, 30-Series, IBM 2780/3780 terminals, DEC PDP-11, VAX-11 or other devices equipped with binary synchronous communications capabilities. (Cat. no. 26-4716, also priced at $995.) For all three of these new packages, synchronous communications are through the A serial port on the Model II, and may operate at up to 19,200 baud, depending on length and type of communication connection used. Only half-duplex communications facilities are required. Installation is also required.

I have some plans for something very interesting next month. . . if I can do it. See you then. ■

80 ACCOUNTANT

by Michael Tannenbaum C.P.A.

The annual New York State Society of CPAs accounting show and conference was recently held in New York City. This show included a series of informative seminars and an exhibition of accounting-related equipment and supplies provided by local and national vendors.

Our firm provided a speaker for one of the scheduled seminars. The seminar, "Word Processing for the First-Time User", was completely filled. Obviously we had picked a topic of great interest.

This impression was amply confirmed as I toured the exhibition hall. It seemed that every other exhibit dealt with a computer related product or computer system. Apparently, the skepticism of the past is quickly yielding to enthusiasm as practitioners rush to embrace automated systems.

As I wandered through the exhibits, I found the large number of competing systems bewildering. Every vendor seemed to have the same software packages on display—word processors and general accounting systems. With each vendor touting the technical superiority of his or her system, I could see how a novice could become quite confused.

Whatever the apparent benefits of any software/hardware configuration, the novice should ask "How many are installed," "Where are they installed," and, "What users can be contacted". A vague or evasive answer to any of these questions should trigger an abrupt retreat.

A new computer user should always pick hardware and software which have the widest distribution. If help is needed, other users can be contacted to provide advice. There is nothing more comforting than an informative talk with someone who has been there.

Business Systems Users Group

Although '80 owners have a lot of company, there is need for an '80 business systems users group. The new COBOL business systems differ significantly from the BASIC systems initially offered for the Model II. These systems have different file handling procedures and require the development of custom interfaces to expedite data entry and processing.

Many users of the new COBOL systems have file problems caused by improper handling of system diskettes. If a job is not terminated by returning to the main menu and exiting to the operating system, an application file in use could be destroyed. The scary thing about this situation is that the bad file can then be backed up without any indication of error. The unfortunate system user finds out about the problem the next time the file is required.

"The skepticism of the past is quickly yielding to enthusiasm."

This situation and others occur because the COBOL business system users are, in a very real way, pioneers. In last month's column, I indicated that COBOL is a business system language that has been in wide commercial use for over 20 years. But until very recently, COBOL systems were only available on large computers. The COBOL system available on the Model II is quite new.

Tandy's decision to implement new Model II business software in COBOL was a reasonable one. There is probably no other language in which more business programming talent is available. Also with COBOL, business system designs can be protected since the source code need not be distributed with an executable system.

ISAM

The COBOL system implemented on the Model II gets much of its speed and flexibility through the use of ISAM (Indexed Sequential Access Method) file structures. This type of file organization stores key words used to control access to random data files in special sequential files called index files. When you want a record, locate the key word in the index file and obtain a pointer to the random file.

Although this method of file access is available to the BASIC programmer, use of the method requires coding. With the COBOL system, the COBOL compiler generates ISAM coding. The use of ISAM file structures is, therefore, transparent to the programmer and easily implemented.

Since the ISAM index must be updated after a random file is altered, the disk must remain in the computer until a job is completely terminated. Obviously, removing a disk in a COBOL system could be dangerous. With this in mind it is a bit difficult to understand why Radio Shack is releasing a three-disk system which requires disk swapping.

How's your love life?

A little dull around the edges? Routine? Predictable? Boring? Maybe all it needs is a little Interlude. Interlude is the most stimulating computer game ever conceived. It combines a computer interview, an innovative programming concept, and a one-of-a-kind manual to turn your love life into exciting, adventurous, delicious fun!

Interlude is: romantic... playful... outrageous... a fantasy. Interlude is: ■ A Bed of Roses (Interlude #1) ■ Mata Hari (Interlude #49) ■ The Chase (Interlude #7) ■ Rodeo! (Interlude #71) ■ The King and I (Interlude #60) ■ Some Enchanted Evening (Interlude #84) ■ Caveman Caper (Interlude #82) ■ From Here to Ecstasy (Interlude No. 30) ■ Satin Dreams (Interlude #72).

More than 100 Interludes are included in the program. Most are described in detail in the accompanying manual, but several surprise Interludes are buried in the program awaiting that very special time when your interview says you're ready. (When you learn secret Interlude #99, your love life may never again be the same!) Interlude can give you experiences you'll never forget. Are you ready for it?

Interlude™ 287
The Ultimate Experience.

INTERLUDE, 10635 Richmond, Houston, Texas 77042. I'm really ready. Send my Interlude today.

Apple II (16K)*
- ☐ Cassette ($18.95)
- ☐ Diskette ($21.95)
- ☐ Diskette—Pascal or DOS 3.3 ($21.95)

TRS-80 (Level II-16K)**
- ☐ Cassette ($18.95)
- ☐ Diskette ($21.95)

Poster
- ☐ 20"x 24" reproduction of this ad without ad copy ($4.95—includes shipping charges)

Available for immediate shipment.

Please enclose your check payable to INTERLUDE or complete the charge information:

Add $1.75 for shipping and handling.

☐ MASTERCARD ☐ VISA All charge customers must sign here. ____________

Account No. ____________ Expiration date ________ MasterCard Bank Code ________

CHARGE CUSTOMERS: Order by phone toll-free! **1-800-231-5768 Ext. 306** (Texas: 1-800-392-2348 Ext. 306)

Name ____________ Age ______

Address ____________

City ____________ State ________ Zip ________

*Apple II is a registered trademark of Apple Computers, Inc. **TRS-80 is a registered trademark of Radio Shack, a Tandy Co.

Examination of the directories of the first three-disk COBOL system, Accounts Receivable, provides the answer. The package is supplied with six diskettes: Three systems diskettes containing programs and three data diskettes. The first systems disk contains 38 programs and six data files, the second disk 14 programs, and the third 27 programs.

Obviously this system is too big to fit on a single diskette. Why three data diskettes? This system is only one element of an order/invoicing system. The other elements—Order Entry, Sales Analysis, and General Ledger use some of the same files. The Accounts Receivable system maintains permanent data files and generates data which the other systems use for user reports.

Advantages of the Receivable System

I am tempted to compare the receivable system to the single-disk receivable system examined last month. If this is done carelessly the new system hardly seems worthwhile. The single-disk system, if expanded to its maximum capacity of three drives, can handle up to 1800 accounts with 4100 open transactions. The latest system can only handle 800 accounts and 2500 open items.

The difference between the two systems lies in the analytical data which the new system can accommodate. The new system increases the number of general ledger account distributions from 26 to 100. With this quantity of general ledger accounts available a more detailed sales analysis is feasible. This should give a system user a better handle on sources of gross profit contribution by product line.

The system also accommodates the accumulation of standard cost data. With this statistic it is possible to develop a customer profitability profile, and there is evidence that such reports will be developed by the sales analysis module.

"Obviously, removing a disk in a Cobol system could be dangerous. . . ."

The new system accommodates up to 100 salespeople. A report is available that can be used for the preparation of sales personnel commission reports. To provide the detail required for this report, a special file is set up which accommodates up to 6000 lines of data.

In short, Tandy has developed a big business receivable system for the small firm. Unfortunately the relatively small number of accounts that the system can handle will limit its application. However, when and if the long rumored hard disk subsystem is released, the software will be ready.

However, if the system fits your needs you get quite a bang for a buck. The system has a tab indexed oversized binder. The indexing divides the documentation into separate sections for each processing operation. Each section preserves detailed instructions, and devotes particular care to specifying the range of acceptable entries required for each item on the CRT screen.

Each section provides sample data to aid the novice in learning systems procedures. The user is urged to enter this data and compare the resulting reports with the sample printouts provided. This method is quite effective in shortening the learning curve.

This method also provides an insight into the enormous amount of company data which must be entered into the system before processing can commence. The problem becomes obvious on the first data screen. In addition to the company name and address the user must decide whether or not to use profit center accounting. A no answer will affect the output from the system and the resulting financial reports generated by the general ledger system.

In the second screen, the user must decide on the format of the aging report, finance charges, and whether or not to use preprinted statements, account distributions, and sales personnel commissions. The answers to some of these questions should not be developed while entering data on the CRT.

Before you install a system with this potential, carefully review company records and the current management information system. Questions concerning profitability improvement and sales management should be considered. If possible, new directions should be defined and the order/invoice system used as the means of implementation.

In operation, the screens are clear and uncluttered. As options are selected, future data entry screens are affected. For example, if profit center accounting is not selected on the first screen, only a four-digit account number would be allowed on the second screen. Consider the interaction of the screens carefully when answering each question.

As data files are built, the files are used to check subsequent entries for validity. When entering a customer's data, the salesman number, ship via code, terms code and tax code are validity tested. All of the above as well as customer sales, cost of sales, commission paid and account balance are part of the customer master file.

The system then allows an open file to be created. At this point the documentation becomes somewhat confusing. The screen refers to two amounts which must be entered for each document. For unexplained reasons, the system requires freight charges to be separately identified on invoices. Discounts and allowances must also be segregated on payments. A note of explanation from the system designers would be helpful at this point. Segregating these balances during conversion will be quite a job and the option of avoiding this messy operation should be available.

This system, like all Radio Shack receivable systems, allows both balance forward and open item accounts. In addition, it can use miscellaneous account numbers. You can use these numbers for one-time or occasional customers. This avoids the trouble of setting up a master file for every new customer and should permit installation of the system in many firms which ordinarily would have too high an account volume to be considered candidates.

"Tandy has developed a big business receivable system for the small firm."

Once the files have been created, the system user will have a good familiarity with operation of the system and the screen design. Unfortunately the screens are not as informative and easy to use as previous Tandy accounting systems.

Some Differences

Unlike the General Ledger, the system doesn't make full use of cursor control for editing. If an error is made and not recognized until after Enter is pressed, backspacing the cursor to the previous line is not possible. The screen must be completed and the offending line number referenced for correction.

In addition, the Tab stop terminates processing and escapes the screen. This differs from procedures in other accounting packages and Scripsit where F1 and F2 are used. This system uses the special function keys during data entry and there may be some confusion for users of other 'Shack products. Regardless, the most regrettable flaw is that these key functions appear only in the documentation and are not displayed on the screen during processing.

Post Invoicing Sales Entry

Although this receivable system is just one module in an order-invoice system, it does contain a post invoicing sales entry routine. If implemented as a stand-alone system, it is the normal method for entering sales invoices. With this routine, entering an invoice is a two-screen operation. First, details of the overall invoice are entered. This consists of the customer number and 11 other pieces of information required by the system.

If the company requires a departmental sales analysis, the routine presents a second screen which allows distribution of the items sold to accounts affected. The user must enter a valid account number and amount. During the data entry procedure, the system displays the total amount and the amount distributed. Exit from the screen is not possible unless the amount distributed equals the amount invoiced.

Clearly this information is best developed during the preparation of an invoice. If invoice volume is substantial and the integrated order entry system is not going to be purchased, a custom invoicing program should be considered. This will require purchase of the COBOL development system and the accounts receivable source code. A COBOL programmer will have to be retained to do the programming.

Printing Capabilities

Much thought has gone into the development of the cash applications module. Unlike most other small systems that I have reviewed, this system allows printing of a cash applications worksheet. It prints a worksheet for all accounts or selected accounts and provides a convenient means of reconciling amounts received with items recorded in the accounts receivable ledger. Once the worksheet has been completed, it will guide the posting process and serve as hard copy documentation of item key-off decisions.

"If the system fits you get quite a bang for a buck."

If a worksheet is not used, the system allows direct inquiry into the accounts affected. If the customer account number is not known, a search routine locates the customer's record. When it locates the customer number, the name of the customer, balance method and payment terms on file are displayed. Because a customer number is required for all cash entries, a miscellaneous customer number should be defined for processing non-receivable collections.

Cash Application Procedures

Cash application procedures for open item and balance forward accounts differ. The system provides additional screens for open item accounts and an automatic cash application routine. With this routine, the program automatically pays off as many documents as it can, allowing discounts on qualifying documents where applicable. If the total amount received is not enough to cover all documents, then it applies a partial payment to the final document processed. If cash is left over, it is listed as an open credit.

All documents entered; cash, invoices, debit or credit memos, and adjustments, are retained in batch input files. While in this state, they can be altered without affecting the receivable balances. Each file can be printed and compared to input controls before posting. Once they have been posted no change is possible. Only a debit or credit memo may adjust a posted entry.

Unlike other systems, there is no automatic purge of keyed-off items. This means that it returns all entries affecting an account for printing or display on demand and the system will rapidly become choked with completed transactions unless a periodic purge is run. The user has complete control over the purging procedure. First prepare an eligibility report. This report details every eligible item to be purged. Once he makes selection of accounts or dates to be purged, the system user can purge the file or purge the file and print the deleted records.

The accounts receivable module is an impressive bit of work; I am quite eager to see the rest of the order-invoice system. I suspect that in combination they will provide a very useful management information system for progressive business executives. ■

✓See List of Advertisers on page 306

EDUCATION 80

by Earl R. Savage

"Many schools find that magazine subscription sales are their most significant fund-raising activity."

As computers become an increasingly significant part of the daily activities in your school, several difficulties are likely to arise. The first will be getting a sufficient quantity of TRS-80s to meet the growing demand—a tough problem in most districts.

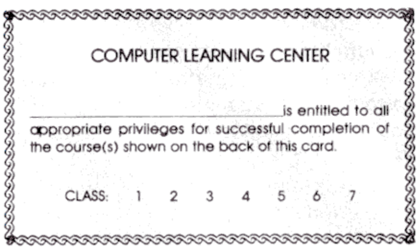
The second difficulty is the ever-present problem of acquiring *good* educational programs in one or many subject fields. Reviews of programs many times are not available resulting all too often in a disappointing purchase. An authoring program as described in the May *Education 80* column can help you write your own programs with a minimum of effort.

Sooner or later you will encounter another difficulty. That is the matter of keeping track of the level of proficiency reached by a growing number of students. When John comes in and asks to use a machine, how can you remember how independently he can function with it? What privileges has he earned? What capabilities has he demonstrated?

In a larger school, especially, this can be a real problem. You can spend a lot of time checking student records—even more in checking with other teachers who may have taught the students. Your best bet is to have a system in which each student carries his own record with him. Here is a description of one such plan.

The system consists of a small wallet-size identification/record card and several programmed instructional courses. It is suitable for use with students of any age.

The front of the student card is shown in Fig. 1. There is a place for a name and an indication of the class of privileges the student has earned. The back of the card lists available courses and provides spaces for additional ones. Beside each is a place for the date of completion and the instructor's initials.


COMPUTER LEARNING CENTER

________________________ is entitled to all appropriate privileges for successful completion of the course(s) shown on the back of this card.

CLASS: 1 2 3 4 5 6 7

Fig. 1 The Student Identification/Record Card

With this system in place, the student simply presents his card when he wishes to use the computer (and/or to sign up to use it). In addition, the card and its use provide a further degree of student motivation for mastery of the courses.

You will be interested in the qualifying programs used by this particular school as listed on the back of its card. Though the first four were locally developed, they will give you some ideas for your own. Of course, you can use any appropriate programs.

The Programs

The first program course is entitled Keyboard and is designed for the young beginner. The purpose of this interactive sound program is to familiarize the student with the keyboard.

Operation is the title of the second program. It, too, has sound and is interactive. Operation provides linear instruction in using the TRS-80. It covers such topics as List, Break, Continue, Reset, and CLOAD. Upon successful completion of this course, the student should be able to operate the machine independently.

The mechanics of writing elementary programs are presented in the third course, Programming I. The student receives instruction in BASIC through the examination of several simple programs. Among the statements covered are Print, Input, IF...THEN, and FOR...NEXT. There are brief discussions of arithmetic functions and variable types.

The fourth course is Graphics I. It presents the rudiments of graphics—both the Set and block varieties. A bit of animation is thrown in for good measure.

The final two listed programs are Part I and Part II of the Radio Shack Level II BASIC Course.

There you have the list of programs which constitute an informal computer literacy program in one school. If you have another type of system, all of us would like to hear about it.

Micro Funds—Another Source

This is not really another source of microcomputer funds but a source of microcomputers themselves. It comes about because of a recent agreement between Radio Shack and QSP. (QSP is the organization which helps put on school sales of magazines.)

Many schools find that magazine subscription sales are their most significant fund-raising activity. QSP will be offering TRS-80s as premiums for the first time. This is a chance for you and your students to earn computers for your school.

If your school already has a magazine subscription campaign, look into getting TRS-80s as premiums. If magazines sales are permitted in your district but your school does not have them, check on this source of TRS-80s. You can get details from William E. Drake, QSP, Inc., Box 2003, Ridgefield, CT 06877.

Math Games

As you plan your computer purchases for the fall, there are two inexpensive, unusual and very useful programs you should consider. These programs are games which help the students develop certain mathematical skills while they are having fun—painless learning, as it were. The presentation methods along with excellent graphics assure high student motivation.

The Estimation Game develops number sense and estimation skills in whole number computations. The student is presented problems in an unusual way and his estimates are followed through by the computer.

The Distance Game permits the student to attempt to locate a hidden point. With each try, he is given a visual and a mathematical hint. The game may be played in either two or three dimensions; the latter is especially challenging because of the mental imagery required.

Both programs were developed in schools by a mathematics educator. You will find that they are several cuts above the typical math program. They are available from Educational Programs, P.O. Box 2345, West Lafayette, IN 47906. ■

THE ASSEMBLY LINE

by William Barden, Jr.

"The budget for the Space Shuttle isn't quite comparable to the budget for this column, but this month's topic almost didn't make it for much the same reason . . ."

Wouldn't it be nice to program without any interface to hardware? I know some of you are heavily into number crunching, but assembly language programmers are exposed to hardware interfacing from time to time. I rose early recently to turn on the TV and watch a piece of hardware that couldn't get off the ground because of an incompatibility between hardware and software. The budget for the Space Shuttle isn't quite comparable to the budget for this column, but this month's topic almost didn't make it for much the same reason—hardware interfacing and a lack of documentation for a hardware glitch!

This month we'll look at that hardware interface and the software that drives it. You can talk directly to your system disk drives with it, bypassing TRSDOS, NEWDOS and LDOS.

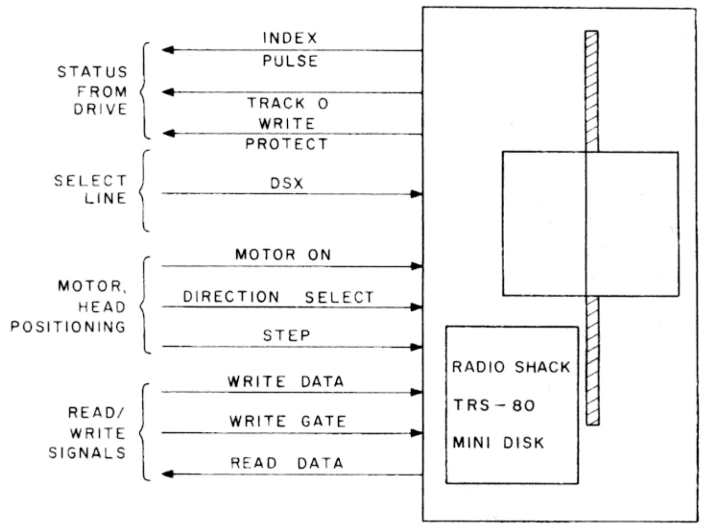


Fig. 2. SA-400 Signals

The Shugart SA-400

Many of you have Percom, Pertec, Micropolis and other disk drives. However, the story starts with Shugart, which has become another of those de facto standards. All disk drives look very much like a Shugart SA-400, and we must look at its specifications to see what's involved in disk I/O.

Lest you forget, data is arranged in 35, 40 or 77 tracks, on one or two sides of a diskette. In single density format, the tracks are divided into ten sectors. Each sector can hold 256 bytes of data, so there are 256•10 or 2560 bytes per track, or 2560•35 = 89,600 bytes of data on a 35-track diskette.

The diskette spins at 300 revolutions per minute, or five revolutions per second. Data passes under the disk head at a rate of 5•2650 = 12,800 bytes per second. A byte of data is available every 78 milliseconds (actually, every 64 milliseconds, as we shall see).

Data is arranged serially on each track, so that each track is made up of 2560•8 = 20,480 bits in a concentric circle. The physical arrangement is shown in Fig. 1.

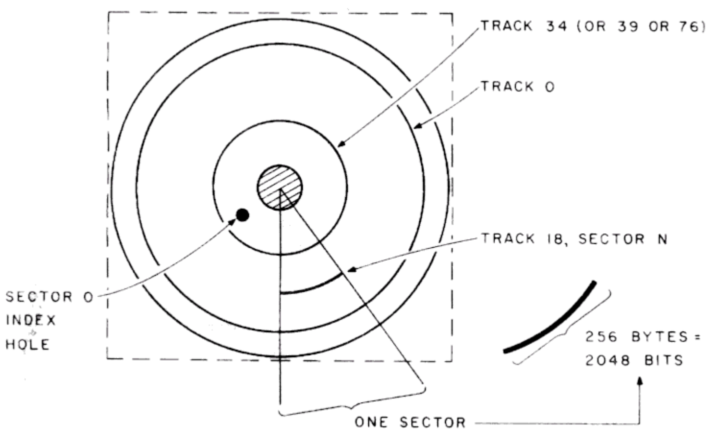


Fig. 1. Physical Format of Diskette

A Shugart SA-400 is another dumb device, at least as far as its primary functions. It can be instructed to turn on its motor, to step the head in or out one track, or to write or read a bit. It returns signals representing a sense of the diskette index hole, a track 0 position, and a write protected disk. The SA-400 has a circuit board full of logic to read and write serial data, but the circuitry is not much more sophisticated than that found in a cassette recorder. The signals that go to the SA-400 are shown in Fig. 2.

Disk Formatters and Controllers

Not too long ago, when a computer manufacturer wanted to use a floppy disk drive with his computer, he had to design a disk controller/formatter. This usually involved about 200 medium scale LSI chips acting as in interface from computer to drive.

First, the controller had to convert a byte to a serial bit stream to be sent to the single Write Data line of the disk drive. A similar eight bits had to be assembled from the Read Data line during disk read operations. Since the disk head can only

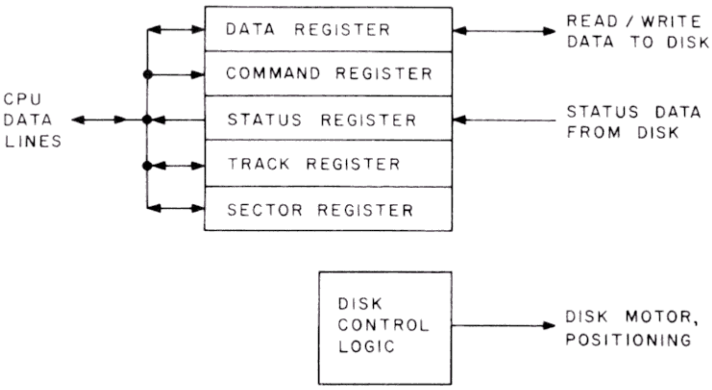


Fig. 3. WD1771 Architecture

be stepped one track at a time, a portion of logic in the controller was devoted to maintaining the location of the current track, and to stepping the disk drive head in and out to find a given track. Another portion of the logic was dedicated to finding a given sector within the track. Other functions included collecting status from the disk sensors. The controller required a clock to time disk actions.

Along with the controller, and usually designed into it, was a formatter. The formatter wrote skeleton tracks onto each diskette. Each skeleton track was divided into sectors, each sector containing an identification (ID) field. This field contained the track and sector number. A data field contained the user data. Special codes were used to mark the start of ID and data fields.

A Single-Chip Disk Controller

Western Digital has a beautiful building off the San Diego Freeway in Orange County, CA. Rumor has it that the second floor landing has a small brass plaque upon which is inscribed, "Dedicated to the 1771 Floppy Disk Formatter/Controller." The 1771 is the single 40-pin LSI chip which replaces those early 200-chip controllers (and presumably 200 design engineers). The 1771 is used in the Model I, and a close relative, the 1793, is used in the Model III.

The 1771 is really a CPU; it performs the following functions:

- It restores the disk read/write head to track 0.
- It automatically positions the head over a specified track. This is called a seek operation.
- It steps the head in and out one track.
- It steps the head in one track.
- It steps the head out one track.
- It reads a sector's worth of data.
- It writes a sector's worth of data.
- It reads the next identification field of the current track.
- It reads an entire track, including formatting data.
- It writes (formats) an entire track.
- It forces an interrupt.

A twelfth function that the 1771 also performs is returning a status byte that contains information about the disk drive and the success or failure of the current operation.

As the reader can see from the above commands, the 1771 requires some information about the track number and sector number before it can issue some of its commands. In fact, the 1771 contains five eight-bit registers that are accessible from a program. These are the status, command, track, sector and data registers. There are additional registers not addressable under program control. The most important of these is the data register, which holds eight bits of data to be shifted out serially (write to disk) or eight bits that have been assembled from the disk (read from disk). The general architecture of the 1771 is shown in Fig. 3.

The addresses assigned to the five addressable controller registers in the Model I are 37ECH, 37ECH, 37EDH, 37EEH and 37EFH, respectively, as shown in Table 1. Note that the status and command register share the same address, 37ECH; the status register is addressed for a read 37ECH, while the command register is addressed for a write 37ECH.

Positioning Commands

The first five commands, Restore, Seek, Step, Step In and Step Out, are head positioning commands. Restore steps the head until it is positioned over track zero. The track register does not have to contain a valid track number for Restore to work, and the track register contains a zero at the end of the operation.

The Seek command must be preceded by an output to the data register (37EFH) of the track number for the Seek. In addition, the track register must contain the current head position. This means that Restore must have been performed first, although other head positioning commands could follow Restore.

Step steps the head to the next track in the last used direction. Step In and Step Out also step the head one track.

Some of these commands share functions, and we *could* get by with only Restore and Seek. No data is written to the disk with any of these commands; they are used only to position the head.

The formats for these commands are shown in Table 2. The V bit, or verify bit,

ADDRESS	INPUT	OUTPUT
37ECH	STATUS REG	COMMAND REG
37EDH	TRACK REG	TRACK REG
37EEH	SECTOR REG	SECTOR REG
37EFH	DATA REG	DATA REG

Table 1. WD177 Controller Addresses

RI	RO	msec
0	0	6
0	1	6
1	0	10
1	1	20

COMMAND	FORMAT								TYPICAL
RESTORE	0	0	0	0	0	V	RI	RO	03H
SEEK	0	0	0	1	0	V	RI	RO	13H
STEP	0	0	1	U	0	V	RI	RO	33H
STEP IN	0	1	0	U	0	V	RI	RO	53H
STEP OUT	0	1	1	U	0	V	RI	RO	73H

Table 2. Head Positioning Commands

COMMAND	FORMAT								TYPICAL
READ SECTOR	1	0	0	M	B	1	0	0	8CH
WRITE SECTOR	1	0	1	M	B	1	0	0	ACH

Table 3. Read/Write Sector Commands

COMMAND	FORMAT								TYPICAL
READ ADDRESS	1	1	0	0	0	1	0	0	C2H
READ TRACK	1	1	1	0	0	1	0	$\overline{S}$	E4H
WRITE TRACK	1	1	1	1	0	1	0	0	F4H

Table 4. Read Address, Read/Write Track Commands

specifies that the track ID will be read and compared with the track register. The R1 and R0 bits define the stepping rate of the head. Some disks are designed to step at a faster rate than the nominal 20 millisecond track-to-track step for the SA-400. (I know you're going to change this rate—just be forewarned that your disk may not be capable of stepping at a faster rate!) Typical settings for the commands are also shown in the table.

Read and Write Sector Commands

The Read sector and Write sector commands are shown in Table 3. They read or write a sector of data. Before a Read or Write command can be given, the head must be positioned over the proper track with a head positioning command, and the sector register must be loaded with the proper sector number by an output to 37EEH. Once this preliminary work is done, the Read or Write sector command is output to 37ECH.

After receiving the Read sector command, the 1771 searches for the ID field that contains the proper sector number. When it finds it, it waits until the data field comes under the head and then starts assembling data bytes into the data register from the serial bit stream. As each byte becomes available, a data request or DRQ flag is set in the status register, and the program can pick up the byte by reading the data register at 37EFH.

Data is transferred from a sector on a byte-by-byte basis in a tight assembly language loop. The loop consists of checking the status (37ECH) for the DRQ bit and performing a read from the data register (37EFH) if data is present. Reading the data register resets the DRQ, and the program loops back again to check for the next byte.

How do we know when to stop? Actually, we don't have to know when to stop: The 1771 knows how long the data field is from the sector length byte in the ID field of the sector. (For an IBM-type format, 01H in the sector length byte of the ID field specifies 256 bytes of data.) When the 1771 reaches the end of the data file, it resets another bit in the status register, called the busy flag. This flag is normally set to indicate that the 1771 is executing a command; it is reset when the command has been completed. On a Read sector command, the busy flag is reset after the last byte has been assembled and read from the 1771.

On a Write sector command, the process is similar. The head must be correctly positioned over the desired track, and the sector register must contain the proper sector number. The DRQ flag in the status byte is used to signal the program that it must send the next data byte so the 1771 can convert it to a serial bit stream. Sending a byte to the 1771 data register resets the DRQ; the DRQ is set again after the byte has been written on disk. The process continues until the last sector byte has been written, at which time the busy flag bit in the status register is reset.

Going back to the format of the Read and Write sector commands in Table 3, we can see there are some microprogram bits that can be manipulated. The M bit specifies either a single record (0) or multiple records (1). If multiple records are specified, the 1771 will keep transferring data until the end of the track. The normal setting in the Model I is 0, or single record.

The B bit specifies an IBM format (1) or non-IBM format (0). The IBM format refers to a standard sector length of 128, 256, 512 or 1024 bytes per sector. The non-IBM format allows for lengths of 16 to 4096 bytes per sector, provided that the diskette was formatted to one of those sector lengths.

The A1, A0 bits define which data address mark the 1771 will use. The data address mark is a byte from F8H to FBH that precedes the user data field.

Read Address, Read Track and Write Track Commands

The remaining commands affect portions of the disk that the TRS-80 user never sees. We have talked about formatting a disk, but what is actually involved in the formatting process?

The 1771 uses the Write track command (Table 4) to format a track. One track at a time is formatted, so there must be 35 separate Write track commands to format a 35-track diskette, each one followed by a Step command to move the head to the next track.

The head is first positioned on track 0. A Write track command (F4H) is sent to the command register through an output to 37ECH. At this point the process resembles the Write sector procedure. The 1771 requests the next data byte by the DRQ in the status. The program responds by outputting to 37EFH, which resets the DRQ.

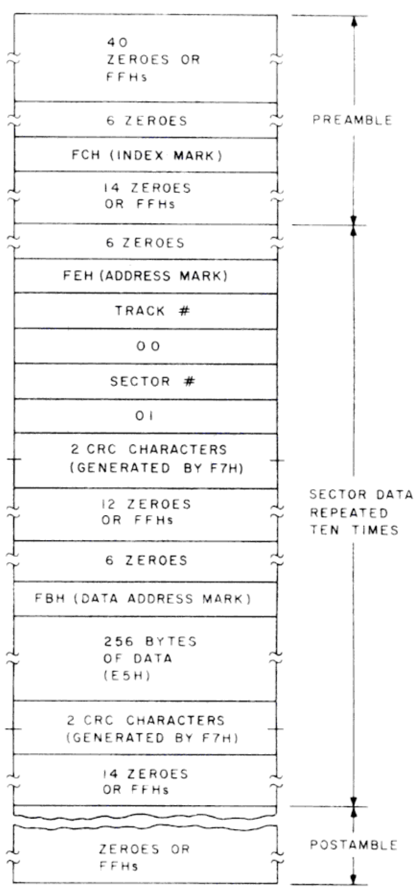


Fig. 4. Formatting Data

BILLIST

. . . organizes bill paying!

NO SYSTEM is the system many people use with their only priority being past dues.

Billist keeps track of bills due, bills paid, check numbers, dates paid and even repetitive cash payments!

Billist will keep track of up to 100 accounts a month. Store data on an inexpensive cassette with a complete listing of accounts, if printer equipped.

Billist allows six priorities.

Sort by priority, name or date due; compress; delete; advises you when an account is late!

Billist becomes indispensable . . . providing current account information without digging through the pile.

—THIS MONTH'S SPECIAL—

5.95 for the listing (1 lb.)
9.95 for the cassette (1 lb.)

Required equipment R.S. Lvl II or III with 16K STK #126

MASTERGUESS

. . . a challenge beyond Mastermind!

Masterguess by Elf II offers a test of logic enhanced by the creative use of TRS 80 graphics.

AVAILABLE NOW FOR ONLY $5.95

Required equipment R.S. Lvl II or III, 16K STK #127

SHIPPING RATES

- $2.50 for first 3 lbs. . . .55¢ ea. add'l. lb.
- COD's add $2.00
- Canadian order, please add 75¢ per lb.
- P.O. Boxes—include phone for UPS

Please allow 2 to 3 weeks for checks to clear.
3% cash discount (cash, check, money order or C.O.D.)

P.O. Box 1567, Wheat Ridge, Colorado 80033
Colorado Residents add 3% Sales Tax
P.O. Box 1017, Medina, Ohio 44258
Ohio Residents add 5½% Sales Tax
24 Hour phone Order Line 1-303-575-8518

LTDATA INC.

COMPUTER ACCESSORIES & EQUIPMENT

STOCK NO.	PRINTERS	PRICE
113	Okidata microline 80	$ 475.00
114	Okidata microline 82	614.00
115	Okidata microline 83	912.00
116	Tractors	70.00
117	RS-232-256 char. buf	180.00
118	RS-232 2K char. buf	264.00
119	Epson Mx-80	525.00
120	Centronics 737-1	720.00
121	MPI 88G	659.00
122	Vista Daisy Wheel	1545.00
123	NEC 5510 or 5530	2845.00

Shipping and insurance included on all Printers to 48 continental states. Blue Label to Hawaii and Alaska available.

STOCK NO.	RIBBONS	PRICE	WT.
109	Zip pack LP I, II, IV	$17.75/4	1 lb.
	Multistrike carbon		
110	Diablo hy type	12.95/3	1 lb.
111	Qume	9.95/3	1 lb.
112	NEC	20.45/3	1 lb.
	VERBATIM w/DATALIFE™ (with hub rings)		
103	MD 525-01,	26.95	1 lb.
104	MD 525-10	26.95	1 lb.
105	MD 525-16	26.95	1 lb.
	(for plastic box, add $1.00/box) 10 boxes, mix or match $252.50		
106	FD 34-8000	39.95 (x10 $360.00)	2 lb.
107	with hub ring and hard box	43.95 (x10 $400.00)	

	CASSETTES TDK/ MAXELL	PRICE	SHPG. WT.
100	D-C30/LN-30	$14.95/10	3 lb.
101	D-C60/LN-60	18.45/10	3 lb.
102	D-C90/LN-90	22.95/10	3 lb.
	TERMINALS		
124	Hazeltine H 1410	$765.00 (free shipping)	
	MEMORY CHIPS		
125	Motorola MCM 4116 AC 20; 16K, 200 ns. memory chips for mod I, expansion interface and mod III	$21.95/8 (shunts not included)	2 lbs.

ONLY 50 SETS AVAILABLE

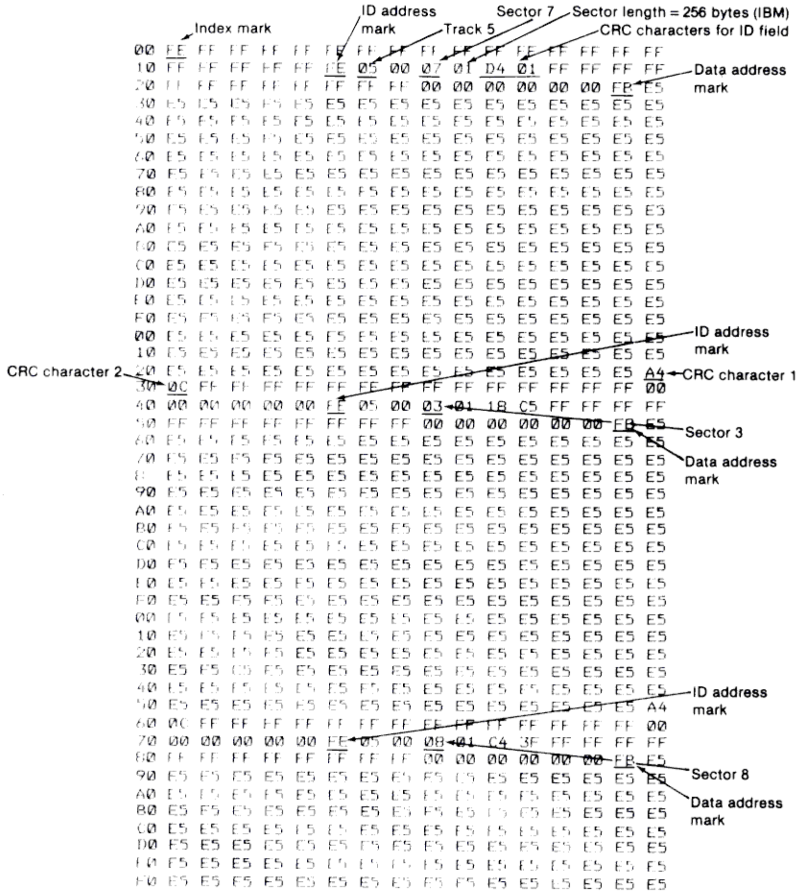


Fig. 5. Formatted Track Segment

The formatting process is done by writing 10 separate blocks of information, for the 10 sectors, for each track formatted. Each block of information consists of the segments shown in Fig. 4. Data is written by the 1771 as soon as it detects the index hole in the diskette.

The program writes a string of 40 zeroes or FFHs, followed by six zeroes. Next, an FCH is written, followed by 14 zeroes or FFHs. This pattern is a preamble that syncs the hardware on subsequent reads. The FCH is used as a sync mark to define the beginning of each track.

It's important to note that any character from F7H through FEH is a special character for the Write track command processing. A character in this range causes a special hardware action in the 1771.

After the preamble, the data for 10 sectors is output. The first six bytes constitute a leader made up of zeroes. The next byte is the special hardware character of FEH. This character is an ID address mark that identifies the beginning of the identification field. The identification field follows. Five bytes are output, but six bytes are actually generated in the field. The track number, zeroes, sector number and sector length (one for the IBM format of 256 bytes, multiples of 16 for non-IBM formats) are all written on the track. The next byte is the special character F7H, which causes two CRC characters to be written to the disk. The CRC (Cyclical Redundancy Check) characters are checksums of the ID field data.

The ID field data is followed by a program output of 12 zeroes or FFHs, followed by six zeroes, followed by the special character FBH. FBH is a data address mark which identifies the beginning of the user data area.

The user data area is reserved for the 256 bytes of sector data (IBM-type format with sector length of 01H) that we normally associate with a disk sector. In the formatting process, no meaningful data is output, but some non-conflicting data, such as E5H, is written instead. By non-conflicting, I mean any data except for the special hardware codes of F7H through FEH.

After 256 bytes of E5Hs are output (or multiples of 16 bytes for non-IBM format), the formatting program outputs an F7H which causes two CRC bytes to be generated. These bytes are a checksum of the data field. Finally, 14 zeroes or FFHs are written as a trailer in the sector.

This process is repeated nine times to make up the 10 sectors of the track. After the last sector has been written, zeroes or FFHs are written until the 1771 busy bit is reset (index mark detected again).

Fig. 5 shows a partial track of data after formatting. The data starts with the FCH index mark and continues for two and one-half sectors or so.

One interesting point about the tracks is that the sectors do not follow each other sequentially. Instead of sectors 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 on the disk, we find 0, 5, 1, 6, 2, 7, 3, 8, 4, 9! The reason for this strange arrangement is that the sectors have been optimized, to allow for some processing between the next sector in sequence. For similar reasons, the tracks do not necessarily start with sector 0.

The Read address command reads the next ID field on the current track. The six bytes of the ID field are read in an operation identical to the Read sector operation, except that the busy flag bit is reset after the six bytes have been read.

The Read track and Write track commands result in operations identical to the Read and Write sector commands, except that the entire track, and not just the data field, is read or written. The Write data command initiates the formatting sequence. The S bar bit in the Read track command is used to synchronize the reading of data to each address mark (0) or to simply read in data (1).

Force Interrupt

The last command can be used to terminate the current 1771 operation upon attaining a specified condition. It would never be used on the Model I. The term interrupt does, in fact, refer to an interrupt from the 1771 to the Model I, but the disk operations in the Model I are not interrupt driven.

Status

The status register at 37ECH holds status dependent upon the current command in progress, as shown in Table 5.

Bit 0 is always a busy flag bit, set to one when a 1771 operation is in progress, and to zero when the operation is over. Likewise, bit 7, Not RDY, is set when the disk drive is not ready.

Write protect is a direct reading from

the write protect notch sensor. (If the notch is covered, the disk is write protected.) The Record type during a Read is the actual data address mark encountered in the data field. Write fault and CRC error indicates errors in reading or writing data.

Track 0 indicates that the head is over track zero. Index indicates that the index hole is passing under the sensor. Lost data indicates that the program could not keep up with the data transfer. During a write procedure, this means the program didn't present the next data byte in time to write it at the next point on the track; during a read, it means the next byte filled the data register before the program read in the previous byte. Either one is a catastrophic condition.

We mentioned earlier that data is transferred at a rate of one byte every 64 microseconds. This figure is derived from the rotational speed of the disk and the amount of data (including format data) on a track. If we assume that each Model I instruction takes five microseconds, we can execute about 13 instructions in a read or write disk loop! It doesn't take too much overhead to fall behind—if the real-time clock interrupts are enabled, this puts additional processing in the loop as the code for the real-time clock routine is entered. It's important, therefore, to keep the DRQ loop very tight to avoid missing data.

A Disk Driver Program

You're now ready to use a disk driver program, which I have aptly named DSKDRV (see Program Listing 1). DSKDRV enables you to read disk status, to read and write a sector, to read the ID field, and to read and write a track. The latter function would allow you to format a track if you have a buffer full of data.

DSKDRV is called with HL pointing to a parameter block that defines the disk parameters to be used. If you are calling this program from BASIC, note that it is not relocatable. If you are not calling DSKDRV from BASIC, NOP the three bytes at F009H, AH and BH with three 00Hs, or reassemble with the Call 0A7FH deleted. I would not advise using this on your non-disk Level II system.

The parameter block is shown in Fig. 6. The first byte is the function code. The next two bytes are the sector and track numbers. The final two bytes are the buffer address for reads and writes. This address, of course, is in standard Z-80 format with least significant byte followed by most significant byte. PARAM + 5 is the type of completion and PARAM + 6 is the status after completion. Both of these bytes are used to return data to the calling program.

BIT POS'N	7	6	5	4	3	2	1	0
HEAD POSITION	NOT RDY	WRITE PROT	HEAD ENGAGED	SEEK ERROR	CRC ERROR	TRK 0	INDEX	BUSY
READ SECTOR		REC TYPE	REC TYPE	REC NOT FOUND		LOST DATA	DRQ	
WRITE SECTOR		WRITE PROT	WRITE FAULT	REC NOT FOUND				
READ ADDRESS		0	0	ID NOT FOUND				
READ TRACK		0	0	0	0			
WRITE TRACK		WRITE PROT	WRITE FAULT	0	0			

Table 5. Status for Commands

Program Listing 1. DSKDRV

```
FØØØ                       ORG     ØFØØØH          ;***CHANGE THIS***
                 ;**********************************************************
                 ;* DISK DRIVER. CHECKS DISK STATUS, POSITIONS HEAD,      *
                 ;* READS AND WRITES SECTORS, READS ID, READS AND WRITES  *
                 ;* TRACKS.                                               *
                 ;*    CALLING SEQUENCE: HL=> PARAMETER BLOCK             *
                 ;*           PARAM+Ø: FUNCTION: Ø=RD STATUS, 1=POSITION  *
                 ;*               HEAD, 2=READ SECTOR, 3=WRITE SECTOR,    *
                 ;*               4=READ ID DATA, 5=READ TRACK, 6=WRITE   *
                 ;*               TRACK                                   *
                 ;*           PARAM+1: SECTOR NUMBER                      *
                 ;*           PARAM+2: TRACK NUMBER                       *
                 ;*           PARAM+3,+4: BUFFER ADDRESS                  *
                 ;*           PARAM+5: TYPE COMPLETION: Ø=OK, 1=POSITION  *
                 ;*               ERROR, 2=READ/WRITE ERROR               *
                 ;*           PARAM+6: LAST STATUS OR STATUS OF FAILURE   *
                 ;*           PARAM+7: BIT 7:Ø=NO WAIT, 1=WAIT  BITS 6-Ø: *
                 ;*               DRIVE NUMBER                            *
                 ;**********************************************************
                 ;
FØØØ F3          DSKDRV  DI                      ;DISABLE INTERRUPTS
FØØ1 F5                  PUSH    AF              ;SAVE REGISTERS
FØØ2 C5                  PUSH    BC
FØØ3 D5                  PUSH    DE
FØØ4 E5                  PUSH    HL
FØØ5 DDE5                PUSH    IX
FØØ7 FDE5                PUSH    IY
FØØ9 CD7FØA              CALL    ØA7FH           ;***GET PB LOC'N***
FØØC E5                  PUSH    HL              ;TRANSFER TO IX
FØØD DDE1                POP     IX
FØØF AF                  XOR     A               ;ZERO A
FØ1Ø DD77Ø5              LD      (IX+5),A        ;ZERO TYPE COMPLETION
FØ13 DD77Ø6              LD      (IX+6),A        ;ZERO STATUS
FØ16 DD7EØ7              LD      A,(IX+7)        ;GET DRIVE #
FØ19 E6Ø3                AND     3               ;MASK OUT WAIT BIT
FØ1B 3C                  INC     A               ;Ø-3 BECOMES 1-4
FØ1C 47                  LD      B,A             ;NOW IN B
FØ1D 3E8Ø                LD      A,8ØH           ;BIT FOR SELECT
FØ1F Ø7          DSKØ1Ø  RLCA                     ;ALIGN
FØ2Ø 1ØFD                DJNZ    DSKØ1Ø           ;CONVERT TO POSIITON
FØ22 32EØ37              LD      (37EØH),A       ;SELECT DRIVE
FØ25 DDCBØ77E            BIT     7,(IX+7)        ;TEST WAIT BIT
FØ29 28Ø9                JR      Z,DSKØ3Ø        ;GO IF NO WAIT
FØ2B 21ØØØØ              LD      HL,Ø            ;WAIT COUNT
FØ2E 25          DSKØ2Ø  DEC     H                ;DECREMENT COUNT
FØ2F 2ØFD                JR      NZ,DSKØ2Ø        ;GO IF NOT Ø
FØ31 2D                  DEC     L                ;DECREMENT COUNT LSB
FØ32 2ØFA                JR      NZ,DSKØ2Ø        ;GO IF NOT Ø
FØ34 3AEC37      DSKØ3Ø  LD      A,(37ECH)        ;GET STATUS
FØ37 ØF                  RRCA                     ;TEST BUSY
FØ38 38FA                JR      C,DSKØ3Ø         ;GO IF BUSY
FØ3A FD21ØØØØ            LD      IY,Ø            ;FOR MULTIPLICATION
FØ3E DD4EØØ              LD      C,(IX+Ø)        ;GET FUNCTION
FØ41 Ø6ØØ                LD      B,Ø             ;NOW IN BC
FØ43 FDØ9                ADD     IY,BC           ;FUNCTION*1
FØ45 FDØ9                ADD     IY,BC           ;FUNCTION*2
FØ47 FDØ9                ADD     IY,BC           ;FUNCTION*3
FØ49 Ø1DCFØ              LD      BC,FTAB         ;FUNCTION TABLE ADDRESS
```

Program continues

THE ASSEMBLY LINE

PARAM + 7 contains the drive numbers. The most significant bit of this byte is set to one if a pause is to occur before the disk is accessed, or to a zero if no pause is to occur. The disk is spinning only when it is accessed. It takes about a second to bring the disk up to speed for read and write operations. Before any disk operation is performed, a disk must be selected and brought up to speed. If no further operations are done, the disk will turn off after about three seconds. As long as the disk is spinning and consecutive disk operations are being done, there is no need to wait. However, if no operation has yet been done, or if the disk has turned off, the wait bit should be on.

DSKDRV Structure

DSKDRV flows from beginning to end (don't laugh—a lot of programs don't). In a concession to structured programming advocates, there are no computed GOTOs.

The order of operations are: Select a drive; position head if necessary; output to sector register if necessary; read or write if

PARAM + 0	FUNCTION	
+ 1	SECTOR NUMBER	
+ 2	TRACK NUMBER	
+ 3	BUFFER ADDRESS	
+ 4		
+ 5	TYPE COMPLETION	*
+ 6	STATUS	*
+ 7	W | DRIVE #	

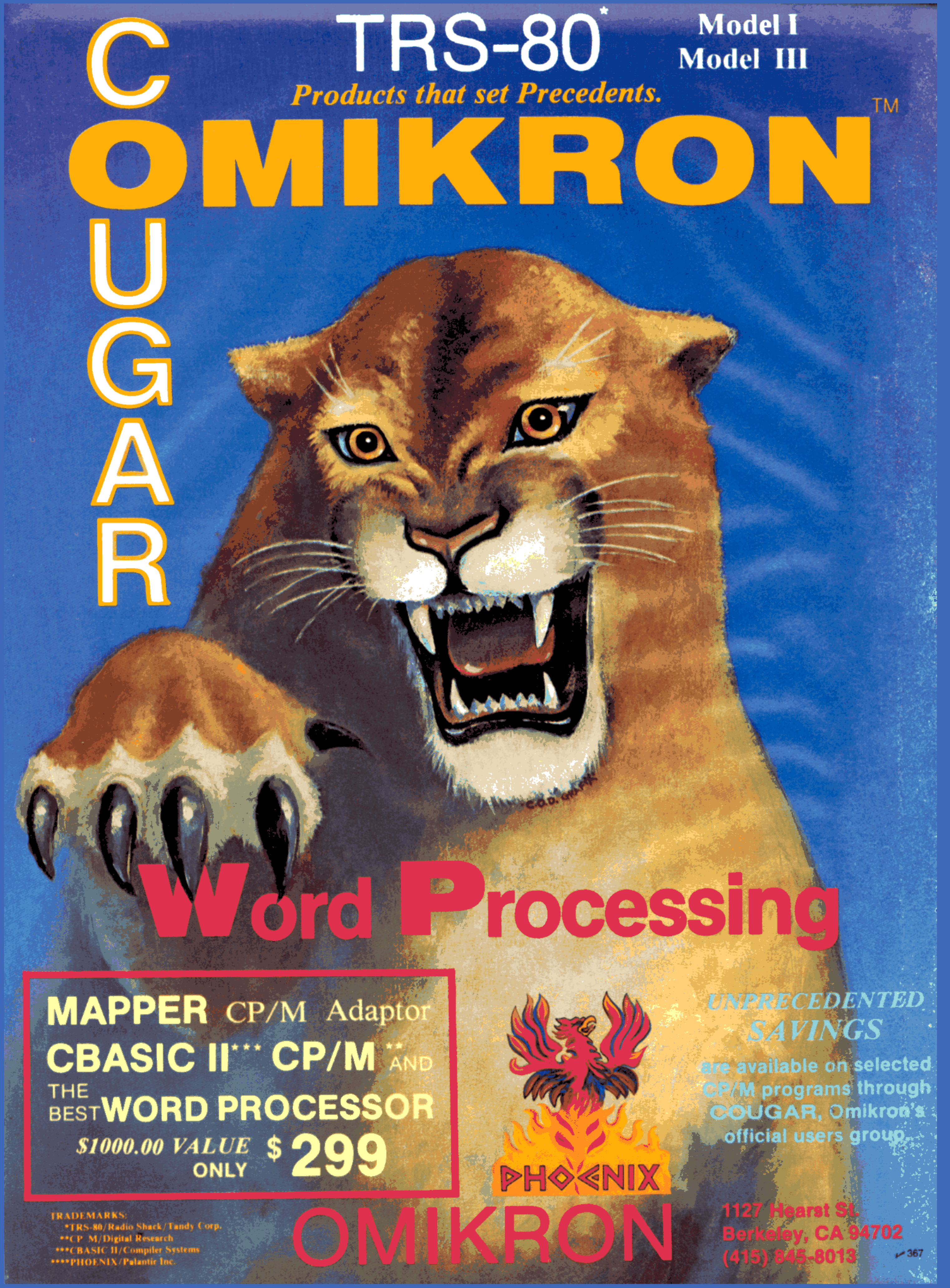
W: 0 = NO WAIT; 1 = WAIT

* SUPPLIED ON RETURN

Fig. 6. DSKDRV Parameter Block Input

```
F04C FD09            ADD   IY,BC          ;POINT TO ENTRY
F04E FD4E00          LD    C,(IY+0)       ;GET COMMAND
F051 FD4601          LD    B,(IY+1)       ;GET SEQUENCE
F054 FD5E02          LD    E,(IY+2)       ;GET STATUS MASK
F057 CB40            BIT   0,B            ;CHECK TRACK BIT
F059 282A            JR    Z,DSK070       ;GO IF NO TRACK ACTION
F05B DD7E02          LD    A,(IX+2)       ;GET TRACK NUMBER
F05E B7              OR    A              ;TEST FOR ZERO
F05F 2004            JR    NZ,DSK040      ;GO IF NOT RESTORE
F061 3E03            LD    A,3            ;RESTORE COMMAND
F063 1807            JR    DSK050         ;GO TO RESTORE
F065 32EF37   DSK040 LD    (37EFH),A      ;OUTPUT TRACK #
F068 D5              PUSH  DE             ;WASTE TIME
F069 D1              POP   DE
F06A 3E17            LD    A,17H          ;SEEK COMMAND
F06C 32EC37   DSK050 LD    (37ECH),A      ;OUTPUT RESTORE OR SEEK
F06F D5              PUSH  DE             ;WASTE TIME
F070 D1              POP   DE
F071 D5              PUSH  DE
F072 D1              POP   DE
F073 3AEC37   DSK060 LD    A,(37ECH)        ;GET STATUS
F076 0F              RRCA                   ;TEST BUSY
F077 38FA            JR    C,DSK060         ;LOOP IF BUSY
F079 07              RLCA                 ;RESTORE STATUS
F07A DD7706          LD    (IX+6),A       ;STORE STATUS
F07D E698            AND   98H            ;TEST STATUS
F07F 2804            JR    Z,DSK070       ;GO IF OK
F081 3E01            LD    A,1            ;POSITION ERROR FLAG
F083 184B            JR    DSK120         ;GO TO STORE
F085 CB48     DSK070 BIT   1,B            ;GET SECTOR BIT
F087 2808            JR    Z,DSK080       ;GO IF NO SECTOR ACTION
F089 DD7E01          LD    A,(IX+1)       ;GET SECTOR
F08C 32EE37          LD    (37EEH),A      ;OUTPUT TO SECTOR REGISTER
F08F D5              PUSH  DE             ;WASTE TIME
F090 D1              POP   DE
F091 CB50     DSK080 BIT   2,B            ;GET READ/WRITE BIT
F093 D5              PUSH  DE             ;SAVE STATUS CHECK BITS
F094 282E            JR    Z,DSK110       ;GO IF NO READ/WRITE ACTION
F096 79              LD    A,C            ;GET COMMAND
F097 DD5E03          LD    E,(IX+3)       ;GET BUFFER ADDRESS
F09A DD5604          LD    D,(IX+4)
F09D 21EC37          LD    HL,37ECH       ;STATUS REGISTER ADDRESS
F0A0 77              LD    (HL),A         ;OUTPUT COMMAND
F0A1 CB58            BIT   3,B            ;TEST READ/WRITE TYPE
F0A3 D5              PUSH  DE             ;WASTE TIME
F0A4 D1              POP   DE
F0A5 D5              PUSH  DE
F0A6 D1              POP   DE
F0A7 01EF37          LD    BC,37EFH       ;DATA REGISTER ADDRESS
F0AA 280C            JR    Z,DSK100       ;GO IF READ
F0AC 7E       DSK090 LD    A,(HL)           ;GET STATUS
F0AD 0F              RRCA                   ;BUSY TO C
F0AE 3014            JR    NC,DSK110        ;GO IF DONE
F0B0 0F              RRCA                   ;DRQ TO C
F0B1 30F9            JR    NC,DSK090        ;GO IF NOT READY
```

Program continues



TRS-80*
Model I
Model III
Products that set Precedents.
OMIKRON™
COUGAR
Word Processing
MAPPER CP/M Adaptor
CBASIC II*** CP/M** AND
THE BEST WORD PROCESSOR
$1000.00 VALUE ONLY $299
PHOENIX
UNPRECEDENTED SAVINGS
are available on selected CP/M programs through COUGAR, Omikron's official users group.
TRADEMARKS:
*TRS-80/Radio Shack/Tandy Corp.
**CP/M/Digital Research
***CBASIC II/Compiler Systems
****PHOENIX/Palantir Inc.
OMIKRON
1127 Hearst St.
Berkeley, CA 94702
(415) 845-8013
367

```
F0B3 1A                     LD      A,(DE)          ;GET BYTE
F0B4 02                     LD      (BC),A          ;OUTPUT TO DISK
F0B5 13                     INC     DE              ;BUMP MEMORY POINTER
F0B6 18F4                   JR      DSK090          ;GO FOR NEXT
F0B8 7E           DSK100    LD      A,(HL)          ;GET STATUS
F0B9 0F                     RRCA                    ;BUSY TO C
F0BA 3008                   JR      NC,DSK110       ;GO IF DONE
F0BC 0F                     RRCA                    ;DRQ TO C
F0BD 30F9                   JR      NC,DSK100       ;GO IF NOT READY
F0BF 0A                     LD      A,(BC)          ;GET BYTE
F0C0 12                     LD      (DE),A          ;STORE
F0C1 13                     INC     DE              ;BUMP MEMORY POINTER
F0C2 18F4                   JR      DSK100          ;GO FOR NEXT
F0C4 3AEC37       DSK110    LD      A,(37ECH)       ;GET STATUS
F0C7 DD7706                 LD      (IX+6),A        ;STORE
F0CA C1                     POP     BC              ;RESTORE MASK BITS
F0CB A1                     AND     C               ;TEST
F0CC 2805                   JR      Z,DSK130        ;GO IF OK
F0CE 3E02                   LD      A,2             ;ERROR CODE
F0D0 DD7705       DSK120    LD      (IX+5),A        ;STORE IN COMPLETION TYPE
F0D3 FDE1         DSK130    POP     IY              ;RESTORE REGISTERS
F0D5 DDE1                   POP     IX
F0D7 E1                     POP     HL
F0D8 D1                     POP     DE
F0D9 C1                     POP     BC
F0DA F1                     POP     AF
F0DB C9                     RET                     ;RETURN TO CALLING PROG
F0DC              FTAB      EQU     $               ;FUNCTION TABLE
F0DC 0000                   DEFW    0000H           ;STATUS
F0DE 00                     DEFB    0
F0DF 0001                   DEFW    0100H           ;SEEK
F0E1 18                     DEFB    18H
F0E2 8C07                   DEFW    078CH           ;READ SECTOR
F0E4 1C                     DEFB    1CH
F0E5 AC0F                   DEFW    0FACH           ;WRITE SECTOR
F0E7 7C                     DEFB    7CH
F0E8 C405                   DEFW    05C4H           ;READ ADDRESS
F0EA 1C                     DEFB    1CH
F0EB E405                   DEFW    05E4H           ;READ TRACK
F0ED 04                     DEFB    04H
F0EE F40D                   DEFW    0DF4H           ;WRITE TRACK
F0F0 44                     DEFB    44H
0000                        END
```

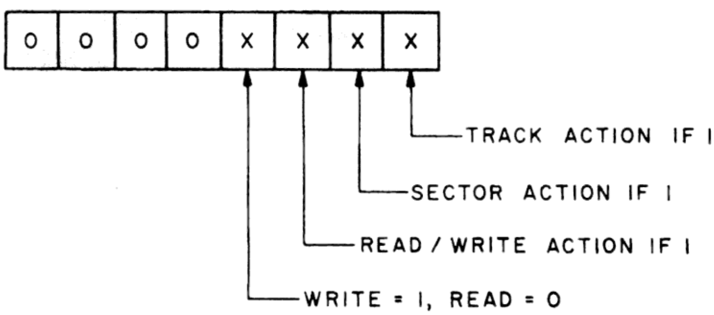


Fig. 7. DSKDRV Sequence Code

necessary; and get status.

First the interrupts are turned off and the registers saved. A Call is made to 0A7FH to get the address of the parameter block. Next, the completion type and status are zeroed.

At about DSK010, the drive number is picked up from PARAM + 7. A possible wait bit is masked out by AND 3. The drive number must be converted to 01H, 02H, 04H or 08H so the proper drive may be selected by an output to address 37E0H. Outputting to this address loads a four-bit latch in the expansion interface and turns on a one shot for about three seconds. The four select lines on the disk cable connect to the latch, and the one corresponding to the drive number will be low after the LD (37E0H),A instruction.

The conversion from zero through three to a bit position is handled by shifting 80H left the number of times corresponding to the drive number. After the drive has been selected, the wait bit is tested. If a wait is specified, a time delay of 65,536 counts is performed at DSK020.

The loop at DSK030 tests the busy status of the disk by reading the status register at 37ECH. If the disk is not busy, the loop falls through.

The code from this point up to DSK040 accesses a function table called FTAB based on the function in PARAM + 0. Each entry in the function table is made up of three bytes. The first byte is a sequence code for the function, the second is the principal command for the function, and the third is a status mask.

The sequence code is shown in Fig. 7. The four lower-order bits define operations for track action, sector action, read/write action and read/write function. They are a type of microcoding for operations in DSKDRV. As an example, the sequence code for Write track is 0DH, specifying head positioning over a track, no sector action, and read/write action with a write.

The command for the function is the actual command to be output. The status byte is, in fact, a mask byte. When the final status is obtained, this value can be ANDed with the final status, and if any one bit falls through, an error has occurred. The sequence, command and status mask are put in B, C and E, respectively.

The code before DSK040 checks for track action in the sequence byte by: BIT 0,B. If track action is called for, the track number from PARAM + 2 is loaded. If this track number is a zero, a Restore function is done; if non-zero, a Seek function is done. The track number is loaded into the data register at DSK040 for a Seek. At DSK050, the Seek or Restore command is output.

That status loop at DSK060 loops until the Seek or Restore operation has terminated (busy reset). The status is then stored in PARAM + 6, and a check is made of the validity of the Seek or Restore action. If any bit represented by 98H is on in the status byte, an error Seek or Restore has occurred, a type completion of 1 is stored in PARAM + 5, and an abnormal return is made.

If everything is proper, DSK070 tests the sector action. If sector action is called for (BIT 1,B), the sector number is picked up from PARAM + 1, and an output is made to the sector register at 37EEH.

The code at DSK080 tests the read/write action (BIT 2,B). If there is to be a read or write, the command from C is output to the command register by: LD (HL),A. The buffer address is put into DE and the data register address into BC; this allows for a tight read/write loop.

Next, bit three of the sequence byte is checked to determine whether the action is to be a read or write. A write action occurs at DSK090, while a read occurs at DSK100. In both cases, the busy bit is tested first to see if the last byte of the action has been transferred. If not, the DRQ bit is tested to see if the 1771 is ready for the next byte (write) or if it has the next byte (read). As the length of the operation is implicit in the command, no check must be made of the number of bytes transferred by the program.

At the end of the read or write, the busy flag is reset and DSK110 is entered. The final status is read from the status register by: LD A,(37ECH). This status is ANDed with the status mask; if the one-bit is set in the result, an abnormal type 2 completion has occurred, and this code is stored in PARAM + 5. Otherwise, the zero initially put in the completion type remains on the return to the calling program.

How to Use DSKDRV

Not all the 1771 commands are used in

```
20 'DISK DRIVER DRIVER
40 DEFUSR0=&HF000
60 CLS
80 INPUT "INPUT FUNCTION: 0=RD STATUS 1=POSITION HEAD 2=READ SEC
TOR
3=WRITE SECTOR 4=READ ID DATA 5=READ TRACK 6=WRITE TRACK";F
100 IF F<0 OR F>6 GOTO 80
120 POKE 61432-65536,F
140 IF F<>0 GOTO 240
160 GOSUB 460
180 A=USR0(61432-65536)
200 GOSUB 860
220 GOTO 80
240 IF F<>1 GOTO 340
260 GOSUB 460:GOSUB 680
280 A=USR0(61432-65536)
300 GOSUB 860
320 GOTO 80
340 GOSUB 460:GOSUB 600:GOSUB 680:GOSUB 760
360 A=USR0(61432-65536)
380 GOSUB 860
400 IF F=2 OR F=3 THEN L=256 ELSE IF F=4 THEN L=6
ELSE L=3000
420 I=B:J=B+L-1:GOSUB 940
440 GOTO 80
460 INPUT "DRIVE #";D
480 IF D<0 OR D>3 GOTO 460
500 INPUT "WAIT(W) OR NO WAIT(N)";W$
520 IF W$<>"W" AND W$<>"N" GOTO 500
540 IF W$="W" THEN D=D+128
560 POKE 61439-65536,D
580 RETURN
600 INPUT "SECTOR #";S
620 IF S<0 OR S>9 GOTO 600
640 POKE 61433-65536,S
660 RETURN
680 INPUT "TRACK #";T
700 IF T<0 OR T>39 GOTO 680
720 POKE 61434-65536,T
740 RETURN
760 INPUT "BUFFER ADDRESS";B
780 IF B<0 OR B>61431 GOTO 760
800 POKE 61435-65536,B-INT(B/256)*256
820 POKE 61436-65536,INT(B/256)
840 RETURN
860 PRINT "COMPLETION=";PEEK(61437-65536)
880 PRINT "STATUS=";PEEK(61438-65536)
900 FOR I=0 TO 300:NEXT I
920 RETURN
940 M=0
960 FOR K=I TO J
980 IF M<>INT(M/16)*16 GOTO 1040
1000 IF M=256 THEN M=0
1020 L=M:GOSUB 1180
1040 IF K>32768 THEN L=PEEK(K-65536) ELSE L=PEEK(K)
1060 GOSUB 1180
1080 M=M+1:IF M=INT(M/16)*16 THEN PRINT
1100 IF INKEY$<>"" GOTO 1140
1120 NEXT K
1140 PRINT
1160 RETURN
1180 L1=INT(L/16):L2=L-L1*16
1200 IF L1<10 THEN PRINT CHR$(L1+48); ELSE PRINT CHR$(L1+55);
1220 IF L2<10 THEN PRINT CHR$(L2+48); ELSE PRINT CHR$(L2+55);
1240 PRINT" ";
1260 RETURN
```

Program Listing 2. Disk Driver Driver

✓See List of Advertisers on page 306

✓107


```
INPUT FUNCTION: 0=RD STATUS 1=POSITION HEAD 2=READ SECTOR
3=WRITE SECTOR 4=READ ID DATA 5=READ TRACK 6=WRITE TRACK? 0
DRIVE #? 0
WAIT(W) OR NO WAIT(N)? W
COMPLETION= 0
STATUS= 0
INPUT FUNCTION: 0=RD STATUS 1=POSITION HEAD 2=READ SECTOR
3=WRITE SECTOR 4=READ ID DATA 5=READ TRACK 6=WRITE TRACK?
BREAK IN 80
READY
>CMD"P"
```

Fig. 8. Disk Driver Screen Dialog

```
INPUT FUNCTION: 0=RD STATUS 1=POSITION HEAD 2=READ SECTOR
3=WRITE SECTOR 4=READ ID DATA 5=READ TRACK 6=WRITE TRACK? 4
DRIVE #? 0
WAIT(W) OR NO WAIT(N)? W
SECTOR #? 0
TRACK #? 5
BUFFER ADDRESS? 40000
COMPLETION= 0
STATUS= 0
00 05 00 09 01 F7 0E
INPUT FUNCTION: 0=RD STATUS 1=POSITION HEAD 2=READ SECTOR
3=WRITE SECTOR 4=READ ID DATA 5=READ TRACK 6=WRITE TRACK?
BREAK IN 80
READY
>CMD"P"
```

Fig. 9. Read ID Function Screen Dialog

DSKDRV. As the Step, Step In and Step Out commands are somewhat redundant with Seek, only Seek (and Restore) are used in DSKDRV. DSKDRV, however, can be used to perform virtually any disk operation performed in the DOSes.

If you are operating with a properly formatted diskette, you will have no problem reading or writing to any sector on the disk. Reading a track is also no problem. You should experiment with the Write track for some time before trying this function on your 2000-name mailing list, however.

Program Listing 2 is designed to give you some experience using DSKDRV. The BASIC program is called Disk Driver Driver and makes it somewhat easier to interface to DSKDRV than Debug or another assembly language program. Disk Driver uses a parameter block area at locations OEFFOH, so be certain to protect memory above that point by answering the memory size question as 61423.

Fig. 8 shows the sample output from Disk Driver Driver. The program first asks for the function to be performed. The code corresponds to the function codes in DSKDRV. Next, the drive number and wait bit status are requested. In this case, the function was simply to read the status, and the program returned the completion type of zero and status of zero.

A sample display for a read ID function is shown in Fig. 9. In this case the sector and track numbers were also requested. Disk Driver Driver always asks for the sector number for a read or write, even though, as in this case, it is not required.

The buffer address can be any address not in use by BASIC, and capable of storing 3000 bytes or so for the read track function. Any read or write operation is followed by completion type and status and the contents of the buffer. The buffer contents printout can be stopped by pressing any key (except shift).

In an earlier figure we saw the appearance of a track directly after formatting. Fig. 10 shows the printout of the same track with data. The first column of the display is the displacement from the start of the buffer in hexadecimal. This value cycles from 00H through FFH and then back to zero again.

Fig. 11 shows the display resulting from a Read sector function. This is the third sector of the directory. While Disk Driver Driver is no Superzap in its sector displays, it will let you look at any sector on the disk. Furthermore, it lets you look at any track to investigate strange formatting or other secrets.

The Strange Case of Lost Data

Now to get back to the glitch I men-

THE ASSEMBLY LINE

Displacement from start of bumper.

```
00 BF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
10 FF FF FF FF FE 05 00 07 01 D4 01 FF FF FF FF FF
20 FF FF FF FF FF FF C0 00 00 00 00 00 FB 88 F5 E6
30 80 C4 7C 55 F1 01 02 00 54 E6 03 C4 A0 55 7C 2F
40 57 7D 2F 5F 13 3A D3 88 4F C9 CD 33 72 EB 13 1B
50 7A 1C 1F 0D 0D 0D 0D 0D 20 2A 2A 20 4C 44 4F 53
60 20 44 49 53 4B 20 42 41 53 49 43 20 52 45 4E 55
70 4D 42 45 52 20 2D 20 56 54 20 32 2E 31 31 20 2A
80 2A 0D 0D 20 20 20 20 20 20 20 4C 69 63 65 6E 73
90 65 64 20 74 6F 20 4C 6F 62 6F 20 44 72 69 76 65
A0 73 20 49 6E 74 27 6C 2E 20 31 39 38 31 0D 20 20
B0 20 20 20 20 20 28 63 29 20 63 6F 70 79 72 69 67
C0 68 74 20 31 39 38 30 20 62 79 20 4C 61 6E 63 65
D0 20 4D 69 63 6B 6C 75 73 2C 20 49 6E 63 2E 0D 20
E0 20 20 20 20 20 20 42 75 72 6C 69 6E 67 74 6F 6E
F0 2C 20 56 74 2E 20 30 35 30 34 31 20 2D 20 41 6C
00 6C 20 72 69 67 68 74 73 20 72 65 73 65 72 76 65
10 64 0D 0D 00 43 68 65 63 6B 69 6E 67 20 66 6F 72
20 20 65 72 72 6F 72 73 0D 00 42 61 64 20 D5 64 FF
30 7F FF FF FF FF FF FF FF FF FF FF FF FF FF 80 00
40 00 00 00 00 FE 05 00 03 01 18 C5 FF FF FF FF FF
50 FF FF FF FF FF FF FF FF FF FF FF FF FB 28 33 37
60 45 43 48 29 09 20 20 3B 47 45 54 20 53 54 41 54
70 55 53 0D B0 B0 B5 B9 B0 20 09 52 52 43 41 09 09
80 09 20 20 3B 54 45 53 54 20 42 55 53 59 0D B0 B0
90 B6 B0 B0 20 09 4A 52 09 43 2C 44 53 4B 30 33 30
A0 09 20 20 3B 47 4F 20 49 46 20 42 55 53 59 0D B0
B0 B0 B6 B1 B0 20 09 4C 44 09 49 59 2C 30 09 09 3B
C0 46 4F 52 20 4D 55 4C 54 49 50 4C 49 43 41 54 49
D0 4F 4E 0D B0 B0 B6 B2 B0 20 09 4C 44 09 43 2C 28
E0 49 58 2B 30 29 09 3B 47 45 54 20 46 55 4E 43 54
F0 49 4F 4E 0D B0 B0 B6 B3 B0 20 09 4C 44 09 42 2C
00 30 09 09 3B 4E 4F 57 20 49 4E 20 42 43 0D B0 B0
10 B6 B4 B0 20 09 41 44 44 09 49 59 2C 42 43 09 09
20 3B 46 55 4E 43 54 49 4F 4E 2A 31 0D B0 B0 B6 B5
30 B0 20 09 41 44 44 09 49 59 2C 42 43 09 09 3B 46
40 55 4F 43 54 49 4F 4E 2A 32 0D B0 B0 B6 B6 B0 20
50 09 41 44 44 09 49 59 2C 42 43 09 09 3B 69 E0 FF
60 FF FF FF FF FF FF FF FF FF FF FF FF F0 00 00 00
70 00 00 0F 05 00 06 01 C4 3F FF FF FF FF FF FF FF
80 FF FF FF FF FF FF FF FF FF FF FB 70 61 72 61 6D
90 65 74 65 72 73 0D 00 01 02 00 55 E5 21 DB 54 CD
A0 6A 58 D1 2A 49 40 F9 EB C3 1C 55 3E 24 C9 21 F0
B0 54 CD 6A 58 C3 B7 57 AF 32 65 58 2B CD 7C 52 7E
C0 FE 21 20 0B 3E 01 32 65 58 CD 7C 52 06 04 DD 21
D0 5B 58 7E FE 0D 28 39 0D 5E 00 DD 56 01 7E FE 2C
E0 28 10 C5 DD E5 CD DF 57 DD E1 C1 DA 13 55 7A B3
F0 28 0A DD 73 00 DD 23 DD 72 00 DD 23 7E FE 0D 28
```

Fig. 10. Track Section with Data

Displacement from start of buffer.

```
00 5D 00 00 00 00 44 49 52 20 20 20 20 20 53 59 53
10 20 C2 96 42 0A 00 11 01 FF FF 00 00 00 00 00 00
20 5F 03 D9 54 00 53 59 53 37 20 20 20 20 53 59 53
30 20 C2 52 C3 22 00 21 22 02 23 FF FF FF FF FF FF
40 80 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00
50 00 00 00 00 00 00 0D 22 FF FF FF FF FF FF FF FF
60 15 03 D9 D3 00 4B 53 4D 20 20 20 20 20 46 4C 54
70 F3 40 96 42 02 00 1A 00 FF FF FF FF FF FF FF FF
80 10 44 B9 00 00 44 53 4B 44 52 56 20 20 4F 42 4A
90 96 42 96 42 02 00 0E 00 FF FF FF FF FF FF FF FF
A0 06 03 D9 5C 00 4C 43 4F 4D 4D 20 20 20 43 4D 44
B0 0D 5F 96 42 07 00 1A 21 FF FF FF FF FF FF FF FF
C0 1E 03 D9 1C 00 43 4D 44 46 49 4C 45 20 43 4D 44
D0 0D 5F 96 42 0C 00 1B 20 1D 01 FF FF FF FF FF FF
E0 05 03 D9 A4 00 52 53 32 33 32 4C 20 20 44 56 52
F0 F3 40 96 42 04 00 1E 00 FF FF FF FF FF FF FF FF
```

Fig. 11. Read Sector Display

tioned at the first part of this column. Using DSKDRV, I consistently got a Lost Data status for the Read track function. The data itself, however, looked valid. Checking around, I happened to talk to Bill Schroeder of Galactic Software. He suggested I talk to one of their people, Tim Mann, the resident 1771 expert. Before I could even describe the problem completely, he said, "Yes, on Read track I noticed that I was consistently getting a Lost Data message. I suspect there is an error in the 1771." Perhaps one of you can define under what conditions the Lost Data message appears. Is there an error in the 1771 logic? Or is it programmer error? If you can ferret out the answer, you'll be mentioned in dispatches in this column.

. . . And Speaking of Programmer Error

I did it again. Another error in a column. (For my only previous error please look at the April, 1923, column of *80 Microcomputing*, where I discussed punched cards.) In the March column, marking should read spacing and vice versa. Thanks to Charles A. King of Techplan Corporation, Falls Church, VA, for this.

That's it until next month. Thanks for all your comments and suggestions on the column. ■

✓See List of Advertisers on page 306

80 APPLICATIONS

by Dennis Kitsz

". . . I knew that time was short for me and my Model I. . . But in parting, I wished to give my trusty Model I a taste of. . . high resolution graphics."

When I first saw the TRS-80 Color Computer, I knew that time was short for me and my Model I. Those flashy graphics, those tiny dots, lines and circles, and all that color won me over. But in parting, I wished to give my trusty Model I a taste of those same high-resolution graphics.

This month's project is the result—a grid of graphic dots as fine as those which make up the letters on the screen: 384 across and 192 down. This month's project is also the most complicated and costly you will likely see in my column. I call it "The Detailer." It demands 30 integrated circuits, many hours of wiring, care in assembly, and about $75 in parts.

After all the work, however, you will have a graphics addition with a resolution higher than any available commercially for the TRS-80, as well as more graphics detail than any other popular home computer. Although no actual grey scale is provided, shades of grey can be simulated by varying the density of the dots used.

Now for the requirements. For greatest ease manipulating the screen, your TRS-80 should have 48K of memory. However, this is not essential. Aside from an edge card connector to attach to The Detailer, you will also have to place two wires directly inside your computer (one to its master clock, and one to its vertical sync line). Finally, the video monitor and the computer's video output will both attach to The Detailer.

Design Thoughts

There were several considerations involved in the design of this board. First, the video signal should relate as closely as possible to that provided by the TRS-80 itself. Next, a relationship should be maintained between the TRS-80 graphics blocks and the higher resolution graphics areas. Finally, both high-resolution drawings and normal text and graphics should be available on the screen simultaneously.

The Detailer is designed so its dot selection and video synchronization are identical to that produced by the TRS-80; this is done simply—by stealing the TRS-80 video circuitry IC by IC. Also, the output signal is of the same amplitude and timing, thereby providing perfect synchronization between the two units. And, finally, it creates its graphics independently of the text and graphics normally within the TRS-80, meaning both may be mixed on a single monitor, or the TRS-80 and The Detailer may be fed to separate video monitors.

This high-resolution addition consists of several major blocks:

1. A video "countdown" chain almost identical to that inside the TRS-80, which provides row and column addresses for the video dots, and a set of horizontal and vertical synchronization and blanking signals for the video monitor.
2. 12,288 bytes of random access memory to store the high-resolution graphics information. Dynamic RAMs (type 4116—the same as those used for normal TRS-80 memory) are used for this.
3. A latch and shift register to capture the graphics dots and shift them out one at a time to the video screen.
4. Write-select circuits to fill the high-res memory from the TRS-80, and read-select circuits for use by The Detailer. Memory refresh (dynamic RAMs are used) is provided automatically by each screen display cycle.

Immediately upon connecting The Detailer, very fine-definition video graphics are available without complicated synchronization and control-twiddling.

Circuit Details

The timing for The Detailer can be achieved in two ways: first, via a self-contained crystal oscillator made up of Z1, sections a/b/c, two resistors, a variable capacitor, and a crystal. The crystal (10.6445 MHz) is available from Radio Shack as a special order item, for $4.95.

The second method is easier and more reliable. The clock timing signal is provided by the TRS-80 itself at Z42, pin 6. By running a wire from this integrated circuit inside the computer case to the high-resolution board, accurate and synchronized timing of the video is possible.

The timing pulse is then fed to Z2. This pulse has three results: It toggles the video shift register, which sends dots to the screen at a 10 MHz rate; it is divided into a video divider chain signal (0.887 MHz) for the column, line and row selection; and it creates the video byte latch signal (1.774 MHz combined with 3.5481 MHz for a short latching pulse).

Z4 through Z7 represent the video divider chain itself. Sixty-four columns, twelve lines and sixteen rows are addressed by these dividers; the dividers also determine the occurrence of horizontal synchronization (at Z5 pin 11) and vertical synchronization (at Z7 pin 11). The simultaneous occurrence of horizontal and vertical sync determines the blanking period (the time during which the video monitor's electron beam is off), provided by Z9a. For details on the operation of the video countdown chain, read the *TRS-80 Technical Reference Handbook*, which contains an excellent description.

High-res memory is selected in much the same way as that in normal video memory. In this case, when address lines 14 and 15 are high, the high-resolution video is selected. The output of Z15b then swings low, switching multiplexers Z10 through Z13 from the video divider chain to the computer address lines.

Simultaneously, the high-res refresh/select lines (found at the input of Z20) are switched to the computer, completing computer selection of high resolution mode. If the computer's write (WR) line is also low at that time, then the high-resolution memory is written into by the TRS-80.

The video byte selected by the high-resolution board is latched into Z16, and shifted out a bit at a time through Z17. This shifted dot pattern moves through Z15d to Z30, where its voltage level is adjusted to be compatible with normal video output. The horizontal and vertical sync signals, meanwhile, are fed through Z27, Z28, and Z29, and subsequently mixed to provide a composite video output.

This output is then mixed with its synchronous partner, the video ouput of the TRS-80. The two are synchronized by Z31, which, when power is applied to The Detailer, switches high. The vertical sync signal from Z66 pin 12 on the TRS-80 clears this circuit, allowing the clock pulses to begin counting through the video divider chain. If for some reason the two boards get out of sync, momentarily depressing S1 will again trigger Z31, restoring synchronization.

Construction Hints

The Detailer is not an easy project to tackle. It requires considerable patience, and you must follow a few important guidelines:

1. Use the power supply as shown. It's crucial that the −5 volt power line be stable first and last, so do not attempt to change the arrangement of parts.
2. Bypass capacitors on all integrated circuits are vital. Use 0.1 microfarad capacitors between +5 volts and ground at each IC, and use good quality glass or tantalum capacitors between +5 and ground, +12 and ground, and −5 and ground on the memory chips.
3. Handle the memory chips with care, and don't put them in their sockets until the circuit is complete. Also, apply power and test the voltages *before* putting the memory chips in place. Then remove power and insert the chips.
4. Wire-wrap carefully, and keep connections short and clean around the memory and multiplex areas, particularly where the CRAS, CCAS, CMUX, MRAS, MCAS, and MMUX lines are found.
5. Put a heavy heat sink on the 7805 voltage regulator, or use a separate five-volt regulator for the memory chips. Without a heat sink, the power rating of the 7805 can be exceeded and the board will show many memory errors.
6. The board cannot be successfully accessed at high speed. If you have a high speed modification in your computer, make sure it returns to normal speed when writing to the high-res memory.
7. Three hundred nanosecond 4116 dynamic RAMs will give the best results. The memory select circuitry during screen access is very fast, and can outrun some of the old standard 450 nS memories.
8. The line marked "HIRES*" on the schematic is the most time-sensitive, and likely to cause memory dropouts. Be sure to use the pull-up and pull-down resistors on this line.
9. Occasionally, random dots will turn on in a wire-wrapped version of this circuit. Since a copy of this memory is stored in your TRS-80's RAM, a

Table 1. POKE Codes for High-Resolution Graphics

DOTS	IN BINARY	HEX	OCTAL	DECIMAL
O O O O O O	Ø Ø Ø Ø Ø Ø	ØØ	ØØ	Ø
O O O O O *	Ø Ø Ø Ø Ø 1	Ø1	Ø1	1
O O O O * O	Ø Ø Ø Ø 1 Ø	Ø2	Ø2	2
O O O O * *	Ø Ø Ø Ø 1 1	Ø3	Ø3	3
O O O * O O	Ø Ø Ø 1 Ø Ø	Ø4	Ø4	4
O O O * O *	Ø Ø Ø 1 Ø 1	Ø5	Ø5	5
O O O * * O	Ø Ø Ø 1 1 Ø	Ø6	Ø6	6
O O O * * *	Ø Ø Ø 1 1 1	Ø7	Ø7	7
O O * O O O	Ø Ø 1 Ø Ø Ø	Ø8	1Ø	8
O O * O O *	Ø Ø 1 Ø Ø 1	Ø9	11	9
O O * O * O	Ø Ø 1 Ø 1 Ø	ØA	12	1Ø
O O * O * *	Ø Ø 1 Ø 1 1	ØB	13	11
O O * * O O	Ø Ø 1 1 Ø Ø	ØC	14	12
O O * * O *	Ø Ø 1 1 Ø 1	ØD	15	13
O O * * * O	Ø Ø 1 1 1 Ø	ØE	16	14
O O * * * *	Ø Ø 1 1 1 1	ØF	17	15
O * O O O O	Ø 1 Ø Ø Ø Ø	1Ø	2Ø	16
O * O O O *	Ø 1 Ø Ø Ø 1	11	21	17
O * O O * O	Ø 1 Ø Ø 1 Ø	12	22	18
O * O O * *	Ø 1 Ø Ø 1 1	13	23	19
O * O * O O	Ø 1 Ø 1 Ø Ø	14	24	2Ø
O * O * O *	Ø 1 Ø 1 Ø 1	15	25	21
O * O * * O	Ø 1 Ø 1 1 Ø	16	26	22
O * O * * *	Ø 1 Ø 1 1 1	17	27	23
O * * O O O	Ø 1 1 Ø Ø Ø	18	3Ø	24
O * * O O *	Ø 1 1 Ø Ø 1	19	31	25
O * * O * O	Ø 1 1 Ø 1 Ø	1A	32	26
O * * O * *	Ø 1 1 Ø 1 1	1B	33	27
O * * * O O	Ø 1 1 1 Ø Ø	1C	34	28
O * * * O *	Ø 1 1 1 Ø 1	1D	35	29
O * * * * O	Ø 1 1 1 1 Ø	1E	36	3Ø
O * * * * *	Ø 1 1 1 1 1	1F	37	31
* O O O O O	1 Ø Ø Ø Ø Ø	2Ø	4Ø	32

Note: Lighted dots are marked with a star (*); blank "undots" are marked with an o (O).

Table continues

$ DISCOUNT $

TRS-80®

DEALER A301

COMPUTER SPECIALISTS

CALL US. . .

SAVE MONEY

MODEL II

26-4002
64K 1 Drive
$3440.00

26-4160 1 Drive Exp.	$1034.00
26-4161 2 Drive Exp.	1574.00
26-4162 3 Drive Exp.	2114.00
26-4530 Scripsit II	265.00
26-4512 Profile II	162.00
26-4511 Visicalc II	265.00
26-4501 Gen Ledger	180.00
26-4506 Mail List	72.00

We carry the full line of TRS-80 Computers. All other R.S. software, furniture, and accessories at discount from catalog price. We stock most items to assure you fast delivery and save you money.

26-1140 Expansion Interface	$249.00
26-1141 16K Exp. Interface	359.00
26-1142 32K Exp. INterface	469.00
26-1145 RS232C Board	84.00
26-1160/1 Mini Disk Drive	419.00
26-1563 Scripsit-Disk	79.00
26-1566 Visicalc	83.00

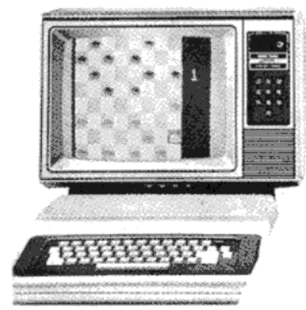
* * * * * *PRINTERS* * * * * *

26-1155 Quick Printer	187.00
26-1167 9½ Dot Matrix Printer	360.00
26-1166 Line Printer VI	1080.00
26-1158 Daisy Wheel II	1799.00
26-1165 Line Printer V	1710.00

MODEL III

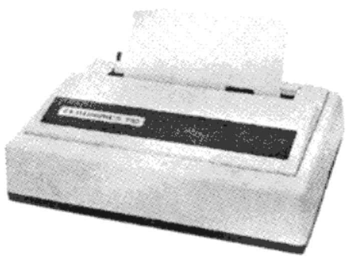
26-1061 4K I	$629.00
26-1062 16K III	865.00
26-1063 32K III W/2 Drives, RS232	2225.00

COLOR



26-3001 4K	$353.00
26-3002 16K Ext. Basic	533.00
26-3008 Joysticks	22.50
26-3010 Color Video	353.00
26-1206 Recorder	54.00

PRINTERS

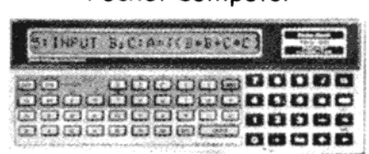


CENTRONICS

BEST PRICES

Fast 100 CPS Centronics 730-1A Printer	$577.00
Text Quality Centronics 737-1 Printer	$737.00

Pocket Computer



26-3501 1.9K P.C.	$221.00
26-3503 Cassette IF	45.00
14-812 Recorder	72.00

commodore

AUTHORIZED DEALER
CALL FOR PRICES

ALL POCKET AND COLOR COMPUTER SOFTWARE SOLD AT DISCOUNT

WRITE US FOR A
FREE CATALOG

1-800-841-0860 Toll Free Order Entry

MICRO MANAGEMENT SYSTEMS, INC. ✓72

No Taxes on Out Of State Shipments

Immediate Shipment From Stock on Most Items

DOWNTOWN PLAZA SHOPPING CENTER
115 C. SECOND AVE. S.W.
CAIRO, GEORGIA 31728
(912) 377-7120 Ga. Phone No. & Export

TRS-80 is a registered trademark of the Tandy Corp.

R.S. 90 Day Limited Warranty
F-48 Form Provided

Largest Inventory
In The S.E. U.S.A.

subroutine to write this copy back into high-resolution RAM can be used:

```
        LD    HL,0C000H
LOOP    LD    A,(HL)
        LD    (HL),A
        INC   HL
        LD    A,H
        OR    L
        JR    NZ,LOOP
        RET
```

How It Works and How to Use It

Look carefully at the dots that make up the letters on your TRS-80 screen. Set a few graphics characters, and turn the contrast and brightness down so the individual dots are clearly defined. You will see that each graphic Set point is really made up of a block of dots, three dots across and four dots down. Now, print a CHR$ (191), and notice that this full block (made up of six of the Set points) is six dots across and twelve dots down.

The Detailer is set up to occupy the top 16K of memory space in the TRS-80. Each byte of high-resolution memory affects six screen dots, in the same way that each byte of normal screen memory affects an area six dots across. The difference is that The Detailer only creates a character one dot deep. That means it takes twelve bytes of memory to create a graphics block six dots across and twelve dots down (like CHR$(191)).

High-resolution graphics act somewhat like a binary window: Each bit of memory lights up one bit on the screen, like this * * * * * *. The right-most bit is bit 0, and the leftmost is bit 5. Bits 6 and 7 are not needed, since each graphics character is only six dots wide. Table 1 shows how a single memory cell would affect the dots on one line of the screen, counting in binary, hexadecimal, octal and decimal.

Chances are you'll never see me use octal numbering in this column again. But it just so happens that these graphics lines are six dots across. Whereas hexadecimal breaks groups of binary digits into fours (0000 0000 0000), octal breaks them up into threes (000 000 000 000), and is mainly a heritage of 12-bit minicomputers. (Note to you machine language programmers—did you know the architecture of the Z-80 is really octal? Check it out.) Octal can be a convenient way of visualizing the six-dot-wide "bytes". If hex or octal doesn't interest you, then decimal will work just fine; cut out or copy Table 1, and refer to the decimal values when drawing lines.

* O O O O *	1 Ø Ø Ø Ø 1	21	41	33
* O O O * O	1 Ø Ø Ø 1 Ø	22	42	34
* O O O * *	1 Ø Ø Ø 1 1	23	43	35
* O O * O O	1 Ø Ø 1 Ø Ø	24	44	36
* O O * O *	1 Ø Ø 1 Ø 1	25	45	37
* O O * * O	1 Ø Ø 1 1 Ø	26	46	38
* O O * * *	1 Ø Ø 1 1 1	27	47	39
* O * O O O	1 Ø 1 Ø Ø Ø	28	5Ø	4Ø
* O * O O *	1 Ø 1 Ø Ø 1	29	51	41
* O * O * O	1 Ø 1 Ø 1 Ø	2A	52	42
* O * O * *	1 Ø 1 Ø 1 1	2B	53	43
* O * * O O	1 Ø 1 1 Ø Ø	2C	54	44
* O * * O *	1 Ø 1 1 Ø 1	2D	55	45
* O * * * O	1 Ø 1 1 1 Ø	2E	56	46
* O * * * *	1 Ø 1 1 1 1	2F	57	47
* * O O O O	1 1 Ø Ø Ø Ø	3Ø	6Ø	48
* * O O O *	1 1 Ø Ø Ø 1	31	61	49
* * O O * O	1 1 Ø Ø 1 Ø	32	62	5Ø
* * O O * *	1 1 Ø Ø 1 1	33	63	51
* * O * O O	1 1 Ø 1 Ø Ø	34	64	52
* * O * O *	1 1 Ø 1 Ø 1	35	65	53
* * O * * O	1 1 Ø 1 1 Ø	36	66	54
* * O * * *	1 1 Ø 1 1 1	37	67	55
* * * O O O	1 1 1 Ø Ø Ø	38	7Ø	56
* * * O O *	1 1 1 Ø Ø 1	39	71	57
* * * O * O	1 1 1 Ø 1 Ø	3A	72	58
* * * O * *	1 1 1 Ø 1 1	3B	73	59
* * * * O O	1 1 1 1 Ø Ø	3C	74	6Ø
* * * * O *	1 1 1 1 Ø 1	3D	75	61
* * * * * O	1 1 1 1 1 Ø	3E	76	62
* * * * * *	1 1 1 1 1 1	3F	77	63

Before drawing lines, it's time to hook up The Detailer. Follow these steps:

- Attach the edge card connector to the TRS-80.
- Attach the video monitor cable to The Detailer.
- Attach a cable from The Detailer to the TRS-80.
- Attach the two wires from inside the TRS-80 to The Detailer, as shown in the schematic.
- Turn on the TRS-80.
- Turn on The Detailer.
- Set the memory size to 49152.

Clearing the Screen and Drawing a Line

The screen will present the memory size query, as usual. If you do not see it, adjust

```
10 CLS : A$ = "12345678901234567890"
20 X = VARPTR(A$) : Y = PEEK(X+1) + 256*PEEK(X+2)
30 Z = Y : FOR N = 1 TO 15 : READ A : POKE X,A : NEXT
40 DATA 175,245,33,0,192,241,119,35
50 DATA 245,124,181,32,248,241,201
60 POKE 16526,PEEK(X+1) : POKE 16527,PEEK(X+2)
70 REM * FOR DISC SYSTEMS USE DEFUSR0=Z
80 M = USR(0) : REM * FOR DISC USE M = USR0(0)
```

Program Listing 1

DOTS	IN BINARY	HEX	OCTAL	DECIMAL
0 0 0 0 0 *	0 0 0 0 0 1	01	01	1
0 0 0 0 * 0	0 0 0 0 1 0	02	02	2
0 0 0 * 0 0	0 0 0 1 0 0	04	04	4
0 0 * 0 0 0	0 0 1 0 0 0	08	10	8
0 * 0 0 0 0	0 1 0 0 0 0	10	20	16
* 0 0 0 0 0	1 0 0 0 0 0	20	40	32

Table 2. Individual Screen Dots

the balance control until it appears. If there is any tearing, twiddle the V-Sync and H-Sync controls until it stabilizes. What you will probably see is a screen filled with garbage dots as well as the memory size question. First, clear the high-resolution screen using Program Listing 1.

Drawing horizontal lines is easy. A solid line is made up of continuous "on" dots, like this: ******/******/******/******/******/******. To draw a horizontal line across the top of the screen, we need to know where the high-resolution memory is. It runs from C000 hex (49152 decimal) to FFFF hex (65535 decimal). But wait—Level II doesn't like integers over 32767, so here's the rule: If an integer number X is greater than 32767, then X − 65536 is the way Level II needs to see it. So, The Detailer's memory runs (in Level II talk) from − 16384 to − 1.

That makes the first graphics line 64 places long, from − 16384 to − 16321. Enter these commands: FOR X = − 16384 TO − 16321:POKE X,63:NEXT. To understand the command to POKE X with 63, refer to Table 1. Sixty-three is the decimal value to set all dots on. How about a dashed line? Try this: FOR X = − 16384 TO − 16321:POKE X,56:NEXT. Or, as a final example, a dotted line: FOR X = − 16384 TO − 16321:POKE X,42:NEXT. By trying different POKE values, the density and character of the horizontal line changes.

Vertical Lines—A Different Story

Vertical lines are a different story, because each vertical dot is 64 memory locations away from the one above and below it. This draws a vertical line from top to bottom in the center of the screen: FOR X = − 16352 TO − 32 STEP 64:POKE X,1: NEXT. There's a trick to doing single vertical lines. If in this example the POKEd value were 16 instead of 1, the line would move to the left (try it). There are 384 dots across each line of the screen, but only 64 memory locations to hold them—six dots each. Table 2 is an excerpt of Table 1, showing only those POKE values with individual dots, from which thin vertical lines can be built.

Now, I know that's not immediately comprehensible; it was hardly clear to me when I built it! Remember that horizontal lines were made up of contiguous groups of six bits, each group in a single memory location. Vertical lines don't have any contiguous bits. Every bit belongs to a *different* memory location, 64 memory locations apart. Look at this:

```
0000*0
0000*0
0000*0
0000*0
0000*0
0000*0
0000*0
```

There's a vertical line, drawn with a statement something like: FOR X = − 16352 TO 0 STEP 64:POKE X,2:NEXT.

Do you see how it carries with it a burden of five other bits representing unlit dots? Now the next question: Let's say we already have a vertical line like the one above. How is a line drawn right next to it? This program won't do it: FOR X = − 16352 TO 0 STEP 64:POKE X,4:NEXT, because it will draw a new line in the right place, but POKEing 4 (dots 000*00) will *erase the line already there*. If you know what line is already in place, you can draw it by POKEing X with 5 (dots 000***0), which will create the new line and redraw the second.

There's a better way, but only if you have 48K memory in your TRS-80. Examine these commands: FOR X = − 16352 TO 0 STEP 64:POKE X,(4 OR PEEK(X)): NEXT.

What? Time for a quick review of the logical OR function. Logical OR says: Given a pair of items, if either the first *or* the second is true, the result will be true. In this case, it can be rephrased: Given a pair of graphics bits, if either the first *or* the second is on, the result will be on. Here is how it looks:

```
Original group of bits (single lines):  0000*0
New group of bits (added line):         000*00
OR function:                            ------
Resulting group of bits (new lines):    000**0
```

To draw another vertical line a few spaces to the left, we would OR the new line with those present in the very same way:

```
Two lines now present:                  000**0
New group of bits (added line):         *00000
OR function:                            ------
Resulting group of bits (new lines):    *00**0
```

The Disappearing Line

Now how about erasing a line? For horizontal lines, the process is pretty simple ... POKE in zeroes, like this: FOR X = − 16384 TO − 16321:POKE X,0:NEXT. That should make the line on the top of the

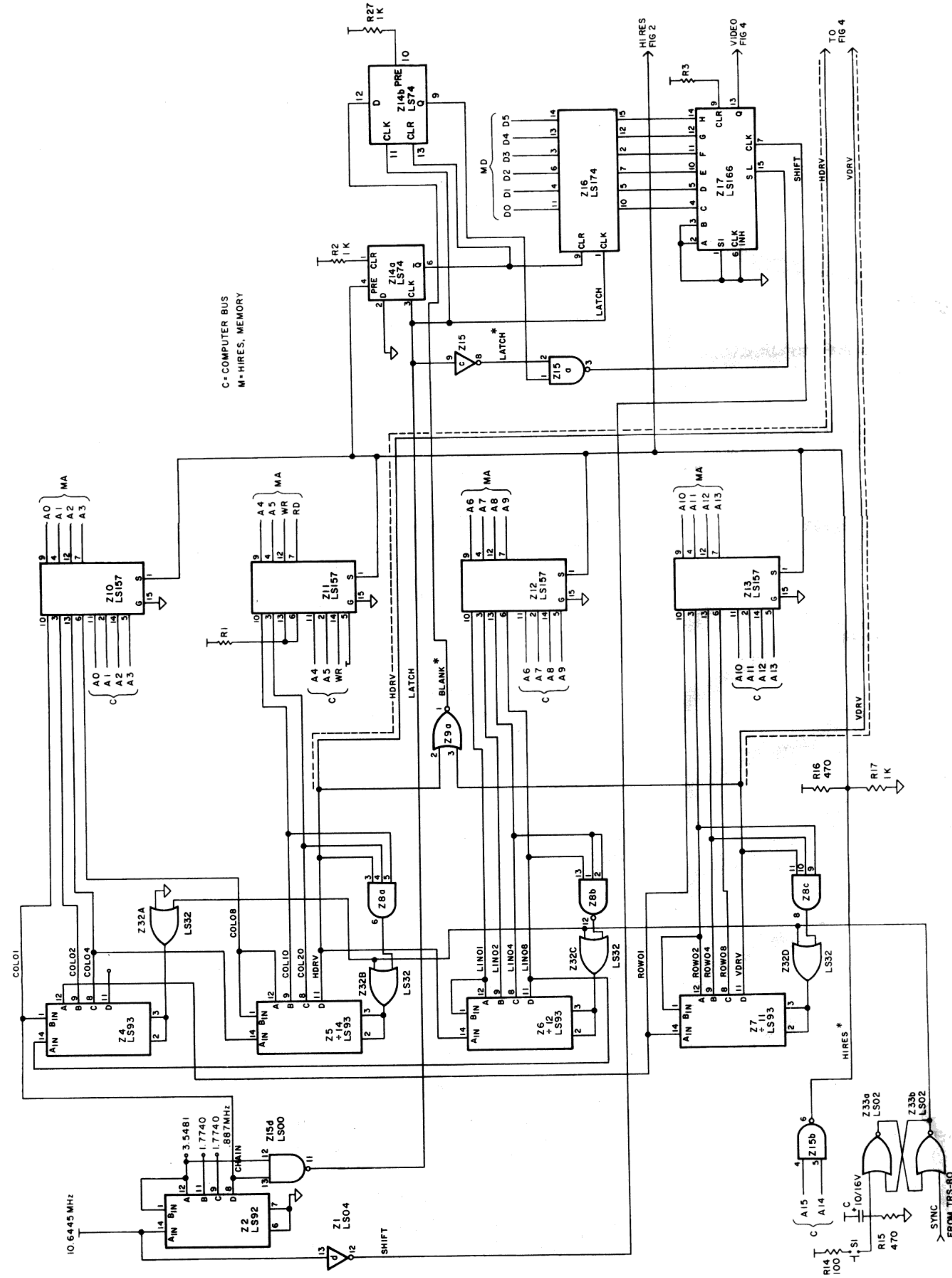


Fig. 1. Video divider chain of the hi-res graphics board uses circuitry almost identical to that in the TRS-80.

```
AF              00100                 XOR     A
F5              00110                 PUSH    AF
21 00 C0        00120                 LD      HL,0C000H
F1              00130     LOOP        POP     AF
77              00140                 LD      (HL),A
23              00150                 INC     HL
F5              00160                 PUSH    AF
7C              00170                 LD      A,H
B5              00180                 OR      L
20 F8           00190                 JR      NZ,LOOP
F1              00200                 POP     AF
C9              00210                 RET
```

Program Listing 2

Program Listing 3. Drawing Random Horizontal and Vertical Lines

```
10 CLS : REM * SAVE THIS PROGRAM BEFORE RUNNING IT (LINE 30!)
20 REM * HI-RES CLEAR SCREEN ROUTINE FOLLOW IN DUMMY STRING
30 A$="12345678901234567890" : REM * SET UP DUMMY M/L STRING
40 X=VARPTR(A$) : REM * DISCOVER INFO. ABOUT A$ LOCATION
50 Y=PEEK(X+1) + 256*PEEK(X+2) : REM * A$ MEMORY LOCATION
60 Z = Y : REM * SET UP VARIABLE FOR USE IN USR ENTRY POINT
70 FOR N = 1 TO 15 : REM * 15 DATA ELEMENT READ/POKE LOOP
80 READ A : POKE Y,A : Y = Y + 1 : NEXT : REM * POKE M/L INFO
90 DATA175,245,33,0,192,241,119,35,245,124,181,32,248,241,201
100 DEFUSR0=Z : REM * USE THIS FOR DISK SYSTEM - LII BELOW:
110 REM * POKE 16526,PEEK(X+1) : POKE 16527,PEEK(X+2)
120 M=USR0(0) : REM * USE M=USR(0) FOR LEVEL II SYSTEMS
130 FOR Q = 1 TO 100 : REM * READY TO DRAW 100 RANDOM LINES
140 Z = -16384 : REM * SEE TEXT FOR DESCRIPTION OF THIS VALUE
150 Y = 64 : REM * THIS OFFSET DEFINES VERTICAL POSITIONING
160 A = RND(Y)-1 : REM * CHOOSE ANY OLD HORIZONTAL BYTE
170 AA = (RND(256)-1)*64 : REM * CHOOSE A VERTICAL AREA
180 B = RND(Y)+A : IF B> 63 THEN B = 63 : REM * HOR. GUIDE
190 C = (RND(256)-1)*64 : REM * CHOOSE ANY OLD VERTICAL BYTE
200 CC = RND(63) : REM * CHOOSE A HORIZONTAL POSITION HERE
210 E = RND(255)+D : IF E > 255 THEN E = 255 : REM * V. GUIDE
```

Program continues

```
220 E = E * 64 : REM * DEFINE WHICH VERTICAL LINE TO BE USED
230 F = RND(7)  : REM * SELECT A RANDOM VERTICAL BIT POSITION
240 IF F = 3 THEN F = 4 ELSE IF F = 5 THEN F = 8 ELSE
    IF F = 6 THEN F = 16 ELSE IF F = 7 THEN F = 32 :
    REM * CONVERSION OF RND(7) TO A VERTICAL BIT (TABLE 1)
250 REM * ACTUAL DRAWING OF RANDOM LINES BEGINS BELOW:
260 FOR X = A+AA+Z TO B+AA+Z : REM * GET HORIZONTAL POSITION
270 POKE X,63 : REM * 63 = 111111 FOR FULL LINE (SEE TEXT)
280 NEXT : REM * DRAWING HORIZ. LINE IS FASTER THAN VERT.
290 FOR X = C+CC+Z TO E+CC+Z STEP 64 : REM * GET VERTICAL
300 POKE X,(F OR PEEK(X)) : REM * SEE TEXT ABOUT OR FUNCTION
310 NEXT : REM * NO LINES ARE ERASED IN THIS DEMO PROGRAM
320 NEXT Q : REM : COMPLETE DRAWING OF 100 HI-RES LINES
330 GOTO 120 : REM * AND REPEAT HI-RES CLS AND DO IT AGAIN
```

```
10 FOR X = 20480 TO 20505
20 READ A : POKE X,A : NEXT
30 POKE 16526,0 : POKE 16527,0
40 M = USR (0)
50 DATA 175,245,33,0,192,241
60 DATA 119,35,245,124,181,194
70 DATA 5,80,1,0,128,205,96
80 DATA 0,241,60,245,195,2,80
```

Program Listing 4. Vertical Line Drawing Program—BASIC Listing

Program Listing 5. Vertical Line Drawing Program—Assembly Listing

```
                  ORG     5000H
AF                XOR     A
F5                PUSH    AF
21 00 C0          LD      HL,0C000H
F1         LOOP   POP     AF
77                LD      (HL),A
23                INC     HL
F5                PUSH    AF
```

Program continues

See List of Advertisers on page 306

```
7C              00170      LD      A,H
B5              00180      OR      L
C2 05 50        00190      JP      NZ,5005H
01 00 80        00200      LD      BC,8000H
CD 60 00        00210      CALL    0060H
F1              00220      POP     AF
3C              00230      INC     A
F5              00240      PUSH    AF
C3 02 50        00250      JP      5002H
```

screen disappear, because all the high-resolution graphics dots on that line are turned off.

But what about the vertical line? Again, it gets just a bit tacky. If we POKE a zero, we will erase any of the six possible vertical lines in that group; here's the status of the three vertical lines drawn above:

```
*00**0
*00**0
*00**0
*00**0
*00**0
*00**0
*00**0
```

. . . and so on. Another logic function will be called up for this task: logical AND, along with the concept of "masking". First, there's the logical AND function, which states: Given a pair of items, only if both the first item *and* the second item are true, will the result be true. Converted to the high-resolution graphics model, it reads: Given a pair of graphics bits, if both the first *and* the second graphics bit are on, then the result will be on. As an example, the vertical group we currently have is ANDed with a group which is completely turned on:

Original group of bits (three lines):	*00**0
New group of bits (six lines):	******
AND function:	......
Result of original AND new group:	*00**0

Essentially, nothing has changed, because wherever a bit was turned on in the original, it is also turned on in the second group. But here's the problem: Let's say the fourth dot of this group is to be turned off. In other words, *00**0 is to be changed to *000*0.

To solve this, consider how a photographer or painter obtains a properly balanced and bordered picture or photo. A photographer will place a cardboard frame around a photo to evaluate how it looks, covering up uninteresting or obtrusive areas. A painter will tape over areas which are not to be painted in order to create a sharp border. The first uses a cardboard mask; the second uses masking

Parts List

Z1	74LS04	R2	1K
Z2	74LS92	R3	1K
Z3	74LS74	R4	910 ohms
Z4	74LS93	R5	910 ohms
Z5	74LS93	R6	1.8K
Z6	74LS93	R7	47 ohms
Z7	74LS93	R8	270 ohms
Z8	74LS11	R9	120 ohms
Z9	74LS02	R10	330 ohms
Z10	74LS157	R11	75 ohms
Z11	74LS157	R12	10K
Z12	74LS157	R13	10K
Z13	74LS157	R14	100 ohms
Z14	74LS74	R15	470 ohms
Z15	74LS00	R16	470 ohms
Z16	74LS174	R17	1K
Z17	74LS166	R18	33 ohms
Z18	74LS157	R19	33 ohms
Z19	74LS157	R20	33 ohms
Z20	74LS157	R21	33 ohms
Z21	up4116	R22	33 ohms
Z22	up4116	R23	33 ohms
Z23	up4116	R24	33 ohms
Z24	up4116	R25	33 ohms
Z25	up4116	R26	33 ohms
Z26	up4116	R27	1K
Z27	74C04	C1	47 pf
Z28	74C04	C2	.01 mf
Z29	74C00	C3	10 mf/16v
Z30	75452	C4	330 pf
Z31	74LS00	C5	.046 mf
Z32	74LS32	Q1	2N3904
Z33	74LS02	Q2	2N3904
R1	1K	X1	10.6445 MHz crystal

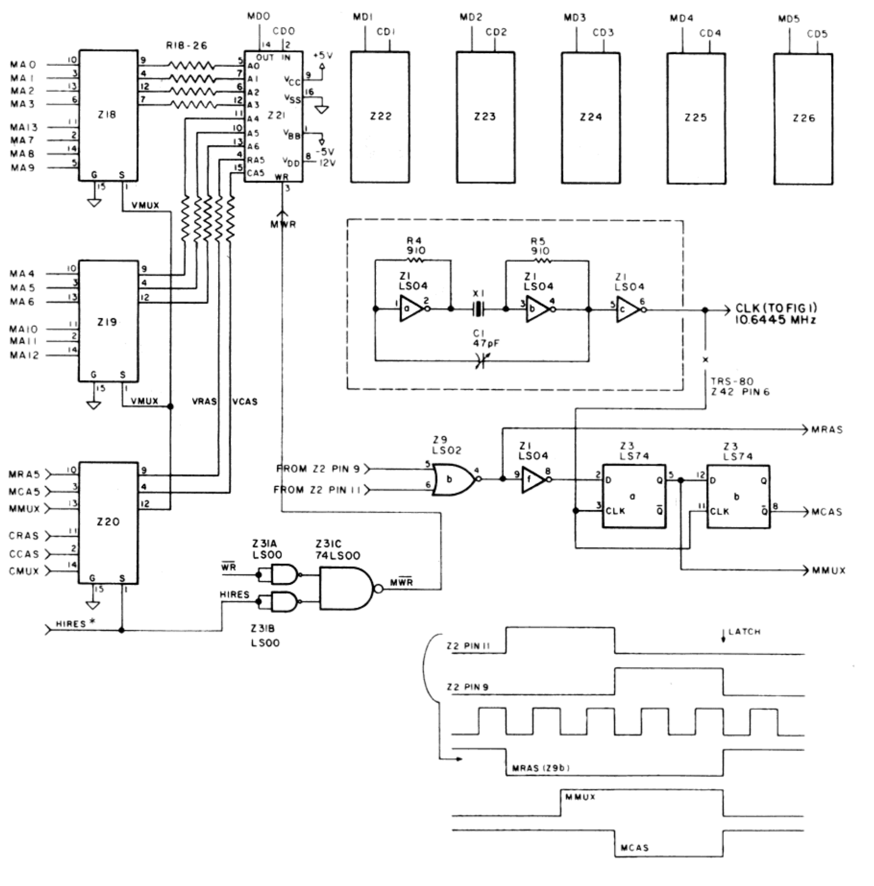


Fig. 2. Memory select-refresh section of the hi-res board. Note that all six memory circuits (Z21 through Z26) are connected in parallel.

✓See List of Advertisers on page 306

```
CØØØ (-16384)...Line #1, 64 bytes (384 dots)...(-16321) CØ3F
CØ4Ø (-1632Ø)...Line #2, 64 bytes (384 dots)...(-16257) CØ7F
CØ8Ø (-16256)...Line #3, 64 bytes (384 dots)...(-16193) CØBF
CØCØ (-16192)...Line #4, 64 bytes (384 dots)...(-16129) CØFF
C1ØØ (-16128)...Line #5, 64 bytes (384 dots)...(-16Ø65) C13F
C14Ø (-16Ø64)...Line #6, 64 bytes (384 dots)...(-16ØØ1) C17F
C18Ø (-16ØØØ)...Line #7, 64 bytes (384 dots)...(-15937) C1BF
C1CØ (-15936)...Line #8, 64 bytes (384 dots)...(-15873) C1FF
C2ØØ (-15872)...Line #9, 64 bytes (384 dots)...(-158Ø9) C23F
C24Ø (-158Ø8)...Line #1Ø, 64 bytes (384 dots)..(-15745) C27F
C28Ø (-15744)...Line #11, 64 bytes (384 dots)..(-15681) C2BF
C2CØ (-1568Ø)...Line #12, 64 bytes (384 dots)..(-15617) C2FF
- - - - - Unused Memory Area Between Line #12 & 13 - - - - -
C4ØØ (-1536Ø)...Line #13, 64 bytes (384 dots)..(-15297) C43F
   .     .        .        .        .        .     .
   .     .        .        .        .        .     .
   .     .        .        .        .        .     .
C6CØ (-14656)...Line #24, 64 bytes (384 dots)..(-14593) C6FF
- - - - - Unused Memory Area Between Line #24 & 25 - - - - -
C8ØØ (-14336)...Line #25, 64 bytes (384 dots)..(-14273) C83F
   .     .        .        .        .        .     .
CACØ (-13632)...Line #36, 64 bytes (384 dots)..(-13569) CAFF
- - - - - Unused Memory Area Between Line #36 & 37 - - - - -
- - - - - - - - Blocks of Memory Continue - - - - - - - - -
FECØ (-32Ø)....Line #192, 64 bytes (384 dots)....(-257) FEFF
```

Fig. 3.

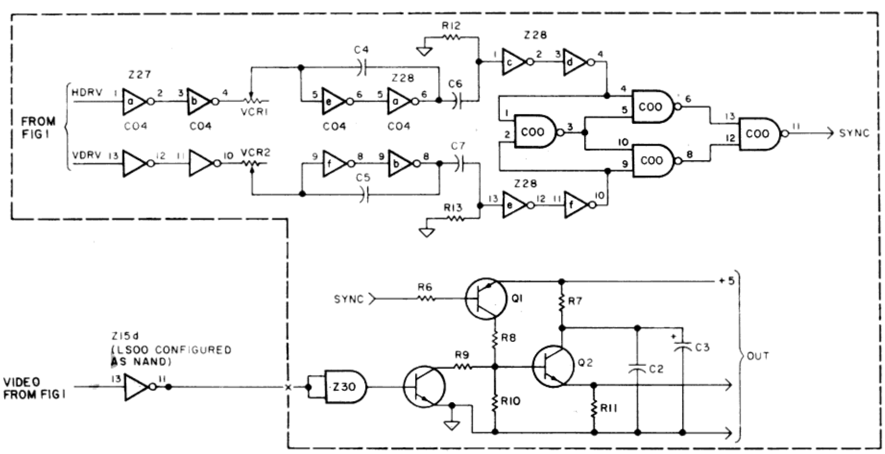


Fig. 4. Horizontal and vertical synchronization circuits and video output circuit. Like the video divider chain, these circuits are almost identical to those used by the TRS-80.

tape. The principle is the same with graphics dots.

A mask of "on" dots is placed over the dots we want to keep, and left off the areas we want to turn off. The AND function is used, like this:

Original group of dots:	*00**0
Mask of on and off dots:	***0**
AND function:	------
Result after masking is done:	*000*0

Using Table 1, you can find that the original group of dots is 38 decimal, and the mask is 59 decimal, and the result is 34 decimal. The BASIC line to accomplish this example would be: FOR X = – 16352 TO 0 STEP 64:POKE X,(59 AND PEEK(X)): NEXT.

With all this in mind, try Program Listing 3, which is a complete program to draw 100 random horizontal and vertical lines using The Detailer.

As a final example, Program Listings 4 and 5 create a stream of vertical lines, drawn using values from 0 to 63, and repeating. You will see fine lines, simulated white and grey areas, and broad bands of the kind you could create with Set/Reset graphics. (Note: You must press Reset to exit from this program).

Monkeys in the Works

Back a few dozen paragraphs, you might recall a mention of 12 horizontal lines per group, for a total of 12,288 bytes. With some fast sleight-of-word, I hoped none of you would have done any quick calculations ahead of time. Because I chose to duplicate Radio Shack's video addressing scheme, the graphics units are six bits across. The two remaining bits are ignored, and don't appear in the circuitry—only six 4116's are used.

However (I begin to sweat here), the 4116 memories are *16K* memories, meaning there are 16,384 bits available. Only 12,288 are used. Where are the rest? Well . . . uh . . . they're . . . how do I say this . . . *invisible*. They are the unaddressed locations between the twelfth line and the sixteenth line.

Instead of one of my hopelessly confusing verbal explanations, look instead at Fig. 3. You'll see that locations – 16384 to – 16321 (hex C000 to C03F) are line number 1; – 16320 to – 16257 (hex C040 to C07F) are line number 2; and continuously down to – 15617 (C2FF hex), which is line number 12. At that point, four lines are unaddressed, which means that – 15616 to – 15361 (C300 to C3FF hex) are unused memory locations. POKE something between – 15616 and – 15361 and nothing appears on the screen. Oh, yes, it does go

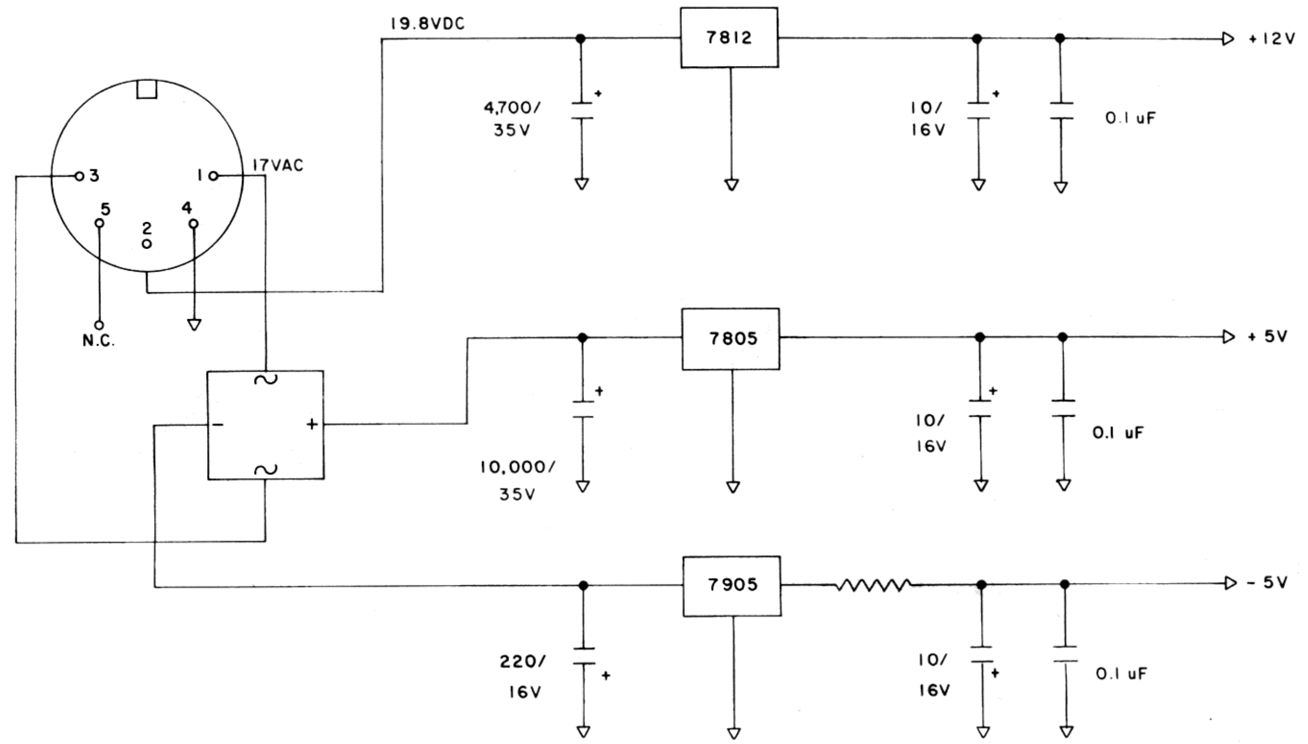


Fig. 5. Power supply for the hi-res board has +12, +5 and −5 volt outputs. This design is critical—see text.

into memory, but that memory isn't displayed (oh, no!).

As long as you're drawing horizontal and vertical lines, the invisible memory can be ignored. Just remember to consider the length of vertical memory when trying to draw such things as truly square boxes. When doing three-dimensional simulations, circles or ellipses, and other drawings where proportion and scale are important, you must take the unused memory blocks into consideration.

High-resolution memory of the kind available from The Detailer can be a pleasure; animations won't come alive as fast, but the clarity and shading made possible can add a new dimension to your home computer use. I would be happy to publish any fast machine language programs created for this board.

Updates

Have you tried to pick up one of the new lowercase chips from Radio Shack? The ones which sold for $12.93? Well, forget it for awhile. The latest price I paid (in May) was $37.50. Anyone for letters to Tandy's president about this curiously steep price increase?

Model III users, please note: Most of the machine language software you see in *80 Microcomputing* that uses cassette input/output will not work unmodified. You can try writing to the authors, but don't expect miracles. Many authors like myself have opted for the Color Computer instead of the Mod III.

A printed circuit board is now available for the Micro Front Panel (May "Applications"); write to me at Roxbury, Vermont 05669 for information. Note that Radio Shack no longer stocks the 74LS373 parts, but they can be obtained from other major suppliers. ■

"Potential problems from non-ionizing microwave radiation have created the most controversy. . . ."

Workers, Unions Concerned Over Video Display Health Issues

Increasing concern among office workers in the U.S. and Canada over the possible harmful effects of video display terminals is growing, as evidenced by several union actions this past spring.

In Toronto, the Communications Workers of Canada (CWC) convinced Bell of Canada that pregnant workers should be allowed to move the non-VDT tasks or take early leaves of absence. The decision followed a work boycott by four pregnant workers who feared that radiation from the terminals could harm their unborn children.

"Many (micros) do not offer detachable keyboards; the screens generally are not adjustable; the nonglare features . . . inadequate. . . ."

In San Francisco, employees at Blue Shield of California ended a 19-week strike that was called partly over what the Office and Professional Employees Union (OPEU) felt were poor working conditions at VDT stations. Concessions by Blue Shield included foot rests, better lighting, nonglare shields for terminals and adjustable chairs.

Potential problems from non-ionizing microwave radiation have created the most controversy, despite tests by the National Institute for Occupational Safety and Health (NIOSH) and the Food and Drug Administration's Bureau of Radiological Health that showed radiation levels below the U.S. standard of 10 milliwatts per square centimeter (10 mW/cm²). The CWC action came closely on the heels of a situation at the *Toronto Star* in which four women working on VDTs gave birth to children with birth defects. Further, the Newspaper Guild, which represents 32,000 newspaper employees in the U.S. and Canada, has seen at least a dozen members develop cataracts, which some doctors feel were caused by the non-ionizing radiation.

"The *Toronto Star* situation remains unexplained," says CWC Health and Safety Officer Gary Cwitco. "While many officials and any number of scientists have said that the terminals were not responsible, they can't tell us what was."

While questions surrounding radiation from VDTs will probably not be answered for years, studies point to a number of other pressing problems. Poorly designed equipment and ill-conceived work places have prompted VDT operators to lodge a variety of health complaints with employers and unions.

At Blue Shield of California, for example, NIOSH found that 90 percent of a group it sampled had experienced back problems during the previous 12 months. Operators also reported tearing or itching of the eyes (79 percent), headaches (89 percent), severe fatigue or exhaustion (83 percent), blurred vision (78 percent), and eyestrain or sore eyes (93 percent).

NIOSH and other researchers say that such problems are caused by poor lighting, heavy glare, improperly adjusted chairs and tables, nondetachable keyboards and poor terminal displays. Both manufacturers and employers must assume partial responsibility—the manufacturers because they've tended to emphasize cosmetics over user comfort, and employers because they've rushed headlong into office automation without considering the impact on workers.

"Some employers have been responsive," says OPEU Research Director Gwen Wells. "But when they put a machine in the office, they put them there to increase production, and that's what they're concerned about. So you can talk about rest breaks, but they want their machines to be running full-time. Some of these changes cost money."

For the moment, microcomputerists have remained relatively unaffected by the controversy. Says Steven Sauter, a psychologist with the University of Wisconsin Department of Preventive Medicine:

"Poorly designed equipment and . . . work places have prompted VDT operators to lodge a variety of health complaints. . . ."

"The types of users you're talking about are generally highly motivated, high-level, well-trained individuals. The work they're doing is more creative. So I think that the problems we're seeing right now in the office will not exist in the home.

"In the office you're talking about a fast pace, routine work, no control at all in the work place, no dedication to or personal interest in or understanding of what they're doing. At home, you can live with the inconvenience for short periods of time."

But as microcomputers are used more in businesses and schools, problems are sure to arise. Says Sauter of equipment for the home, "that stuff is still in the Neanderthal Age." Many do not offer detachable keyboards; the screens generally are not adjustable; the nonglare features are often inadequate; the quality of the terminal display—especially when a TV set is used—can cause a great deal of eyestrain.

While consumer demand has compelled many mainframe and minicomputer manufacturers to pay attention to human engineering, microcomputer firms have faced little pressure. In fact, some see little or no problem at all.

"It used to be a big problem that there was no software," says a Tandy engineer. "Now we've got software for the machines, and people start nit-picking. The major points are solved, and they're looking for the minor points."

NIOSH researcher Dr. Marvin Dainoff takes a different approach. "People are so excited about these, and so impressed by their capabilities, that they'll overlook the problems," he says. "But I would guess that home users sooner or later will want better design."

by Eric Maloney
Kilobaud Microcomputing

American Comes Home from Asia To Head Tandy's Manufacturing

Seymour Bogitch, a top manufacturing executive from Tandy's Asian subsidiary, has been promoted to the new position of Senior Vice President for Electronics Manufacturing for Tandy/Radio Shack, Fort Worth, TX. He will be responsible for Tandy's 26 manufacturing facilities in the U.S., Canada and Asia, and Tandy's two product development engineering groups in the U.S. and Asia. He will report directly to Tandy President John V. Roach.

Bogitch said his new position is a combination of two existing posts.

"We have plants in the U.S. and Canada and in the Orient," he said. "In the past we had a vice president for North America in manufacturing and the equivalent in the Orient. No one was officially on top of both. John Roach used to do that."

The new position, therefore, was a result of Roach's promotion. Bogitch was a natural choice for the job. An electrical engineer with a master's degree from Northeastern University, Boston, MA, he has been representative-director for manufacturing for Tandy Electronics—Asia, Tandy's Japanese-based Far Eastern subsidiary, for 10 years. Before that he installed radio stations. He worked directly under the Japanese president of the subsidiary and was responsible for Tandy plants in Japan, Korea and Taiwan.

Bogitch said they make a variety of things in the 1,500-employee Korean and the new 1,000-employee Taiwanese facilities. They make Tandy's CB equipment, many small radios such as the Weather Radios, almost all Tandy's hi-fi equipment, public address systems and multimeters. The Japanese plant, originally a manufacturing factory, has been turned into a research and development and purchasing installation, he said. The staff, once close to 200, has shrunk to 25 in response to changing economic conditions. Bogitch said it just isn't economical to make these things in Japan because of rising wages.

Although born in New York, Bogitch has lived on and off in Japan, "since I became a big boy." He was living in Japan when Tandy hired him. He is married to a Japanese woman, and they have two children. He said the boy, Ray, 4, understands English but prefers to speak Japanese. The girl, Yoko, 8, is completely bilingual. They have lived in Hiroshima, Nagoya and Tokyo. They now live in Fort Worth. He said they are all adapting quickly, although Japan still "feels like home" to him.

Bogitch is a job-oriented man who spends his personal time reading and watching television. He said he gets the greatest on-the-job satisfaction from being involved with new products.

"I'm proud I contributed to the new things we made in the Orient," he said. "When we came out with our digital receivers, for instance, we had one of the lowest-priced on the market. With telephone equipment we started from almost nothing and became one of the world's largest suppliers. It was the same with CBs. We are one of the largest suppliers of multitesters."

He said he hopes to continue to expand Tandy's manufacturing capabilities and add new items like the TRS-80 microcomputers. Tandy manufactures all of these except the Pocket Computer, which they buy from Sharp.

"I'd like to come up with a few others like those," Bogitch said.

In fact, Tandy's biggest computer manufacturing problem is keeping up with demand. Bogitch said they have not caught up yet, but he hopes to solve that problem in the next few months.

Seymour Bogitch

In that vein, he said the new Texas Peripherals plant Tandy established last year in conjunction with Datapoint Corp. is going well. Under that unusual arrangement, each firm owns half the facility. So far its only product is a Radio Shack disk drive, but he said it will be making products for Datapoint soon as well.

"We are technically oriented," Bogitch said. "We are generally considered mostly a retail operation, but we do have a considerable manufacturing capability."

by Bert Latamore
80 Microcomputing *Staff*

Dutch to Air BASIC Program

In what may be a first, an international shortwave broadcasting station will soon broadcast a machine readable computer program around the world.

On Sept. 10, the Dutch World Radio Service, Hilversum, Holland, intends to broadcast a brief BASIC program in computer ready, CLOADable form as part of a weekly science segment called "Media Network". The show features microcomputers as its topic, and the BASIC program broadcast will be a housekeeping program. It will be broadcast in TRS-80, Apple and Pet compatible formats.

The broadcast may herald a new era in information exchange for microcomputerists. Should the reception of computer programs over the shortwave bands by listeners equipped with ordinary receivers turn out to be a straightforward process, the dissemination of software for popular microcomputers could take a large leap forward: A leap made at the expense of the many cable network facilities now being planned. In addition, the public broadcast of machine readable code could pose new legal questions for the precedent-poor microcomputer software industry.

The key to the success of the experiment lies in the Dutch station's signal strength in the targeted reception area. If the received signals are strong, free from fading and phase distortion, and atmospheric noise levels are low, listeners around the world stand a good chance of successfully recording the computer program.

According to Johnathan Marks, the producer of the Media Network show, similar experiments have successfully been performed within Holland by the Dutch domestic broadcasting service. A weekly program called "Hobbyscope" has used FM transmissions to broadcast several BASIC programs to its listeners.

This experiment on the international shortwave band will be conducted in the AM transmission mode, however. The resultant loss of fidelity and increased susceptibility to noise inherent in AM transmissions may cause problems for listeners in weak signal regions of the world. With this in mind, the Dutch Broadcasting Service will use its remote transmitting facilities in Bonnaire in the Caribbean and in Madagascar to ensure adequate signal levels in North America and throughout the world.

It is the Dutch Broadcasting Service's hope that computerists around the world tune in to the Media network segment at the proper time and frequency (see Table 1) and make an effort to record the BASIC program. Listeners are encouraged to report their results to Radio Netherlands as soon as possible at the following address: Computer Experiment, Media Network, Radio Netherlands, P.O. Box 222, 1200 JG Hilversum, Holland. If the transmission is a success, additional shortwave computer program transmissions are planned.

Several measures can improve your chances to successfully receive the computer program. First, use a good quality, highly selective shortwave receiver. Since adjacent channel interference in the form of heterodyne tones and cross-talk is common in the crowded international broadcast bands, a receiver that minimizes the amount of this interference is desirable. Also, a good quality antenna that maximizes received signal strength is a must. While a directional dipole antenna cut for the specific frequency of the transmission is ideal, a 50 to 100-foot length of wire is satisfactory. Finally, the received audio signal should be routed directly from the external speaker jack of the receiver to the input jack of the recorder.

By following a few precautions, and if atmospheric conditions are right, the Sept. 10th experiment may have far-reaching implications in the computing world. Tune in.

by Chris Brown
80 Microcomputing *Staff*

TARGET RECEPTION AREA	FREQUENCY (KHz)	TIME (GMT) +
Eastern N. America	9590 & 6165 KHz	02:47*
Western N. America	9715 & 6165 KHz	05:47*
Australia	9770 & 9715 KHz	07:47
Australia	9715 KHz	08:47
Europe	15560, 11930, 9895, 6045 & 5955 KHz	09:47
Europe	17605, 11930, 9895, 6045 & 5955 KHz	13:50
S. E. Asia	11735, 15560, 21480 KHz	14:47
East Africa	15220 & 6020 KHz	18:47
West Africa	21685, 17695, 17605, 15220 & 9715 KHz	20:47

+ Note: Greenwich Mean Time is five hours ahead of Eastern Standard Time.
* Times indicated are early Friday morning GMT. Note that it is still Thursday evening in target area.

Table 1. Radio Netherlands Transmission Specifics for Media Network Program Segment

Retired Tandy/Radio Shack President Granted Honorary Boston U. Doctorate

Retired Tandy/Radio Shack President Lewis Kornfeld has received an honorary Doctor of Humane Letters (LHD) degree from Boston University, Boston, MA.

BU said Kornfeld, who is vice chairman of the Fort Worth, TX, based Tandy Corporation, played an "exceptional role in the development of a great corporation that has had a profound influence on American life."

Kornfeld joined the original Boston Radio Shack store in 1948 as advertising manager. In 1954, by which time the store's business had tripled to $3 million a year and the staff grown from 30 to 60 people, he was named vice president of advertising. In 1958, with Radio Shack sales at $6 million, he become vice president of merchandising and advertising.

✓See List of Advertisers on page 306

Kornfeld stayed with the firm when it was bought out by Tandy Corp. in 1963 and later said his decision to do this was a turning point is his life.

Kornfeld is considered by the company to be the "father of manufacturing at Radio Shack."

He said bringing Tandy into the manufacturing end of the electronics business when he did was a significant development.

"There wouldn't be any TRS-80 computer if we hadn't. . . had seven or eight years of manufacturing experience."

Lewis Kornfeld

"There wouldn't be any TRS-80 computer if we hadn't already had seven or eight years of manufacturing experience under our belts," he said.

Kornfeld also received the Distinguished Professional Achievement Award from the University of Denver earlier this year.

by Bert Latamore
80 Microcomputing *Staff*

Radio Shack April Sales Jump 31 Percent over '80

Tandy/Radio Shack, Fort Worth, TX, had 31 percent higher sales this April than it did a year ago, and Garland P. Asher, director of financial planning for Tandy, sees it as a sign that the retail market is firming up. Asher said April was the third straight month of unexpectedly high sales figures. He said this may be a reflection of the unexpected strong upturn of the U.S. economy in the first quarter of 1981.

On the other hand, he said, Tandy's ups and downs are generally not tied as much to economic figures as they are to technological trends. For instance, 1974 was a recession year, but it was a good year for Tandy, which was riding the CB boom. In 1977 positions were reversed. Tandy, tied to a CB market bust, performed sluggishly, while the economy generally was strong.

Asher said whatever is fueling the present increase, it seems to involve a broad cross section of Tandy products. He said he didn't have any inventory breakdowns for April, but the indications were that while computer and telephone products were among the leaders for Tandy, other Radio Shack products were also attracting larger markets. Tandy's stereo equipment, which has suffered from depressed sales for more than a year, for instance, has picked up considerably, he said.

One interesting part of this phenomenon, he said, is that the Great Lakes industrial cities including Detroit and Pittsburg, which have been depressed market areas for some time, are showing 20 percent sales gains over a year ago. He said he had no idea why this is happening.

The figures as released by Tandy showed a consolidated sales totaling $138,048,000 for April, a 31 percent increase over the $105,179,000 figure of a year ago. These figures include both sales in the U.S. and overseas. The U.S. figures were $111,562,000 for April, up 32 percent from $84,238,000 for a year ago. Sales in U.S. Radio Shack stores in existence more than one year rose 20 percent in April over a year ago.

by Bert Latamore
80 Microcomputing *Staff*

Personal Micro to Fight Shack Suit, PM Prexy Terms It Scare Tactic

Personal Micro Computers, Inc. (PMC), Mt. View, CA, has issued a statement promising vigorous defense in a suit filed by Tandy/Radio Shack, Fort Worth, TX, charging copyright infringements.

"It is quite obvious that the purpose of this suit is to intimidate. . . PMC dealers and. . . 'scare off' potential dealers..."

Dr. Lester Lee, PMC president, said, "It is quite obvious that the purpose of this suit is to intimidate present PMC dealers and, most importantly, to 'scare off' potential dealers who might be interested in carrying the PMC-80 product line."

At issue is the PMC-80, a Z-80 chip-based microcomputer compatible with most TRS-80 software and peripherals. Tandy, in the suit filed Feb. 19 in U.S. District Court, San Francisco, CA, claims the machine's I/O routines are copies of TRS-80 routines and violate Tandy

✓See List of Advertisers on page 306

copyrights. They also maintain the PMC-80 trademark is "confusingly close" to the TRS-80 name and is a trademark infringement. Named in the suit with Personal Micro Computers is EACA, the Hong Kong-based manufacturer of the microcomputer, and several U.S. distributers.

Lee said PMC-80 "will give. . . independent software and hardware suppliers an alternative vendor to proliferate their product in the true spirit of free enterprise."

As evidence that the suit is aimed basically at discouraging potential dealers, Dr. Lee cited "the fact that Tandy did not inform PMC or EACA about any possible infringement by the product prior to filing the suit."

Lee said Tandy is trying to eliminate competition.

"While PMC-80 has been delivered for only about six months here in the United States, the same product has been widely accepted in Europe and other countries as the Video Genie marketed by distributors for EACA," Lee said. "In many countries the Video Genie has been out-selling the TRS-80, and this has probably prompted Tandy to take action in an attempt to quench PMC before it gets a strong foothold in this country."

Atty. Gary Pat, a Tandy spokesman on legal matters, said Tandy's policy is to refuse to comment on the case outside court.

by Bert Latamore
80 Microcomputing *Staff*

Computer Literacy Made Requirement for Graduation

Nolan Catholic High School in Fort Worth, TX, has established a unique requirement for graduation. Along with the more traditional areas of academic competence, students will soon be expected to demonstrate computer literacy.

Although computer literacy will not be mandatory until the 1982-83 school year, courses in BASIC programming began in August 1980. Currently in use are one Level II and 17 Level Is which were donated to the school. Another Level II will be donated for use in September, 1981.

This year, approximately 150 students have taken the courses which emphasize a hands-on approach to programming. Over 200 students are expected to sign up for next year's offerings which, in addition to beginning level courses in BASIC, will also include word processing and data processing for more advanced pupils.

Parents as well as students have responded positively to the computer courses offered by the school, according to Brother Tony Pistone, principal at Nolan Catholic. This year more than 60 parents attended an introductory computer class held at the school at night.

A major source of difficulty has been the lack of high quality, commercially prepared educational software, according to Brother Pistone. Two professional programmers have been hired for the 1981-82 school year to fill the gap created by the lack of good software and help teachers learn new ways to utilize the micros in their courses.

"Two professional programmers have been hired. . . to fill the gap created by the lack of good software. . . ."

Brother Pistone is enthusiastic about the current computer literacy program at Nolan Catholic and the school plans to expand the program in the coming years. According to Pistone, computer literacy is now "as critically important in learning as reading, writing and arithmetic."

by Lise Markus
80 Microcomputing *Staff*

80 CALENDAR

July

July 5-31 The Hill School, Pottstown, PA, will conduct **four one-week computer workshops** using the school's PDP 11/34 system and will offer students maximum hands-on experience. The first three workshops will be open to students of Grades 7-12. The last will be for teachers and other professionals.

Contact John E. Parnell, The Hill School, Pottstown, PA 19464, for information.

July 13-14 will see a **seminar on using the OASIS operation systems on Z-80 microcomputers** at Phase One Systems, Oakland, CA.

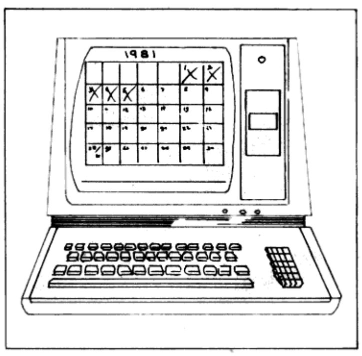
Classes will be limited to 20-30 students with plenty of "hands-on" activities. Price is $195. Information is available from Phase One Systems, 7700 Edgewater Dr., Suite 830, Oakland, CA, 415-562-8085.

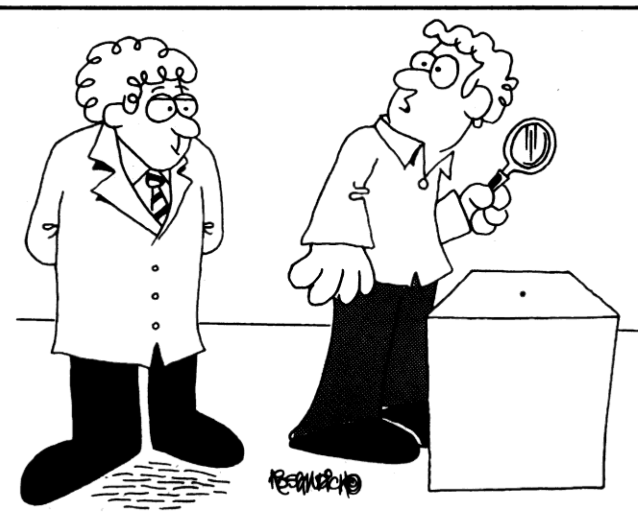
Motorola Technical Training Headquarters is offering seminars on their 6809 microprocessor chip July 21-24 in Phoenix, AZ; July 7-8 in Los Angeles, CA; and July 9-10 in San Diego, CA. They will cover all aspects of chip operation including software design. Seminar cost varies from between $300 and $450. Information is available from Ron Bishop, Motorola Technical Training Headquarters, TOM-57, PO Box 2953, Phoenix, AZ 85062.

August

Aug. 28-29 the **International Microcomputer Fine Arts Festival** will combine artists using or interested in using microcomputers with programmers and other technical people who have done work applicable to the needs of artists at the Teela-Wooket Camp, Roxbury, VT.

The event is sponsored by Trans/Media Inc., a non-profit artists' cooperative; Green Mountain Micro; Wayne Green, Inc.; and individual artists and programmers. Information is available from Dennis B. Kitsz, festival director; and Richard B. Fredette, festival coordinator, both of Roxbury, VT 05669 (802) 485-6112.



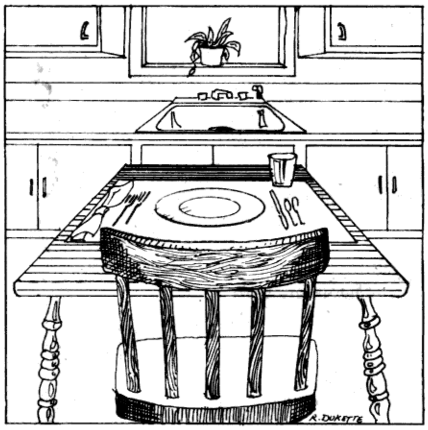


"Could I see one of your bigger models?"

✓See List of Advertisers on page 306

News From KITCHEN TABLE SOFTWARE, INC.

Ultimate DOS: Still not available from Kitchen Table Software



Last night, I dreamt that I was sent a review copy of the ultimate disk operating system for the TRS-80 Model I. Unfortunately, I woke up before I was able to produce a backup disk of this wonder product, called DROSSDOS 1.1. Even though no copies are known to exist in the real world, I thought I'd go ahead and write the review, anyway.

DROSSDOS 1.1 from the Kitchen Table Software Corporation contains many useful utility programs. It might be helpful to explore some of their features before delving into the actual operating system and Disk BASIC enhancements.

Superduperzap

Superduperzap is a great multi-purpose machine language disk utility program, which works in conjunction with a small hardware modification recommended by Kitchen Table Software.

When used in a compatible computer, Superduperzap will read or write to any disk sector, or main memory, including the ROM! To write to the latter, the user must make a few changes to the keyboard, which include replacing the Radio Shack ROM with an EPROM supplied by Kitchen Table, adding a small ultraviolet lamp and several printed circuit cards. The entire installation was completed by the author in only 17 hours.

The work was well worth it. Superduperzap repairs Parity Error During Write, Directory Error During Write, disk-destroying blunders and performs proper repairs to sectors. No user input is required. To invoke, the abbreviation FT is entered or (optionally) the full command Fix That! may be used.

This program will automatically recover accidentally killed disk files. The syntax for this function is as follows: Resurrect filename/ext, :? Superduperzap will search through all available sectors on all disk drives until the remains of a file by that name (or anything similar) are found. The file will be restored and the directory updated. If some of the sectors have already been overwritten, Superduperzap will invent machine or BASIC code that looks good and fits in with the rest of the program. This reconstruction may be entirely transparent to the end user.

Superduperzap has too many capabilities to discuss here. For example, the program can be commanded to ignore read and write protected sectors entirely, forever, and will list the contents of a given track to a printer, the CRT screen, your color television set, or the face of any LCD display digital watch within three feet of the keyboard.

Editor/Disassembler is another machine language program. When used with any optical character reader, it will capture assembly language programs printed in microcomputer magazines, and assemble them into flawless command files. It also will take Z-80 object code and disassemble it, providing helpful remarks which explain the purpose of each instruction. If the program, as entered, conflicts with other machine language programs that will be loaded simultaneously, this module will re-write the code to relocate to other addresses.

The user may also input code, which is automatically debugged. I liked this editor's flexibility; I didn't have to be excessively precise when entering instructions. If I was close, Editor/Disassembler would make a good guess as to what I meant and supply the correct op code.

Programmers with a Sense of Humor

Lodecodeoffset is another program which was also written in machine language. (You've really got to hand it to the Kitchen Table boys for entirely eliminating the utilities originally supplied with DROSSDOS 1.0, which were written in Pilot.)

Lodecodeoffset takes any machine language program, and, if it conflicts with DOS, some other binary module, ROM, or your automatic phone dialer, relocates it into unused memory. The Kitchen Table wizards have found an extra few hundred bytes of memory that is included in the 64K that can be addressed by the Z-80, but which are not included in the 12K ROM, video memory, or user RAM. Eight IC memory chips are supplied with DROSSDOS 1.1, and can be installed by the user in a few weekends.

Warning!! Each time Lodecodeoffset is run, the program adds an appendage to the code. This appendage has no purpose, except that when appendages equal to 255 bytes have been added, the object code is automatically destroyed beyond recall, even with Superduperzap. My sources tell me that this feature is a joke dreamed up by the Kitchen Table staff late one Friday night. It's always refreshing to see programmers with a sense of humor.

Adventure Disassembler is a 36K program written in BASIC, with no purpose other than to solve Adventures. It will take any adventure written by Scott Adams, and, through sophisticated byte crunching, provide a printout of the location of all treasures, secret words, and proper use of each object. We tried it on a couple of our own adventures and discovered that by stuffing the Mongoose/Squirrel into the tape recorder, and throwing it through the window of the Mystery Fun House, you can gain access to the dumb waiter!

Dirfink is a program that will test a target disk's directory and, if any errors are found, provide the name of the operator who was running the TRS-80 when the problem occurred. When the messages Bad Gat Sector Byte and Extent Space Overflows Diskette are displayed, Dirfink explains what these mean, and automatically invokes Superduperzap to correct the problem. Another handy program.

TI59/CMD is a program that allows the TRS-80 to emulate a TI-59, with the added capability of storing the programs to disk. I tested this module carefully, and found that it worked. Using it, I was able to combine all the power of a TI-59 programmable calculator with the portability of a 48K TRS-80 with four disk drives in one machine. Somebody should have thought of this one a long time ago.

Middlecase/DVR allows users who have neither upper nor lowercase character generators in their machines to make use of the less popular middlecase character set. A specialized program, to be sure, but with 300,000 copies of the TRS-80 Model I sold before Radio Shack decided to discontinue it, you're bound to find several hundred of almost any wild configuration you can think of. I personally have seen several TRS-80s whose owners have installed inverted keyboards.

DOS Library Commands in DROSSDOS 1.1

Some unusual library commands in this DOS include Checkout, Overdue, Cannot Renew and Reference Only.

The more usual commands such as Chain, Clock, Kill, List, Load, etc., are also included. One caution: Just because you have read the TRSDOS manual, do not assume that you know everything there is to know about DROSSDOS commands. There are some subtle differences. For example, in DROSSDOS, Kill causes the computer to energize the keyboard with 110 volts. The correct DOS command to get rid of a file is Erase. Even here, Kitchen Table has built in some valuable error checking. Below is a sample computer-human interchange:

```
Erase TestFile/BAS:1
Do you really mean that?
Yes
O.K. Enter the password.
No
Then say "please."
```

As you can see, it is nearly impossible to kill a file by mistake, or even on purpose. DROSSDOS has certain other "friendly" characteristics. For example, the following input might be used: BASIC 64000 RUN "STARWARS/BAS". The computer responds: "Excuse me. I found BASIC, but do you really want me to keep 64,000 buffers open for I/O files? Or should I have looked for a program called BASIC 64,000? Should I load STARWARS/BAS before I run it, or what?"

Some DOS commands can be run without any user input whatsoever. Dump will cause the computer to spill paper from any attached printer onto the floor. Purge commands the system to build a blacklist from any mailing lists on its disks. DIR compiles a list of all programs on a disk, but it won't tell you what they are.

Other handy DOS commands: Verify asks the operator's name twice and then checks both answers to see if they are the same. MDcopy produces duplicates of patient invoices for physician's accounts receivable programs. Rename allows the operator to enter different names when asked by Verify.

Sysgen is a very powerful command that deserves an article of its own. Using a complicated series of switches (numbered SW1 through SW255), the user can configure the operating system to suit specialized needs. A few of the many options are explained below.

SW14 = n, where n is a number between one and four. This allows the operator to specify which disk drive is prone to failure, and the system will automatically avoid using this drive whenever possible. Also, whenever a Parity Error During Read or Data Record Not Found During Read error is caused by this drive, the system will ignore data supplied by the suspect drive, and use something interesting from the same sector on some other drive.

SW18 = Y or N. This switch is used to flag the marital status of the primary operator of the computer. Thereafter, all programs using gender and titles will address the operator as Mr., Mrs., Miss or Ms, as preferred.

SW20 = n, where n is a number between 0 and 255. The computer will always use this number as a seed for generating pseudo-random numbers, thus ensuring predictability when the operator wishes to win dice, cards and other computer games.

SW103 = message string. The message can be any cute saying the operator wishes to display whenever the system crashes. It also relieves boredom, and can be changed as frequently as necessary.

Most features of Disk BASIC remain the same under DROSSDOS 1.1. Several new capabilities have been added. Program lines can be renumbered using a simple Renum command. If no values are specified, the program will be renumbered beginning with line 0, to the end, in increments of 10, but in reverse order. That is, a renumbered program might begin at line 10000, and end at line 10.

Renum R directs the system to renumber the lines in random order. However, because all GOTO's and GOSUB's are changed to the correct new line number, the program will still work. The BASIC interpreter has been altered so that it is not confused to find line 69 following line 186½.

Renum P renumbers a program using only prime numbers as line numbers, while Renum F employs, you guessed it, only Fibonacci numbers for renumbering. Hats off to the Kitchen Table gang for another stroke of originality.

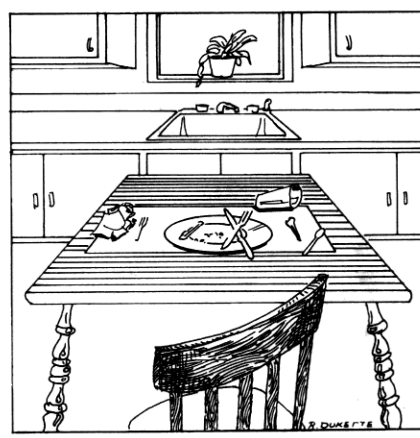
The popular Ref command has been expanded greatly. Under DROSSDOS 1.1, it provides an alphabetized listing of every word, number, command and punctuation mark in a program, with a cross-reference of the lines in which each appears.

Editing a program line has been made very simple. The operator enters a line number, and the relevant line is completely deleted from the program. Corrections are made just by typing in a new line. Level I had this feature—why has it taken so long to come to us Level II and Disk BASIC users?

DROSSDOS File Handling Capabilities

In their quest for simplification, the Kitchen Table software crew has come up a winner again. I found DROSSDOS' file handling perhaps the simplest to learn of any DOS, ever. There aren't any! Instead, all values are stored in variables within the programs themselves. Then, when an input session is completed, BASIC simply transfers the entire contents of memory—all 32 or 48K of it—to disk. The next time the program is run, DOS loads all of memory back where it belonged with all registers intact. In fact, the system uses buffers only to keep the CPU from getting a headache.

There isn't room in one article to explore all the features of DROSSDOS 1.1. If this innovative new operating system proves popular, you can be sure that there will be followup reports. ■



NEW PRODUCTS

edited by Bert Latamore

Lifeboat Publishes Buyer's Guide

Lifeboat Association's new buyers' guide and catalog lists 50 media formats, CP/M compatible disk operating systems, hard disk integration modules, system tools, telecommunications systems, languages, language and application tools, word processing systems and aids among other subjects.

It is available from the Catalog Department, Lifeboat Assoc., 1651 Third Ave., New York, NY 10028.

Reader Service ✓ 330

MOG Automates Mail Ordering

MOG (Mail Order Generator) for the TRS-80 Model I and III completely automates making mail orders.

It prompts you for all order information, allows a review of your order and complete editing, sorts the order by any of five variables, totals the order, adds tax and postage, saves the order on cassette, and drives a 32-, 40- or 80-column printer.

It supports lowercase on the Model III and requires at least 16K of memory, but will use up to 48K RAM for longer orders. It costs $10 from Practical Programs, 1104 Aspen Dr., Toms River, NJ 08753.

Reader Service ✓ 331

Program Figures Feed Costs

A new user-developed program for the Model I or III with 48K memory and a 132-column printer will store animal feed formulae, figure protein, fat, fiber, etc. for each formula and keep the price of each formula up-to-date with grain market prices.

The formulae are easily edited, deleted, added, displayed and printed and ingredients can be changed easily.

The program is available for $75 from Thomas R. Broussard, PO Box 2577, Lafayette, LA 70502.

Reader Service ✓ 332

Free Catalog Lists 200 Products

Creative Computing's new 48-page catalog lists more than 200 computer-related products including 20 books on programming, games and educational applications, 160 software packages, three magazines, five graphics and music peripherals, an LP record, a board game, eight T-shirts and an assortment of other products.

The 48-page publication is free from Creative Computing, 39 E. Hanover Ave., Morris Plains, NJ 07960.

Reader Service ✓ 184

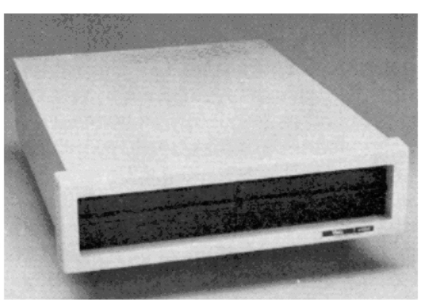
Dual 8-Inch Drive Offered

The V1000 dual eight-inch drive subsystem accomodates mass storage units ranging from single-sided eight-inch floppies to 20-megabyte streaming tape cartridges and 40-megabyte Winchester disk drives.

A sliding-chassis design gives easy access to the drives.

It is available with a choice of drive configurations. Prices range from $1,095 to $2,295 from Vista Computer Co., 1317 E. Edinger Ave., Santa Ana, CA 92705.

Reader Service ✓ 329



V1000 Dual 8-Inch Disk Drives

Data Conversion Achieved

Prestige Marketing Corporation is offering a data conversion service which will convert data from one system disk to that of another system.

For instance, Prestige can convert data from IBM 3740, DEC RT-11 and Apple II to TRS-80 disks and vice versa.

Information is available from Prestige Services Division, 909 N. Coliseum Blvd., Fort Wayne, IN 46805.

Reader Service ✓ 333

Program Tracks Accounts Receivable

ACCT-M3 carries the accounts receivable functions for a small business or medical clinic using initialization, account manager and report generator programs.

Data bases are limited only by the number of disks you own.

Designed for a dual-disk, 32K minimum memory, Model III with TRSDOS, it costs $69 from Micro Architect Inc., 96 Dothan St., Arlington, MA 02174.

Reader Service ✓ 171

Programs Published

A book of BASIC programs for the TRS-80 has been released by Sams Books.

The 168-page book has completely tested and debugged programs in home use, educational and business use areas ranging from an automatic telephone dialer to a checkbook balancer.

The book *Mostly Basic: Applications for Your TRS-80* costs $10.95 from Howard W. Sams & Co., 4300 W. 62nd St., Indianapolis, IN 46268.

Reader Service ✓ 334

Program Tracks Portfolio

Options-80 allows the TRS-80 to analyze investments for maximum return.

The program handles buying and selling listed call and put options, spreads and shares, analyzes impact of commissions, cost of money, dividend and risk exposure, and projects an annualized percentage return on investment as a

✓See List of Advertisers on page 306

function of annualized percentage growth in share value. It displays results in graphical and tabular form on video display or printer.

The program, for the Model I or III, 32K, on disk or cassette, costs $125 from Options-80, PO Box 471, Concord, MA 01742.

Reader Service ✓ 335

Book Discloses Structures

Structured Requirements Definition by Ken Orr is a presentation of recent advances in systems theory, tools and methodology in a readable text.

The book is available from Ken Orr and Associates, Inc., 715 E. 8th St., Topeka, KS 66607, for $25.

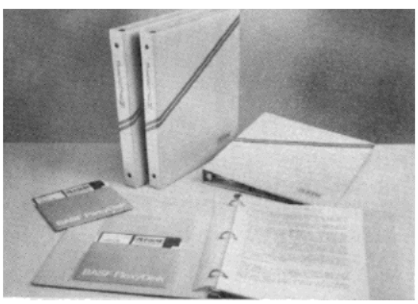
Reader Service ✓ 336

Percom Sells Binders

Percom Data Co. is offering three-ring binders with two inside clear plastic diskette pockets designed for software documentation.

The light tan binders, decorated with the Percom logo and designer stripes, sell for $4.95 from Percom Data Co., 211 N. Kirby, Garland, TX 75042

Reader Service ✓ 172



Percom Binders

Target Helps Business Planning

The Target business planning system is designed to replace time-sharing and large-scale systems for the small business in the analysis of past business activities and projection of future performance.

Advanced features include an ability to display or print the entire set of data entries, calculation rules and report specifications for error correction and the ability to follow English commands instead of matrix algebra commands.

It runs on the Model II with at least 56K memory and 200K disk storage using a CP/M operating system.

Created by Advanced Management Strategies Inc., Atlanta, GA., it is available from WESTICO, 25 Van Zant St., Norwalk, CT 06855 for $195.

Reader Service ✓ 174

Muse is Word Processing Program

Muse provides extensive word processing operations for ANSI Fortran-compatible computers including full-screen cursor control, automatic or manual pagination, letter, report and manual formats, block editing and erasure, page headers and feet, superscripts and subscripts, and multiple overlays for composite characters.

It runs on a variety of CRT terminals and outputs to all popular correspondence-quality printers.

A Muse package supporting one to three work stations and one terminal printer on perpetual lease is available for a one-time payment of $6,800 from Marc Software International, 260 Sheridan Ave., Suite 412, Palo Alto, CA 84306.

Reader Service ✓ 173

Centronics Offers Graphics Printer

The Model 739 is a graphics printer with a 74- by 72-dot per inch resolution, a monospaced print speed of 100 cps, and an acoustical top cover for single sheet loading and noise suppression.

It is available for less than $1,000 from Centronics Data Computer Corp., Hudson, NH 03051.

Reader Service ✓ 337

Program Interfaces PC, Models I and III

Pocket Tape I is a machine language program allowing the TRS-80 Models I and III to read data tapes recorded by the Pocket Computer.

The Model III versions may be used directly on a cassette- or disk-based machine. The Model I versions require an external hardware interface to the PC tape format. Minimum system memory is 16K.

The program costs $14.95 on cassette and $24.95 on disk, and the Model I cassette interface is available for $49.95 assembled and tested (or it may be built from Radio Shack parts according to the included schematic) from Green River Systems, PO Box 552, Auburn, WA 98002.

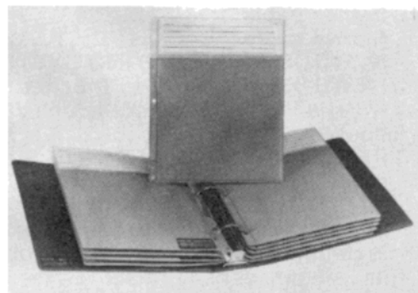
Reader Service ✓ 185

Envelope Protects Data

Data-Safe is a metallic shielding alloy envelope designed to shield two floppy disks from magnetic fields during storage. Each page fits in a three-ring binder.

They are $8.95 each for orders of five, or less with discounts for larger orders from Data-Safe Products Inc., 1926 Margaret St., Philadelphia, PA 19124.

Reader Service ✓ 160



Data-Safe

Select Word Processor Features Easy Operation

Select is a word processing program compatible with CP/M using microcomputers.

It features single-key instruction entry and the creators claim it takes only 90 minutes to learn to use it.

It costs $600 from Select Information Systems Inc., 919 Sir Francis Drake Blvd., Kentfield, CA 94904.

Reader Service ✓ 180

Program Teaches Even-Odds Play

Tired of losing money at casino blackjack tables? Basic Strategy Tutor I will teach you even-odds play using Las Vegas strip, Las Vegas downtown and Reno-Tahoe rules, with or without double down after splits option, with single or four decks.

It features moderate and advanced speed play options and tracks the number of hands played, number of hands played correctly, number of blackjacks dealt and strategy accuracy percentage.

It is available for $24.95 on cassette for a 16K machine or $29.95 on disk for a 32K machine plus $2 shipping and handling from Micro BlaJak Systems, Inc., 2800 N. Ellen St., Flagstaff, AZ 86001.

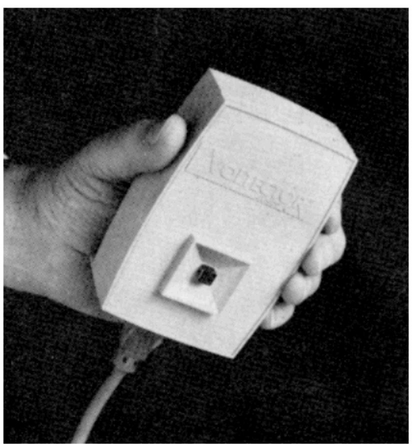
Reader Service ✓ 175

Regulator Stops Volt Surges

The Voltector protects microprocessor-based equipment from power surges, spikes, transients and high frequency interference. It meets latest industry surge voltage standards and gives two-way protection, preventing the microcomputers from causing line problems as well.

It is available for $79.50 from Pilgrim Electric Co., 29 Cain Dr., Plainview, NY 11803.

Reader Service ✓ 161



Voltector

M-Zal is Editor/Assembler

M-Zal is a modular editor/assembler for the TRS-80 Models I and III which includes full screen option menus, full screen text editor, and object module linker.

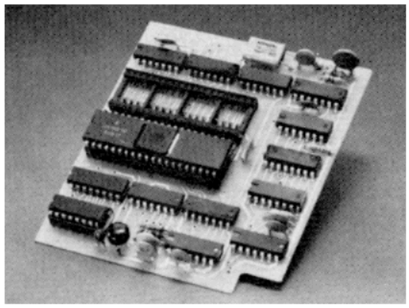
M-Zal is available for $149 from Computer Applications Unlimited, PO Box 214, Rye, NY 10580.

Reader Service ✓ 165

Run Model III Disks On Model I System

Doubler II is an update of Percom's Doubler double-density disk system adaptor for Model I computers.

Like the original, the new version allows



Doubler II

as much as 364K bytes of storage per side of a five-inch diskette, four times that provided by an unmodified Tandy Model I drive.

Unlike the original, this version allows the Model I to use Model III diskettes. The Model I cannot do this without modification.

Doubler II costs $219.95 including a DBLDOS Diskette. The upgrade kit for Doubler I costs $30 with proof of purchase of Doubler I. It is available from Percom Data Co., 211 N. Kirby, Garland, TX 75042.

Reader Service ✓ 181

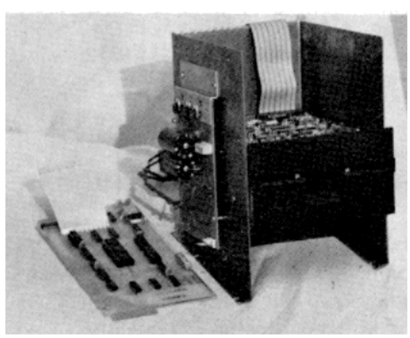
VR Data Announces Disk III

Disk III, from VR Data, is a five and one-quarter-inch disk storage subsystem that is 100 percent compatible with the Model III hardware and software.

Installation can be done by VR Data or by any mechanically-inclined person with hand tools. Disk III options include a second internal 40-track drive, an 80-track disk driver, a two-sided 40-track drive, and a two-sided 80-track drive.

The price for a basic unit is $599, the second 40-track drive is $265. Disk III is available from VR Data Corp., 777 Henderson Blvd., Folcroft, PA 19032.

Reader Service ✓ 328



Disk III

Convert Color Programs to Cartridge

TRS-80 Color Computer owners can have their programs on cassette put into a ROM cartridge, giving them instant loads. Eigen Systems will do this transfer with any Color BASIC or Extended Color BASIC program.

The ROM cartridge plugs into the Color Computer's external port. The program will run instantly upon power-up, and all memory can be used for data storage or graphic displays.

Prices start at $45 from Eigen Systems, Box 10234, Austin, TX 78766.

Reader Service ✓ 338

Board Doubles Color Computer Memory

Ramcharger is a completely assembled and tested printed circuit board which fits inside the Color Computer to increase its memory from 16K to 32K.

Completely compatible with Color BASIC, it requires no special software, no soldering or hardware modifications and leaves the ROM Pak port free.

It comes completely documented for $99.95 from Spectral Associates, 141 Harvard Ave., Tacoma, WA 98466.

Reader Service ✓ 177

Raiders Has Fast-Moving Animation

Space Raiders is a high-speed space battle game which puts the operator inside a spaceship on a search and destroy mission to intercept a Klingon convoy.

It features arcade-style simulation and gives a different game each time using a TRS-80 Model I, 16K Level II machine.

The game costs $24.95 from Bosen Electronics, 445 East 800 North, Spanish Fork, UT 84660.

Reader Service ✓ 164

A Walk on The Monster Side

Crush, Crumble and Chomp gives the game player the opportunity to be a monster, literally.

The player chooses from among several famous Grade B Movie monster greats or, in the disk version, he may create his own monster to invade New York, Washington DC., San Francisco, or Tokyo. He battles tanks, infantry, helicopters and mad scientists while trying to achieve his goals.

The player can choose from five goals: destroy buildings, destroy combat units, survival, eat or just blast everything in his path.

The game comes on disk (TRSDOS 32K) or cassette for $29.95 from Automated Simulations, PO Box 4247, Mountain View, CA 94040.

Reader Service ✓ 183

ASC Has New Program for Livestock

Agricultural Software Consultants, Inc., offers a new least-cost ration balancing program called Mixit-1. Mixit-1 can be set up for any type of livestock and comes with a 15-day money-back guarantee.

The program uses a machine language linear programming model to get a true least-cost ration quickly. In 16K Level II BASIC you can run Mixit-1 with 30 feed ingredients and 10 restrictions, and it is expandable to 32K.

The program costs $95 on cassette or $99 on diskette from Agricultural Software Consultants, Inc., 1706 Santa Fe, Kingsville, TX 78363.

Reader Service ✓ 326

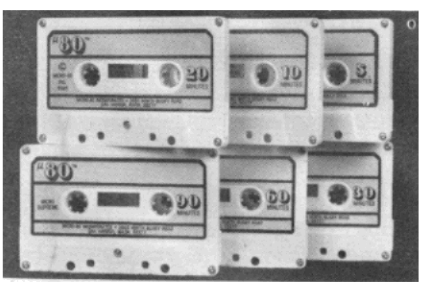
C-5 Tape is Quicker

Micro-80 now offers a cassette tape format designed for higher baud-rate systems. The Micro-Data C-5 five minute cassette allows over 24,000 bytes of storage per side at 1500 baud.

Compared to the C-10 cassettes used on the Model I, the C-5 will store more data in less space and less time. These capabilities make the C-5 ideal for the TRS-80 Model III and the Color Computer.

For more details write Micro-80, Inc., E-2665 North Busby Road, Oak Harbor, WA 98277.

Reader Service ✓ 325



C-5 Cassettes

Exchange Offers Demo Tapes

Computer Information Exchange Inc. is offering demonstration tapes on music synthesis on the TRS-80 and compilers in BASIC and Tiny Pascal, a tape head azimuth alignment tape, and an update on their SuperPimx data-base management program.

The music tape, designed for play on a high fidelity system, compares the performance of two synthesizer boards in popular and classical music including Bach, Handel, Mozart and Rossini pieces.

The compiler tape demonstrates how use of a compiler speeds play of two games, one in BASIC and the other in Tiny Pascal.

The head alignment tape, which has a 10K Hertz tone on one side and white noise on the other, allows the user to adjust his tape recorder head for maximum alignment by simply adjusting it to the maximum volume.

These tapes are $3.95 each.

The data-base manager is an update of a popular management system. The update has added pagination, easy accommodation of machine-language drivers, ease of editing fields or records, merge or split files, menu-driver memory management and user-chosen limits of number of fields.

SuperPimx is available on cassette for $19.95.

All these tapes are from Computer Information Exchange, Inc., Box 159 San Luis Rey, CA 92068.

Reader Service ✓ 179

OASIS Publishes Programs

Volume I of the OASIS Users' Group public domain software collection includes 12 games, a purge utility, a poetry generator and a loan amortization program on eight-inch diskettes.

Membership costs $35 including the package and is available from OASIS, PO Box 2400, Santa Barbara, CA 93120.

Reader Service ✓ 176

Memory Expander Plugs In

International Memory (IM) is a memory expansion board that will give the TRS-80 Model I up to 48K without an expansion interface, soldering or trace cutting and without software or any alteration of the micro's functions.

Two versions of the board, which plugs into the RAM sockets inside the keyboard unit, are available. The IM-1 gives a 32K byte capacity with 4K or 16K RAM chips and costs $47.50; the IM-2 gives up to 48K bytes with 16K RAM and costs $79.50. RAM chips are $32 per 16K bytes.

All are available from Holmes Engineering, 6246 W 3705 S, Salt Lake City, UT 84120.

Reader Service ✓ 163

Graphing Program Available

Automatic Graphing of Functions is a fast, low priced program which graphs equations in the form $Y = mx + b$ and $Y = (x)$.

It can graph simple formulas, multiple equations, summations, etc., automatically scales its axis for screen display size, has an LPRINT option for a lineprinter and has error handling to take care of tricky equations. It has manual or automatic range selection.

The program with user's manual for the TRS-80 Model I, Level II and Model III BASIC are $19.95 on cassette from David L. Modney, 4144 N. Via Villas, Tucson, AZ 85719.

Reader Service ✓ 169

Printer Uses 7 × 7 Dot Matrix

The Bytewriter I is an 80-column dot matrix printer that accepts single sheet or roll paper up to 8½ inches wide and prints at 60 lines per minute using a 7X7 dot matrix.

Designed to interface with all TRS-80 models, it retails for $299 with a limited 90-day warranty from Microtek, Inc., 9514 Chesapeake Dr., San Diego, CA 92123.

Reader Service ✓ 162

Bytewriter 1

Invasion Orion: Can You Defeat The Klaatu and Your Computer?

Look your computer straight in the eye, load in the new **Invasion Orion** and suddenly you are the Fleet Admiral, the Commander-In-Chief. And only you can stop the alien forces: the robotic Klaatu who have just invaded your Stellar Union space.

You command as many as nine starships; each ship spends energy on moving, on shielding itself, on firing its three weapon systems—destructor beams, missiles, torpedoes. There are 30 different types of starships—all armed with such a fantastic array as to intimidate the Klaatu.

But it isn't as easy as it seems. Your ships have only a limited amount of energy and you must decide how to allocate that energy to destroy the enemy.

Will your ship's armor be enough to stop the enemy's torpedo? Or should you divert energy from your beam to your shield? Move in for the kill on your weakened opponent and risk a beam attack? Or outrun the enemy? With each turn you take, your energy is replenished. Can you defeat the Klaatu? Or is your computer smarter than you are?

Invasion Orion is an EPYX game. Like all EPYX games, you will never get bored playing. Not in your or your computer's lifetime. Every game is different and fresh. Choose from three levels of skill: beginner, intermediate, expert. Ten fully tested scenarios, from one-on-one starship combat to full scale battle. Two programs: the first uses your pre-created scenarios to play the game; the second lets you create your own scenarios and design your own ships. A game that is infinitely expandable.

Yet so very easy to learn. With any of the ten scenarios, the computer takes care of all the details; no complex rules to remember. The screen shows prompts for your battle orders. Just concentrate on your strategy for victory.

Complete with superb graphics (if you have either an Apple or an Atari, you can enjoy color and sound!) and with battle manual, game program, scenario creation program, data files for your computer.

Invasion Orion. Another bug-free, easy-loading lifetime computer game from EPYX. With the unique EPYX lifetime warranty: If anything happens to your cassette or disk at any time and for any reason, send it back with just $5.00 for shipping and handling and we will send you a brand new one.

(Of course, there is also our 30-day unconditional guarantee: if your EPYX game has any defect whatsoever within 30 days of purchase, return it to us or your dealer and we will replace it free. No questions asked.)

Visit your dealer now and pick up **Invasion Orion** in its good-looking, protective box with the best instruction book you've ever read. Now available on disk for the Apple II (48K RAM with Applesoft) and Radio Shack's TRS-80 (32K RAM). Or on cassette for the Atari (32K), TRS-80 (Level II, 16K), Apple (16K and Applesoft) and Pet (16K). **Only $24.95, disk or cassette.**

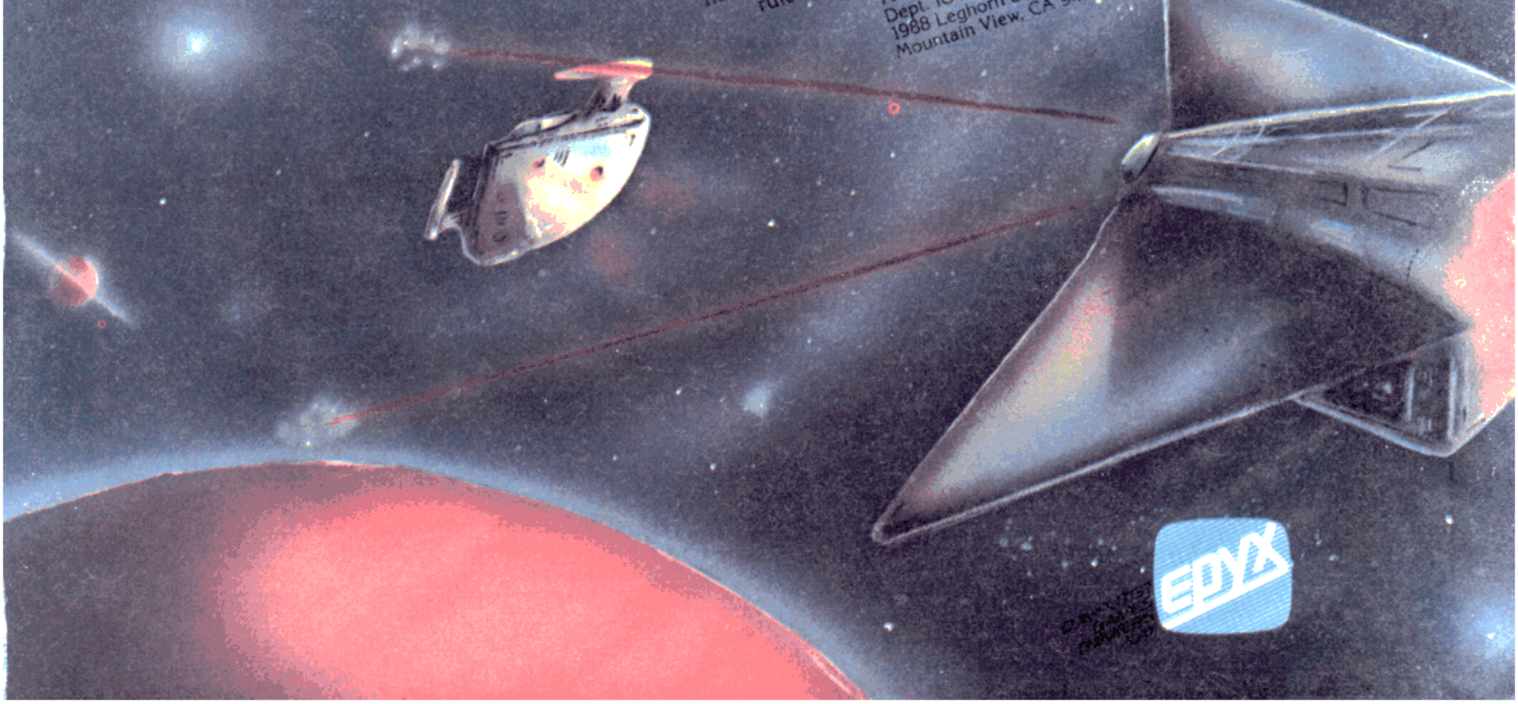
If your dealer is out of stock and you can't wait, order directly from Automated Simulations. $24.95 plus $2.00 for shipping and handling (and sales tax if you are in California).

We guarantee you will be delighted with **Invasion Orion.** (If it is not exactly what you want, return it to us for full refund. Again, no questions asked.)

Enclose your check. Or if you order by Visa or Master-Card, use our toll-free phones: In the United States: operator 861 (800) 824-7888; In California: operator 861 **(800) 852-7777**; In Hawaii and Alaska: operator 861 **(800) 824-7919.**

Order today. You and your computer deserve the fun.

AUTOMATED SIMULATIONS, INC.
Dept. 101, P.O. Box 4247
1988 Leghorn Street
Mountain View, CA 94040



NEW PRODUCTS

Program Analyzes Phone Bills

Long Distance Analyzer streamlines telephone bill accounting by identifying where your calls go, grouping calls, and totalling those groups for cost accounting, client billing and investigating unfamiliar numbers.

Usage patterns are identified by area code, state and WATS zone.

Cassette version for Models I and III is $95; disk is $135. Disk for Model II is $155, from Golden Braid Software, PO Box 2934, Sarasota, FL 33578.

Reader Service ✓ 166

Micro Link Manages Communication

Micro Link enables inter-computer communication by allowing files to be prepared in advance and transmitted automatically, automatic data-base scanning, the recording of items of interest to the user for later reading, and several options with default settings and simple, fast user commands.

It runs on a 16K Z-80 machine using Micropolis DOS or CP/M 1.4 and up (inquire about TRSDOS, etc.) and costs $89 from Wordcraft, c/o Microcomputer Software Assoc., 1122 B. St., Hayward, CA 94541.

Reader Service ✓ 182

Development System Put on Model III

The PDS assembly language development system uses TRSDOS on the Model III to provide a macro assembler, linkage editor/linking loader, string-oriented text editor, interactive editor/assembler, trace debug/monitor, disk disassembler and several other utilities.

The system is available on five-inch double-density disks with 100 pages of documentation for $99 from Allen Ashley, 395 Sierra Madre Villa, Pasadena, CA 91107.

Reader Service ✓ 178

Foto-File is For Photographers

Tape-Tronics is offering two software packages for photographers.

Foto-File organizes slide, negative or print album files by title, location, category or code and costs $19.95 for cassette and $29.95 for disk.

Darkroom Assistant is a three-program package covering prints from slides, negatives or Cibachrome process giving correct filtration values, exposure times and developing temperatures. Cost is $59.95 for tape or diskette from Tape-Tronics, 346 N. Western Ave., Los Angeles, CA 90004.

Reader Service ✓ 168

Radio Shack Made the TRS-80® Color Computer Even Better!

At $399*, it's no wonder the 4K TRS-80 Color Computer is ideal for beginners. Now there's a TRS-80 Color Computer designed for more advanced applications, too.

TRS-80 Extended Color BASIC Computer. This high-performance computer includes a 16K ROM Extended BASIC with advanced graphics, eight brilliant colors, and sound for an unprecedented low price! You can draw fine lines, circles, rectangles, boxes and more with *easy-to-use one-line commands.* Four graphic modes with two color sets allow up to 49,152 programmable screen points (pixels). There's 225 separate tones for music or sound effects, too. All this on a 16K RAM machine (including video memory) loaded with the dynamic features a serious programmer demands. You get a 32-cpl x 16-line screen, multi-character variable names (two significant), editing, tracing, user-definable keys, 255-character string arrays, floating point 9-digit accuracy, and even machine language routines.

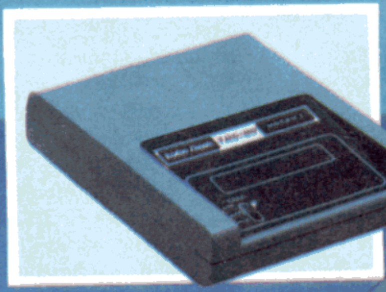
Priced at Only $599, the TRS-80 Extended Color BASIC Computer is useful, entertaining and educational. Yet using it can be as simple as plugging in one of Radio Shack's instant-loading Program Paks. The computer attaches to your TV, or our own $399 TRS-80 Color Video Receiver. For just $24.95, you can add a pair of joysticks which add flexibility to games and video displays. A built-in serial interface lets you attach a printer or a modem. A pair of tutorial, Extended Color BASIC instruction manuals are included, as well.

More Good News. Extended Color BASIC is also available as an upgrade kit ($99) for your 4K Color Computer (16K RAM required—$119). There's a modest installation charge for each kit.

New TRS-80 VIDEOTEX Software (with the modem shown below) offers quick, affordable access to various information and data services. The CompuServe® Information Service gets you local, national and international news, weather and sports from major newspapers, like The New York Times and The Washington Post, plus the Associated Press News Service; info on stocks and bonds; educational reference service; nationwide Electronic Mail and much more! The Dow Jones® Information Services provides stock exchange quotes—as recent as 15 minutes—plus feature selections from The Wall Street Journal and Barron's.

Only $29.95 Buys You VIDEOTEX Software including a free hour on both CompuServe and Dow Jones. Come see the new TRS-80 Color Computer, its programs and accessories, at your nearby Radio Shack today!

Radio Shack®
The biggest name in little computers™
A DIVISION OF TANDY CORPORATION
6100 STORES AND DEALERS, 140 COMPUTER CENTERS AND 135 SERVICE CENTERS NATIONWIDE



NEW

A Low-Cost, Direct-Connect Modem. A convenient alternative to an acoustic coupler. The TRS-80 Modem I lets you enter the world of microcomputer communication for only $149. Cable extra.

Specifications. Low-Power CMOS Circuitry; Full duplex 300 baud, 103 compatible ANS/ORIG. Sensitivity: −48 dB/−43 dB. Connectors: DB25/4-pin DIN; includes interface for Model I cassette port. FCC approved.

*Retail prices may vary at individual stores and dealers.
® CompuServe is a trademark of CompuServe, Inc.
® Dow Jones is a trademark of Dow Jones & Co., Inc.

Disk Drives and Interface—top end hardware for Model I connoisseurs.

Lobo Connections

LX-80
Lobo Drives International
Goleta, CA
$862
SA800
$1342

by Jake Commander

Though Lobo Drive's LDOS may already be a familiar name to disk operators, the equipment for which it was designed—the LX-80 expansion interface and the SA800 dual eight-inch disk drive—remain unexplored.

Why bother with a more expensive alternative to the Radio Shack expansion interface and drives? First, an unmodified Radio Shack interface won't accommodate eight-inch disks, whereas the LX-80 will. Also, since production of the Model I has ended, it's anyone's guess as to how long the attendant interface equipment will remain available. Despite such a threat the Model I is thriving, with reports of units being sold for more than they cost new. Manufacturers of ancillary gear are not merely continuing to support the hardware, but in some cases are offering superior alternatives.

Offers Reliability

I've been operating the Lobo equipment on my system for over two months without a single glitch. This reliability, reflected in the stiffer price, has also been built into the equipment.

Lobo engineers employed a good degree of overkill in their designs. For instance, try lifting the interface. Instead of a plastic case, both interface and drive units are enclosed in one-eighth-inch thick steel. Not that many users are likely to try it, but you could quite literally drive a car over the interface without damaging it.

The unit measures just under three inches high (lower than the RS expansion interface), by 19 inches wide, by 12 inches deep. Photo 1 shows how the interface looks in typical setup.

The LX-80 comes with a user manual which not only describes set up procedures, but offers the reader simple step by step diagrams. An enthusiastic owner who dabbles first, without reading, can plug in either the five or eight-inch drive connector cables the wrong way; neither the multi-pin plug nor the socket have a keying notch. Not that this is likely to damage anything, but, considering Lobo's high degree of engineering, this is a curious omission.

The manual also tells you how to change the LX-80's parameters to fit your needs. Furthermore, owners are actually told how to open the box and install up to 32K of their own RAM. The only criticism I have of the manual is that there is no circuit schematic. I don't care how secret Lobo's circuit design is, if someone wants this sort of quality, and pays this sort of price, he should have the option to maintain his own equipment. This is obviously impossible without the circuit diagram.

The interface may be powered from either 117 VAC at 60 Hz, or 235 VAC at 50 Hz. The manual tells you how to make the change by replacing a strap inside the unit. This dual-standard power supply is a blessing to foreign users. I speak from experience. I blew two Radio Shack power units while in Europe.

An extra bonus from the LX-80 is that it can also supply power directly to the TRS-80 keyboard, allowing it to run cooler.

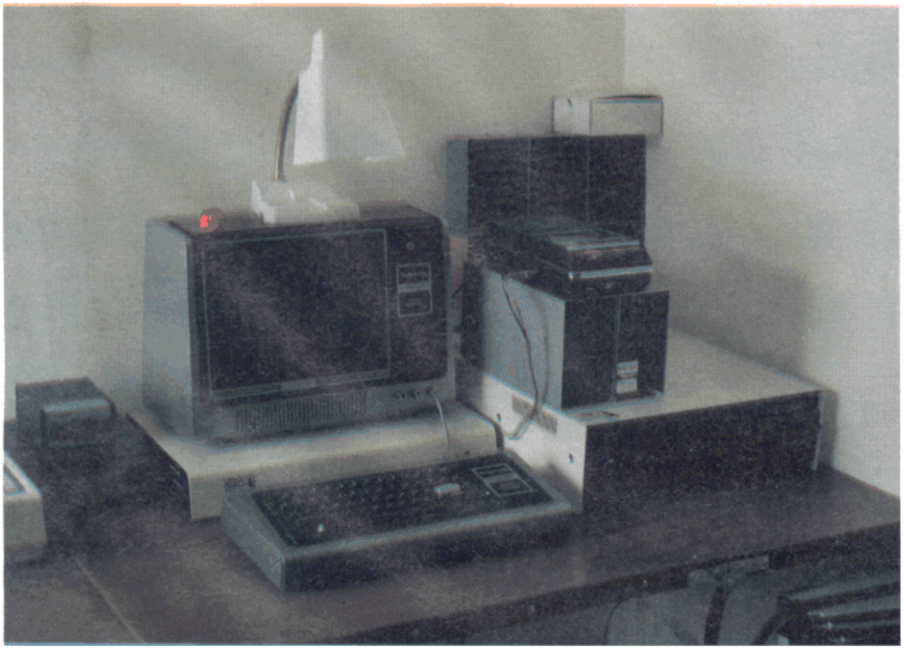


Photo 1. Lobo's LX-80 Shown in Typical Setup

"The LX-80. . .can also supply power directly to the TRS-80 keyboard, allowing it to run cooler."

Thus, you can exchange Tandy's plastic power units for Lobo's higher reliability. This is especially useful when operating the TRS-80 in a high ambient temperature when the back of the keyboard can become frighteningly hot.

Disk Mix

The unit is extremely flexible regarding the disks you can use. If you want to use four five-inch mini-floppy drives or four eight-inch standard floppy drives, the LX-80 will support it, or any combination. Not only that, but it's possible to configure the interface to boot up from either five- or eight-inch drives.

Even this doesn't cover all possibilities because the LX-80 can also support hard disks, specifically Lobo's 1850T dual fixed/floppy. The fixed disk comes in five- or ten-megabyte versions, the floppy gives up to 1.6 megabytes per disk.

This can be configured as the bootstrap drive. Though this may sound complicated, it's not. The whole point is merely to tell the interface which is drive zero, and this is done by setting small DIP switches at the rear of the unit.

In summation, the LX-80 supports floppies in double or single density, single or double-sided, up to four five-inch disk drives, plus up to four eight-inch drives, plus any number of hard disk drives, and in any combination. Try that with your average expansion interface. You now have the possibility of a huge data base of tens of megabytes connected to the TRS-80.

I need to add one small caveat to this glorious mixing of drives. The LX-80 doesn't map its disk input/output in the same way as does Radio Shack in their expansion interface. Lobo's unit addresses the drives via I/O ports. Radio Shack memory-maps them. This is minor, as that's the way both the Model II and the Model III computers access their drives.

Lobo decided to follow the same course by port-mapping I/O, which is no mean feat, as the ROM bootstrap loader routine in the keyboard unit expects the disk controller to be mapped in memory. The LX-80 appears to cope with this anomaly by flipping to memory-mapped I/O upon reset (which suffices to read the bootstrap sector) then flopping automatically to port-mapped I/O for all subsequent disk accesses.

The upshot of all this is that the few pieces of software written for the TRS-80 that don't perform disk I/O through the disk operating system will not work. Included in this small category are such programs as Super-Utility and certain adventure games which performs their own disk I/O.

This is the reason Lobo International

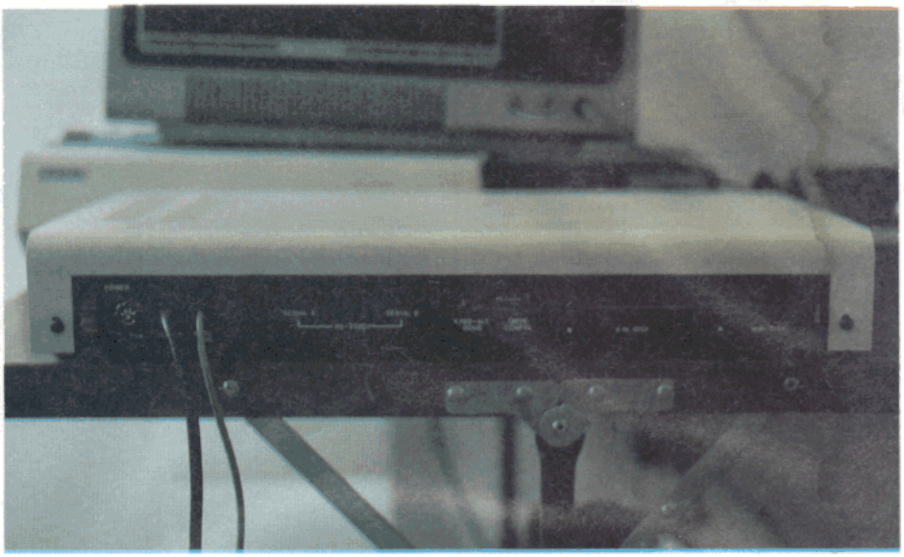


Photo 2. The LX-80 features serial I/O ports, alternate ROM switch, drive zero configuration switch and sockets for connecting five and eight-inch floppies.

needed to have a DOS developed especially for their hardware. All BASIC programs will work fine because the disk accesses are made via the disk operating system, in this case, LDOS. All machine code programs, such as Visicalc or Profile, will work as well. They, too, access disk files via the DOS.

For the curious, here are the ports used for disk I/O in the LX-80. Experienced TRS-80 hardware users will notice several extra options compared to the regular interface.

Port	Input	Output
E0	Hard disk data	Hard disk data
E1	Floppy FIFO data	Floppy FIFO data
E2	FIFO counter	FIFO mode (0-read, 1-write)
E3	DIP switch	Floppy select and modes
E4	FDC status register	FDC command register
E5	FDC track register	FDC track register
E6	FDC sector register	FDC sector register
E7	FDC data register	FDC data register

Other Capabilities

An interesting feature of the Lobo setup is the ability to override the keyboard Read Only Memory. A switch at the rear of the unit switches out the Level II BASIC ROM and switches in an alternate set which can be plugged into three sockets inside.

With the usual flexibility that seems to be part of Lobo's design philosophy, numerous sorts of ROM can be added by reconfiguring a set of jumper wires near the alternate ROM sockets. The kinds of ROM Lobo accommodates are 2708s, 2716s, or 2732s. This should allow the whole TRS-80 to operate with any dedicated application in mind. Possibilities include a Pascal or Pilot that can be available on power-up, or any number of industrial or mechanical applications.

The LX-80 contains the usual real-time clock which provides interrupts to the Z-80 once every 25 milliseconds in the same manner as the Radio Shack interface. Another similarity is the expansion port which replicates the pinout from the back of the keyboard, except for the five-volt supply on pin 37. This allows you to use the wide range of peripherals available for the TRS-80.

A Centronics-type parallel printer port is memory-mapped to the same address as the ordinary expansion interface, so printing is unaffected. Two serial output ports driven by a Z-80-SIO/2 controller will drive a serial printer. These RS-232 I/O channels can be configured by the user to interface with just about any serial device.

Opening the cabinet, you will see two jumpers, one for serial port A and the other for port B. These jumper plugs can be soldered to allow any serial custom configuration. Anyone who has need of serial I/O has probably come to realize just how non-standard a standard RS-232 bus can be. Lobo's configuration allows handshaking, other control signals or data to appear on different pins to the cable from the outside world.

Baud rates can be set from software, and range from 12.5 baud to 316.8 kilobaud. That should cover just about every possible serial device imaginable.

A description of the jumper plug follows. Note that the left side shows the signals available from the interface, while the right side shows which pins these signals can be

"The drives themselves function so perfectly it's almost boring."

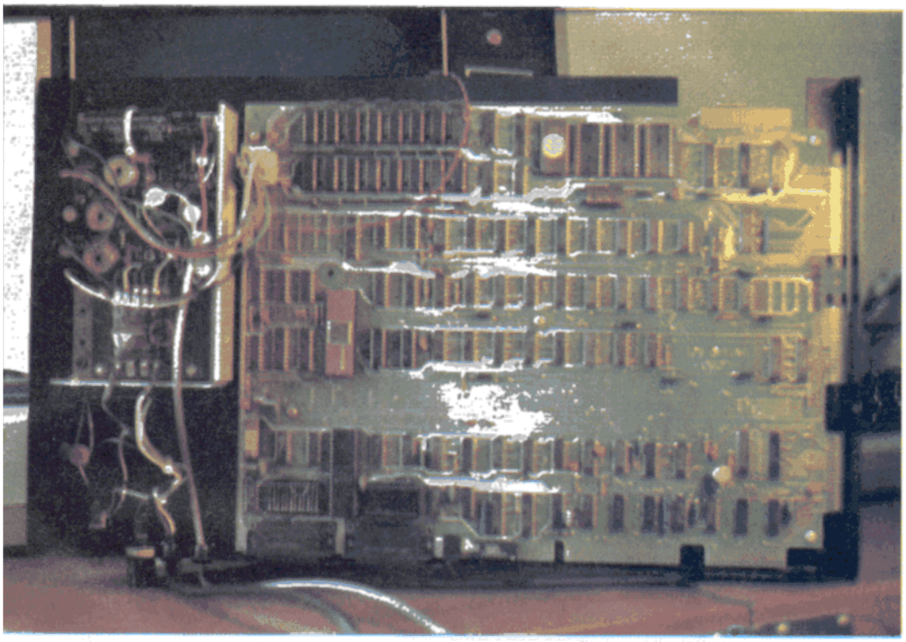


Photo 3. Inside LX-80

routed to on the DB-25S socket at the rear of the interface.

1. + 12 volts	24. Not used
2. Transmit data	23. DB-25S pin 2
3. Transmit clock	22. DB-25S pin 24
4. Request to send	21. DB-25S pin 4
5. Data terminal ready	20. DB-25S pin 20
6. Receive data	19. DB-25S pin 3
7. Receive clock	18. DB-25S pin 17
8. Clear to send	17. DB-25S pin 5
9. Data carrier detect	16. DB-25S pin 8
10. − 12 volts	15. DB-25S pin 6
11. Not used	14. Not used
12. Not used	13. Not used

Photo 2 is a shot of the rear of the interface showing the serial I/O ports, alternate ROM switch, drive zero configuration switch and sockets for connecting five- and eight-inch floppies.

Access couldn't be simpler to the 10.5 × 14-inch printed circuit board for adding RAM or changing jumpers. The board is rigid thick glass fiber held down by five screws which should eliminate any flexing problems.

To the right of this PCB is the power supply mounted inside an alloy shield. Photo 3 shows you the inside of the interface. The alternate ROM sockets are visible just right of center at the top of the main PCB, whereas the serial I/O jumpers are at bottom left of the same board. Notice the hefty power transformer.

The SA-800

Though the LX-80 steals the show, the dual SA-800 eight-inch single-sided floppy disk drive offers over one megabyte of storage in double density. The units are tried and trusted Shugart SA800 soft sector drives which boast a head life of 15,000 hours and a disk life of a phenomenal 3.5 million passes per track.

The cabinet is fairly large by microcomputer standards, measuring 17.5 inches × 22 inches × 4.5 inches with the same hefty construction techniques used for the interface.

Unlike five-inch floppies, eight-inch drive motors are constantly running, whether or not the drive is selected. For this reason, Lobo installed a cooling fan at the back of the cabinet. Unfortunately, for my taste, the fan was noisy to the point of distraction.

The drives themselves function so perfectly it's almost boring. I haven't noticed a soft error in two months.

A Shugart OEM manual is supplied with the drive unit, as well as a maintenance manual that does contain the circuit schematics.

Conclusion

Though the Lobo expansion interface offers much greater versatility versus others currently available, their drives just outperform, or are equal to others now on the market. However, both units are of the highest quality and seem to be aimed at the professional microcomputer user.

OK, Lobo International, when you are going to come out with a TRS-80 compatible keyboard unit? The industry could use one. ■

like the golf pros

THE STOCKING SOURCE

comes through with lower prices

- ***Your Closest Link to the Manufacturer***
- ***All Items Listed are Stocked***
- ***We Support & Service Products Purchased From Us***

NEW EPSON Model MX-80FT List $745.

Friction Feed has been added to the popular MX-80. All other features the same, including an adjustable tractor.

Model MX-80 .... List $645.
NEW — Friction Drive Kit ... $60.

Anadex Model DP-9001 ... List $1550.
80 Column
Model DP-9501 .... List $1650.
132 Column

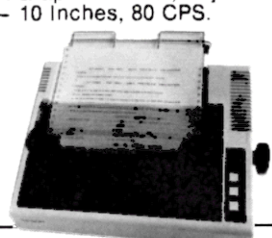
Unbelievable Dot Matrix Printer with All Functions Program-Accessible. Has 4-Character Font, Hi-Res Graphics, Switch Selectable Parallel, Serial & Current Loop Interface, Operator Selectable Protocal.

PRINTERS

Okidata —
Microline 83 .. PRICE REDUCED
Up to 15" Paper Width, 136 Columns Standard, 132 Compressed, Vertical Tab, Top of Form, Parallel & RS-232C Interfaces & Tractor Feed Included, Friction & Pin Feed.

NEW MODELS 82A & 83A
Available soon —
9 wire Print Head & 120 CPS

Microline 82 .. PRICE REDUCED
Microline 80 .... List $599.
Tractor Feed for Microline 80 & 82 ...... List $50.
Hi-Speed Interface for Microline 82 & 83 .. List $210.
256 Character Buffer, Expandable to 2K
RS-232C Interface for Microline 80 ...... List $150.
256 Character Buffer
Epson —
Model MX-70 .... List $445.
Bit Plot Graphic Printer, Adjustable Tractor 4 — 10 Inches, 80 CPS.



NEW — INTERFACES

Hi-Res Bit Plot Graphics for Epson Printers .... List $90/Set.
Plug In Prom Set
Vardon 1K Serial Interface Buffer for Epson MX-80 ... List $175.
(Complete with Cable)
Tymac — Apple Interface Using 8th Data Bit .... List $139.
(Complete with Cable)

INTERFACES

Apple Plug-In Interface & Cable .... List $110.
Serial Interface & Cable List $90.
IEEE 488 Interface & Cable .... List $80.
TRS-80* Expansion Interface Cable .... List $35.
*A trade name of Radio Shack — Tandy Corp.
Model 232 Universal Serial Interface Cable .... List $25.

NEW MONITORS

BMC & NEC — Black & White and Color

master charge VISA

No Penalty for Credit Card Orders

NEW — PRINTER LINES

Centronics Printers In stock.
NEW — with Graphics Model 739
Model 739-1 (Parallel) - List $995.
Model 739-3 (Serial) - List $1045.
Model 704-9 (Serial) - List $1795.
Model 704-11 - List $18.70
Also Ribbons & Accessories
All other Centronics Printers are Available.

We now stock NEC Spinwriters, Models 5510, 5515, 5520, 5525 & 5530. Also Thimbles & Ribbons. All other models are available, including Band Printers.

TEST EQUIPMENT

We stock Beckman, Fluke, Global Specialties, Keithley & NLS DMM's and probles. Also Hayes Technical Breakout Boxes.

MICRO-COMPUTERS

We stock the ALTOS Computer Line Featuring Hard Disk Memory.

FURNITURE & PRINTER STANDS

Systems Furniture and Printer manufacturers

ASK FOR OUR INSTANT DISCOUNT
From Roy Hawthorne
Talk To Bill Tokar On Applications

CALL TOLL FREE
U.S.A.
1-800-521-2764
MICHIGAN
1-800-482-8393

Reminder:
We are open
8:30 to 6:00 PM EST
Monday through Saturday

WRITE TO: ✓438
"The Stocking Source"
24069 Research Drive
Farmington Hills, MI 48024
313-474-6708

Though the pundits differ, the users seem clear on the issue.

Microcomputers— Business or Pleasure

by Bert Latamore
80 Microcomputing *Staff*

TRS-80 microcomputers are business machines, not home computers, according to the official view of Tandy/Radio Shack, Fort Worth, TX. While this idea might surprise many home users, Tandy's market surveys apparently back it up.

"Our primary thrust has always been for the business user," Ed Juge, Tandy's director of computer merchandising, said. "We only advertise to homeowners during Christmas time."

Tandy doesn't make its detailed market breakdowns public, so it is hard to determine just how large Tandy's business market is. However, its last annual report shows its total computer-related sales for fiscal year 1979-80 was 12.7 percent of its approximately $1.4 billion gross corporate sales or $175,848,890. This is up from $19,678,320 in 1978 or 1.8 percent of Tandy's sales.

Yankee Group, Boston, MA, a market research firm, estimates 250,000 business microcomputers of all kinds were in use at the beginning of the year.

Actually this is just the tip of the potential business market. The question is not whether a potential market exists—it is how much of this market microcomputers like the TRS-80 can capture. The answer depends on several variables including the amount of resistance the market has to change, the availability of adequate programs, and the ease with which the machine can be used by the average businessperson.

The experts are split on this issue. Each seems to have a different answer based on a different theory of the future. While they agree computerization is our common destiny, they are not all sanguine over the future of the TRS-80 in business. In fact, some are downright pessimistic.

Not so, for Ed Juge—he believes Radio Shack computers have a great future in the business market based on their record.

"Sales are growing by leaps and bounds," he said. "I think there's not any question that they earn their keep. I talk to people who say they saved the $4,000 for a Model I disk system in the first three weeks."

Juge said the Model III is designed as a desktop unit for an administrator who wants such things as data base management and word processing at his elbow. The Model II, he said, is intended to run a larger work station.

"The mainframes serve a particular market, and we serve a particular market," he said. "As the micros grow they will undoubtedly move into the mini market."

Even in complex mathematical applications, Juge said, the desktop unit has an advantage over the typically heavily utilized mainframe. The big machine may not get to your program in the three hours it takes the micro to solve it.

Juge predicted business microcomputers will do more communicating with each other and their larger cousins in the future. At Yankee Group, Senior Analyst George Colony expects the local network concept to grow. He predicts desktop computers in the future will normally be connected to a four-level system. The first level will be a mainframe computer—either company-owned or on timesharing. It will store data and run the network, which is the second level. The network will connect it with minicomputers which act as remote processing stations. These are connected with a microcomputer on each desk.

Most actual programs would run on either the micro or the minicomputers.

"The advanced work station (the desktop unit) must be fully compatible with Level III (the minis) and, perhaps, Level I," Colony said. "This cuts out TRS-80 and Apple."

Colony predicted both IBM and Xerox will enter the microcomputer market with units designed to interface with their larger machines. In fact, he said he had heard IBM will have two micros, one in the $700 range and one costing about $1,500. If Colony's highly-structured vision proves correct, these computers will have an overwhelming advantage over micros that are not specifically designed to work as part of such a system.

Colony said the ideal business microcomputer would be multifunctional, componentized and highly intelligent, probably at least 32K bits. It would be very easy to use and self-teaching to the point that a new person could start using it five minutes after he first sat down in front of it. It would be available through retail stores and may have a flat-plane screen.

Colony, like many others, also predicted a strong Japanese entry into the U.S. market this year.

Micros Not Ready

Extensive interfacing capabilities aren't the problem foreseen by Francis O'Reilly, an independent market analyst. His report for Business Communications Co., Stamford, CT, is the most pessimistic of those surveyed. He said the microcomputer and the business market just aren't ready for each other and won't be for 10 more years.

The microcomputer, he said, still requires technical expertise. Most businessmen do not have technical minds. For instance, he talked with one small-business owner who did not understand the need for programs. He thought he merely had to feed the computer data and ask it questions. To be acceptable to such people, the system has to be easier to use, he said.

This is part of a larger problem, he said. Today's mircomputers are too limited in their abilities.

"If you look at larger systems, they have a megabyte of memory and can do a whole chain of functions on a single command," he said. "You have to get more function into the system."

This kind of power, which will allow micros to handle many more applications and which will support a much friendlier system, will not be available until the next generation of microcomputers, he said. He

"...people are learning to use it (microcomputer)... because of what it can do."

does not expect this to be on the market in this decade.

Therefore, he predicts only 2,355,000 microcomputer units with a total retail value of $10,007,000,000, will be sold in the 1980s.

The next decade will be a different matter, he said. By then the new generation of microcomputers will be available. A new generation of people, the children who are learning to use the microcomputer in school today, will be entering the business market as well.

"In the 1950s, when a woman went into her first job from secretarial school, she told her boss she needed an IBM typewriter like the one she learned on," O'Reilly said. "He got her one. The same system will work here."

In O'Reilly's model of the future many microcomputers will stand alone. Therefore, while easy interfacing with larger machines is desirable, he doesn't see it as being of overwhelming significance.

Limited Only by Programs

The microcomputer market is here, according to Tom Arnett, market analyst for Creative Strategies, San Jose, CA. His views are much more optimistic. He believes the market is limited mainly by available programming, but also by the ease, or lack of ease, with which you can use the machine.

"The typical machine has atrocious documentation," he said, "but people are learning to use it anyway because of what it can do."

Arnett defined the microcomputer as a machine costing less than $15,000. He said in 1979 about 350,000 of these were sold with a total retail value of $650 million. By 1984 he predicted sales will grow to $3.8 billion annually.

Tandy, he said, has a large portion of this market although their machines are on its low end. He said they are sixth in number of units shipped but second in profit generated.

"The reason Radio Shack has been so successful is their 7,000 outlets," he said.

He said microcomputers will be used more and more in both network and standalone situations. If IBM does enter the mar-

---

In 1979, Dr. Henry Lee, President of Lee Pharmaceuticals, South El Monte, CA, bought a single TRS-80 Model I so he could learn something about programming.

His main aim was to learn enough to allow him to arbitrate between the head of his data center, an MIT graduate with an MBA from Harvard, and that man's assistant, who holds a Ph.D. in chemical engineering from the California Institute of Technology. The two often disagree, based on different visions of the role the firm's Basic/Four 730 minicomputer should play in the business.

"The question was whether we would have several small computers or one mini in the company," Dr. Lee said.

From that modest initial commitment, Lee Pharmaceuticals' involvement in microcomputers has grown. Today, they own about 45 Model Is, Dr. Lee said. They are used by everyone from scientists to salesmen and for everything from research to direct computer to computer supply ordering.

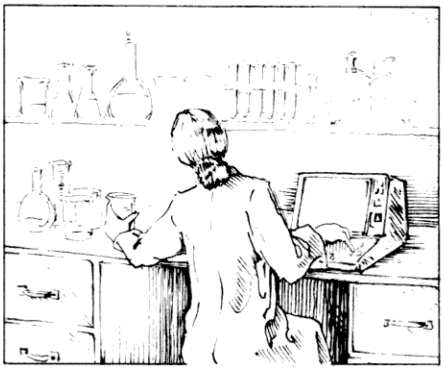
The first thing Dr. Lee tried with his micro was word processing. They had just tried and failed to add a word processing capability to the Basic/Four.

"Once we got into them and modified my first one for upper and lowercase, I discovered what a fine typewriter it was," Dr. Lee said. "I decided if I could buy them cheap enough I would go to them and have my scientists do their own typing."

He hoped this would speed up the process of getting reports out and eliminate the chronic problem of late reports. Dr. Lee bought them in lots of 10.

They solved the report problem, but Dr. Lee found this was just the tip of the iceberg.

"The sales reps and manufacturing people and even our secretaries wanted them he said.



The word processing application, alone, has made them worth their expense, Dr. Lee said.

"It beats distributive processing and terminals on big machines," Lee said. "You don't have to worry about the response time."

At the same time, he said, they "liberated the executive from the tyranny of the secretary."

Dr. Lee and several of his executives have Model Is at home, and others sometimes take them home overnight, so they can work at any time they find convenient.

In fact, he said, the company has eliminated half its 12 secretarial positions by attrition since the Model Is came in.

But word processing is only the start for Lee. Virtually every department of this manufacturer of dental and orthodontic materials, biomedical adhesives and artificial fingernails has found uses for them.

Ten are equipped with modems. These are used to talk with Lee's Ventura County plant, 75 miles away, get chemical information from Lockheed's Dialog time-sharing system and place orders directly to the IBM 370/168 of Van Waters & Rogers, a chemical supplier. Dr. Lee said this has eliminated problems caused by frequent misunderstanding of long chemical names in telephone orders.

The machines are so light that the engineers carry them from building to building and salesmen take Model Is on appointments, Dr. Lee said.

Lee does not buy any hardware from Radio Shack, having found equivalent items at less cost elsewhere. The Model Is are equipped with Matchless disk drives, kit-built expansion interfaces and upper/lowercase adaptors. They use seven Matchless dot matrix printers, six TRS Line Printer IIIs for high-speed printing and eight daisy-wheel letter-quality printers. They also bought five used, modified IBM Selectrics, which turned out to be too slow, and a used Data Trans that has never worked.

On the other hand, Lee depends on Radio Shack for software support. With its 7,000 stores open seven days a week to 9 p.m., Dr. Lee said, Tandy gives him better service than he can get for his big computer, even though he pays a $1,500 monthly retainer to a service company.

"Most of the utilities are great," Dr. Lee said. "The good ones include NEWDOS+, NEWDOS 80, Electric Pencil. We like Xtra Special Delivery (a mail sorting program). We are just getting started with The Creator.

"But at least a third of the chaps are writing their own programs," Dr. Lee said. "That's where the real power comes: You can write programs for your own situation."■

by Bert Latamore
80 Microcomputing *Staff*

"They all agree business software is inadequate."

ket, he said, they will have a great impact. The reason, however, will not be a presumed greater ability to communicate with larger IBM computers. Rather IBM's great marketing ability combined with its reputation will allow it to blanket the market.

"If you were a businessman," he asked, "would you be more likely to buy a computer from IBM or a Radio Shack store?"

Micros have a long-term viability in the business area, he said. Rather than being replaced by minicomputers because businesses want to upgrade their equipment, micros will replace minis in some applications.

"The thing is contagious," he said. "People get hooked on it and discover how useful the thing really is."

Karen Horowitz, market research analyst for Venture Development Corp., Wellesley, MA, sees the business microcomputer market as evolving from the personal computer hobby. She said Venture has not done a study of the business computer market, but she has written a report on personal computers.

"Personal computers were thought of as a personal hobby at one time," she said. "Then people wanted to play games or write programs. Now businesses are finding if they have more software available they will get more use. . . The home market is becoming secondary."

Inadequate Software

They all agree business software is inadequate. Quality has to improve, for one thing.

"You have to understand who your potential purchaser is," Juge said. "I think a lot of the secret is to keep it very simple. Computerize what he is probably doing now with a pencil."

Horowitz said businessmen might be resistant to standardized programs that enforce a standardized way of doing things.

"I think the businessman likes to think he's unique," she said. "However, if you can change a few things you do, you can buy a package for a third of the cost of a specially produced program."

Most of the experts agreed standardized programs fitting basic business uses were needed. Juge said, while needs may vary greatly from one industry subject area to another, needs of different businesses within an area—different print shops, for instance—do not. Therefore, programs could be written for each industry.

O'Reilly disagrees. To earn general acceptance, he says programs must adapt to the businessman, not vice versa.

Canned programs won't be attractive to large numbers of firms, he said, "because every company does things a little differently."

Two years ago, Roy F. Weston Inc., West Chester, PA, was in the market for a way to get computer power to 10 regional offices and a variety of temporary field sites around the country.

Their Univac 90/30, a mainframe computer which they bought in 1977, was doing the job at its main office.

However, the only way for the 400 person firm to make this machine available to engineers in the field was to provide them with terminals and use the telephone lines.

Instead, according to Donald R. Milner, manager of computer services for Weston, they decided to try the TRS-80 as a less expensive way of doing the job.

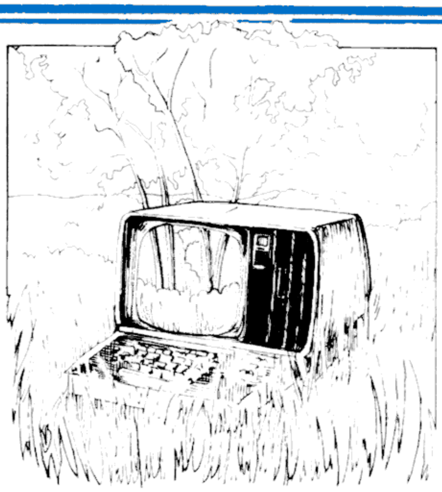
As a result of this decision, four Model Is of various configurations and six Model IIs, each with a 64K disk drive and one diskette, are operating in a very demanding environment. While problems have appeared from time to time, Milner said the experience has been generally positive.

Weston's micros primarily handle engineering applications. They do operate as word processors at times, but Milner said they are not used for any other normal business functions.

Weston does a lot of environmental engineering. This involves constant air, groundwater and soil sampling.

"Our initial application was to record instrumentation readings (from sampling test instruments)," Milner said. "We've developed a field data management system for sample logging and reporting."

Normally raw data is hand-fed into the machine. However, Weston does have one



automatic setup in which the measuring instruments record their data as audio signals on tape that can be fed directly into one of their Model Is.

From there it was a natural step to have them perform data reduction and other preliminary data preparation, Milner said. They have moved beyond this to perform sanitary sewer system evaluating and laboratory management on the Model IIs.

"I'm sure they are being used by some of the people to do exotic formulae and so forth," Milner said.

In fact, he said, sometimes they have pushed the micros beyond their limits. But then, he said, they have sometimes exceeded the limits of their 393K Univac and had to go to a time-sharing service to get their problem solved.

They have had to write most of their own programs because they haven't been able to buy programs for their applications, Milner said. Their main problem in this area, he said, is "human engineering" the system. That mainly involves using the KISS Principle (Keep It Simple, Stupid).

"There are applications that are built to be used by programmers," he said. "You can't give one of those systems to a layman and expect it to work."

On the other hand, he said, the engineers have no trouble adapting to the microcomputers, mainly because they learn to use computers in school.

"An engineer cannot go through school without being exposed to a large amount of on-line computing," Milner said.

The micros face a different kind of challenge in interfacing with a variety of other machines. Wesson has already written a program for interfacing the Model IIs with the Univac. The firm also owns a Tektronix 4045 with two dual-diskette drives, a four-color flatbed plotter and a Wang WP30 dedicated word processor. They plan to tie them together into an integrated system.

Milner said he was basically happy with the Model II as it is. The main change he would like to see in it is a price reduction.

"The situation we will probably continue to run into is matching the micro to the application," Milner said. "We have had the situation where we have asked the micro to do too much and it failed, but that was not the fault of the micro, it was the fault of us. They have their place."

by Bert Latamore
80 Microcomputing *Staff*

What has nine lives, three forms, multiple faces and a price tag that almost disappears?
OKIDATA
The Magical Microline 80 Printer
It's magic! Well, almost. The Microline 80 will run all day at 80 cps with no duty cycle limitations. The head is warranted for 200,000,000 characters. That translates to over nine years on your TRS-80,™ APPLE® or other small computer.
Want to change forms? The magical Microline 80 is three printers disguised as one. There is a whisper-quiet rubber platen for cut sheets and roll paper, pins on nine inch centers for pin feed stock and optional snap-on tractors that adjust to suit all your other forms. The Microline 80 also saves paper by letting you tear off as close as one inch from the last print line.
Want to change your image? The magical Microline 80 really does tricks. It prints upper and lower case, condensed and double width characters *and* block graphics for charts, graphs and diagrams.
The Microline 80 is not a toy. With two motors, a rugged cast aluminum base and a head you never have to throw away, the Microline 80 is built to handle the most demanding business applications.
Which brings us to the biggest magic of all, the price tag, the one that almost disappears. If we're not the lowest, we are so close that it doesn't matter. There are stocking Microline distributors throughout the country. Call or write today for the name of the one near you and the price of the Magical Microline 80.
OKIDATA
Okidata Corporation, 111 Gaither Drive, Mount Laurel, New Jersey 08054 609-235-2600
Okidata is a subsidiary of Oki Electric Industry Company, Ltd.
245

"These market experts see the business microcomputer as still in its infancy."

He advocated programs with the flexibility to adapt to variations in business activities. For instance, he said, Radio Shack has a program which creates a matrix to hold data, programs, etc., that the user chooses to enter. In his report he predicts business software will generate only $8,192,000,000 in gross sales in this decade.

In general, these market experts see the business microcomputer as still in its infancy. Eventually it will gain an important place in business. The questions that remain to be answered are how quickly this will happen and just what their place will be. ■

One of the biggest headaches in retailing is inventory. It must be performed continuously or you may find yourself in the embarrassing and costly position of running out of your most popular items. But keeping a running inventory up to date can involve hours of work daily. Even at that you will make mistakes and have problems; no system is perfect.

This was exactly the situation at Bond Discount Wine & Liquor in New York, where for 25 years two generations of the Schneider family have labored, devoting three man-hours a day to maintaining a perpetual inventory.

Enter the TRS-80.

"One day my father read about them in *The Wall Street Journal*," Paul Shneider said. "He went right out and bought one." He wanted a machine to do his inventory. What he ended up with was that and more.

For a start it replaced the cash register. When a sale is "rung up," it automatically adds the tax. It keeps complete unit pricing information on all items in the store. If the customer could get a better buy from a different size of the item he selected, the computer shows this on its screen where the customer can see it.

Bond was the first liquor store in New York state to offer unit pricing. The story was carried on two New York television stations.

The micro adds the dollar amounts from each sale to the appropriate bookkeeping categories—it can accept 15—thereby taking care of another major problem, the daily bookkeeping postings.

At the same time, the machine deducts each item purchased from two lists: one a record of what is out on the shelves, and the other a total inventory. The computer can list any items which have fallen below a predetermined minimum on the shelves so the stock boy will know what to bring up from the basement.

It will also list any items which have fallen below a predetermined minimum in total inventory and should be reordered.

Using a TRS-80 Model I with two disk drives, the system can handle 2,500 separate stock items, with minimums set separately for each one, Schneider said. However, double-density disk drives would increase the system's capacity to 5,000—10,000 items.

The system goes another step, Schneider said. It keeps records of the total monthly sales of each item for the last 15 months. It then uses this information to calculate trends and seasonal fluctuations to project sales on each item for the next three months.

This item, alone, has saved Bond several times the cost of the system and the professionally written program, Schneider said, because it virtually eliminates costly over and under-stocking situations.

Schneider said the system is simple to use and requires no training. In fact, he said, on Christmas Eve, which is the busiest day of the year for package stores, his wife, who has no previous experience with micros, came in and ran the checkout all day with no problems.

Schneider said he finds the microcomputer totally adequate.

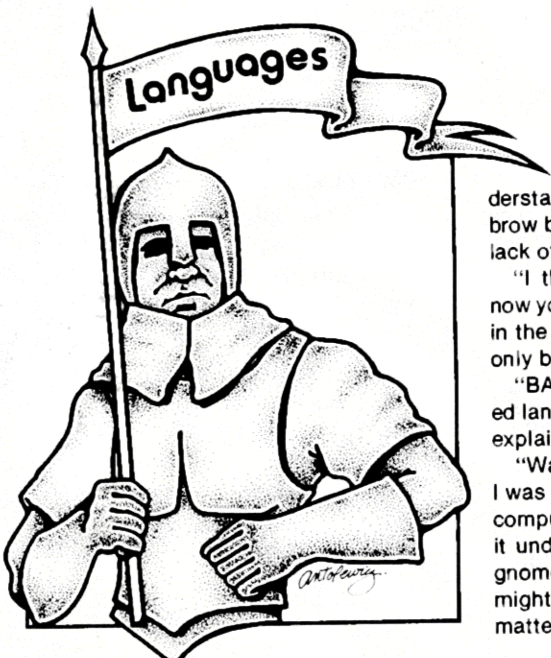
"It's a matter of making the machine do what you want it to do," he said.

Schneider is selling his program which, he said, will run on either a Model I or Model III. It is available through Accurate Business Computers, 800 Preston Road, East Meadow, NY, 11554. ■

by Bert Latamore
80 Microcomputing *Staff*

In search of the pathway to computer enlightenment.

Language Quest '81



by G. Michael Vose
80 Microcomputing *Staff*

I knew that I would have to join the twentieth century sooner or later. Besides, I'm a college graduate and I can even handle myself in a disco. So I went down to the local computer store and started to browse. I sat down at a computer and typed HELLO on its keyboard and pressed this big white button. The screen showed ?SN ERROR and I figured I must have committed some kind of mortal sin. I was just about to try to sneak out the door when a salesperson in a snappy three-piece suit came up and asked if he could help me. I told him I had tried to talk to the computer but that things were not going well.

"That's because the computer only understands BASIC," he explained and my brow began to furrow in the early stages of lack of comprehension.

"I thought computers were brainy and now you tell me to stick to basics?" The guy in the three-piece suit gave me what could only be described as a tolerant smile.

"BASIC is a high-level, procedure-oriented language like FORTRAN or COBOL," he explained further.

"Wait a minute," I retorted. "Do I look like I was born yesterday? First you tell me the computer needs BASIC and now you claim it understands the language of a German gnome called a kobold." I began to think I might need to look more deeply into this matter of communicating with computers.

What Is a Computer Language?

My search to discover a way to communicate with the computer led me to some fascinating discoveries. First I learned that a computer language is simply a set of rules, representations and conventions used to transmit and convey information. Computer languages are classified as low-level or high-level. But low-level does not necessarily mean simple or easy.

As it turns out, the computer actually understands one and only one language. That language is called machine language. Machine language is a low-level language simply because it is at the level at which the computer can directly recognize and manipulate numbers. All other languages must be converted into machine language to be understood by the computer.

Computers can only understand numbers represented by groups of 1s and 0s called binary digits. Groups of these digits are called a binary code. Machine language is essentially binary code modified to allow you to enter and manipulate numbers using the more standard numeric form of decimal, octal or hexadecimal notation. A number in machine language will stand for a specific instruction (or memory address) that the computer can recognize and execute. This number is determined by the electronic architecture, or design, of the microprocessor. In other words, the machine was built to handle these numbers in a certain way. This process is accomplished using electronic devices, often called gates, which can only be opened or closed. These two states can represent a one or a zero, the digits of binary code.

One step up from machine language is assembly language. This language is also a low-level language and differs from machine language only in its code. Assembly language allows you to substitute certain mnemonics for numbers. Since these mnemonics are not directly recognizable by the

COMPILED	INTERPRETED
COBOL	BASIC
FORTRAN	PILOT
ALGOL	LOGO

Table 1

Vista

Model V1000

. . . Looking Out For You.

Eight Inch Floppy Disk Drive Subsystem Model V1000

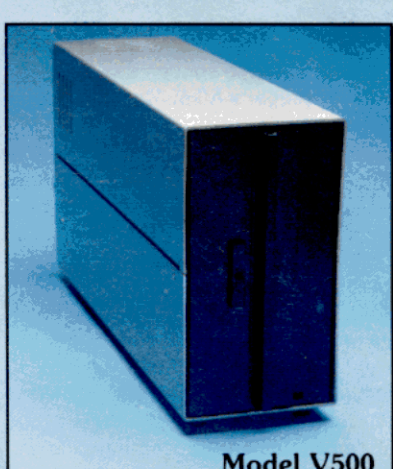
The V1000, Vista's sophisticated new disk drive subsystem, sets new standards for ease of access and use. Its innovative design permits disk drives to be mounted or removed quickly and easily for system reconfiguration or servicing.

Features:

- Deluxe chassis with internal slide allows easy access.
- Storage capacity from 250 kilobytes to 2.5 megabytes.
- Desk or rack mountable.
- Accomodates both single-sided and double-sided drives.
- Industrial quality cabinet with die cast front bezel.
- Drives pull out for easy service and maintenance.

Prices:

Cabinet with (2) single sided drives w/power supply	$1595.00
Cabinet with (2) double sided drives w/power supply	$2295.00
Cabinet w/power supply	$ 495.00



Model V500

Vista's Line of High Performance, High Reliability Products also Includes these Advanced Components

Single Eight-Inch Drive Subsystem Model V500

Features:

- Up to 1.2 megabyte on-line storage capacity
- Compatible with Radio Shack Model II, Apple II® and most popular S100 based computer systems.
- Highly reliable power supply provides ±5% regulation and over-voltage protection.

Prices: Starting as low as $900.00

Single Minifloppy Disk Drive Subsystem Model V80/800/8000

Features:

- From 102K to 816K bytes on a single drive.
- Fastest track-to-track access time on any minifloppy drive — 5 milliseconds.
- Dual-head technology on the V800/8000 allows on-line use of both sides of diskette

Prices: Starting as low as $395.00

Model V80/800/8000

65 **Vista Computer Company** 1317 E. Edinger Avenue • Santa Ana, CA 92705 • (714) 953-0523

Dealer Inquiries Call Toll Free (800) 854-8017

"The question may have occurred to you, 'which language is the best one?' "

computer, they must be translated by an assembler. An assembler is a program that takes these non-machine language instructions and converts them into the numeric code understood by the computer.

Pseudo-Language For the Human Pseudo-Brain

Since low-level does not necessarily mean simple or easy when you are talking about computer languages, I assumed that a language called BASIC must be a low-level language. As is often the case when one makes assumptions, this turned out to be wrong. BASIC is a high-level language.

High-level languages can be understood more readily by you and I. They are symbolic languages which use recognizable English words, groups of numbers, special words and standard mathematical notation. It is important to remember that these alphanumeric symbols are *not* the internal language of the computer. The symbolic code of high-level languages must be either compiled, or converted, into machine language, or interpreted for the computer by a separate component called an interpreter.

There are several major differences between compiled and interpreted languages. Operationally, the difference between compiling and interpreting a source language is substantial. Each technique requires a separate component, either a compiler or an interpreter.

A compiler is a program that converts source language code symbols into executable machine language code. This is done after the program has been written, or coded, by the programmer. The compiled program can then be run. This process is analogous to that of a book being translated from French to English.

An interpreter is a component of the computer's permanent memory that actually interprets each character of source language code for the computer's brain (the microprocessor) as the program is being run. This process is analogous to the speech translators at the United Nations who translate words as they are being spoken.

Compiled and interpreted programs each have their advantages and disadvantages:

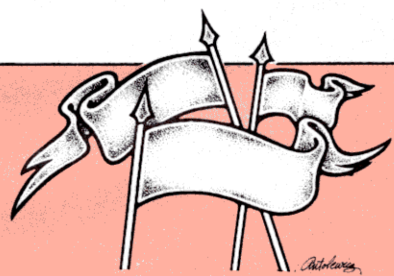
- Compiled programs cannot be executed until they are compiled, therefore they cannot be tested until they are written and converted. This compilation can often take a substantial amount of time.
- Compiled programs execute faster than interpreted programs because they eliminate the interpretive step and because they are, once compiled, machine language.
- Interpreted programs are interactive. This means that the programmer can experiment with different commands and instructions and discover almost immediately if that command or instruction will work. This is because the command can be executed immediately, without having to be compiled.

Some examples of compiled and interpreted languages are shown in Table 1.

Problems, Procedures— Procedures, Problems

The language picture was beginning to get a little clearer when I ran into yet another classification. This time I discovered that languages are further classified as either problem-oriented languages, or procedure-oriented languages. These are pretty fancy-sounding categories, but they simply mean that languages are designed to allow us to solve specific classes of problems, or wide classes of problems. In the latter case, we have to develop a procedure to solve our problem.

Problem-oriented langauges are designed to solve a specific class of problems. For example, the language whose acronym is ICES (Integrated Civil Engineering System) is designed to solve specific problems in civil engineering. The engineer has only to input certain types of data and the program computes the specifics for building a solid concrete wall. The engineer cannot use the program to balance his



Language Primer for the Novice

Compiler—A computer program that produces a machine language program from a source program that is usually written in a high-level language by a computer user. The compiler is capable of replacing single source program statements with a series of machine language instructions or with a subroutine.

Compile—To prepare a machine language program (or a program expressed in symbolic coding) from a program written in another high-level programming language, such as FORTRAN, PL/1 or COBOL.

Metacompiler—A compiler for a language that is used primarily for writing compilers, usually syntax-oriented compilers. A special purpose metacompiler language is not very useful for writing general programs.

Compiler Language—A source language that uses a compiler to translate the language statements into an object language.

ALGOL—An acronym for ALGOrithmic Language, an international high-level programming language designed for scientific programming. ALGOL is used primarily in Europe.

APL—A mathematically-structured language developed by IBM Corporation. In its simplest mode, APL performs the functions of an intelligent calculator. The power of the language is demonstrated by its extended single operators which allow a user to directly perform such things as taking the inverse of a matrix, or solving a set of linear equations. APL is a powerful tool for the scientist or engineer.

Assembly Language—A programming language which allows a computer user to write a program using mnemonics instead of numeric instructions. It is a low-level symbolic programming language which closely resembles machine code language.

Assembler—A computer program that takes non-machine language instructions prepared by a computer user and converts them into a form (binary) that may be used by the computer.

ATOLL—A special language used by NASA on the Apollo space missions.

BASIC—Beginner's All-purpose Symbolic Instruction Code.

COBOL—COmmon Business Oriented Language. Every COBOL source program has four divisions, whose names and functions are: Identification division, which identifies the source program and the output of a compiliation; environment division, which specifies those aspects of a data processing problem that are dependent upon the physical characteristics of a particular computer; data division, which describes the data the object program as output; and procedure division, which specifies the procedures to be per-

"The answer to that question will differ depending on who you ask."

checkbook. At the same time, the engineer does not have to know anything about programming to obtain the results he needs. The procedures for solving his problem are built into the program language.

Other problem-oriented languages include APT, RPG, COGO, GPSS, STRESS and others. One specialized group of problem-oriented languages is the list-processing group which includes LISP and SNOBOL. These languages are designed specifically to process non-numeric data, such as lists of names and addresses.

Procedure-oriented languages are more versatile. They allow the programmer to write routines to solve any problem that he can define and subsequently devise a solution for. FORTRAN is the grandfather of this class of languages and is still one of the most powerful of the procedure-oriented languages. Designed to perform mathematical, scientific and engineering computations, FORTRAN exists today in five or six versions. Other procedure-oriented languages include COBOL, BASIC, Pascal and PL/1.

Will We Ever Use English?

The question may have occurred to you, "Which language is the best one?" The answer to that question will differ depending on who you ask. Programmers and manufacturers will argue, with merit, that one languages is better than another. There can be different versions of the same language, very often to accomodate the unique design of a particular manufacturer's machine.

To some degree, determination of the best language will depend on what application is planned. There probably shouldn't be one universal computer language in much the same way that there probably shouldn't be one universal automobile manufacturer. No one group can satisfy the needs of a large population.

One thing can be said with certainty. While the English language is too complex to adapt for use as a computer language, there will be other languages developed. Many will be simple, user-oriented languages while others will be used to solve specific problems. All will allow us to communicate more effectively with this powerful machine called the computer. ■

formed by the object program by means of English-like statements.

COGO—COordinate GeOmetry, a language used by engineers.

Cross Compiling/Assembling—A technique where one uses a minicomputer, large-scale computer, or time-sharing service to write and debug programs for subsequent use on microcomputers.

DDL—Data Description Language, a language for declaring data structures in a data base.

Firmware—Software that is hard-wired into a computer, usually as read-only memory (ROM). Changes can only be made by changing the chips.

FORTRAN—FORmula TRANslator, a high-level programming language used to perform mathematical, scientific and engineering computations. There are two versions, FORTRAN and Basic FORTRAN.

High-Level Language—A programming language oriented toward the problem to be solved or the procedures to be used.

Interpreter—A computer program that translates each source language statement into a sequence of machine instructions and then executes these instructions before translating the next source language statement.

JOVIAL—Jules' Own Version of the International Algorithmic Language, a scientific language used by the U.S. Air Force.

Language—A set of rules, representations, and conventions used to convey information.

LOGO—A language suited to and used by grammar and junior high students. Developed at MIT by Seymour Papert and staff.

Low-Level Language—A machine-dependent programming language translated by an assembler into instructions and data formats for a given machine. Same as assembly language.

Machine Code—An operation code that a machine is designed to recognize.

Machine Instruction—An instruction that a computer can directly recognize and execute.

Machine Language—The basic language of a computer. Programs written in machine language require no further interpretation by the computer.

Metalanguage—A language which is used to describe a language.

Mnemonic—Pertaining to a technique used to aid human memory. A word or name which is easy to remember.

Mnemonic Code—An easy-to-remember assembly language code, for example, a code that uses an abbreviation such as MPY for multiply.

Native Language—A language peculiar to the machines of one manufacturer.

Object Code—Output from a compiler or assembler which is itself executable machine code or is suitable for processing to produce executable machine code.

PL/1—A high-level programming language designed to process both scientific and business applications. It contains many of the best features of FORTRAN, COBOL, ALGOL and other languages as well as a number of facilities not available in previous languages.

Problem-Oriented Language—A high-level, machine-independent programming language designed for the convenient expression of procedures used in the solution of a wide class of problems, e.g., FORTRAN, COBOL, PL/1, etc.

Programming Language—A language used to express computer programs.

Symbolic Language—A pseudolanguage made up of letters, characters and numbers which are *not* the internal language of the computer system.

RPG—Report Program Generator, a popular business-oriented programming language. The language will allow the user to program many business operations as well as generate reports. A fairly simple RPG program can perform a rather sophisticated business task. It is relatively easy to learn.

SNOBOL—StriNg Oriented SymBOLic Language, a string manipulation programming language used primarily in language translation, program compilation and combinatorial problems. The language stresses the ability to manipulate symbolic rather than numeric data.

Source Program—A computer program written in a source language such as BASIC, COBOL, etc. It is converted to the machine code object program by a special processing program, a compiler, interpreter or assembler.

Syntax—The grammatical and structural rules of a language. All assembly and high-level programming languages possess a formal syntax.

WATFOR—A version of FORTRAN.

At 5½, she's reading and thinking at a 4th grade level.

Is she a prodigy? Not at all! When learning is fun, your child can rapidly increase reading comprehension and reasoning power beyond current age level.

Now you can advance your child's educational level—without coaxing or hiring an expensive tutor. Everyone knows the best student-teacher ratio is 1:1. In modern classrooms, this is impossible. But in your own home, it's as easy as C.A.I. (Computer-Aided Instruction). Without the distractions of a crowded classroom, students can progress, without pressure, at their own pace.

Computers are fun, and once the fun is introduced, studying becomes a pleasure—instead of a chore. The ability to interact with the computer will motivate your child to learn much more than from a book that cannot respond—a book that is too easily closed. These Instant Software programs will guide and teach each lesson with the infinite patience only a computer can provide. Each of the twelve packages have been written and edited by professional educators who know what it takes to make a child want to learn. You can introduce your child to the world of learning by computer with Instant Software's Educational Software Series. All 12 packages are available now.

ARCHIMEDES' APPRENTICE

Archimedes Apprentice is a geometry package that will teach your child the formulas used to find the volume of any solid object, including parallelopipeds (cubes and rectangular solids), prisms, pyramids, cylinders, cones and spheres. It will even quiz the student on how well the lesson was learned. Give your child the advantage of learning these concepts at home, before being faced with them in a classroom. **Order No. 0092R $9.95**

TYPING TEACHER

A complete seven-part package that guides you from familiarization of the keyboard through typing words and phrases to mastery of touch-typing. Your video monitor becomes a bottomless page for typing practice, and your own private TYPING TEACHER, ready to teach when you're ready to learn. **Order No. 0099R $9.95**

TEACHER'S AIDE

Now you can have the benefit of Computer Assisted Instruction (CAI) in your own home. The Teacher's Aide program allows you to create tailor-made lessons for your child. The features of this program include the ability to review material before taking the lesson, a provision for hints to help answer questions and graphic displays as a reward for answering all questions correctly. Once you've created a lesson, you can save it and create an entire sequence of lessons.

The Teacher's Aide package is perfect for parents, teachers, and students who need the unlimited patience only a computer can provide. **Order No. 0214RD (disk-based) $39.95**

VIDEO SPEED READING TRAINER

With this package, students can increase their reading speed and comprehension. How? With practice! This three part program will flash characters, words and phrases onto the monitor screen. You must then type what you saw on the keyboard and enter it. You can begin at a relatively slow speed. As your speed and comprehension increase, the computer will automatically advance your speed. **Order No. 0100R $9.95**

IQ TEST

Are you smart enough to buy this package? IQ TEST will administer and score an intelligence test in 30 minutes. There are three equivalent tests, each consisting of 35 questions, designed to test your general knowledge and problem-solving abilities. Most of us claim a "touch of genius"—here's your chance to prove it! **Order No. 0157R $9.95**

VOCABULARY BUILDER

One of the most valuable tools your child can possess is a comprehensive vocabulary. With Instant Software's VOCABULARY BUILDER package, you can help your child develop a better understanding of the English language—and the fun is in the learning! The crossword puzzle format of these programs will hold the attention of the most reluctant student. The first program in this package is "Polonius," offering 140 crossword puzzles. The second is "VeeBeeGee," a program that provides the letters and allows the student to construct a puzzle on the game board. The VOCABULARY BUILDER package is educational entertainment for all ages—and it's never at a loss for words. **Order No. 0198R $9.95**

Prices subject to change without notice.

Instant Software™

PETERBOROUGH, N.H. 03458
603-924-7296

See List of Advertisers on page 306

Here's how to optimize BASIC and magnify computing power by calling in a Macro.

A Macro Processor For BASIC—Part I

The opening segment of this article, the first of a series, is part of a letter that accompanied Alan Olmstead's manuscript to our offices. The editors felt that his thoughts were sufficiently perceptive to preface his article, particularly in our examination of languages.—Eds.

To perceive the microcomputer as a kind of calculator with delusions of grandeur is to miss the central point of its potential for changing the way mankind lives. The microcomputer is the first form of artificial intelligence applied on a large scale to hundreds of thousands of individuals. Furthermore, it has been applied under conditions of virtual natural selection—only those people capable of realizing what it is come forward to acquire one.

As demonstrated so admirably in the lifetime work of J. J. Bachoffen during the last century, intelligence—natural or artificial—is inconceivable without language, if a measure of intelligence is the ability to classify real things into abstract, invented categories. Thus, the importance of the microcomputer will not be found in mathematical applications, but in linguistic ones. Aside from theoretical and technical applications, there is essentially little need for improving the microcomputer's capacity for solving computational problems. But in terms of the existence and use of language—and its corollary, intelligence—we are little more than infants.

The only computer languages available to us are dinosaurian manifestations of essentially electrical, not electronic, logic. These ponderous beasts all function according to the elementary formula "this input equals that output." A specific and limited command repertoire is first recognized, then equated to a modestly variable form of output. If the repertoire lacks an exactly appropriate command syntax, we must have an alternative, leaving the compiled language entirely and switching into an assembly language subroutine.

The reason for such a limited concept of language (I would go so far as to question that they are even languages) is not lack of creativity. The languages serve the needs of their developers, and their developers are interested primarily in selling computer hardware. Thus, we saw Radio Shack introduce the Model II before correcting the built-in deficiencies of the Model II, and IBM deliberately sabotaging the Model 5110 to keep it out of the System 34 marketplace. As long as language remains in the private domain of large corporations the microcomputer customer will continue in his role as the expedient servant of next year's fiscal planning.

As the translation of the Bible contributed to the splintering of the Church of Rome when its promulgation was intended to preserve unity, so the rigid structure of predetermined languages must prove eventually counterproductive. Application needs change, and those languages which do not keep pace will doom the machines on which they run. So-called "dead" languages die because they no longer serve a purpose. Alternatively, newly evolved languages remain alive through their continuous use. The language of artificial intelligence must be capable of doing likewise. Like a beautiful bird, language must be set free before it will sing its best.
language must be set free before it will sing its best.

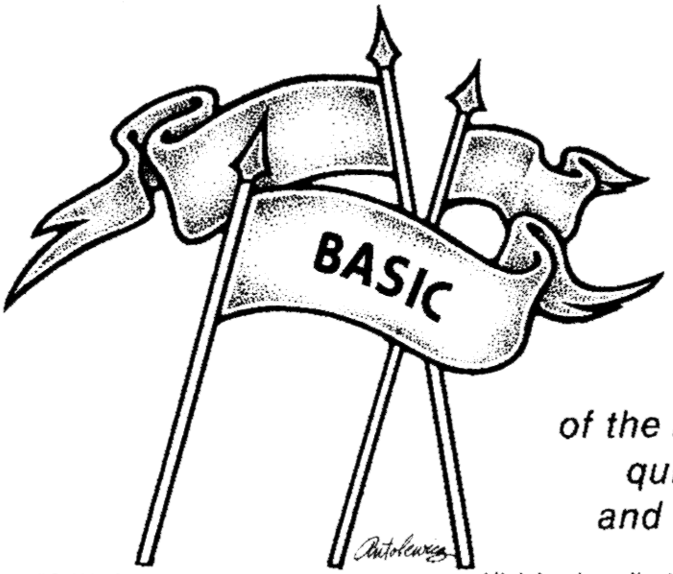
Language consists of two parts, form and content. The form, its parts of speech—nouns, verbs, prepositions, etc.—has remained nearly unchanged for centuries. But the content of language, the actual words and their meanings, change continuously through time, so rapidly, in fact, that it is commonplace for parents to complain that they do not understand their children.

Achieving such independence for the individual user is the whole point of MetaBASIC. It is not merely a simpler method of moving from a high level language like BASIC into assembled code, or even merely to expand the existing command repertoire. It is rather to give the user the actual ability to make up his own command words and sentences according to his needs—even if he is the only user in the world with such a need. I have provided a general repertoire of MetaBASIC command sentences to serve very common needs, such as:

```
LOAD STR A$ (FROM) B$
PLOT FROM(X1,Y1) TO(X2,Y2) WIDTH(W)
```

However, these command sentences are merely electronic babytalk. It is my sincere hope that users will invent their own command sentences with such enthusiasm that in a matter of a few years even I, the originator, will not recognize the language. Best of all, they will do so governed only by their own needs and desires, without having to ask my permission or obtain my cooperation in the effort.

Just as we cannot foretell the style of the next major poet or novelist, none of us is capable of imagining what will become of this industry when 50,000 intelligent young people are turned loose to build anything they are capable of imagining. ■



"BASIC might be called the guitar of the rock 'n' roll computer set, because it is quite willing to respond with good results and requires. . .only modest talent to use."

MACROEDT
Model I
$50
Financial Engineering Systems
J. Olmstead
Phoenix, AZ

J. Alan Olmstead
3843 West St. Moritz Lane
Phoenix, AZ 85023

At the end of 1980, the microcomputer industry came to an early maturity in many areas of its development. Witness the emergence of BASIC as the most common programming language in the world. BASIC was clearly designed with straightforward syntax in mind by which computer novices might understand the workings of a computer in a short period of time. In this regard, the language is a total success, as there never was an easier language to learn and use.

BASIC might be called the guitar of the rock 'n' roll computer set, because it is quite willing to respond with good results and requires practically no formal education and only modest talent to use. But in truth, BASIC is an astonishingly wasteful language which, in some environments, simply cannot be used.

Because of certain applications like communications (600 baud is the practical limit under interpreted BASIC) and business problems involving reiterative mathematical treatments, the BASIC cross-compiler emerged. The cross-compiler was touted as the cure of BASIC's ills as a run-time device. However, the greatest deficiency among compilers is their uniform failure to recognize that BASIC, as it is used, is a totally different language from its high-level predecessors, such as COBOL and FORTRAN. Because of this difference, it requires additional features.

High-level applications programmers of the sixties and seventies were individual members of large teams which were backed up by small battalions of assembly language specialists. Individual high-level programmers concentrated on developing applications, while assembly language teams concentrated on optimizing the whole computer system. When an application method became too expensive for these individual high-level programmers, the team developed new tools to ease their burden. These tools are brought into operation within the application programs by means of Call commands.

Contemporary BASIC programmers are one-person computer departments. Even among the small percentage which are able to program in assembly language, who has the time? Assembly language programming requires ten times more programming hours than BASIC. The BASIC programmer is committed only to accomplishing his tasks using BASIC—aside from the few utilities which appear on the market from time to time. Yet the programmer's demand is for ever more speed.

The first part of this series will emphasize new tools and techniques for optimizing BASIC beginning with the BASIC macro processor.

What Is a Macro?

The definition of a macro varies from manufacturer to manufacturer, but for our purposes, the following definitions will suffice:

- A macro is a program module of varying size, which cannot execute by itself for two reasons: It is incomplete, being written to perform a specific kind of task; and it contains specialization commands which, when acted upon, change the form of the macro into a specific kind of sub-module which can then be executed.
- A macro is always in source language which resides in a separate library file from any given application program into which it will later be incorporated.
- A macro is capable of including, omitting and changing the form of its source code based upon the instructions received from the applications programmer at the time of the macro call.
- A macro is called into the user's program during the program writing phase, before interpretation, compilation or assembly.

The macro processor is an interpreter program that calls the macro from its library into the new applications module, shaping it according to the user's instructions contained in the macro call command syntax (see Fig. 1).

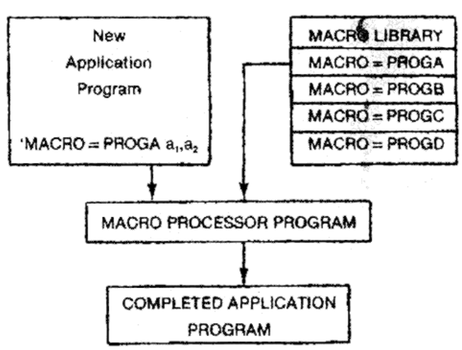


Fig. 1.

The specialization of a called macro takes place when two programming elements interact in the macro processor program. First, there are arguments, or parameters as they are sometimes called, appended to the end of the macro Call command. Second, there are various tests coded directly into the lines of BASIC within the macro library module (in this case, PROGA) which tell the macro processor what to do if the call arguments are: present, absent, relational to a given integer value, or in AND/OR relationships between two of the arguments.

"The macro processor is an interpreter program that calls the macro from its library into the new applications module."

To more clearly illustrate this, examine the generalized code illustrated in Example 1. It appears to be BASIC code, but it cannot be executed either in interpreter or compiler form. At line 1000, a remark line, appears the pseudo-command word 'MACRO=, which signals to the macro library processor that a library module is being defined. The definition continues until an 'ENDMAC pseudo-command word is encountered (at line 1100). The name of the library module follows the pseudo-command. It does not use any form of a BASIC reserved word, which might present a problem to some interpreters and compilers, even though it appears inside a comment line.

Finally, a series of arguments appear, each separated by a comma. The arguments are not legal BASIC values, but are two numbers preceded by the flag symbol &&.

The arguments could continue up to ninety-nine—from 01 to 99. Argument 00 is used by the library processor as internal work space. The arguments are in ascending order only for convenience. The same is true of sequence, which may contain skips or gaps in which argument numbers are used. Line 1000 indicates to the library processor the outer dimensions of the specialization problem about to be undertaken.

Throughout the macro appearing in Example 1, there are true BASIC statements, such as line 1010, and other almost-BASIC statements, like line 1020. Line 1020 is a standard Radio Shack TRSDOS Open command, but in generalized form. It will be specialized into a true command when the user presents the actual values of arguments one, two and three to the library processor program. During specialization, the true arguments presented to the library processor are substituted into their physical counterpart locations (one into one, two into two, etc.), and the resulting BASIC code is then included into the user's program in its

```
1000 'MACRO = DSKOPN &&01,&&02,&&03,&&04,&&05,&&06
1010 ON ERROR GOTO 1050
1020 OPEN &&01,&&02,&&03
1030 ON ERROR GOTO &&04
1040 RETURN
1050 ON ERROR GOTO &&04
1060 'BOOL &&05 + &&06
1070 &&05 = &&06
1080 'ENDB
1090 GOTO 1040
1100 'ENDMAC
```

Example 1. Example of a Type of Generalized Code

```
100 'PROGRAM MODULE "TESTPROG"
110 CLS:CLEAR1000:DIM A$,B$,C$:ON ERROR GOTO 260
120 INPUT "ENTER DISK FILE NAME: ";A$
130 INPUT "ENTER DRIVE NUMBER: ";B$
140 A$ = A$ + ":"B$
150 B$ = "R":C$ = " "
160 GOSUB 220:IF C$ = " "THEN 170 ELSE 240
170 FIELD1,64 AS X1$,64 AS X2$,64 AS X3$, 64 AS X4$
180 GET1,1
190 PRINT X1$;X2$;X3$;X4$
200 CLOSE1
210 END
220 'MACRO = DSKOPN B$,1,A$
230 '-260,C$,"CAN'T OPEN FILE "
240 PRINT C$;A$
250 GOTO 200
260 PRINT "ERROR ";ERR;" AT LINE ";ERL
270 STOP
```

Example 2. User's Program Before Specialization

Powerful Utilities To Get More Out Of Programming your TRS-80*...

BOSS 2.2 Back to work everybody the BOSS is BACK! A BASIC Operated Single Stepper -monitor your basic programs, line by line, while they are executing!	c	$	24.95
	d		29.95
VFU Visual File Utility - Allows you to: EXECUTE or PURGE a basic or machine language file; COPY a file from one disk drive to another or HARD copy your directory, ALL with single key entries	c	$	19.95
	d		24.95
ULTRAZAP Disk modification utility - Read, Write, Copy, Purge or ZAP! any byte or sector on your diskette. The same utility that comes with ULTRADOS	c	$	19.95
	d		24.95
SPOOL Parallel port, print spooler - Define spooler buffer size, lines per page, and characters per line, allows continued use of CPU during the printing process. A True RAM Spooler!	c	$	19.95
	d		24.95
GEDIT Global Editor - Allows the user to search and replace any keyword command, variable or string with incredible speed, even pack strings for high speed graphics	c	$	19.95
	d		24.95
RENUM-80 Program Renumberer - Tape or Disk operation, ultra fast, accurate operation	c	$	5.95
Lower Case Driver - Will work with all Model I lower case conversions. Shortest available, only 68 bytes	c	$	5.95
Epson Driver - For the Epson MX-80 printer, allows the use of all the functions available on today's most popular printer - Works with the Model I or Model III, cassette or disk	c	$	5.95

OR,

Get most of these utilities and much, much more with...

ULTRADOS!

Now only $ 94.95!

LEVEL IV PRODUCTS ✓14

For Orders Only 1(800) 521-3305
For Information (313) 525-6200
32461 Schoolcraft Rd.,
Livonia, Michigan 48150

VISA MasterCard

* **A trademark of Tandy Corporation**

TRS-80 is Trademark of Tandy Corp.

✓See List of Advertisers on page 306

"Anything and everything appearing between two commas will be considered to be the value of the argument, including both single and double quotes."

new, interpretable/compilable form.

The programmer's new program gives the specialization commands to the library processor program. Example 2 shows a program, which opens up a named file and prints the first record on the screen. Beginning at line 220, and including line 230, the sample macro illustrated in Example 1 is called by the pseudo-command 'MACRO =, in the exact same form as previously seen. This time, however, instead of following the pseudo-command with dummy arguments, the actual proper values are given. The proper values must be in ascending sequential order, and if any arguments are optionally omitted, the parameter slot must be preserved. This is done by inserting the comma which would have followed the argument if it were present.

Lengthy Arguments

In the case where the arguments are too lengthy to fit on a single line, they may be continued in sequence to the next line with the continuation pseudo-command word '-. (See line 230 in Example 1.) Of further interest is the delimiter character comma. Anything and everything appearing between two commas will be considered to be the value of the argument, including both single and double quotes. However, the one value which may not appear between two commas is another comma.

Note also that the dummy argument &&01 (and others) require only four character positions, while literal arguments (with actual proper values) that are substituted into that place may take either fewer or more character positions. This is of no consequence; the specialized output code will be expanded within the same line, not to exceed a total of 128 character positions.

The reason why the dummy argument is required to take four positions, even if the leading zero must be inserted (&&01), is for flexibility. For example, if the generalized code directed the last two numbers of the lines to be used as a GOTO, it would look as follows:

```
1000 ON &&01 GOTO &&0210,&&0220,&&0230
```

When specialized, this line will test the value of the numeric variable specified as argument one and conditionally jump to one of three lines whose numbers are relative to a base line number like 5000. The desired specialization would be:

```
1000 ON A GOTO 5010,5020,5030
```

If the leading zero were not present in &&02, the library processor would either look for &&21, &&22, &&23 or for &&210, &&220, &&230, the latter group being, of course, illegal.

In line 1060 of the generalized code in Example 1, there is another pseudo-command word, 'BOOL, which is logically grouped with the pseudo-command word 'ENDB at line 1080. This pseudo-command set incorporates elementary Boolean logic in a method by which the library processor decides whether or not certain lines or sets of lines are supposed to be included into the specialized output. The Boolean method tests for truth. Accordingly, line 1060 is interpreted to mean "If *both* argument five *and* argument six were specified in the specialization pseudo-command line, then *do* include all lines until the next 'ENDB line is found." If either argument five or six had been optionally omitted by the calling programmer, line 1070 would not appear in the specialized output.

"The Boolean method tests for truth."

The forms of the 'BOOL pseudo-command permit quite complicated and flexible condition testing. The OR combination is indicated by a minus sign between two arguments:

```
'BOOL &&23-&&03
```

This means "If argument 23 *or* argument 3 is present, *include* the lines which follow until the next 'ENDB pseudo-command." Remember that the 'BOOL command is always a test of truth for inclusion.

In addition to the logical AND and OR possibilities, combinations of present and absent argument conditions may be tested in the 'BOOLean test for inclusion. The examples of the arguments illustrated so far have actually implied a " + " just before the numerals of the argument. For example, the illustration above actually means && + 23—&& + 03, that is, a test for the *presence* of argument numbers 23 or three. A test for absence could be written:

```
'BOOL && - 23 - && + 03
'BOOL && - 23 + && - 03
```

✓See List of Advertisers on page 306

"It's important to write BASIC functions with the same deliberate and meticulous care as when writing assembly language."

In the first example, inclusion will occur if argument 23 is absent, or, if argument three is present. In the second example, inclusion will occur if agrument 23 is absent, and if argument three is absent. The OR condition could as easily test for the absence of either argument:

```
'BOOL && – 23 – && – 03.
```

In order to obtain additional complexity and flexibility, nest combinations of code lines to 10 levels of 'BOOL. The rules applying to subroutines also apply to nesting.

The code lines appearing in Example 3 illustrate the results of running the user's program (Example 2) against the macro library through the macro processor program. A number of quite interesting things did and did not take place.

First, note that only the named macro, DSKOPN, is included into the user's final program. Other macros cataloged in the same library file with DSKOPN are not called.

Second, the macro is inserted into the user's calling program in the exact sequential place where its pseudo-command occurred. Thus, it is the programmer's responsibility to document this macro library, knowing which modules may be dropped through and which must be accessed by the GOSUB command. The macro processor program makes no attempt to analyze program logic (except for common macros, described below).

Third, all arguments are inserted into their corresponding argument locations. The format of each line is expanded or shortened, accordingly.

Fourth, since both arguments five and six are specified, line]070 of the original macro is included, as line 280 of the final, specialized output. Both the 'BOOL and 'ENDB pseudo-command lines are not included, nor the 'ENDMAC.

Fifth, the entire program is renumbered.

Additional Features

This simple example illustrates the tremendous increase in programming power a macro processor makes possible. It's important to write BASIC functions with the same deliberate and meticulous care as when writing assembly language. Such authorship is expensive in programming time and, therefore, should be placed in a macro library for economical reuse.

Since economy is the principal purpose of the macro processor, there are two additional devices which conserve both memory and time. These include the common module designator and the list-printing control commands.

A common macro module's call is common to several sections of the user's program, or is called through a nested macro by two or more macros. A good example of this might be a disk I/O command which, for economy of space, serves several different logically designated files. The macro appears in the library as:

```
1000 'MACRO = DSEQP &&01,&&02
1010 PUT&&01,&&02
1020 RETURN
1030 'ENDMAC
```

This macro is a simple Put to a sequential disk file. Arguments one and two are, respectively, the logical file number and the record number. But the arguments could be specified as the numeric variable containing the logical file number, and the numeric variable containing the record number. In that event, the macro name could read:

```
1000 'MACRO = DSEQP-C &&01,&&02
```

The macro would then be declared common to any and all places in the program where the same macro call is found. The first form of the macro would appear every time it is named in the user's program or is nested within any other macros called by the user's program. But the second, common form would appear in the user's program only once, no matter how many times it is called.

The common designation -C causes the macro processor to search through the macro for the Return command. If it is found, any subsequent call to the common macro will be replaced by the GOSUB nnnn command. If no Return command is found, every subsequent call to the common macro is replaced by the GOTO nnnn command.

The argument values that specialize a common macro are taken from the first macro call encountered (lowest line number). The argument values specified in subsequent calls are completely ignored. However, it is a good idea to include them each time because it is easy to go back and insert a new macro call. It is also easy to forget that it nests a call to a common macro whose specialization arguments are now in the second encountered macro call.

This warning also points up another feature of the macro processor's 'BOOL logic. the present/absent switch operates within every line, even if there is no 'BOOL pseudo-command. If a line of macro code calls for an argument, and if that argument is not provided with the macro call, the dummy argument &&nn remains in the line and an error message is printed at the left of the line during the final printing.

The second economical feature permits

```
100 'PROGRAM MODUEL "TESTPROG"
110 CLS:CLEAR1000:DIM A$,B$,C$:ON ERROR GOTO 320
120 INPUT "ENTER DISK FILE NAME: ";A$
130 INPUT "ENTER DRIVE NUMBER: ";B$
140 A$ = A$ + ":" + B$
150 B$ = "R":C$ = " "
160 GOSUB 220:IF C$ = " "THEN 170 ELSE 300
170 FIELD1,64 AS X1$,64 AS X2$,64 AS X3$,64 AS X4$
180 GET1,1
190 PRINT X1$;X2$;X3$;X4$
200 CLOSE1
210 END
220 'MACRO = DSKOPN B$,1,A$,320,C$,"CAN'T OPEN FILE "
230 ON ERROR GOTO 270
240 OPEN B$,1,A$
250 ON ERROR GOTO 320
260 RETURN
270 ON ERROR GOTO 320
280 C$ = "CAN'T OPEN FILE "
290 GOTO 260
300 PRINT C$;A$
310 GOTO 200
320 PRINT "ERROR ";ERR;" AT LINE ";ERL
330 STOP
```

Example 3. User's Program After Specialization

"Often, the (program) writer calls as many as 30 or more macros in each BASIC program, none of which is being seen for the first time."

selective control over printed listings. After the macro library modules are debugged and used several times, the user has no interest in seeing them again. Furthermore, with practice in writing macros, the user finds ways to standardize his programming procedures, so that the macro modules in the library become large and numerous. Often, the writer calls as many as 30 or more macros in each BASIC program, none of which is being seen for the first time.

Selective listing controls are illustrated in these two examples:

```
1000 'LISTF#22-#255
1010 ....
1020 ....
1030 'LISTN
1040 'LISTF#33+#255
1050 ....
1060 ....
1070 'LISTN
```

At line 1000 (either in the user's program or in a macro), the list-off command states, "Do not print the following lines until the next list-on command, unless print suppression is overridden by operator's keyboard switches number 22 or 255." These switches are entered either at the beginning of the second pass (after macro specialization but before printing) by the operator, or they may be entered in the program itself by the pseudo-command:

```
1000 'SWITCH #22,#33,#255
```

(The commas are included only for readability, since the # is the controlling delimiter.)

The switch numbers mean anything the programmer wants. For example, switch 255, the highest available switch, might mean GLOBAL PRINT. When switch 255 is entered, every section of code containing the simple switch command #255 or the OR switch command #nnn-#255 would print.

Operating Requirements

Given the 256-byte character of nearly all disk systems, you should have 64K (48K user) main memory with two disk drives. The macro library and user application program are input to the macro processor from one drive. The specialized output file is written to this same drive. The other drive is reserved as working space for the macro processor program.

The macro library should not contain more than 160 macros, and a five-inch disk system should limit the number of arguments per macro to 32. Macro nesting should also be limited to ten levels in such a small system. Visually illustrated, a 10-level macro call nest would appear:

```
1000 'MACRO=PROGA &&01,&&02,&&03
1010 'MACRO=PROGB &&..,&&..
1020 'MACRO=PROGC &&..,&&..
1030 'MACRO=PROGD &&..,&&..
1040 'MACRO=PROGE &&..,&&..
1050 'MACRO=PROGF &&..,&&..
1060 'MACRO=PROGG &&..,&&..
1070 'MACRO=PROGH &&..,&&..
1080 'MACRO=PROGI &&..,&&..
1090 'MACRO=PROGJ &&..,&&..
1100 'CODING FROM PROGJ
1110 'ENDMAC
1120 'CODING FROM PROGI
1130 'ENDMAC
1140 'CODING FROM PROGH
1150 'ENDMAC
1160 'CODING FROM PROGG
1170 'ENDMAC
1180 'CODING FROM PROGF
1190 'ENDMAC
1200 'CODING FROM PROGE
1210 'ENDMAC
1220 'CODING FROM PROGD
1230 'ENDMAC
1240 'CODING FROM PROGC
1250 'ENDMAC
1260 'CODING FROM PROGB
1270 'ENDMAC
1280 'CODING FROM PROGA
1290 'ENDMAC
```

Program line lengths should be kept at or below 64 bytes, including the line numbering characters. However, this is not a restriction, since programming for macros usually involves so many 'BOOLean operators that lines are short anyway.

The total number of lines per program should be kept at 2500 maximum, including all specialized macros. However, with longer lines it probably would not be possible to contain 2500 lines within the computer's user memory under the interpreter, and cross-compilation into executable machine language would definitely not be possible.

As indicated above, the macro processor operates in two passes. If any called macros have not been found in the named library, a second library may be named and two more passes result. This continues until all macro calls have been honored. If any called macro cannot be found when the operator calls for a print listing, they are flagged with an error indicator at the left of the line number. Error-flagging also overrides print suppression, in case one of the unfound macro calls is nested inside another macro.

Line number references are also important. The user program may reference any

"A line number which is the name of a line may not be a parameter."

of its own lines, some of which are macro calls. But the user may not reference any line believed to be inside a macro or an undefined line error will be flagged.

When writing a macro library module, lines within the macro may be numbered at the convenience of the programmer—they will never remain similarly numbered after specialization. A macro may reference only its own lines, unless the line number is supplied as an argument to resolve a GOTO, GOSUB, Then or Else command word in the macro with one of the user's program lines. An argument line number may not attempt to reference a line number internal to another macro.

A line number which is the name of a line may not be a parameter. For example:

```
&&03 GOTO &&07
```

This line is illegal, because it does not begin with a number. However, as described above, the following is legal:

```
1000 GOTO &&07
```

The value for argument number seven is a line in the user's program.

Any line number which is equal to or less than the original (library file) line number of a macro, but which is not supplied as an argument value, is considered to be a Restart command.

For example:

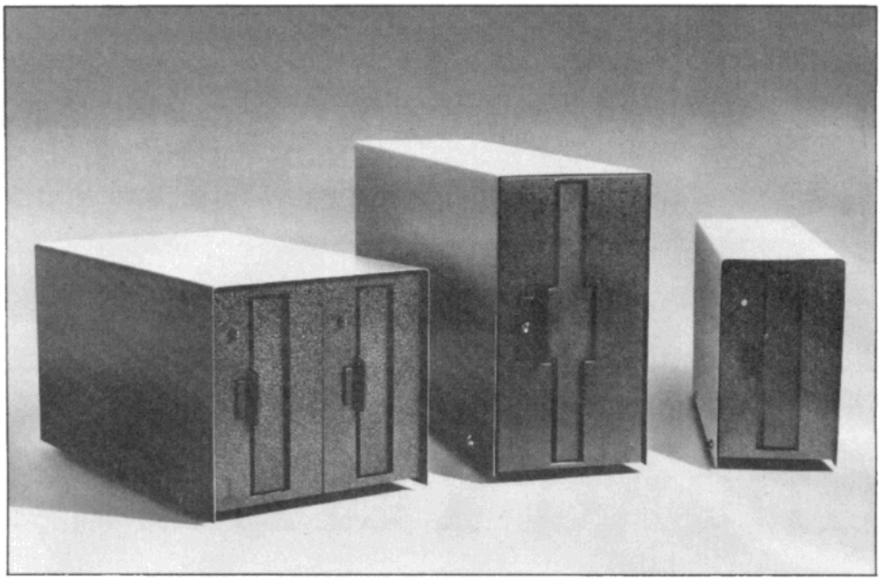
```
1000 'MACRO = PROGA-C &&01,&&02,&&03
1020 PRINT@&&01,&&02:&&01 = &&01 + 64:&&03 =
     &&03 + 1
1030 ON &&03 GOTO 0,0,0,0,0,1040
1040 RETURN
1050 'ENDMAC
```

This macro provides repeated printing of a message (from one to five times) depending upon the value of the numeric constant named as argument three. The zero line numbers in line 1030 will be replaced with line number 1000, as would be any number from 0–999.

Conclusions

Although writing a macro processor is not easy fare for any but experienced programmers, it can and should be written in BASIC. If made a macro itself, the library processor program may be specialized for such diverse purposes as macro library processing for COBOL and for word processing applications, such as filling in pre-printed forms and contracts.

Once debugged, it should be cross-compiled and run as executable machine code in order to increase its speed (by as much as 30 times) in some of its math/logic functions. ■



See List of Advertisers on page 306

Some buzzwords defined.

Coming to Terms

Joe D. Fugate
8979 Mandan Ct.
Boise, ID 83709

The word *computer* is becoming a common buzzword. However, along with the microcomputer has come many not-so-common concepts that are both unfamiliar and confusing to the layman.

Let's lay some ground work: The TRS-80 (and all microcomputers) is based on a particular chip (integrated circuit) or microprocessor. The TRS-80 microprocessor chip is called the Z-80. To program the Z-80 directly, one must use numeric codes known as machine-code. This is not very convenient for humans who are not accustomed to talking to each other in strings of numeric codes.

To ease our communication problem, assemblers, compilers and interpreters were developed. The basic job of these three is the same: to decode your programs into something the machine can understand and operate on. How they each accomplish this is where the difference between them lies.

Your Level II or Level I ROM is a BASIC interpreter. That is, the ROM is coded to interpret the computer language known as BASIC. Another ROM might be designed to interpret FORTRAN, another Pascal.

Interpreter vs Compiler

Let's talk about the difference between your BASIC ROM and a BASIC compiler. The following are concise descriptions of the functions of an interpreter and a compiler:

Interpreter

1. Examines BASIC program statements.
2. Determines the action requested.
3. Calls machine code subroutine(s) that do the action.

Compiler

1. Examines BASIC program statements.
2. Generates equivalent machine codes.
3. Saves generated machine code.

A compiler generates a genuine machine code replacement for a BASIC program, while an interpreter only fakes it.

The outstanding feature of an interpreter is its ability to make the computer seem ready to do your every bidding—to be interactive. It immediately does what is requested. This nicety is not free, though; it costs in speed. For example, when you code a FOR-NEXT loop, an interpreter must re-interpret each instruction every single time it goes through the loop! To enhance the interactive feature, the interpreter is constantly scanning the keyboard (taking valuable time) so the computer will always remain ready to do your bidding.

With a compiler, getting from the program coding stage to the program running stage is less convenient, but what the compiler lacks in convenience, it makes up for in program execution speed. Compare the programming steps required with an interpreter versus that of a compiler:

Interpreter Programming Steps

1. Code program statement(s).
2. Run program.

(The program doesn't work the first time? Locate the offending statement and repeat the above steps until it does work).

Compiler Programming Steps

1. Code program statement(s) (source).
2. Run statements through compiler to produce machine code output (object).
3. Machine code program is saved on tape or disk.
4. Save source program statements from step 1.
5. Reload machine code program from tape or disk (step 3).
6. Run machine code program.

(The program doesn't work right? Uh-oh. You're going to have to reload the compiler program, reload your saved source code from step 4, and repeat all the steps to replace the offending statements. You must repeat this until the program works).

Using a compiler seems like a lot of hassle, doesn't it? Ah, but the final result makes it all worth it! The machine code output from a compiler can be run directly, without the need for an interpreter standing between your BASIC program and the machine. As a result, compiled programs will typically run 10 to 20 times faster than the same source program running through an interpreter. Sure beats that 2X clock mod!

Assemblers

An assembler is similar to a compiler in function. The difference is that an assembler is used to make coding direct machine code a little easier for us humans. For example, to the TRS-80, 128 (hexadecimal 80) means "take the contents of the B register and add it to the A register." You have to have a really good memory to remember all 256 possible machine code instructions by their number codes! An assembler simply allows you to code "ADD A,B" and it will replace that mnemonic phrase with 128 when you run the assembler against your mnemonic machine code. See Figs. 1 and 2.

DIAGRAM OF INTERPRETER FUNCTION

Fig. 1. The interpreter stands between your program and the computer. Your program simply tells the interpreter which machine code subroutine(s) to run to get the result you want.

DIAGRAM OF COMPILER OR ASSEMBLER STEPS

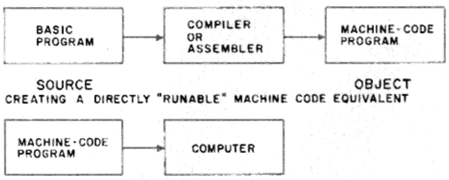


Fig. 2. In the compiler method, there is ultimately nothing standing between the computer and your program. As a result, execution is much faster. However, a compiler involves many more steps to get from the source program to the directly executable object machine code.

Subroutines and Macros

There are methods available to you as a programmer that will spare you the tedium of repetitive coding. The most commonly used is the subroutine. Compilers and assemblers also often incorporate an additional method—the macro.

Using subroutines, you code only a single copy of your repetitive code, and trigger its execution with a subroutine call (GOSUB in BASIC). At the end of your subroutine code you must return control back to your calling program (RETURN in BASIC). You need to code your subroutine only once, yet you can use it often. Subroutines reduce the over-all size of the program as well as provide relief for the programmer's poor numb fingers.

With all its virtues, a subroutine does cost time. When the computer is told to execute a subroutine, it takes time for the computer to figure out where the subroutine is, execute the subroutine, and then determine where to resume processing again. With some compilers or assemblers, there is an alternate approach to the subroutine: the macro.

The macro is best thought of as a kind of in-line subroutine. A macro inserts the subroutine code into your program each time it is used. Using a hypothetical programming language, let's illustrate how a macro is used.

This code tells the computer this is a macro routine.

```
ADDEMUP MACRO:
INPUT X
Y = X + Y
END MACRO
```

Before you run the compiler against your program, the following shows how you might code ADDEMUP to sum three input numbers and display a total:

```
Y = 0
ADDEMUP
ADDEMUP
ADDEMUP
DISPLAY Y
STOP
```

When the compiler is run against your program, it will see ADDEMUP and replace ADDEMUP with the actual code it represents:

```
After compiler macro expansion:
Y = 0
ADDEMUP      •MACRO REPLACED•
INPUT X
Y = X + Y
ADDEMUP      •MACRO REPLACED•
INPUT X
Y = X + Y
ADDEMUP      •MACRO REPLACED•
INPUT X
Y = X + Y
DISPLAY Y
STOP
```

After macro expansion, the compiler will convert the expanded code into machine code.

With the compiled program loaded into the computer and run, the computer will never know we coded all those commands with the help of a macro.

Why not always code a subroutine for repetitive code instead of a macro? After all, subroutines take less memory! Again, the reason goes back to the memory/run-time trade off. Even though a subroutine saves memory, it's slower. All those macro expansions for each use of the macro take memory, but save time!

Did you know when you compile a BASIC program, each BASIC statement could generate several equivalent machine code instructions? What does this mean? Your BASIC program statement is actually a *macro*! Ah-ha!

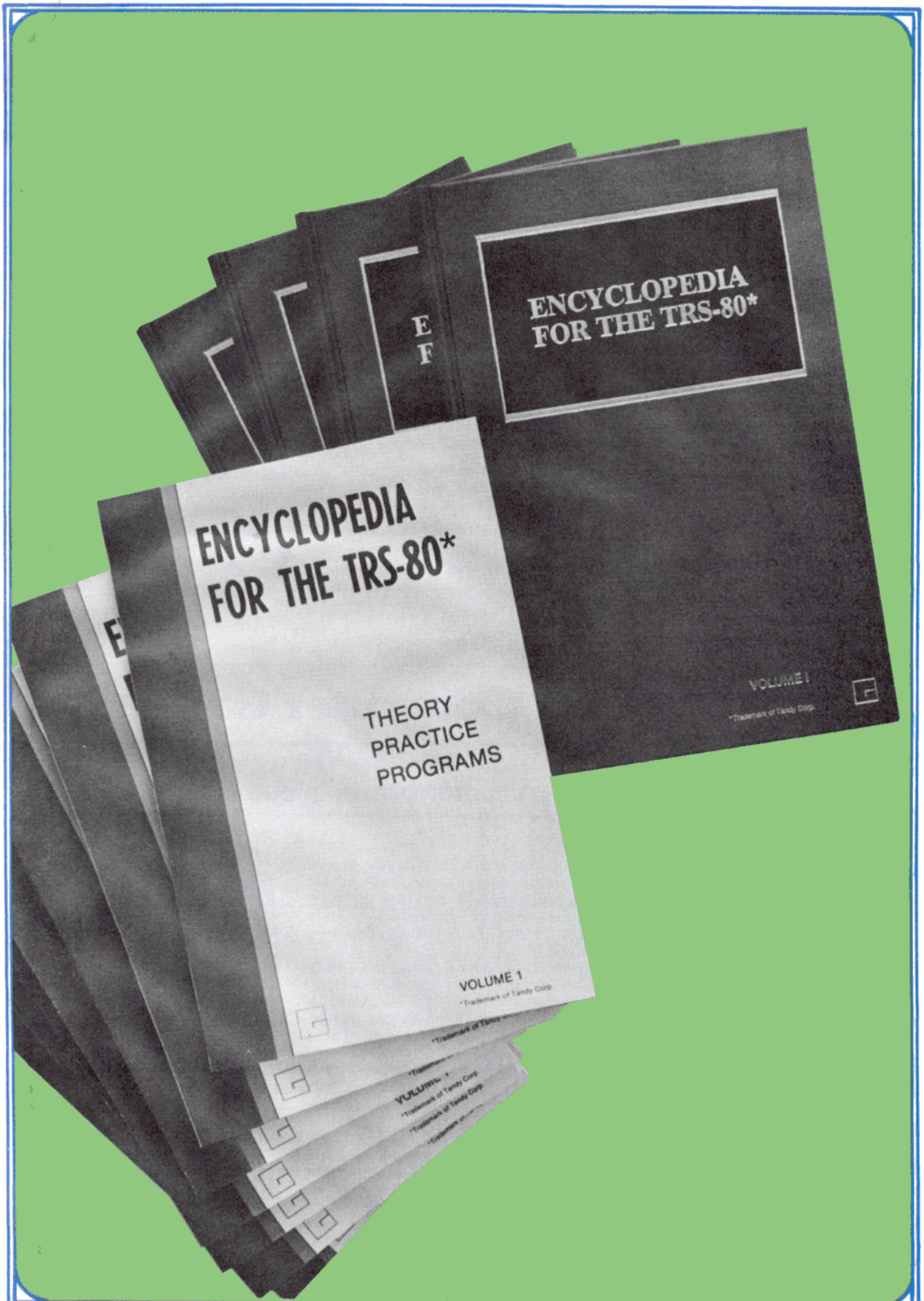
Link Editors and Loaders

A loader is a special loading program that will load your machine-code and move it around in memory. A link-editor is a fancy loader that will tie two or more programs with references to each other together into one big program.

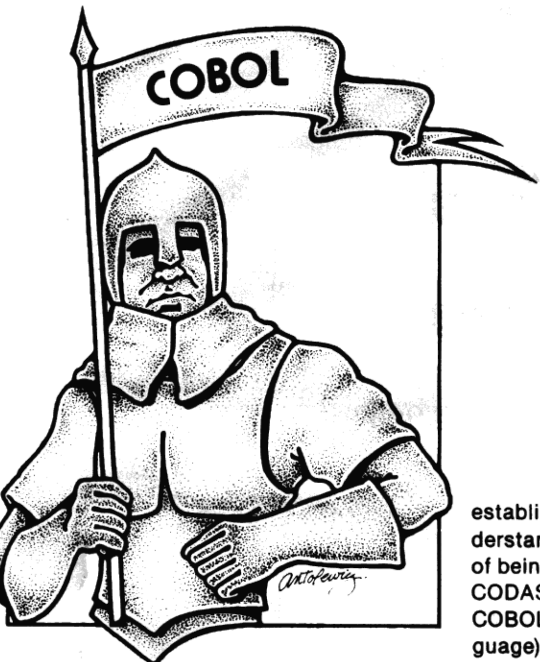
The link-editor allows you to code your program in modules if you like (modular programming). You can code each module separately and then link-edit them together.

You should now understand these computer terms a bit better. At least, better than the fellow who typed this graffiti on a computer in a Radio Shack store: Byte My Baud. ■

✓See List of Advertisers on page 306


ENCYCLOPEDIA
FOR THE TRS-80*
VOLUME I
*Trademark of Tandy Corp.
ENCYCLOPEDIA
FOR THE TRS-80*
THEORY
PRACTICE
PROGRAMS
VOLUME 1
*Trademark of Tandy Corp.

A mainframe tool for the businesslike. It's waiting in the wings.



COBOL: Ready and Waiting

Robert L. Bradley
10720 Danbury
Baton Rouge, LA 70809

Several languages are vying for primacy among microcomputer users. What we call BASIC is really a generic term for each manufacturer's version of an equation-oriented language. The diversity of BASIC precludes its consideration as a single entity. We need fewer, not more, languages.

The need for a minimum number of languages is apparent. If we are to communicate ideas effectively, we must have an understanding of the rules. The more languages there are, the greater the number of rules. Program portability becomes more difficult with the increasing number of languages. The more conversions required, the higher the probability of errors and misinterpretations.

Ideally, an industry committee should be formed to standardize a language, as did the Conference on Data Systems Language (CODASYL) more than 20 years ago, which established an English-based language understandable to businessmen and capable of being easily changed. The language that CODASYL defined and developed was COBOL (COmmon Business Oriented Language).

COBOL is now the most commonly used language for business applications on mainframes. What has this to do with microcomputers? As memory gets cheaper and the trend toward larger memories continues, COBOL is waiting in the wings for serious business programmers. Several independent software firms have released COBOL compilers, and Tandy Corporation recently announced one for the TRS-80 Model II.

The Radio Shack version has the greatest potential for widespread acceptance because of Tandy's marketing power. The compiler is software-based; there are no hardware modifications needed to run it on the Model II. Although some of the commands found in standardized COBOL are not present in the Radio Shack compiler, the TRS-80 version represents a giant stride in providing mainframe computing tools to micros. The documentation is well written and, except for the lack of an index, provides an excellent guide to the rules of the language.

COBOL Structure

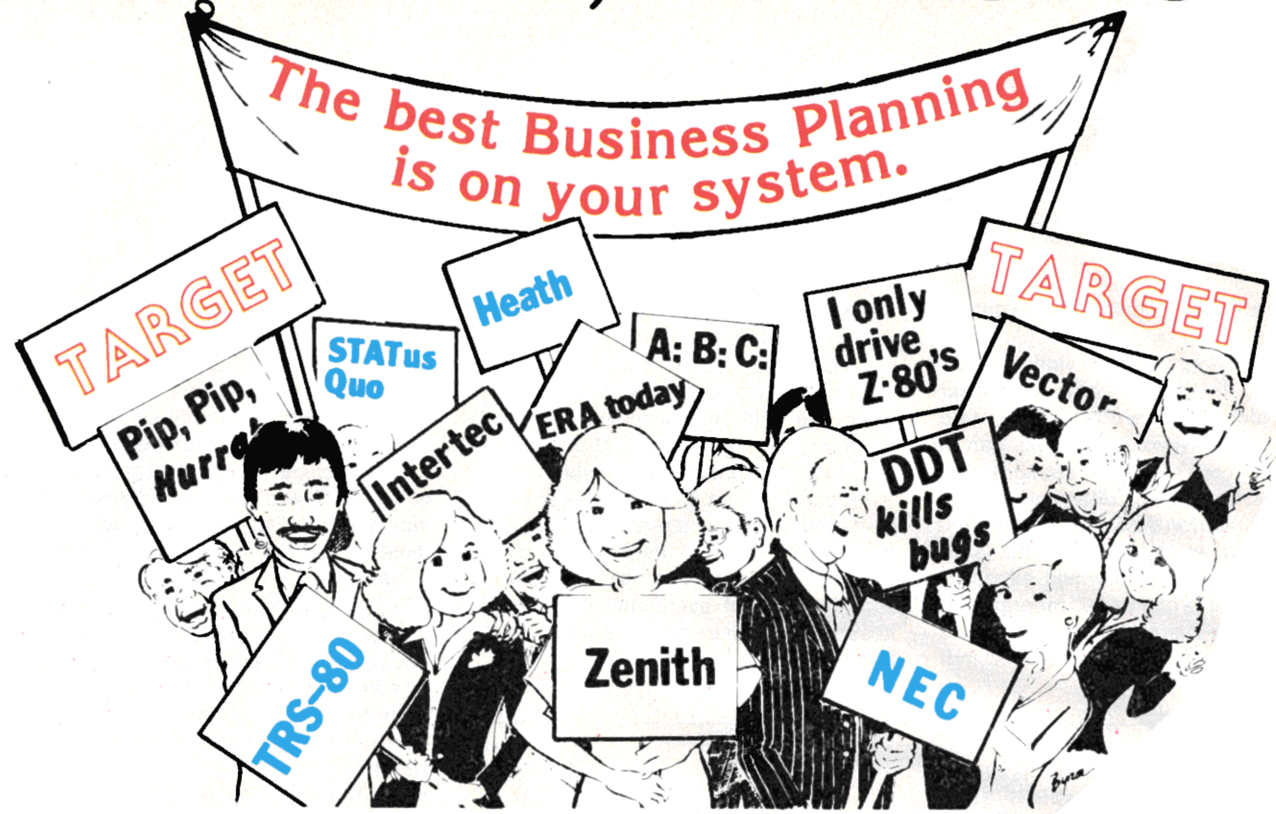
I wish to impart the flavor of the language and make a few comparisons with what we call BASIC. The application which I shall use to illustrate COBOL is simple, but by refining the programming technique I hope to demonstrate the power of the language. The purpose of the program is to calculate a few commonly used business financial ratios:

(1) Profit margin = net income/net sales
(2) Current ratio = current assets/current

```
       IDENTIFICATION DIVISION.
         PROGRAM-ID.                 FINRATIO.
         AUTHOR.                     R L BRADLEY.
         INSTALLATION.               HOME.
         DATE-WRITTEN.               JULY 80.
      *
      *  CALCULATE BUSINESS FINANCIAL RATIOS
      *
```

Program Listing 1. Identification Division.

STAND UP, CP/M USERS



For the past few months you have patiently endured the indignity of watching your friends show off their flashy visible number cruncher on their game-playing computer and longed for something as slick.

You seriously considered buying *their* computer, but you just couldn't give up the benefits of the CP/M operating system.

Well, say hello to TARGET™, the best business planning system available on micros today (and some big computers besides).

Talk about a blank ledger sheet or columnar tabulator! TARGET™ is much more than that. Sure, just like their product, you can fill in the rows and columns as you see fit and the numbers will whiz by you on the screen. But, we let you create your report in English, not in some combination of reverse Polish notation and algebraic matrix languages.

Our TARGET on *Your* System. . .

```
LINE 1 SALES = 100 200 300 400
LINE 2 EXP   = GROW 50 BY 15%
LINE 3 NET   = SALES  –  EXP
```

Their Product on *Their* System . . .

```
SALES 100 200 300 400
EXP 50  + B2*1.15  + C2*1  + D2*1.15
NET  + B1-B2  + C1-C2  + D1-D2  + E1-E2
```

At least, that is what their product might look like if you could see all of your data and calculation rules at the same time, *which you can't.* If you think that it is an easy approach for debugging, guess again.

TARGET™ displays a full screen of results or data and calculation rules at your command. And, it runs on your system. How much more could you ask?

Lots! And TARGET™ delivers. We give you the ability to:

- Obtain hard copy printouts of *both* the results *and* your set of data and calculation rules used to create those results.
- Write the results of data and calculation rules to the disk as formatted files for word processing.
- Use full conditionals (IF . . . THEN . . . ELSE) in your set of rules and stack conditionals within conditionals.
- Build powerful models with commands such as GROW, MIN, MAX, AVE, CUM, GREATER, LESSER, SUM.
- Edit lines with ease.
- Enter rules and data in any order you wish. TARGET™ will automatically sort rows in ascending order *and* process your model correctly even if LINE 33 SALARIES needs to be calculated before LINE 3 EXPENSES.

There is so much more about the TARGET™ Planner that will impress you. Our manual explains not just what a command or function is, but how to use the functions in everyday business situations. Our manual and our newsletter give you illustrative examples of business planning problems and how TARGET™ can help you solve those problems. For example, we will show you how to quickly determine the payback period for a proposed project.

TARGET™ is a compiled system running under CP/M. There is no need to buy BASIC or FORTRAN or any other system software. And the price is *only $195* for the TARGET™ Planner, our basic system.

Stand up and be proud, CP/M users. The best business planning system runs on the best operating system on *your* computer.

Advanced Management Strategies, Inc.
1935 Cliff Valley Way, N.E., Suite 200
Atlanta, Georgia 30329
404/634-9535 ✓260

P.S. When all of your friends start drooling over *your* product, tell them to cheer up. With Microsoft's great SoftCard and 16-K memory board, your friend's Apple can move up to your operating system and run TARGET™.

"The Radio Shack version (of COBOL) has the greatest potential for widespread acceptance, because of Tandy's marketing power."

liabilities

(3) Quick ratio = quick assets/current liabilities

All COBOL programs are divided into four major parts. They are, in order of appearance, the identification, environment, data, and procedure divisions. Divisions can be partitioned into sections, sections into paragraphs, paragraphs into sentences.

The identification division for the sample program is shown in Listing 1. This is the type of information one would expect to find in the remarks of a BASIC program. The difference is that here it has been standardized. A COBOL comment statement, indicated by the *, states the purpose of the program.

The environment division, shown in Program Listing 2, describes the hardware environment. If we were using files in this program, the relationship between their program representations and their physical counterparts would be established here.

Program Listing 3 depicts the data division. Here we define the variables which are used in the program, as indicated by the code 77. Variable names can be quite descriptive without sacrificing their uniqueness; in fact, names can be up to 30 characters in length.

```
ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
  SOURCE-COMPUTER.               MODELII-64K.
  OBJECT-COMPUTER.               MODELII-64K.
```

Program Listing 2. Environment Division.

```
DATA DIVISION.
 WORKING-STORAGE SECTION.
   77  NET-INCOME                PIC S9(7)  COMP-3 VALUE 0.
   77  NET-SALES                 PIC S9(7)  COMP-3.
   77  NET-INCOME-DISP           PIC $Z,ZZZ,ZZZ-.
   77  NET-SALES-DISP            PIC $Z,ZZZ,ZZZ.
   77  PROFIT-MARGIN             PIC S99V999  COMP-3.
   77  PROFIT-MARGIN-DISP        PIC -Z9.9.
```

Program Listing 3. Data Division.

```
PROCEDURE DIVISION.
  PRINT-HEADING.
    DISPLAY "FINANCIAL RATIOS CALCULATION", LINE 1, POSITION 27,
    ERASE. DISPLAY "ENTER:", LINE 3, POSITION 27.
    DISPLAY "1. NET INCOME", LINE 5, POSITION 27.
    DISPLAY "2. NET SALES", LINE 7, POSITION 27.
  ENTER-DATA.
    ACCEPT NET-INCOME, LINE 5, POSITION 50, PROMPT ".", CONVERT.
    MOVE NET-INCOME TO NET-INCOME-DISP.
    DISPLAY NET-INCOME-DISP, LINE 5, POSITION 50.
    ACCEPT NET-SALES, LINE 7, POSITION 50, PROMPT ".", CONVERT.
    MOVE NET-SALES TO NET-SALES-DISP.
    DISPLAY NET-SALES-DISP, LINE 7, POSITION 50.
  CALCULATE-RATIO.
    DIVIDE NET-INCOME BY NET-SALES GIVING PROFIT-MARGIN ROUNDED.
    MULTIPLY PROFIT-MARGIN BY 100 GIVING PROGIT-MARGIN.
    MOVE PROFIT-MARGIN TO PROFIT-MARGIN-DISP.
  PRINT-RATIO.
    DISPLAY "PROFIT MARGIN =", LINE 10, POSITION 27.
    DISPLAY PROFIT-MARGIN-DISP, LINE 10, POSITION 43.
    DISPLAY "%", LINE 10, POSITION 49.
END PROGRAM.
```

Program Listing 4. Procedure Division.

The use of the PIC (or PICTURE) clause permits the definition of both the type and format of the variable. There are essentially two types of PICTURE clauses, those for internal use (unedited) and those for input/output (edited). An example of the unedited numeric type, as seen in the definition of the net-income variable, is signed (indicated by the S) and has seven digits (the nine indicates any digit and the seven in parentheses indicates the number of digits permissible).

A decimal position would be indicated by a V, as illustrated in the definition of profit-margin. The variable net-income-disp uses an edited numeric clause. The $ symbol will be printed in the first print position. The Z indicates that if the resulting digit is a nonsignificant zero, nothing will be printed (including any intervening commas). Thus, a number such as 15872 would be printed as $ 15,872. If the number were negative, $ 15,872- would result. The clause COMP-3 accompanies the unedited items; this indicates packed decimal representation. Variables may be initialized by use of the VALUE clause.

Several advantages of the data division are evident. First, the format establishes a variable list, an optional exercise in BASIC. If the variable names chosen are descriptive ones, this list is excellent documentation. Secondly, the format of any variable can be easily changed simply by modifying the PICTURE clause. The power of the editing features permits virtually any format of input/output. One disadvantage of COBOL is that the representation of a variable for program internal use must be distinct from that for input/output purposes when the value is numeric. Thus, two variables are sometimes required to denote the same value.

The procedure division, shown in Program Listing 4, is the part that makes things happen. Statements are grouped into paragraphs, for which the programmer must select names. A glance at the listing therefore enables one to obtain a quick summary of the actions that are taken. In this first example we are going to calculate only the profit margin. Additional statements could be used to calculate the other two ratios.

Each statement in the procedure division begins with a command called a verb. In the PRINT-HEADING paragraph the DISPLAY verb outputs messages to the CRT at the screen positions indicated. The ENTER-DATA paragraph receives the data into the program by the use of the ACCEPT verb.

"Because (COBOL) is virtually self-documenting, the time required to understand someone else's program is minimal."

The CONVERT clause in the ACCEPT statement changes the input to the format specified by COMP-3 in the data division. In order to display the edited input on the CRT, the data must be MOVEd to a variable defined for that format. The purpose of the statements in the CALCULATE-RATIO and PRINT-RATIO paragraphs can be easily interpreted.

A Better Way

Although this technique will accomplish the desired task, if additional ratios are calculated, the amount of coding increases almost proportionally. We must find a more compact way to calculate several ratios. In Program Listing 5 we have selected the variable names to reflect their more generalized use. Each variable will assume several values during the course of the program. This change enables us to utilize the PERFORM verb, a command analogous to GOSUB in BASIC.

PERFORM transfers control to the paragraph specified. After the statements in that paragraph are executed, program flow is resumed at the statement following PERFORM. In this program version we have two subroutines, or paragraphs, to be PERFORMed, one for entering data and the other for calculating a ratio. With the power of the PERFORM verb we have noticeably increased the throughput of the program. We are now ready to expand the program to calculate all three ratios. The data and procedure divisions for this effort are shown in Program Listing 6. The sample run appears in Program Listing 7.

This was only a very brief glimpse at what COBOL can do. There are dozens of additional features and commands too numerous even to mention.

Your first reaction may be that it is much too wordy and time consuming, compared to BASIC. Yet this is one of COBOL's great strengths. Because it is virtually self-documenting, the time required to understand someone else's program is minimal.

A second advantage is that COBOL is easier to learn than other languages. One can memorize the rules of BASIC quickly, but effective techniques can take months to master. Because of its more rigid format and its English language syntax, the learning time for COBOL is shorter.

Your final argument against COBOL may be that you simply don't need it. The language was designed for business applications, and unless you are writing business programs which may be modified by someone else, you really don't need COBOL. However, the day may be approaching when microcomputer users need a standardized business language that provides easily understood programs which can be quickly modified. Many people, including

```
DATA DIVISION.
 WORKING-STORAGE SECTION.
   77  ENTRY                    PIC S9(7)  COMP-3.
   77  ENTRY-DISP               PIC $Z,ZZZ,ZZZ-.
   77  LINE-NO                  PIC 99  COMP-3.
   77  VAR1                     PIC S9(7)  COMP-3.
   77  VAR2                     PIC S9(7)  COMP-3.
   77  RESULT                   PIC S99V999  COMP-3.
   77  RESULT-DISP              PIC -Z9.9.
PROCEDURE DIVISION.
  PRINT-HEADING.
    DISPLAY "FINANCIAL RATIOS CALCULATION", LINE 1, POSITION 27,
    ERASE. DISPLAY "ENTER:", LINE 3, POSITION 27.
    DISPLAY "1. NET INCOME", LINE 5, POSITION 27.
    DISPLAY "2. NET SALES", LINE 7, POSITION 27.
  ENTER-AND-CALCULATE.
    MOVE 5 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR1.
    MOVE 7 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR2.
    PERFORM CALCULATE-RATIO.
  PRINT-RATIO.
    DISPLAY "PROFIT-MARGIN =", LINE 10, POSITION 27.
    DISPLAY RESULT-DISP, LINE 10, POSITION 43.
    DISPLAY "%", LINE 10, POSITION 49.
    STOP RUN.
  ENTER-DATA.
    ACCEPT ENTRY, LINE LINE-NO, POSITION 50, PROMPT ".",
    CONVERT.
    MOVE ENTRY TO ENTRY-DISP.
    DISPLAY ENTRY-DISP, LINE LINE-NO, POSITION 50.
  CALCULATE-RATIO.
    DIVIDE VAR1 BY VAR2 GIVING RESULT.
    MULTIPLY RESULT BY 100 GIVING RESULT.
    MOVE RESULT TO RESULT-DISP.
END PROGRAM.
```

Program Listing 5.

Program Listing 6.

```
DATA DIVISION.
 WORKING-STORAGE SECTION.
   77  ENTRY                    PIC S9(7)  COMP-3.
   77  ENTRY-DISP               PIC $Z,ZZZ,ZZZ-.
   77  LINE-NO                  PIC 99  COMP-3.
   77  VAR1                     PIC S9(7)  COMP-3.
   77  VAR2                     PIC S9(7)  COMP-3.
   77  RESULT                   PIC S99V999  COMP-3.
   77  RESULT-DISP              PIC -Z9.9.
   77  PERCENT                  PIC 9  COMP-3.
PROCEDURE DIVISION.
  PRINT-HEADING.
    DISPLAY "FINANCIAL RATIOS CALCULATION", LINE 1, POSITION 27,
    ERASE. DISPLAY "ENTER: ", LINE 3, POSITION 27.
    DISPLAY "1. NET INCOME", LINE 5, POSITION 27.
    DISPLAY "2. NET SALES", LINE 7, POSITION 27.
    DISPLAY "3. CURRENT ASSETS", LINE 9, POSITION 27.
    DISPLAY "4. CURRENT LIABILITIES", LINE 11, POSITION 27.
    DISPLAY "5. QUICK ASSETS", LINE 13, POSITION 27.
  PROFIT-MARGIN.
    MOVE 5 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR1.
    MOVE 7 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR2.
    MOVE 1 TO PERCENT. PERFORM CALCULATE-RATIO. MOVE 0 TO PERCENT.
```

Program continues

See List of Advertisers on page 306

"The day may be approaching when microcomputer users need a standardized business language that provides easily understood programs."

```
           DISPLAY "PROFIT MARGIN =", LINE 16, POSITION 27.
           DISPLAY RESULT-DISP, LINE 16, POSITION 43.
           DISPLAY "%", LINE 16, POSITION 50.
       CURRENT-RATIO.
           MOVE 9 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR1.
           MOVE 11 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR2.
           PERFORM CALCULATE-RATIO.
           DISPLAY "CURRENT RATIO =", LINE 20, POSITION 27.
           DISPLAY RESULT-DISP, LINE 18, POSITION 43.
       QUICK-RATIO.
           MOVE 13 TO LINE-NO. PERFORM ENTER-DATA. MOVE ENTRY TO VAR1.
           PERFORM CALCULATE-RATIO.
           DISPLAY "QUICK RATIO =", LINE 20, POSITION 27.
           DISPLAY RESULT-DISP, LINE 20, POSITION 43.
           STOP RUN.
       ENTER-DATA.
           ACCEPT ENTRY, LINE LINE-NO, POSITION 52, PROMPT ".",
           CONVERT.
           MOVE ENTRY TO ENTRY-DISP.
           DISPLAY ENTRY-DISP, LINE LINE-NO, POSITION 52.
       CALCULATE-RATIO.
           DIVIDE VAR1 BY VAR2 GIVING RESULT.
           IF PERCENT=1, MULTIPLY RESULT BY 100 GIVING RESULT.
           MOVE RESULT TO RESULT-DISP.
       END PROGRAM.
```

those at Tandy, are preparing for that day. When the time arrives, COBOL will be waiting.■

```
FINANCIAL RATIOS CALCULATION

ENTER:

1. NET INCOME               $      900
2. NET SALES                $    6,500
3. CURRENT ASSETS           $   15,400
4. CURRENT LIABILITIES      $   20,800
5. QUICK ASSETS             $    7,600

PROFIT MARGIN= 13.84%
CURRENT RATIO=  0.74
QUICK RATIO  =  0.36
```

Program Listing 7.

ALPHA BYTE DOES IT AGAIN...

Call us for our never undersold price.

ALPHA BYTE STORES

69

(213) 883-8594

5115 Douglas Fir Road • Suite B • Calabasas, CA 91302

An introduction to an interpreter listing for the exciting new education language.

Pilot—The Language of Computer Aided Instruction

Randy Hawkins
6214 Hidden Cove
Corpus Christi, TX 78412

One of the most exciting applications for the home computer is computer aided instruction (CAI). Not only is the computer infinitely patient with the user, but the novelty of using a computer in learning, especially for children, keeps the user intrigued and involved during the entire lesson. A well written CAI program can both teach and entertain the student in virtually any subject area.

However, to communicate ideas and concepts to the student, the programmer must first organize his ideas and, in turn, communicate them to the computer. This involves an organized set of commands which make up a computer language. The TRS-80 uses BASIC which is both powerful and flexible, but takes time and effort to become proficient in. Often it takes months for the novice to become truly skilled in BASIC programming. This presents a problem; an expert in foreign language, for example, would be best suited for preparing a CAI

Program Listing 1. BASIC Program to Create the Pilot Interpreter.

```
10 CLEAR1000:CLS:PRINT"THIS PROGRAM WILL PREPARE A SYSTEM TAPE T
HAT CONTAINS THE":PRINT"PILOT INTERPRETER FOR THE TRS-80.  THE I
NFORMATION IS NOW BEING":PRINT"PREPARED ..."
20 LN=90:FORI=1TO26:CS=0:LN=LN+10
30 FORJ=1TO8:READX:A$=A$+CHR$(X):CS=CS+X:NEXTJ:READXX:IFCS<>XXTH
ENPRINT"CHECKSUM ERROR IN LINE";LN;"-- CHECK YOUR ENTRIES.":STOP

40 NEXTI:FORI=1TO25:CS=0:LN=LN+10
50 FORJ=1TO8:READX:B$=B$+CHR$(X):CS=CS+X:NEXTJ:READXX:IFCS<>XXTH
ENPRINT"CHECKSUM ERROR IN LINE";LN;"-- CHECK YOUR ENTRIES.":STOP

60 NEXTI:A$=A$+"":ML$=CHR$(229)+CHR$(205)+CHR$(127)+CHR$(10)+CHR
$(205)+CHR$(132)+CHR$(2)+CHR$(6)+CHR$(3)+CHR$(126)+CHR$(205)+CHR
$(100)+CHR$(2)+CHR$(35)+CHR$(254)+CHR$(0)+CHR$(32)+CHR$(245)+CHR
$(16)+CHR$(245)+CHR$(205)+CHR$(248)+CHR$(1)+CHR$(225)+CHR$(201)
70 POKE16526,PEEK(VARPTR(ML$)+1):POKE16527,PEEK(VARPTR(ML$)+2):X
X=PEEK(VARPTR(A$)+1)+256*PEEK(VARPTR(A$)+2):XX=XX+65535*(XX>3276
7)-1
75 PRINT:PRINT"WHAT SIZE COMPUTER ARE YOU":INPUT"MAKING THIS TAP
E FOR (4K,16K,32K)";MS$:IFMS$="16K"THEN90
80 IFMS$="4K"THENPOKEXX+12,50:POKEXX+13,48:POKEXX+14,49:POKEXX+1
5,55:POKEXX+16,55:POKEXX+18,42:POKEXX+24,78:POKEXX+25,60:POKEXX+
29,78:POKEXX+32,78:POKEXX+44,78:POKEXX+162,79:POKEXX+267,79:POKE
XX+295,79:POKEXX+158,26:POKEXX+291,150:POKEXX+400,83:GOTO90
85 IFMS$="32K"THENPOKEXX+12,52:POKEXX+13,56:POKEXX+14,56:POKEXX+
15,52:POKEXX+16,57:POKEXX+18,58:POKEXX+24,190:POKEXX+25,172:POKE
XX+29,190:POKEXX+32,190:POKEXX+44,190:POKEXX+162,191:POKEXX+267,
191:POKEXX+295,191:POKEXX+158,106:POKEXX+291,118:POKEXX+400,195
86 IFMS$<>"32K"THEN75
90 XX=XX+1:PRINT:PRINT"PREPARE YOUR CASSETTE BY PLACING IT IN TH
E RECORD MODE.":INPUT"PRESS ENTER WHEN YOU ARE READY TO BEGIN";Z
Z$:XX=USR(XX)
```

Program continues

"Pilot is amazingly simple, allowing experienced programmers and non-computerists alike to prepare useful programs."

program in this area, but is not likely to devote the time and energy necessary to learn BASIC programming.

Enter the Pilot language. Pilot (Programmed Inquiry Learning Or Teaching) is written expressly for computer aided instruction applications. The structure and commands of Pilot are extremely versatile and, above all, easily learned in a single session.

Pilot was developed by Dr. John Starkweather in the mid-70's. From its origins, as a method of teaching pharmacology to medical students, it has expanded to many different dialects and systems. The version presented in this article implements most of the standard features of Pilot and also utilizes some of the best features of BASIC as well as Level II text editing capabilities. Unlike several other TRS-80 Pilot interpreters, this Pilot interpreter makes it very simple to construct, load, save and execute CAI programs.

The Pilot interpreter is a machine language program which resides in the highest 300 bytes of memory. However, absolutely no knowledge of machine language is necessary to use Pilot. A BASIC program is presented which will construct a System tape containing the machine language Pilot program and even teach you how to load it. An assembly language listing of the interpreter is also included for those who may wish to study the program and possibly improve it. Finally, a Pilot program is presented to teach you the finer points of Pilot programming. This is the ultimate use of Pilot—a Pilot program to teach Pilot.

An Introduction to the Language

Let us first begin with an introduction to the Pilot language. Pilot is strictly a dialogue-oriented computer language. It will not balance your checkbook, solve trigonometry problems, or prepare a mailing list. It does, however, deal in interactive question-answer exchanges which are required for CAI applications. Pilot is amazingly simple, allowing experienced programmers and non-computerists alike to prepare useful programs. We will assume you know absolutely nothing about computer programming.

When programming the computer, you must first decide how to present the information to the student along with what information is to be expected of the students in response to the computer's questions. The program might begin by presenting a paragraph of information, and then asking a series of questions to see if the student understands. This sequence can be reduced to the following steps:

- Type the introductory message on the screen for the student to study and learn.
- Clear the screen and ask a question to see how well the student has learned the material.
- Compare the student's response with the correct answer. If the answer is correct, proceed to the next section or question.
- Compare the student's answer with some incorrect responses and explain why, in each case, the answer was wrong. If a complete review is necessary, return to step 1. If a second try is appropriate, return to step 2.

If you understand the process illustrated above, then preparing CAI programs using Pilot should be very easy. All Pilot commands are represented by a single letter. The letters correspond to easily remembered phrases as listed below:

T—Type the following message on the TV screen.

A—Ask the student the following question and wait for an answer.

M—Match the student's answer with the list of valid responses and decide whether or not they match.

J—Jump to the listed step number.

E—End this program.

C—Clear the screen before typing this message.

W—Clear the screen and type the following message in wide letters.

A Pilot command consists of one of the above command letters, followed by a quotation mark and any text needed with that command. The quotation mark is a special delimiter which separates the command letter and the rest of the statement. (Some versions of Pilot use the colon.) Only one quotation mark can be found in one Pilot statement; if quotes are needed you can substitute apostrophes for them.

Every Pilot statement must also have a line number. These numbers correspond to the step numbers used in the above discussion. The first step must have the lowest line number, the second step the second line number, and so on. When programming use multiples of 10. This way you may add

```
100 DATA85,80,73,76,79,84,32,60,569
110 DATA6,232,65,51,50,52,54,53,563
120 DATA0,45,60,2,22,64,152,126,471
130 DATA108,60,128,152,126,33,255,126,988
140 DATA34,143,65,33,227,3,34,22,561
150 DATA64,33,182,126,205,167,40,62,879
160 DATA13,205,42,3,175,33,231,65,767
170 DATA195,192,0,28,31,84,82,83,695
180 DATA45,56,48,32,80,73,76,79,489
190 DATA84,32,73,78,84,69,82,80,582
200 DATA82,69,84,69,82,32,38,34,490
210 DATA0,205,201,1,24,121,205,201,958
220 DATA1,62,23,205,42,3,24,111,471
230 DATA35,126,254,34,32,91,229,197,998
240 DATA35,126,254,0,40,3,205,167,830
250 DATA40,205,179,27,254,1,40,18,764
260 DATA193,225,24,74,42,164,64,14,800
270 DATA89,213,209,58,64,56,254,4,947
280 DATA32,1,118,126,254,0,35,32,598
290 DATA5,126,254,0,40,170,60,128,783
300 DATA24,127,244,35,94,35,86,213,858
310 DATA35,126,254,89,40,34,254,78,910
320 DATA40,30,254,84,40,38,254,65,805
330 DATA40,179,254,77,40,91,254,74,1009
340 DATA40,98,254,69,40,82,254,87,924
350 DATA40,153,254,67,40,144,24,56,778
360 DATA185,40,213,35,126,254,0,32,885
370 DATA250,35,24,179,35,126,254,34,937
380 DATA32,38,229,197,35,126,254,0,911
390 DATA40,21,254,64,40,6,205,42,672
400 DATA3,35,24,241,229,33,232,65,862
410 DATA205,167,40,225,35,24,230,62,988
420 DATA13,205,42,3,193,225,24,203,908
430 DATA33,242,127,205,167,40,225,205,1244
```

Program continues

"All Pilot commands are represented by a single letter."

```
440 DATA154,10,205,189,15,205,167,40,985
450 DATA118,14,89,35,126,254,34,32,702
460 DATA231,229,246,60,104,152,127,24,1173
470 DATA48,35,126,254,34,32,222,35,786
480 DATA197,205,90,30,42,164,64,125,917
490 DATA180,40,27,78,35,70,197,35,662
500 DATA126,187,40,3,225,24,240,35,880
510 DATA126,186,40,3,225,24,232,43,879
520 DATA43,43,193,193,24,137,193,24,850
530 DATA180,35,17,232,65,126,254,47,956
540 DATA40,28,254,0,40,24,235,70,691
550 DATA235,184,32,4,35,19,24,237,770
560 DATA35,126,254,0,40,6,254,47,762
570 DATA40,223,24,244,14,78,225,24,872
580 DATA138,69,82,82,79,82,32,73,637
590 DATA78,32,76,73,78,69,0,131,537
600 DATA120,0,0,0,0,0,0,0,120
700 PRINT:PRINT"THE MACHINE LANGUAGE PROGRAM HAS BEEN SAVED ON Y
OUR TAPE.":INPUT"PRESS ENTER WHEN YOU ARE READY TO LEARN HOW TO
USE IT";ZZ$
710 CLS:PRINT"MEMORY SIZE?":PRINT@704,"THIS IS THE WAY THE SCREE
N LOOKS AFTER YOU FIRST TURN ON THE":PRINT"TRS-80.  AT THIS POIN
T, YOU SHOULD JUST PRESS ENTER -- NO NUMBERMUST BE ENTERED TO PR
ESERVE MEMORY.  PRESS THE ENTER KEY NOW.":PRINT@13,CHR$(95)
720 IFINKEY$<>CHR$(13)THEN720ELSEPRINT@13," ":PRINT"RADIO SHACK
LEVEL II BASIC":PRINT"READY":PRINT">";CHR$(95);CHR$(31)
730 PRINT@704,"THE PILOT TAPE IS A MACHINE LANGUAGE PROGRAM SO Y
OU MUST USE":PRINT"THE SYSTEM COMMAND TO ENTER IT. AT THIS POINT
 YOU WOULD TYPE THEWORD 'SYSTEM' LIKE THIS ..."
740 PRINT@194,;:A$="SYSTEM":FORI=1TO6:FORTI=1TO500:NEXTTI:PRINTC
HR$(24);MID$(A$,I,1);CHR$(95);:NEXTI:PRINT@862,"AND NOW PRESS EN
TER ..."
750 IFINKEY$<>CHR$(13)THEN750ELSEPRINT@199," ":PRINT@320,"*? ";C
HR$(95);CHR$(31);:PRINT@704,"THE COMPUTER IS WAITING FOR THE NAM
E OF THE PROGRAM YOU WISH TO":PRINT"LOAD -- WHICH IS OF COURSE '
PILOT'.  YOU SHOULD TYPE IN THE WORD'PILOT' LIKE THIS ...";
760 PRINT@324,;:A$="PILOT":FORI=1TO5:FORTI=1TO500:NEXTTI:PRINTCH
R$(24);MID$(A$,I,1);CHR$(95);:NEXTI:PRINT@862,"AND THEN PRESS EN
TER ..."
770 IFINKEY$<>CHR$(13)THEN770ELSEPRINT@329," ";CHR$(31):PRINT@70
4,"THE RECORDER WILL TURN ON (IF YOU HAVE IT IN THE PLAY SETTING
)":PRINT"AND LOAD THE SHORT PROGRAM.  THE ASTERISKS WILL BLINK L
IKE THIS.":FORTI=1TO500:NEXTTI
780 PRINT@62,"**";:FORI=1TO7:FORTI=1TO100+RND(200):NEXTTI:PRINT@
63," ";:FORTI=1TO100+RND(200):NEXTTI:PRINT@63,"*";:NEXTI
790 CLS:PRINT"TRS-80 PILOT INTERPRETER &":PRINT"RADIO SHACK LEVE
L ii BASIC":PRINT"READY":PRINT">";CHR$(95):PRINT@704,"THE SCREEN
 WILL IMMEDIATELY CLEAR LEAVING A DISPLAY LIKE THE":PRINT"ONE AB
OVE.  THE PILOT INTERPRETER IS NOW ACTIVATED AND READY TO"
800 PRINT"USE. THE COMMAND 'NAME' EXECUTES THE PILOT PROGRAM.":F
ORTI=1TO1000:NEXTTI:PRINT@384,"WOULD YOU LIKE TO SEE THESE INSTR
UCTIONS AGAIN (Y/N) ?"
810 ZZ$=INKEY$:IFZZ$="Y"THEN710ELSEIFZZ$="N"THEN820ELSE810
820 PRINT"ARE YOU READY TO TRY YOUR PILOT TAPE (Y/N) ?"
830 ZZ$=INKEY$:IFZZ$="Y"THEN840ELSEIFZZ$="N"THENENDELSE830
840 POKE16526,0:POKE16527,0:X=USR(0)
```

Program Listing 2. Pilot Program to Teach Pilot Programming.

```
10 W"P I L O T  8 0 :
T H E  L A N G U A G E   O F
C O M P U T E R   A I D E D
I N S T R U C T I O N . . .

20 T"PILOT IS A LANGUAGE THAT WAS
CREATED BY DR. JOHN STARKWEATHERIN THE MID-70'S AND INTENDED TO
BE THE IDEAL METHOD OF USING THECOMPUTER AS A TEACHING DEVICE.
```

Due to a printing error, the backward apostrophes in lines 310-330 should be SHIFT @s.

Program continues

statements later between steps if it becomes necessary.

Let's look at the shortest possible example program in Pilot. To print a simple message, a program of this type is needed: 10 T"THIS IS A SAMPLE PROGRAM. Note the line number, the command letter T, the quote mark and the statement text. When executed by Pilot, it will print the message and then stop with no further commands found.

To clear the screen and print the message in the upper left corner, type: 10 C"THIS IS A SAMPLE PROGRAM. Similarly, 10 W"THIS IS A SAMPLE PROGRAM. will clear the screen and print the message in wide letters (32 characters per line) in the upper left.

Although it is not required, one could add a statement to signal the end of the program. 10 T"THIS IS A SAMPLE PROGRAM. 20 E".

These three statements—T, C and W—are very useful for displaying information on the screen. In practice, more than half of the statements in a Pilot program will probably be one of these three varieties.

To ask the student a question, use the A, for ask. Since the A command will always be a question, the question mark need not be typed. For example, 10 A"WHAT IS YOUR NAME will print "WHAT IS YOUR NAME?" on the screen and wait for the student to type in his answer and then Enter. The A command can also be used to stop the program and wait for the student to Enter to continue. For example, 10 A"ARE YOU READY TO CONTINUE will print the question and also wait for the student to Enter. Until this key is pressed, the computer will wait indefinitely for that response.

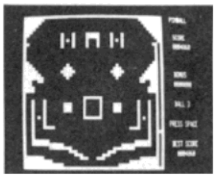
When the A statement is used to halt program execution, the answer the student gives is not important. But when a specific question is asked, the student's answer is of prime importance. The input answer is stored in a special portion of the TRS-80's memory, and until another question is asked, the student's answer stays untouched in that memory location. The most recent answer can be echoed back to the student by using the @ symbol in a T statement. Whenever the computer encounters the @ symbol, it is replaced by the characters in the most recent answer. For example, 10 A"WHAT IS YOUR NAME, 20 T"HELLO @, WELCOME TO PILOT.

The first statement asks "WHAT IS YOUR NAME?" and waits for user response. The answer is stored, and when the Type command in line 20 is executed, the answer is

ACORN ARCADE!

FOR TRS-80*

NOW FOR TRS-80* MODEL III !



PINBALL

By John Allen

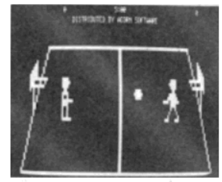
Get your flipper fingers ready for action in this real-time machine language game.

Lots of sound and flashing graphics make this fast action game so much like the real thing that you'll have to remind yourself not to shake your TRS-80.* Choose from five playing speeds to match your skill—but be prepared for a lot of practice if you ever hope to master the fastest speed.

Can you beat your friends' scores? Will you avoid the dreaded "Bermuda Square?" Get PINBALL today and find out. Available for $14.95 on tape or $20.95 on disk.

BASKETBALL

By John Allen

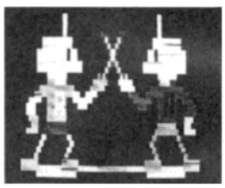


Fast machine language action game, with sound, from the author of the acclaimed "PINBALL"!

Try to keep up with the action and outscore your opponent in five minutes of one-on-one basketball. Steal the ball, duck around your opponent and slant toward the basket for a lay up! The graphics are based on a 3-dimensional depiction of a basketball court, and ball dribbling sounds add to the realism. Compete against a friend or your computer.

It's all there but the cheers—on tape for only $14.95, disk version for $20.95.

DUEL -N- DROIDS

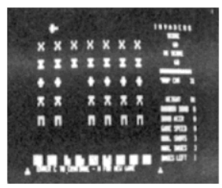


By Leo Christopherson

Your 'droid has already learned NIM, so now it's time to teach it how to wield a laser sword! For DUEL-N-DROIDS, a new type of animation and high-quality sound has been developed by Leo Christopherson, author of "Android NIM," "Dancing Demon" and other animations.

Your 'droid starts out as a lowly clown. You teach it how to use a laser sword by controlling its movements. After it attains the rank of "Grand Master," enter it in the tournament against the program. Entertainment for all ages!

Available for $14.95 on tape, $20.95 for disk.



INVADERS FROM SPACE

By Carl Miller

A new and faster machine language approach to this classic (and addictive) space game. As you play, the aliens drop bombs, move from side to side, and try to overrun your bases. All you have to do is shoot them down. It's easy—until they speed up their action!

You choose the speed, enemy bomb frequency and accuracy, and how many shots and bases you have. Unlike other games of this type, you can move your base and simultaneously fire at the invaders. Fun for all ages and skill levels, it has full sound effects for even more excitement.

Available for only $14.95 on tape or $20.95 on disk.

Available now from these and other fine Acorn dealers

ALL ACORN ENTERTAINMENT SOFTWARE IS NOW AVAILABLE FOR TRS-80* MODEL III !

ADVENTURE INTERNATIONAL
Box 3435
Longwood, FL 32750

COMPUTER CENTER
28251 Ford Rd.
Garden City, MI 48135

DIGIBYTE COMPUTER CENTER
31 East 31st St.
New York, NY 10016

E.B. GARCIA & ASSOCIATES
203 North Wabash
Chicago, IL 60601

HOBBY WORLD ELECTRONICS
19511 Business Center Dr.
North Ridge, CA 91324

H & E COMPUTRONICS
50 North Pascack
Spring Valley, NY 10977

MICRO MANAGEMENT SYSTEMS
115-C Second Avenue, S.W.
Cairo, GA 31728

THE PROGRAM STORE
4200 Wisconsin Ave.
Washington, DC 20016
and
W. Bell Plaza
6600 Security Blvd.
Baltimore, MD 21207

Acorn
Software Products, Inc.
(202) 544-4259
634 North Carolina Avenue, S.E., Washington, D.C. 20003

34

*TRS-80 is a trademark of Tandy Corp.

```
30 T"THERE ARE ONLY A FEW COMMANDS
USED WITH PILOT, BUT THEY ARE
POWERFUL ENOUGH TO CREATE INTER-ACTIVE LEARNING PROGRAMS THAT
ARE EFFECTIVE FOR THE USER AND
40 A"EASY TO CREATE. PRESS ENTER
50 W"PILOT IS TEXT ORIENTED RATHER
THAN MATHEMATICALLY ORIENTED SO
THAT ALMOST ANY SUBJECT AREA CANBE PRESENTED IN A DIALOGUE-TYPE
APPROACH.  THE COMMANDS ARE
SIMPLE AND EASILY LEARNED, SO
THAT THOSE UNFAMILIAR WITH
60 T"HIGHER LEVEL LANGUAGES CAN
QUICKLY ADVANCE PAST THE DETAILSOF PROGRAMMING AND CONCENTRATE
ON THE INFORMATION TO BE TAUGHT
TO THE STUDENT.
70 T"THIS TAPE WILL TEACH YOU PILOT
PROGRAMMING BY USING THE PILOT
LANGUAGE ITSELF.
80 A"PLEASE PRESS ENTER TO BEGIN
90 C"PILOT PROGRAMS USE ONLY 9 SPECIAL COMMAND SYMBOLS: T,A,M,J,
E,
C,W,Y, AND N.  THE SYMBOLS CORRESPOND TO EASILY REMEMBERED
COMMAND WORDS AS SUMMARIZED BELOW:
        T = TYPE          A = ASK          M = MATCH
100 T"       J = JUMP          E = END          C = CLEAR SCREEN
        Y = YES           N = NO           W = WIDE LETTERS
110 T"PILOT PROGRAMS (USING THIS INTERPRETER) USE LINE NUMBERS.
 EACH
PROGRAM STATEMENT IS NUMBERED IN ANY MANNER YOU WISH.  USING THE
AUTO COMMAND OF THE TRS-80, YOU AUTOMATICALLY GET LINE NUMBERS
120 T"STARTING WITH 10 AND INCREMENTING BY 10.  THE PILOT
PROGRAM WILL FOLLOW THESE NUMBERS WHEN IT IS RUN -- DOING THE
INSTRUCTIONS AT LINE 10 FIRST, THEN THOSE AT LINE 20, LINE 30,
AND SO ON.  IN ADDITION, IF YOU WANT THE PROGRAM TO JUMP PAST
130 T"A SECTION OR JUMP BACK TO A PREVIOUS COMMAND WE WILL SEE
HOW THE JUMP COMMAND LOOKS FOR THE LINE NUMBER YOU SPECIFY.
140 A"PLEASE PRESS ENTER TO CONTINUE
150 C"PILOT STATEMENTS ALL LOOK BASICALLY THE SAME. THERE IS ALW
AYS
A LINE NUMBER, FOLLOWED BY A COMMAND LETTER, A QUOTATION MARK,
AND THE INSTRUCTIONS OR MESSAGE TO BE DISPLAYED. FOR EXAMPLE,
160 T"THIS SIMPLE ONE LINE PROGRAM WOULD DISPLAY THE MESSAGE
<<THIS IS THE PILOT LANGUAGE.>>
           10 T"THIS IS THE PILOT LANGUAGE.
170 T"THIS INTRODUCES THE T FOR TYPE COMMAND. WHATEVER MESSAGE I
S
FOUND AFTER THE QUOTATION MARK WILL BE TYPED OUT OR DISPLAYED ON
THE VIDEO SCREEN.  THERE MUST BE NO EXTRA SPACES BETWEEN THE
180 T"LETTER T AND THE QUOTATION MARK. ALSO, THE QUOTATION MARK
MUST NOT BE USED IN YOUR MESSAGE.  IT CAN ONLY BE USED ONCE --
TO DESIGNATE THE BEGINNING OF THE MESSAGE TO BE DISPLAYED. YOU
190 T"CAN USE SUBSTITUTE QUOTE MARKS, SUCH AS <<THESE>>.
200 A"
PLEASE PRESS THE <<ENTER>> KEY TO CONTINUE YOUR INSTRUCTIONS
210 C"

THE C AND THE W COMMANDS ARE ACTUALLY ALTERNATE TYPE COMMANDS
WITH SPECIAL FEATURES.  THE C COMMAND CLEARS THE SCREEN AND THEN
TYPES YOUR MESSAGE AT THE TOP LEFT HAND CORNER. FOR EXAMPLE,
220 T"  10 C"THIS WILL ERASE THE SCREEN PLUS TYPE THIS MESSAGE.
230 T"
THE W COMMAND WILL DO ALL OF THE ABOVE PLUS CONVERT THE DISPLAY
TO THE WIDE LETTERS (32 CHARACTERS PER LINE). FOR EXAMPLE,
  10 W"THIS WILL ERASE PLUS TYPE THE MESSAGE IN WIDE STYLE.
240 A"
PLEASE PRESS ENTER WHEN YOU HAVE STUDIED THE ABOVE INFORMATION
250 W"THIS IS AN EXAMPLE OF THE WIDE
LETTER OUTPUT.  THE A COMMAND
STANDS FOR <<ASK THIS QUESTION>>AND WAIT FOR A REPLY.  AFTER
260 T"TYPING YOUR MESSAGE THE TRS-80
WILL TYPE A QUESTION MARK AND
```

Program continues

```
ALLOW THE USER TO TYPE IN AN
ANSWER.  FOR EXAMPLE,
270 T"
  10 A"WHAT IS YOUR ANSWER
280 T"
WILL TYPE THE MESSAGE AND A
QUESTION MARK AND ALLOW THE USERTO TYPE IN HIS ANSWER AND PRESS
290 A"ENTER WHEN COMPLETED.
PRESS ENTER TO CONTINUE
300 C"WHAT HAPPENS TO THE ANSWER TYPED IN BY THE USER?  IT IS ST
ORED
IN A SPECIAL AREA OF THE COMPUTER'S MEMORY AND UNTIL ANOTHER
QUESTION IS ASKED IT STAYS IN THAT SPECIAL MEMORY LOCATION.
310 T"THE PROGRAM CAN RE-PRINT THE USER'S ANSWER BY USING THE `
SIGN IN THE PROGRAM.  WHEN THE ` SIGN IS FOUND IN A TYPE
STATEMENT, THE MOST RECENT USER'S ANSWER IS PRINTED RATHER THAN
THE ` SIGN.  FOR EXAMPLE,
320 T"      10 A"WHAT IS YOUR ANSWER
325 T"      20 T"YOUR ANSWER OF ` IS RIGHT!
326 T"BE VERY CAREFUL WHEN TYPING -- THE ` SYMBOL AND THE SHIFT-
`
ARE TWO DIFFERENT CHARACTERS TO THE COMPUTER.  DO NOT
ACCIDENTALLY HOLD THE SHIFT KEY DOWN WHILE TYPING THE ` SYMBOL.
330 T"WHATEVER THE USER TYPES IN RESPONSE TO THE QUESTION OF LIN
E 10
WILL BE ECHOED BACK IN LINE 20 IN PLACE OF THE ` SYMBOL.
340 A"PLEASE PRESS ENTER TO CONTINUE
350 C"THE NEXT QUESTION IS HOW DO WE USE THE ANSWER SUPPLIED BY
THE
USER -- HOW DO WE CHECK TO SEE IF IT IS RIGHT OR WRONG?  THIS IS
DONE BY THE MATCH STATEMENT REPRESENTED BY THE M COMMAND.
360 T"AFTER ASKING A QUESTION BY USING THE A COMMAND, WE CAN CHE
CK
TO SEE IF IT MATCHES A LIST OF POSSIBLE RESPONSES. FOR EXAMPLE,

     10 A"WHAT IS YOUR ANSWER
     20 M"YES/OF COURSE/SURE/OK

370 T"THIS SIMPLE PROGRAM WAITS FOR THE USER'S ANSWER.  IT THEN
COMPARES HIS RESPONSE TO THE LIST OF VALID RESPONSES.  AS YOU
380 T"CAN SEE, EACH DIFFERENT RESPONSE IS SEPARATED BY A / AND A
NY
NUMBER OF ANSWERS CAN BE INCLUDED AS LONG AS SLASHES SEPARATE
THEM.  SO THE ANSWER MATCHES ONE OF THE CORRECT RESPONSES --
390 A"WHAT NEXT ?  PRESS ENTER TO CONTINUE
400 C"IF THE USER ANSWER MATCHES ANY ONE OF THE ITEMS IN THE MAT
CH
STATEMENT, A 'FLAG' IS SET TO 'YES'.  IF THE ANSWER DOES NOT
MATCH ANY ITEM, THE 'NO FLAG' IS SET.  ONCE A 'FLAG' IS SET
410 T"IT REMAINS AT THAT VALUE UNTIL THE NEXT MATCH STATEMENT.
WE NOW INTRODUCE TWO NEW COMMAND LETTERS -- Y AND N -- FOR YES
AND NO.  WHEN THE Y LETTER IS FOUND AT THE BEGINNING OF A
420 T"PILOT STATEMENT, IT MEANS 'IF THE YES FLAG HAS BEEN SET TH
EN
DO THIS STATEMENT -- OTHERWISE SKIP THIS STATEMENT AND MOVE TO
THE NEXT NUMBERED STATEMENT'.  SIMILARLY, THE N COMMAND MEANS
'IF THE NO FLAG HAS BEEN SET DO IT -- ELSE SKIP IT'.
425 A"
WOULD YOU LIKE TO SEE AN EXAMPLE OF THE Y AND N STATEMENTS
430 C"      10 A"WHAT IS YOUR ANSWER
     20 M"TEXAS
     30 YT"THAT IS RIGHT!
     40 NT"NO, THAT IS WRONG!
440 T"
IF THE USER TYPES IN 'TEXAS' IN RESPONSE TO THE QUESTION OF
LINE 10, THE COMPUTER WILL WRITE THE MESSAGE OF LINE 30; IF IT
450 T"DOES NOT MATCH, IT WRITES THE MESSAGE OF LINE 40. ANY
STATEMENT CAN BE USED IN CONJUNCTION WITH THE Y AND N COMMANDS.
AFTER DECIDING IF THE CORRECT FLAG HAS BEEN SET, THE STATEMENT
IS EITHER EXECUTED NORMALLY OR SKIPPED COMPLETELY.
455 A"
PRESS ENTER TO CONTINUE
```

Program continues

✓*See List of Advertisers on page 306*

"Any Pilot statement can be either executed or skipped depending on the current value of the yes/no flag."

```
460 C"
THE FINAL PILOT COMMAND IS J FOR JUMP. A JUMP STATEMENT MAY LOOK
LIKE THIS
     10 J"30
470 T"THIS WOULD TELL THE COMPUTER TO JUMP AHEAD TO THE SET OF
INSTRUCTIONS BEGINNING AT LINE 30.  THE PILOT STATEMENT
     50 YJ"90
480 T"WOULD JUMP TO LINE 90 IF THE YES FLAG HAD BEEN SET.
490 A"PRESS ENTER TO CONTINUE
500 C"HERE ARE SEVERAL OTHER MISCELLANEOUS HINTS AND TIPS:

> ALL STANDARD LEVEL ii TEXT EDITING COMMANDS SHOULD BE USED
TO ENTER YOUR PILOT PROGRAM.  THESE INCLUDE LIST, AUTO, EDIT,
NEW, AND DELETE.  SEE THE LEVEL II MANUAL FOR DETAILS.
510 T"
> THE PILOT PROGRAM IS EXECUTED WITH THE 'NAME' COMMAND. THIS
COMMAND IS NOT NORMALLY USED BY THE BASIC INTERPETER.  REMEMBER
THAT BASIC IS STILL AVAILABLE AND USES THE 'RUN' COMMAND.
520 T"
> THE PILOT PROGRAMS CAN BE SAVED AND LOADED USING THE CSAVE,
CLOAD, AND CLOAD? COMMANDS OF LEVEL ii BASIC.  AGAIN, SEE THE
LEVEL ii MANUAL FOR DETAILS.
530 A"
```

Program continues

replaced at the location of the @ symbol.

The answer can be compared to a list of other alphanumeric characters by using the Match command: 10 A"WHAT TYPE OF COMPUTER DO YOU HAVE 20 M"TRS-80. The answer supplied in line 10 is compared to the value in line 20; that is, TRS-80. If the student's answer matches the word or words or numbers in the M statement exactly, then a yes flag is set. If it does not match, a no flag is set. The response to the user's answer is remembered until another Match statement is found. More than one answer may be valid. Multiple answers can be handled by separating them with slashes: 20 M"TRS—80/ATARI/APPLE. If any one answer matches, the computer remembers yes.

Any Pilot statement can be either executed or skipped depending on the current value of the yes/no flag by adding the letter Y or N in front of the Pilot command letter. If

The Resurrection of IAGO T.M.

It is the hour. The clanging of swords is silent—only far off sounds of men struggling are heard. Othello is down. IAGO stands astride—beckoning to all challengers.

IAGO, the classical Shakespearean antagonist to Othello has returned as the ultimate Reversi/Othello game. IAGO contains one of the most powerful strategies ever developed for a Microcomputer system.

IAGO is for the novice to advanced player, and with 7 levels of play that will provide years of challenging enjoyment. So, if computers are your game then ask your Datasoft dealer for a demonstration or contact us for details.

TRS-80 Models I & III
(Diskette) $24.95
(Cassette) $19.95

ARCADE 80

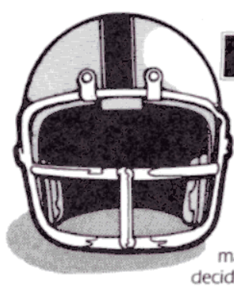
Three high quality arcade games in one simple format. This is home entertainment at its best.

Game 1: Falling Bricks—Balances accuracy against time. If you're not quick enough the bricks come crashing down on you.

Game 2: AstroMines—Seek and destroy the enemy's mines before fuel runs out. Be careful or you will destroy yourself.

Game 3: Star Run—A Star Raiders type game filled with refreshing graphics.

TRS-80 Model I (Diskette) $24.95 **(Cassette)** $19.95



FOOTBALL CLASSICS

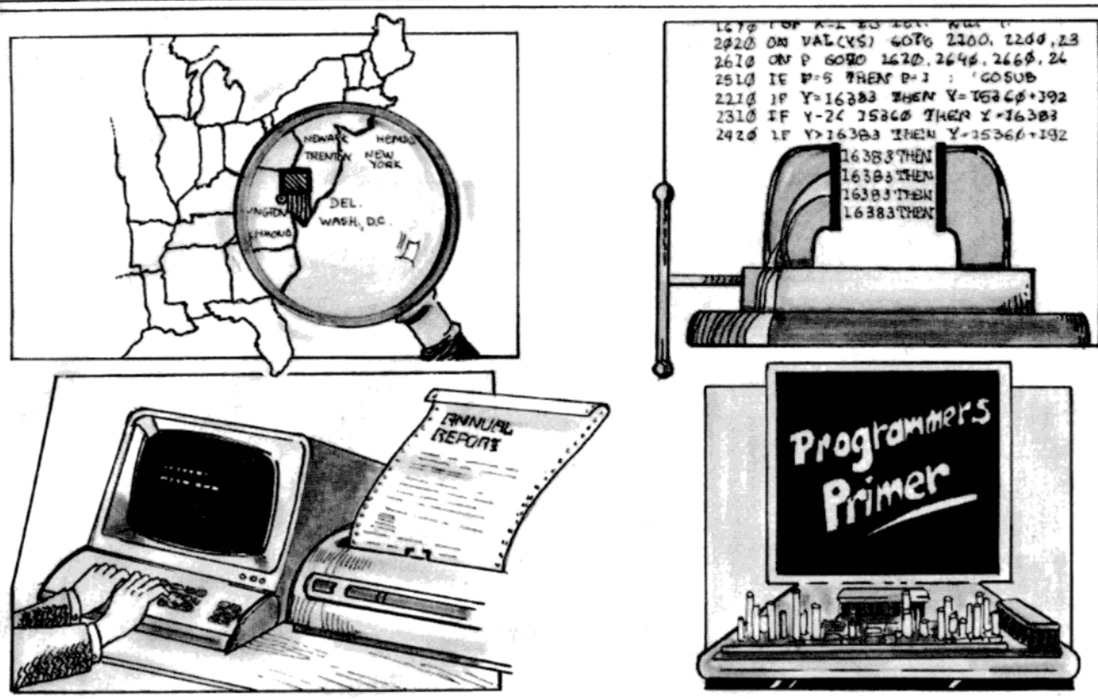
Goes beyond ordinary computer football programs. You can recreate sports history by pitting top teams against each other in their best years. You make the plays and the statistics decide the outcome. And that's not all!!! With the easy to use updated input file, you can update or create your own statistics. Included with the program are six teams of the past.

TRS-80 Models I & III (Diskette) $24.95 **(Cassette)** $19.95

Datasoft Inc. T.M.
COMPUTER SOFTWARE

407

16606 Schoenborn Street, Sepulveda, Calif. 91343—(213) 894-9154



Four New Programs in July for Your TRS-80*

GEOGRAPHY EXPLORER: USA

The Geography Explorer: USA package is the most fascinating (and least painful) way of learning social studies that we've seen yet.

The program displays computer generated maps of the United States, its seven regions, and its individual states. The student then answers questions about the states of a given region. These questions may be in multiple choice recognition of fill-in format.

Geography Explorer has a unique TEACHER mode which allows the teacher or parent to choose multiple options of how the material is to be presented. This permits directed learning for the student.

The package is also just plain fun! When the student answers the questions correctly, there are a variety of graphic rewards that flash on the screen to provide immediate, positive reinforcement.

As a bonus, this educational package can use a compatible Light Pen! The union of sophisticated software with the speed and novelty of the Light Pen represents a milestone in Computer Assisted Instruction.

Order No. 0071RD (Model I, Level II 16K, E/I 16K + disk drive) $49.95 DISK

RENUM/COMPRESS

RENUM/COMPRESS is a machine-language utility which adds two commands to TRS-80 Level II BASIC. Occupying the top 767 bytes of memory, the program allows you to renumber and/or compress any BASIC program. In renumbering, all GOTO and GOSUB commands are re-addressed to the new line numbers. With simultaneous renumbering and compression, all lines are renumbered, GOTO and GOSUB commands are re-addressed, and the entire BASIC program is reduced in size by elimination of all spaces not enclosed in quotes and all REM statements. This program allows you to expand your number-to-number intervals to accommodate insertion of new lines, and then to compress your program to free up additional memory.

Order No. 0133R $14.95 TAPE

THE WORDSLINGER

At last . . . simplified word processing! The Wordslinger is an economical word processing program designed for the individual user or small business. The program has automatic formatting for letterhead stationary and envelope addressing. You can write and edit letters, reports, forms, even schoolwork!

One of the Wordslinger's major advantages is that you don't need disk drives or an expansion interface. The minimum system required is the TRS-80 Level II with 16K of RAM and appropriate printer (you may need an expansion interface if your printer can't operate with just a printer interface cable, such as Radio Shack's part #26-1411 or 26-1416).

Complete text editing is yours for the asking—just ask for the Wordslinger.

Order No. 0129R (Model I, Level II, 16K RAM) $29.95 TAPE

PROGRAMMER'S PRIMER

There is a well known cliche, "One picture is worth a thousands words." That time tested message has served as the impetus for this program that demonstrates, for the novice programmer, the logic used in six common computer operations. The program is designed to function as a CAI device.

The six routines presented include Decimal to Hexadecimal Conversion (and vice-versa); Subscription of a Variable and Three Dimensional Array data storage operations; the For-Next-Loops demonstration of logical program flow; and the Bubble Sort sorting routine.

All of these routines are clearly presented with on-screen step-by-step instructions and are displayed with a sequence of the logic used to arrive at the desired goal. Several of the routines can be modified to suit individual needs.

Order No. 0245R (Model I, Level II 16K, Model III 16K) $9.95 TAPE

Instant Software™

PETERBOROUGH, N.H. 03458

A DIVISION OF WAYNE GREEN INC.

✓531

To Order: See your local Instant Software dealer. If these programs are unavailable, order directly.

Call Toll-Free
1-800-258-5473

*A registered trademark of Tandy Corp.

"New programs can be constructed using the standard Level II BASIC editing commands."

the statement begins with Y and the previous match was a yes result, the rest of the statement will be executed normally. A statement with a Y prefix will be skipped if the no flag is set. Study the following example:

```
10 A"WHAT TYPE OF COMPUTER DO YOU HAVE
20 M"TRS-80/RADIO SHACK/LEVEL II/MODEL 1
30 YT"80 MICROCOMPUTING IS THE MAGAZINE FOR
YOU!
40 NT"SORRY, YOU ARE ON YOUR OWN.
```

If the student answers any of the phrases of line 20, the message of line 30 will be printed. If some other answer is entered, the message of line 40 appears.

A J(ump) causes the computer to jump to the statement indicated. The command 10 J"99 causes the computer to jump immediately to line 99. Similarly, 10 YJ"99 will jump to line 99 if the previous match statement

```
PLEASE PRESS ENTER TO SEE AN EXAMPLE PROBLEM
600 C"THIS IS AN EXAMPLE PROBLEM USING THE PILOT INTERPRETER.
610 A"FIRST OF ALL, WHAT IS YOUR NAME
620 T"OK, @, HERE IS YOUR QUESTION.  WHAT IS THE BEST
MAGAZINE FOR THE TRS-80 MICRO-COMPUTER --
        A = BETTER HOMES AND GARDENS
        B = PSYCHOLOGY TODAY
        C = 80 MICROCOMPUTING
630 T"CONSIDER THE CHOICES CAREFULLY @ AND TELL ME
640 A"IS YOUR ANSWER A , B , OR C
650 M"A
660 YT"ONLY IF YOU ARE USING YOUR COMPUTER AS A FLOWER POT! TRY
AGAIN
670 YJ"640
680 M"B
690 YT"COMPUTER PROGRAMMING MUST BE DRIVING YOU CRAZY! TRY AGAIN
.
700 YJ"640
710 M"C
720 YT"YOU ARE ABSOLUTELY CORRECT!  GOOD LUCK AND ENJOY USING PI
LOT.
730 YE"
740 T"YOU ARE NOT FOLLOWING THE DIRECTIONS!
750 J"640
```

"The use of a second set of quotation marks is totally unpredictable and is normally bad programming practice."

resulted in a yes flag.

Constructing New Programs

New programs can be constructed using the standard Level II BASIC editing commands, including Auto, Edit, New and Delete. They can be saved and loaded to cassette with the CSAVE, CLOAD and CLOAD? commands. Any error found in a program will be noted with an Error in the Line XX message.

Program Listing 1 is in BASIC and will prepare the machine language System tape that you will use at the beginning of every Pilot programming session. Type the program in as shown and run it. Follow the directions and prepare your new Pilot interpreter tape. The program also contains many double checks to make sure that there are no mistakes in your data statements. If you find a checksum error when running the program, go back and check the entries in the line number indicated.

It might be wise to devote an entire cassette to Pilot. On one side, keep the Pilot tape-maker. You can use this program to give others the Pilot interpreter. On the other side, you will create your own Pilot system tape. CSAVE the BASIC program before you actually make the Pilot interpreter. If you have made a mistake the computer will return all the way back to the memory size question. A backup copy on tape will keep you from losing the program in case of disaster.

Once you have prepared the Pilot interpreter program and followed the directions for entry, the screen should clear and TRS-80 Pilot Interpreter and Radio Shack Level II BASIC should be displayed along with the Ready prompt. Type in a simple one-line Pilot program, such as 10 T"WELCOME TO PILOT.

In BASIC we use the command Run to begin program execution. In Pilot we use the command Name. Type Name and Enter and you should see the message above printed. Why use the word Name? The TRS-80 recognizes about 100 different words. Among these are Print, For, Next, Edit, List and others. The TRS-80 also recognizes Name, but unlike the other commands, Name has no specific function assigned to it. As a result, Pilot can borrow the Name command without affecting normal BASIC execution.

Program Listing 2 when typed in and executed, will provide an introduction to Pilot. The program occupies about 8K of memory, so 4K owners will need to break the progam into smaller pieces.

Type the program in exactly as listed. To make an attractive display, the down arrow (or linefeed) is used at the end of lines. When encountered in the program, they accomplish the same thing you see on your video display—the printing jumps to the beginning of the next line. See, for example, lines 10, 20 and 30. In addition, there appear to be run-on words in line 20. However, when printed in wide characters as instructed in line 10, the display looks fine.

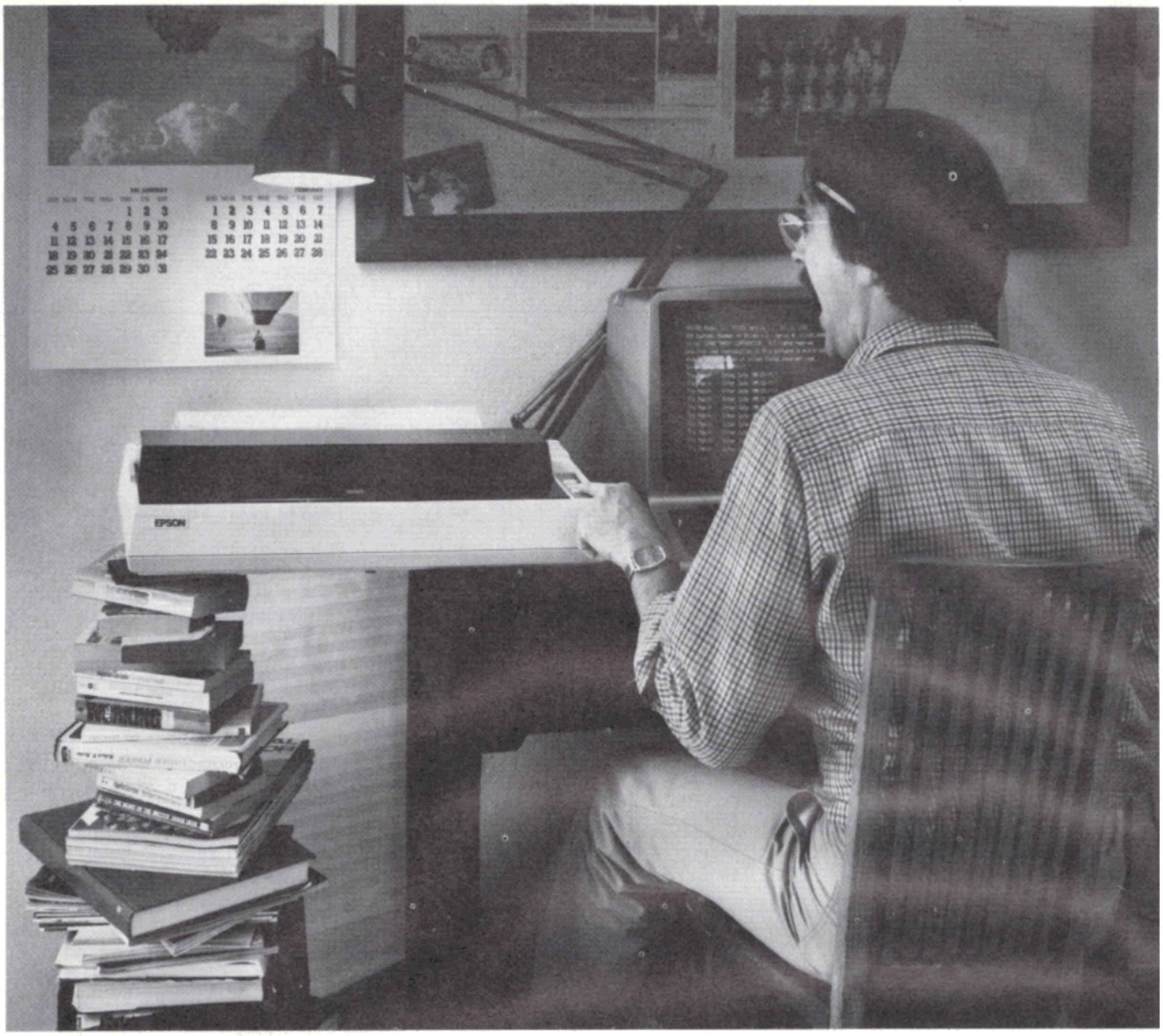
Broken Rules

Several rules are broken in the program, but for a good reason. One should not normally use a second set of quote marks in a Pilot program statement. You will find two sets in lines 160, 220 and 230. The use of a second set of quotation marks is totally unpredictable and is normally bad programming practice. Yet, in the lines above they are absolutely essential to illustrate a point and have been thoroughly tested and are

Program Listing 3. Pilot Interpreter Source Code.

```
        ORG     41E8H           ;SET MEMORY SIZE
        DEFM    '32465'
        DEFB    00H
        ORG     4016H           ;RESET KEYBOARD DRIVER TO
        DEFW    7E98H           ;INITIALIZATION ADDRESS
        ORG     7E98H
        LD      HL,BEGIN        ;RESET NAME COMMAND TO
        LD      (418FH),HL      ;PILOT INTERPRETER
        LD      HL,03E3H
        LD      (4016H),HL      ;RESET KEYBOARD DRIVER
        LD      HL,TITLE
        CALL    28A7H           ;WRITE SIGNON MESSAGE
        LD      A,0DH
        CALL    032AH
        XOR     A
        LD      HL,41E7H
        JP      00C0H           ;JUMP TO LEVEL II START
TITLE   DEFB    1CH
        DEFB    1FH
        DEFM    'TRS-80 PILOT INTERPRETER &'
        DEFB    00H
        ORG     7ED4H
CLS     CALL    01C9H           ;CLEAR SCREEN AND JUMP TO TYPE
        JR      TYPE
WIDE    CALL    01C9H           ;CLEAR SCREEN, SWITCH TO
        LD      A,17H           ;WIDE LETTERS AND JUMP TO
        CALL    032AH           ;TYPE ROUTINE
        JR      TYPE
ASK     INC     HL
        LD      A,(HL)          ;CHECK FOR QUOTES
        CP      22H
        JR      NZ,WRONG
        PUSH    HL
        PUSH    BC
        INC     HL
        LD      A,(HL)          ;IF THERE IS A MESSAGE,
        CP      00H             ;PRINT IT
        JR      Z,ASK2
        CALL    28A7H
ASK2    CALL    1BB3H           ;WAIT FOR USER RESPONSE
        CP      01H
        JR      Z,STRT1         ;IF BREAK KEY THEN STOP
        POP     BC
        POP     HL
        JR      SKIP
BEGIN   LD      HL,(40A4H)      ;BEGIN EXECUTION, SET HL AT
        LD      C,'Y'           ;BEGINNING OF TEXT, SET FLAG
        PUSH    DE
START   POP     DE
        LD      A,(3340H)       ;CHECK FOR BREAK KEY AND STOP
        CP      04H
        JR      NZ,STRT2
STRT1   HALT
```

Program continues



The MX-100. Not just better. Bigger.

Epson.

Our MX-80 was a pretty tough act to follow. I mean, how do you top the best-selling printer in the world?

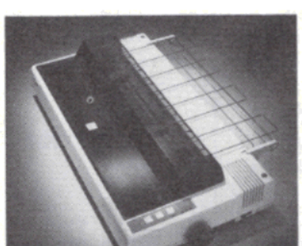
Frankly, it wasn't easy. But the results of all our sleepless nights will knock your socks off.

The MX-100 is a printer that must be seen to be believed. For starters, we built in unmatched correspondence quality printing, and an ultra-high resolution bit image graphics capability. Then we added the ability to print up to 233 columns of information on 15″ wide paper to give you the most incredible spread sheets you're ever likely to see. Finally, we topped it all off with *both* a satin-smooth friction feed platen *and* fully adjustable, removable tractors. And the list of standard features goes on and on and on.

Needless to say, the specs on this machine — and especially at under $1000 — are practically unbelievable. But there's something about the MX-100 that goes far beyond just the specs; something about the way it all comes together, the attention to detail, the fit, the feel. Mere words fail us. But when you see an MX-100, you'll know what we mean.

All in all, the MX-100 is the most remarkable printer we've ever built. Which creates rather a large problem for those of us at Epson.

How are we going to top this?



Your next printer.

EPSON

EPSON AMERICA, INC.

23844 Hawthorne Boulevard • Torrance, CA 90505 • (213) 378-2220

✓404

See the whole incredible Epson MX Series of printers at your Au

✓See List of Advertisers on page 306

"Several of the major computer manufacturers have announced plans to release versions of Pilot for their products in mid-1981."

guaranteed to work. Try to avoid their use in your programs.

The @ symbol in lines 310, 325 and 326 is actually a shift-@. This must be done to make the @ symbol appear on the video display. Try to use only the unshifted @ symbol for its intended purpose and avoid the shifted-@ symbol. The normal use of @ is shown in lines 620 and 630.

More experienced machine language programmers will be interested in Program Listing 3. Any editor/assembler program could be used to produce a workable object code program from this listing. Of special interest are the use of several ROM routines to both conserve memory and take advantage of the efficient coding available. Details of a few of these commonly used subroutines are outlined below:

CALL 28A7H	Prints a message on the video screen. HL must point to first byte of string. String must end with a zero byte. AF, BC, DE and HL are used.
CALL 032AH	Prints the character contained in register A on the screen at current cursor location. Uses AF.
CALL 01C9H	Clears screen and positions cursor at location 0.
CALL 1BB3H	Prints a question mark and allows user entry as in BASIC's INPUT statement. The entry will be stored starting at 41E8H, A contains a 0 if Enter was the terminating character, 1 if the Break key was used. HL and BC are used.
CALL 0A9AH	Transfers the value stored in HL to memory location 4121H and 4122H.
CALL 0FBDH	Convert value in 4121H and 4122H to ASCII string and store at location 41E8H + length of string.
CALL 1E5AH	Converts string starting at location HL into a numerical value and stores in DE. Uses HL, BC, and DE.

The use of absolute jumps is minimized so the code will be easy to relocate. The initialization section not only automatically sets memory size at a value sufficient to protect the routine, but also begins execution immediately upon loading. The slash and Enter keys are not needed.

The Pilot language is a fun and interesting alternative to BASIC. A wealth of information on Pilot is available in past microcomputing journals which can be used with a minimum of translation for this system. Anyone from a first-grader up should be able to learn and use Pilot after only a short introduction.

Several of the major computer manufacturers have announced plans to release versions of Pilot for their products in mid-1981, which will provide the opportunity to see what Pilot is all about.■

```
1530 STRT2  LD     A,(HL)
1540        CP     00H
1550        INC    HL
1560        JR     NZ,STRT4
1570        LD     A,(HL)
1580        CP     00H          ;TWO ZEROES IN A ROW MEAN END
1590        JR     Z,STRT1
1600 STRT4  INC    HL
1610        LD     E,(HL)       ;SAVE LINE NUMBER IN DE
1620        INC    HL
1630        LD     D,(HL)
1640        PUSH   DE
1650 BACK   INC    HL           ;LOAD A WITH CHARACTER AND JUMP
1660        LD     A,(HL)       ;TO APPROPRIATE SECTION
1670        CP     'Y'
1680        JR     Z,FLAG
1690        CP     'N'
1700        JR     Z,FLAG
1710        CP     'T'
1720        JR     Z,TYPE
1730        CP     'A'
1740        JR     Z,ASK
1750        CP     'M'
1760        JR     Z,MATCH
1770        CP     'J'
1780        JR     Z,JUMP
1790        CP     'E'
1800        JR     Z,END1
1810        CP     'W'
1820        JR     Z,WIDE
1830        CP     'C'
1840        JR     Z,CLS
1850 WRONG  JR     ERROR        ;IF NONE MATCH THEN ERROR
1860 FLAG   CP     C            ;SEE IF CONDITIONAL MATCHES
1870        JR     Z,BACK       ;CURRENT FLAG IN C
1880 SKIP   INC    HL           ;NO MATCH SO SKIP COMMAND
1890        LD     A,(HL)
1900        CP     22H
1910        JR     NZ,ERROR
1920        PUSH   HL
1930        PUSH   BC
1940        INC    HL
1950 SKIP2  JR     START
1960 TYPE   INC    HL           ;TYPE STATEMENT TO VIDEO
1970        LD     A,(HL)
1980        CP     22H          ;CHECK FOR QUOTES
1990        JR     NZ,ERROR
2000        PUSH   HL
2010        PUSH   BC
2020        INC    HL
2030 TYPE2  LD     A,(HL)
2040        CP     00H
2050        JR     Z,TYPE3
2060        CP     40H          ;CHECK FOR @ SIGN
2070        JR     Z,BUFFER
2080        CALL   032AH        ;OUTPUT CHARACTER TO SCREEN
2090        INC    HL
2100        JR     TYPE2
2110 BUFFER PUSH   HL           ;TYPE MOST RECENT
2120        LD     HL,41E8H     ;ANSWER TO SCREEN
2130        CALL   28A7H
2140        POP    HL
2150        INC    HL
2160        JR     TYPE2
2170 TYPE3  LD     A,0DH
2180        CALL   032AH
2190        POP    BC
2200        POP    HL
2210 TYPE4  JR     SKIP
2220 ERROR  LD     HL,ERRMES    ;TYPE ERROR MESSAGE AND STOP
2230        CALL   28A7H
2240        POP    HL
```

Program continues

```
2250          CALL    0A9AH
2260          CALL    0FBDH
2270          CALL    28A7H
2280 END1     HALT
2290 MATCH    LD      C,'Y'           ;COMPARE ANSWER TO LIST AND
2300          INC     HL              ;SET APPROPRIATE FLAG
2310          LD      A,(HL)
2320          CP      22H
2330          JR      NZ,ERROR
2340          PUSH    HL
2350          JR      MATCH1
2360 JUMP     INC     HL              ;JUMP TO LISTED LINE NUMBER
2370          LD      A,(HL)
2380          CP      22H
2390          JR      NZ,ERROR
2400          INC     HL
2410          PUSH    BC
2420          CALL    1E5AH           ;CONVERT STRING TO NUMBER IN DE
2430          LD      HL,(40A4H)
2440 JUMP2    LD      A,L
2450          OR      H
2460          JR      Z,JUMP4         ;LINE NOT FOUND SO PRINT ERROR
2470          LD      C,(HL)
2480          INC     HL
2490          LD      B,(HL)
2500          PUSH    BC              ;SAVE NEXT LOCATION
2510          INC     HL
2520          LD      A,(HL)
2530          CP      E
2540          JR      Z,JUMP3
2550          POP     HL
2560          JR      JUMP2
2570 JUMP3    INC     HL
2580          LD      A,(HL)
2590          CP      D
2600          JR      Z,LINE          ;IF MATCH GOTO LINE
2610          POP     HL
2620          JR      JUMP2
2630 LINE     DEC     HL              ;RESET HL TO PROPER LOCATION
2640          DEC     HL
2650          DEC     HL
2660          POP     BC
2670          POP     BC
2680          JR      SKIP2
2690 JUMP4    POP     BC
2700          JR      ERROR
2710 MATCH1   INC     HL              ;POINT TO TEXT ANSWER
2720          LD      DE,41E8H        ;POINT TO USER'S ANSWER
2730 MATCH2   LD      A,(HL)
2740          CP      2FH             ;CHECK FOR SLASH
2750          JR      Z,MATCH3
2760          CP      00H             ;CHECK FOR END OF LINE
2770          JR      Z,MATCH3
2780          EX      DE,HL
2790          LD      B,(HL)
2800          EX      DE,HL
2810          CP      B
2820          JR      NZ,FAIL
2830          INC     HL
2840          INC     DE
2850          JR      MATCH2
2860 FAIL     INC     HL              ;NO MATCH THIS TIME
2870          LD      A,(HL)
2880          CP      00H             ;CHECK FOR END OF LINE
2890          JR      Z,SETFLG
2900          CP      2FH             ;CHECK FOR SLASH
2910          JR      Z,MATCH1
2920          JR      FAIL
2930 SETFLG   LD      C,'N'           ;SET FLAG TO NO
2940 MATCH3   POP     HL
2950          JR      TYPE4
2960 ERRMES   DEFM    'ERROR IN LINE'
2970          DEFB    00H
2980          END
```


✓See List of Advertisers on page 306

Program Listing 4.

```
                 ; TRS-80 PILOT INTERPRETER
                 ;
                 ; BY RANDY HAWKINS
                 ;    6214 HIDDEN COVE
                 ;    CORPUS CHRISTI, TEX
                 ;
                 ; THIS PROGRAM LOADS THE PILOT INTERPRETER
                 ; INTO HIGH MEMORY OF A 16K CASSETTE BASED
                 ; SYSTEM ... ABSOLUTE JUMPS ARE KEPT TO A
                 ; MINIMUM TO MAKE RELOCATION SIMPLEST.
                 ;
41E8                     ORG     41E8H     ;LOAD BUFFER WITH
41E8 33                  DEFM    '32465'   ;PROPER MEMORY SIZE
41ED 00                  DEFB    00H       ;ZERO MARKS END
4016                     ORG     4016H     ;RESET KEYBOARD DCB
4016 987E                DEFW    7E98H     ;TO INITIALIZATION
7E98                     ORG     7E98H
7E98 21FF7E              LD      HL,BEGIN     ;DEFINE 'NAME'
7E9B 228F41              LD      (418FH),HL   ;COMMAND TO XEQ
7E9E 21E303              LD      HL,03E3H
7EA1 221640              LD      (4016H),HL   ;RESET KEYBOARD
7EA4 21B67E              LD      HL,TITLE     ;DISPLAY TITLE
7EA7 CDA728              CALL    28A7H        ;ROM ROUTINE
7EAA 3E0D                LD      A,0DH
7EAC CD2A03              CALL    032AH
7EAF AF                  XOR     A            ;JUMP INTO ROM
7EB0 21E741              LD      HL,41E7H     ;AFTER MEM SIZE
7EB3 C3C000              JP      00C0H        ;QUESTION
7EB6 1C          TITLE   DEFB    1CH
7EB7 1F                  DEFB    1FH
7EB8 54                  DEFM    'TRS-80 PILOT INTERPRETER &"
7ED3 00                  DEFB    00H
7ED4                     ORG     7ED4H
7ED4 CDC901      CLS     CALL    01C9H    ;CLEAR SCREEN
7ED7 1879                JR      TYPE
7ED9 CDC901      WIDE    CALL    01C9H    ;CLEAR SCREEN
7EDC 3E17                LD      A,17H    ;AND SWITCH TO
7EDE CD2A03              CALL    032AH    ;WIDE LETTERS
7EE1 186F                JR      TYPE
7EE3 23          ASK     INC     HL
7EE4 7E                  LD      A,(HL)   ;CHECK FOR QUOTE
7EE5 FE22                CP      22H
7EE7 205B                JR      NZ,WRONG
7EE9 E5                  PUSH    HL
7EEA C5                  PUSH    BC
7EEB 23                  INC     HL       ;IS THERE A MESSAGE
7EEC 7E                  LD      A,(HL)   ;AFTER THE QUOTE?
7EED FE00                CP      0        ;IF NOT, THEN GO
7EEF 2803                JR      Z,ASK2   ;AHEAD TO ASK2 ...
7EF1 CDA728              CALL    28A7H    ;TYPE MESSAGE
7EF4 CDB31B      ASK2    CALL    1BB3H    ;THEN USE ROM
7EF7 FE01                CP      01H      ;ROUTINE TO ACCEPT
7EF9 2812                JR      Z,STRT1  ;ANSWER
7EFB C1                  POP     BC
7EFC E1                  POP     HL
7EFD 184A                JR      SKIP
7EFF 2AA440      BEGIN   LD      HL,(40A4H) ;FIND TEXT START
7F02 0E59                LD      C,'Y'      ;SET YES/NO FLAG
7F04 D5                  PUSH    DE
7F05 D1          START   POP     DE
7F06 3A4038              LD      A,(3840H)  ;CHECK FOR BREAK
7F09 FE04                CP      04H
7F0B 2001                JR      NZ,STRT2
7F0D 76          STRT1   HALT               ;JUMP TO 'READY'
7F0E 7E          STRT2   LD      A,(HL)
7F0F FE00                CP      00H        ;CHECK END OF
7F11 23                  INC     HL         ;TEXT ... IF NOT
7F12 2005                JR      NZ,STRT4   ;THEN CONTINUE
```

Program continues

CONVERT YOUR SERIAL PRINTER TO PARALLEL

NEW MODEL UPI-3 SERIAL PRINTER INTERFACE MAKES IT POSSIBLE TO CONNECT AN ASCII SERIAL PRINTER TO THE PARALLEL PRINTER PORT ON THE TRS-80.

Software compatibility problems are totally eliminated because the TRS-80 "THINKS" that it has a parallel printer attached. NO MACHINE LANGUAGE DRIVER NEEDS TO BE LOADED INTO HIGH MEMORY BECAUSE THE DRIVER ROUTINE FOR THE UPI-3 IS ALREADY IN THE LEVEL II ROM! SCRIPSIT, PENCIL, RSM 2, ST80D, NEWDOS, FORTRAN, BASIC etc. all work as if a parallel printer was in use.

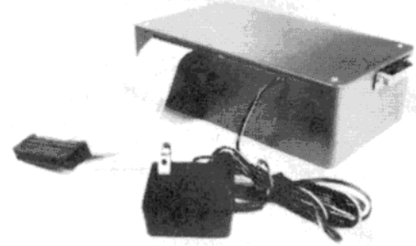
The UPI-3 is completely self contained and ready to use. A 34 conductor edge card connector plugs onto the parallel printer port of the model I Expansion Interface or onto the parallel printer port on the TRS-80 III. A DB25 socket mates with the cable from your serial printer. The UPI-3 converts the parallel output of the TRS-80 printer port into serial data in both the RS232-C and 20 MA. loop formats.

SPEEDWAY ELECTRONICS
1354 AUBURN
SPEEDWAY, IN 46224 ✓275

Also available from
YE ODDE SHOPPE
(317) 925-0496

VISA MasterCard

TRS 80 is a trademark of Tandy

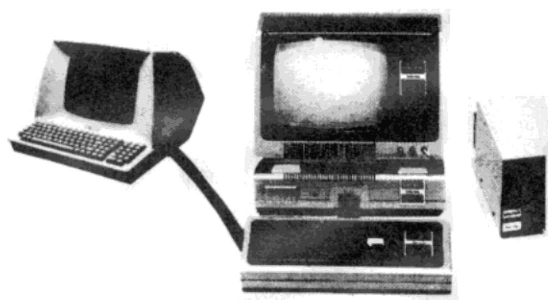


Switch selectable options include:

- Linefeed after Carriage Return
- Handshake polarity (RS232-C)
- Nulls after Carriage Return
- 7 or 8 Data Bits per word
- 1 or 2 Stop Bits per Word
- Parity or No parity
- ODD or EVEN Parity

UPI-3 assembled with 90 day warranty	$139.95
Kit with all parts, sockets, cabinet, etc.	$ 99.95
UPI-4 for use with Model 1 and RS Printer Interface Cable	$149.95
Kit with all parts, sockets, cabinet, etc.	$104.95
Manual only (may be applied to order)	$ 5.00
Ten day return privilege on assembled units	
Shipping and Handling on all orders	$ 4.00
Specify BAUD rate 50-9600 BAUD	

TWO USER MULTI-TASKING



$89 TSHARE VI.3
A SPLIT PERSONALITY FOR YOUR TRS – 80 MODEL I

- Interrupt driven executive patches to parent operating system NEWDOS 2.1 or TRSDOS 2.3.
- Allows two active users to share a single TRS-80.
- Execute BASIC or MACHINE LANGUAGE (above 7700 Hex).
- Joint access of disk files.
- Second user ties to HOST via modem, terminal, or 2nd TRS-80.
- Options for Expansion Interface, RS-232, TRS232, or HUH interface.
- SIMPLEX MODE for non serial-port users. Requires only a printer to act as second "screen". Jobs share the keyboard under user control.
- CONFIGURE UTILITY partitions memory usage and selects I/O.

DBM5 $49
MULTI – FEATURE DATA MANAGER

- Up to 20 user defined fields.
- Files extend across multiple diskettes.
- Supports up to four drives.
- Automatic single drive disk mount requests.
- Four data types and computational fields.
- Flexible REPORT GENERATOR.
- Fast assembly language sorts.
- MULTI-FIELD sorts and searches.
- Keyed access on any sorted field.
- Indexed relations between files.

TRS-80 tm Radio Shack/Tandy Corp.
NEWDOS tm Apparat, Inc.
TRS232 tm Small System Software
HUH tm California Computer Systems.

TO ORDER

ComSoft
203 ARDEN STREET ✓204
GLENDALE, CA 91203
(213) 649-0369

```
7F14 7E                       LD     A,(HL)      ;THIRD ZERO IN A
7F15 FE00                     CP     00H         ;ROW MEANS GO TO
7F17 28F4                     JR     Z,STRT1     ;HALT AT STRT1
7F19 23               STRT4   INC    HL
7F1A 5E                       LD     E,(HL)    ;SAVE PRESENT LINE
7F1B 23                       INC    HL        ;NUMBER IN DE
7F1C 56                       LD     D,(HL)    ;REGISTER PAIR
7F1D D5                       PUSH   DE        ;AND SAVE IN STACK
7F1E 23               BACK    INC    HL        ;LOAD A WITH THE
7F1F 7E                       LD     A,(HL)    ;COMMAND CHARACTER
7F20 FE59                     CP     'Y'       ;AND JUMP TO THE
7F22 2822                     JR     Z,FLAG    ;INDICATED SECTION
7F24 FE4E                     CP     'N'
7F26 281E                     JR     Z,FLAG
7F28 FE54                     CP     'T'
7F2A 2826                     JR     Z,TYPE
7F2C FE41                     CP     'A'
7F2E 28B3                     JR     Z,ASK
7F30 FE4D                     CP     'M'
7F32 285B                     JR     Z,MATCH
7F34 FE4A                     CP     'J'
7F36 2862                     JR     Z,JUMP
7F38 FE45                     CP     'E'
7F3A 2852                     JR     Z,END1
7F3C FE57                     CP     'W'
7F3E 2899                     JR     Z,WIDE
7F40 FE43                     CP     'C'
7F42 2890                     JR     Z,CLS
7F44 1838             WRONG   JR     ERROR     ;SORRY, NO MATCH
7F46 B9               FLAG    CP     C         ;COMPARE TO FLAG
7F47 28D5                     JR     Z,BACK
7F49 23               SKIP    INC    HL        ;SKIP FORWARD TO
7F4A 7E                       LD     A,(HL)    ;START OF NEXT
7F4B FE00                     CP     0         ;LINE SINCE FLAGS
7F4D 20FA                     JR     NZ,SKIP   ;DID NOT MATCH
7F4F 23                       INC    HL
7F50 18B3             SKIP2   JR     START
7F52 23               TYPE    INC    HL
7F53 7E                       LD     A,(HL)    ;CHECK FOR QUOTE
7F54 FE22                     CP     22H           ;& JUMP TO
7F56 2026                     JR     NZ,ERROR      ;ERROR MESS
7F58 E5                       PUSH   HL
7F59 C5                       PUSH   BC
7F5A 23                       INC    HL
7F5B 7E               TYPE2   LD     A,(HL)    ;END OF LINE?
7F5C FE00                     CP     0
7F5E 2815                     JR     Z,TYPE3
7F60 FE40                     CP     40H       ;CHECK FOR @
7F62 2806                     JR     Z,BUFFER
7F64 CD2A03                   CALL   032AH     ;PRINT CHAR. IN
7F67 23                       INC    HL        ;A REGISTER AND
7F68 18F1                     JR     TYPE2     ;CONTINUE ...
7F6A E5               BUFFER  PUSH   HL          ;PRINT THE
7F6B 21E841                   LD     HL,41E8H    ;CONTENTS OF
7F6E CDA728                   CALL   28A7H       ;THE ANSWER
7F71 E1                       POP    HL          ;BUFFER
7F72 23                       INC    HL
7F73 18E6                     JR     TYPE2
7F75 3E0D             TYPE3   LD     A,0DH       ;PRINT A
7F77 CD2A03                   CALL   032AH       ;CARRIAGE
7F7A C1                       POP    BC          ;RETURN
7F7B E1                       POP    HL
7F7C 18CB             TYPE4   JR     SKIP
7F7E 21F27F           ERROR   LD     HL,ERRMES ;IDENTIFY
7F81 CDA728                   CALL   28A7H       ;LINE NO.
7F84 E1                       POP    HL          ;THAT HAS
7F85 CD9A0A                   CALL   0A9AH       ;ERROR
7F88 CDBD0F                   CALL   0FBDH
7F8B CDA728                   CALL   28A7H
7F8E 76               END1    HALT
```

Program continues

6 ways to give your TRS-80* a +PLUS

Exhaustive research and professional standards bring you the best in business software.

EASYTRAK tm

KEEP YOUR SALES CURVING UPWARD WITH EASYTRAK tm, a powerful, all-purpose activity tracking system. User-specified options enable you to monitor sales activity by territory, product line, salesman, or by combinations. Can monitor inquiries, sales calls and order booking. Production activity (by shift, by product, by day, by team, etc.), spending or income activity are simple to set up with EASYTRAK tm. Extensive video displays make available individual, group or total summaries, relative averages, and percentage data all at the touch of a button. Use of an 80-column printer is optional.

MODEL I/III $100
MODEL II $140

New!

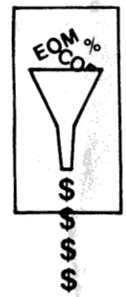
+ACCOUNTS RECEIVABLE

PUT MORE CASH IN YOUR BANK ACCOUNT with +Account Receivable's many collection/credit features, including average payment days, aging schedule and statements. Open-item system records sales activity this year and last, optional automatic discounts and salesman assignment. Cash receipts and A/R adjustments feature extensive on-line validation. And, yes, +Accounts Receivable handles the pesky new 9-digit zip code. Other optional features include service charges to selective customers; partial and "on-account" payments. Posts, at your option, to +General Ledger.

MODEL I/III $250
MODEL II $300

+ACCOUNTS PAYABLE

FUNNEL THAT STACK OF BILLS into +Accounts Payable system and create an orderly, manageable flow of vendor histories, cash requirements forecasts, check registers and checks, aging schedules, etc. Will accept manual checks, reversals, and adjustments. Pay by selected vendor, invoice, due date, discount dates, and optionally post directly to your +General Ledger. Supports 5-digit vendor numbers.



MODEL I/III $250
MODEL II $300

+PAYROLL

KEEP YOUR EMPLOYEES HAPPY and smiling with prompt, accurate and professional payroll records. Will handle weekly, biweekly, semimonthly, monthly payrolls, produce checks, check registers, labor reports, earnings reports, Form 941's, W2's, etc. Can handle fixed/voluntary deductions, hourly/salary employees, bonuses, city and state taxes, etc. Easy to use and update tax tables. Automatic +General Ledger posting.

MODEL I/III $250
MODEL II $300

+GENERAL LEDGER

SLICE OUT MORE PROFIT by using +General Ledger to monitor your financial status. Features include fully-flexible financial statements, both budget and last year comparisons, special report options, monthly and quarterly trial balances, manual and/or automatic journal entries all with extensive audit trails. You structure the chart of accounts with up to 5 digit account numbers.

MODEL I/III $250
MODEL II $300

+FIXED ASSETS

A POWERFUL SET OF TOOLS TO CALCULATE and monitor fixed asset expenditures, book and tax depreciation, investment tax credit, etc. Will even forecast future depreciation for you. (Handy at budget time!) Code structure enables you to classify assets by departments, buildings, groups or other categories. Handles seven depreciation methods. Can post to +General Ledger for you.

MODEL I/III $250
MODEL II $300

SPECIAL! All of the above: MODEL I/III $1250
MODEL II $1500

+PLUS SOFTWARE comes to you feature-packed from on-line validation to machine-language file handling. Extensive documentation and thorough installation guide accompany each order. Source code is included. Model II requires 64K. Model I/III requires 32K, 2 disks.

Dealer inquiries invited

*TRS-80 is a registered trademark of Tandy Corp
EASYTRAK is a trademark of Plus Computer Tech. Inc

VISA

To order, call TOLL-FREE 24 hours
1-800-528-6050 EXT. 1562
(In Arizona, 1-800-352-0458 Ext. 1562)

+PLUS

✓131

PLUS COMPUTER TECHNOLOGY, INC
POST OFFICE BOX 1152
ANGLETON, TEXAS 77515 (713) 849-1108

Please send me the following **PLUS Software:**
☐ EASYTRAK TM ☐ + G/L ☐ + A/R ☐ + A/P
☐ + P/R ☐ + F/A ☐ FREE Catalog

AMOUNT (Tx. residents add 5% sales tax): $

Plus Computer • PO Box 1152 • Angleton, TX 77515

NAME ______
ADDRESS ______
CITY ______
STATE ______ ZIP ______
☐ Model I ☐ Model II ☐ Model III
☐ M/C ☐ VISA Card # ______
Bank # ______ Expiration Date ______
Signature ______ OC-5

✓See List of Advertisers on page 306

```
7F8F 0E59     01370 MATCH   LD     C,'Y'    ;SET FLAG TO YES
7F91 23       01380         INC    HL
7F92 7E       01390         LD     A,(HL)   ;LOOK FOR QUOTES
7F93 FE22     01400         CP     22H
7F95 20E7     01410         JR     NZ,ERROR
7F97 E5       01420         PUSH   HL
7F98 1830     01430         JR     MATCH1   ;CONT AT MATCH1
7F9A 23       01440 JUMP    INC    HL
7F9B 7E       01450         LD     A,(HL)   ;LOOK FOR QUOTES
7F9C FE22     01460         CP     22H
7F9E 20DE     01470         JR     NZ,ERROR
7FA0 23       01480         INC    HL
7FA1 C5       01490         PUSH   BC       ;PUT TARGET LINE
7FA2 CD5A1E   01500         CALL   1E5AH    ;NUMBER IN DE
7FA5 2AA440   01510         LD     HL,(40A4H)
7FA8 7D       01520 JUMP2   LD     A,L      ;HAVE WE REACHED
7FA9 B4       01530         OR     H        ;END OF TEXT?
7FAA 281B     01540         JR     Z,JUMP4
7FAC 4E       01550         LD     C,(HL)   ;NEXT LINE ADDRESS
7FAD 23       01560         INC    HL       ;IS SAVED IN THE
7FAE 46       01570         LD     B,(HL)   ;BC REGISTER PAIR
7FAF C5       01580         PUSH   BC
7FB0 23       01590         INC    HL
7FB1 7E       01600         LD     A,(HL)   ;IS THIS LINE
7FB2 BB       01610         CP     E        ;THE SAME AS
7FB3 2803     01620         JR     Z,JUMP3  ;TARGET NUMBER
7FB5 E1       01630         POP    HL
7FB6 18F0     01640         JR     JUMP2
7FB8 23       01650 JUMP3   INC    HL       ;FIRST NUMBER DID
7FB9 7E       01660         LD     A,(HL)   ;MATCH -- DOES THE
7FBA BA       01670         CP     D        ;SECOND?
7FBB 2803     01680         JR     Z,LINE
7FBD E1       01690         POP    HL
7FBE 18E8     01700         JR     JUMP2
7FC0 2B       01710 LINE    DEC    HL       ;TRANSFER CONTROL
7FC1 2B       01720         DEC    HL       ;TO THIS LINE AFTER
7FC2 2B       01730         DEC    HL       ;RESTORING POINTER
7FC3 C1       01740         POP    BC
7FC4 C1       01750         POP    BC
7FC5 1889     01760         JR     SKIP2
7FC7 C1       01770 JUMP4   POP    BC
7FC8 18B4     01780         JR     ERROR
7FCA 23       01790 MATCH1  INC    HL       ;COMPARE THE ANSWER
7FCB 11E841   01800         LD     DE,41E8H   ;IN THE STANDARD
7FCE 7E       01810 MATCH2  LD     A,(HL)     ;LEVEL II BUFFER
7FCF FE2F     01820         CP     2FH        ;WITH THE LIST
7FD1 281C     01830         JR     Z,MATCH3   ;IN THE PROGRAM
7FD3 FE00     01840         CP     0
7FD5 2818     01850         JR     Z,MATCH3
7FD7 EB       01860         EX     DE,HL
7FD8 46       01870         LD     B,(HL)
7FD9 EB       01880         EX     DE,HL
7FDA B8       01890         CP     B
7FDB 2004     01900         JR     NZ,FAIL
7FDD 23       01910         INC    HL
7FDE 13       01920         INC    DE
7FDF 18ED     01930         JR     MATCH2
7FE1 23       01940 FAIL    INC    HL       ;THIS DID NOT MATCH
7FE2 7E       01950         LD     A,(HL)
7FE3 FE00     01960         CP     0        ;END OF LINE?
7FE5 2806     01970         JR     Z,SETFLG
7FE7 FE2F     01980         CP     2FH      ;OR JUST A SLASH?
7FE9 28DF     01990         JR     Z,MATCH1
7FEB 18F4     02000         JR     FAIL
7FED 0E4E     02010 SETFLG  LD     C,'N'    ;SET FLAG TO NO
7FEF E1       02020 MATCH3  POP    HL
7FF0 188A     02030         JR     TYPE4
7FF2 45       02040 ERRMES  DEFM   'ERROR IN LINE'
7FFF 00       02050         DEFB   00H
0000          02060         END
00000 TOTAL ERRORS
```

✓See List of Advertisers on page 306

SAVE!

EPSON MX-80

$489.00

THE REAL THING

Yes this is the EPSON MX-80 printer you have read so much about... the one everyone's buying. Better get yours while they are available at this terrific price! One of the most advanced 80 column dot matrix printers available featuring the world's first disposable print head. There are lots of printers to choose from but, once you have compared, you will find it hard to pass up the EPSON MX-80.

FEATURES

- Bidirectional Printing
- Logical seeking of shortest lines
- 80 Characters per second
- 64 Block Graphic Characters
- Forms Handling
- Disposable print head
- Adjustable Tractor-type pin feed
- Paper width 4 to 10 inches
- Inked ribbon cartridge with 3 million character life
- Self test mode
- Full 96 character ASCII with decenders
- Printing Modes: Standard, Double, Emphasized and Double Emphasized
- Correspondence Quality

Add $6.00 shipping (Cont. US). Does not include cable. See opposite page for ordering information.

EXTRAS FOR EPSON

TRS80 Cable (I & III) $29.95
APPLE Cable & Interface $96.00
RS232 Interface $69.95
Ribbon Cartridge, Black $13.95
Interface & Cable (from Mod. I Keyboard) $99.95

ORDER TODAY!!

Save Even More EPSON MX-70

$389.00

NO FRILLS PRINTING AT NO FRILLS PRICE

FEATURES

★ Unidirectional printing
★ 80 Characters per second
★ Bit-plot graphics
★ Forms handling
★ Disposable print head
★ Adjustable tractor-type pin feed
★ Paper width 4 to 10 inches
★ Inked ribbon cartridge with 3 million character life
★ Self test mode
★ Full 96 character ASCII
★ Printing modes: normal & normal expanded

Add $6.00 shipping (Cont. US). Does not include cable. See opposite page for ordering information.

ALL NEW! MX-80 F/T

All the features of the MX-80 plus friction feed.

$599.00

LIMITED QUANITIES

CABLES, SOFTWARE & OTHER GOODIES

★ DRIVE CABLES

2-Drive $24.95
3-Drive $29.95
4-Drive $34.95
Extender Cables $16.95

★ OPERATING SYSTEMS

DOSPLUS 3.3, 3.3D (spec. Mod I or III) $89.95
LDOS $135.00
NEWDOS/80 .... $135.00
TRSDOS 2.3 DISK & MANUAL $17.95
DOUBLE ZAP II $44.00

Add $1.25 per pound shipping ($1.00 minimum).

★ DISKETTES (5-1/4 in., softsectored, box of 10)

Cert. one side, single density $29.95
Cert. two sides, double density $39.95

★ OTHER SOFTWARE & BOOKS

SCRIPSIT (disk) $89.95
SPECIAL DELIVERY $119.95
XTRA SPECIAL DELIVERY $189.95
SUPER UTILITY $39.95
MICROSOFT BASIC DECODED & OTHER MYSTERIES
.... $25.95
TRS80 DISK & OTHER MYSTERIES .. $19.95
MAKE 80 .... $14.95

SPECIAL WORD PROCESSING PACKAGE

Includes the following:
★ EPSON MX-80 PRINTER
★ CABLE
★ SCRIPTSIT)disk)
★ SPECIAL DELIVERY (disk)
★ freight and insurance.

SPECIAL

$725.00

(Reg $917.00)

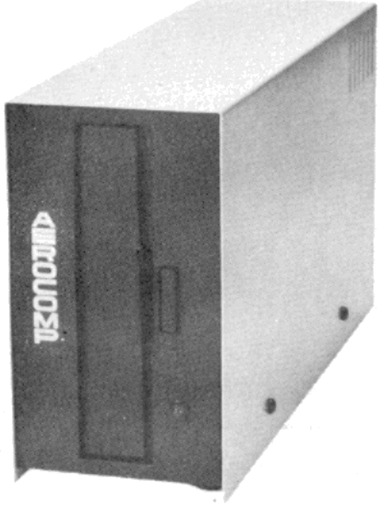
Order today from

AEROCOMP

Redbird Airport, Bldg. 8
P.O. Box 24829
Dallas, Texas 75224 ✓387

See Opposite Page

AROCOMP5/23



TRS 80*

DISK DRIVES

40 & 80 TRACK

as low as $299.95

NEW LOW PRICES

Thanks to you our sales volume has allowed us to cut costs and we're passing the savings along. We offer the best disk drive value in the market place. Reliability, features and cost tough to beat. We deliver...and we stand behind our products. AEROCOMP is the only manufacturer to offer FREE TRIAL! Buy Aerocomp drives today. You deserve the best.

BEST FEATURES

★ Fast 5 ms. track-track access time.
★ Operates single or double density.
★ "FLIPPY" feature for lower media cost (40-1 & 80-1).
★ Head load solenoid.
★ Disk ejector.
★ Easy entry door.
★ <u>NEW</u> EXTERNAL DRIVE CABLE CONNECTION. (no longer need to remove the cover to hook up cable)

MODEL III INTERNAL DRIVE KIT

Includes the following:
(1) Disk Controller Board, Double Density; (2) Dual Power Supply (handles two drives), (3) Drive Mounting Brackets & Shield, (4) all necessary hardware, cables and connectors and (5) installation instructions.
May be used with AROCOMP bare drives. DOSPLUS operating system available $89.95.

SPECIAL PRICE

$399.95

Add $3.00 for shipping & handling.

ORDER TODAY!

★ 40-Track Drive............ $299.95
★ 80-Track Drive............ $429.95
★ 40-Track "FLIPPY" Drive.... $329.95
★ 80-Track "FLIPPY" Drive.... $449.95
★ 40-Track Dual-Head Drive.. $449.95
★ 80-Track Dual-Head Drive.. $579.95

All above drives are complete with silver enclosure, power supply and external drive cable connection. 115 VAC. 50/60 Hz. 230 VAC. 50/60 Hz available on special order

★ 40-Track Bare Drive........ $269.95
★ 80-Track Bare Drive........ $399.95
★ 40-Track "FLIPPY" Bare Drive $299.95
★ 80-Track "FLIPPY" Bare Drive $419.95

★ SPECIAL PACKAGES ★

STARTER A................. $339.95
40-Track Drive, 2-drive cable, TRSDOS 2.3 Disk & Manual, Freight & Ins.

SRARTER B................. $369.95
40-Track "FLIPPY" Drive,cable, TRSDOS 2.3 Disk & Manual, Freight & Ins.

COMBO C................. $465.00
40-Track "FLIPPY" Drive, 2-drive cable, NEWDOS/80 or LDOS, Freight & Ins.

COMBO D................. $585.00
80-Track "FLIPPY" Drive, 2-drive cable, NEWDOS/80 or LDOS, Freight & Ins.

COMBO E................. $709.00
Two 40-Track "FLIPPY" Drives, 4-Drive cable, TRSDOS 2.3 Disk & Mannual, Freight & Ins.

COMBO F................. $999.00
Two 80-Track "FLIPPY" Drives, 4-drive cable, NEWDOS/80 or LDOS, Freight & Ins.

COMBO G................. $909.00
Two 40-Track Dual-Head Drives, cable, TRSDOS 2.3 Disk & Manual, Freight & Ins.

COMBO H................ $1299.00
Two 80-Track Dual-Head Drives, cable, NEWDOS/80 orLDOS, Freight & INS. DOSPLUS available with any special package.

Add$5.00 per drive for shipping & handling (Cont. US)

FREE TRIAL OFFER

Order your AEROCOMP Disk Drive and use it with your system for up to 14 days. If you are not satisfied for ANY REASON (except misuse or improper handling), return it, packed in the original shipping container, for a full refund. (Special packages will be perforated). We have complete confidence in our products and we know you will be satisfied! **ORDER TODAY!!**

* TRS80 is TRADEMARK of TANDY CORP

WARRANTY

We offer you a 120 day unconditional warrenty on parts and labor against any defect in materials and workmanship. In the event service, for any reason, becomes nescessary, our service department is fast, friendly and cooperative.

100% TESTED

AEROCOMP Disk Drives are completely assembled at the factory and ready to plug in when you receive them. Each drive is 100% bench tested prior to shipment. We even enclose a copy of the test checklist, signed by the test technician, with every drive. AEROCOMP MEANS RELIABILITY!!

ORDER NOW!!

To order by mail, specify Model Number(S) of Drive, cable, etc. (above), enclose check, money order, VISA or MASTERCHARGE card number and expiration date, or request C.O.D. shipment. Texas residents add 5% sales tax. Add $5.00 per drive for shipping & handling (Cont. US). Please allow 2 weeks for personal checks to clear our bank. No personal checks will be accepted on C.O.D. shipments-cash, money orders or certified checks only. You will receive a card showing the exact C.O.D. amount before your shipment arrives. Be sure to include your name and shipping address. You will be notified of the scheduled shipping date. Your bank credit card will NOT be charged until the day we ship.!!

WRITE AEROCOMP TODAY FOR MORE VALUES !!!

CALL TOLL FREE FOR FAST SERVICE (800) 824-7888, OPERATOR 24
FOR VISA/MASTERCHARGE/C.O.D. ORDERS
California dial (800) 852-7777, Operator 24. Alaska and Hawaii dial (800) 824-7919, Operator 24.
TOLL FREE LINES WILL ACCEPT ORDERS ONLY!
For Applications and Technical information, call (214) 337-4346 or drop us a card.

Dealers inqiries invited

AEROCOMP
Redbird Airport, Bldg. 8
P.O. Box 24829
Dallas, TX 75224 ✓387

AROCOMP/23

Teach yourself a little Pascal.

Join the Pascal Crusade

Margaret M. Grothman
5117 Denton Place
Madison, WI 53711

Radio Shack's Tiny Pascal comes with a user's manual which is a model of brevity. The manual wasn't intended to be an instructional text on Pascal programming: If you are a beginner, you will have to learn by the discovery method.

The user's manual defines Tiny Pascal as "a complete, self-contained operating system for creating, running, saving and loading Pascal programs for the TRS-80." The minimum system requirement is Level II, 16K, although a 32K version is provided on the other side of the tape. The system consists of three subsystems loaded together and simultaneously present in RAM. These are the monitor, the compiler and the editor. The Run, Save and Load commands are given from the monitor mode. The monitor also provides access to the compiler and the editor.

Although Tiny Pascal employs the syntax of standard Pascal, it is a limited subset of the standard language. Several variable types and library functions are not present in the tiny version. Even so, Tiny Pascal is fast, efficient and easier to read than BASIC.

Getting Started

Tiny Pascal is loaded via the system command, using Pascal as the file name. After a successful load, you will see the name and version number followed by the prompt ".". You are now in the monitor mode. A sample program is always loaded with the system.

There are 10 monitor commands listed in the manual, one of which is R for Run. Since you are in the monitor mode now, and there is a program present, enter R. The message returned is: "P-Code not in memory." What's wrong?

Only the source code is present in memory. Pascal is a high-level language that is not executed statement by statement, like BASIC. The code interpreted by the Pascal system is a low-level language called P-Code. To compile, enter C. As compilation takes place, the source code appears line by line. Finally, a summary line appears, containing the number of codes and the memory locations of the compiled code.

Now enter R to run the program.

There are two remaining editor commands, R and X. X has the same function as in the Level II editor, extending the line to add additional characters, or deleting characters at the end of the line by backspacing. R is used for replacing the current line. It is not followed by Enter, but by the string which is to replace the current line.

There is no editor command to change characters within a line. If an error has been made toward the end of a line, the X command can be used to make the necessary correction. Otherwise, there is no alternative to retyping the line (use the R command). For this reason, it is a good idea to keep lines short and avoid multiple statement lines.

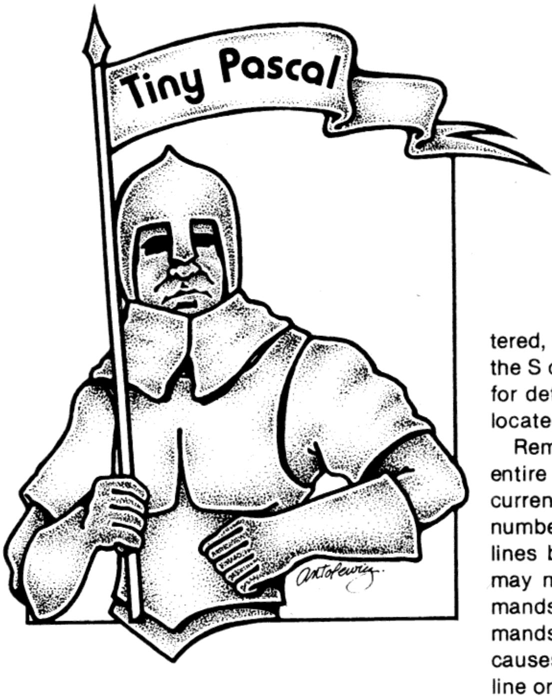
Entering a New Program

To become familiar with the syntax of Pascal, let's delete the existing program and enter a new one, observing various features as they occur.

```
(*SUM AND DIFFERENCE PROGRAM*)
CONST FIRST = 12;SECOND = 3;
VAR SUM, DIFF:INTEGER;
BEGIN
  SUM: = FIRST + SECOND;
  DIFF: = FIRST - SECOND;
  WRITE('THE SUM OF ',FIRST#,' AND ',SECOND#,' IS ',
  SUM#,13);
  WRITE('THE DIFFERENCE BETWEEN ',FIRST#,' AND ',
  SECOND#,' IS ',DIFF#)
END.
```

Although the Pascal editor is line oriented, observe that no line numbers are used. For easier reference to the program, however, I will refer to the lines by number.

The first line is a comment and is not executed. Parentheses and asterisks are needed to enclose remarks.



"Larger versions of Pascal support several data types, but Tiny Pascal supports only integers and integer arrays."

The words First and Second are constants which will be used in the program. In Pascal, all identifiers must be named before being used, in order to reserve memory locations for them. The second line, which begins with the keyword CONST, is the declaration statement for the constants. If we had written the numbers 12 and 3 directly into the program, they would be called "literals" rather than constants.

The advantage of using the CONST statement is more apparent in a program which uses constants repeatedly. To change the value of a constant, simply change the declaration statement rather than each occurrence of the constant.

The third line is the declaration line for variables to be used in the program. The word Integer following the list of names indicates the data type of the variables. Larger versions of Pascal support several data types, but Tiny Pascal supports only integers and integer arrays. (All punctuation in the statement is necessary.)

Letters to the Editor

To gain access to the editor, enter E from the monitor mode. You will see a status report: "FILE HAS 12 LINES 344 BYTES (498E-4AE5) PTR AT LINE 1." This is followed by > which is the prompt for an editor command. Enter P* and you will see all twelve lines of the sample program on the screen. Now enter Q to exit from the editor mode and return to the monitor.

If you try to run the program again, you will find that the P-Code is no longer in memory, and the program must be recompiled. This time after you enter C to compile, notice the location of the compiled code. The source code was located between 498E and 4BB2; the object code occupies the memory immediately above the source code.

Enter E again. The status report, which is displayed as soon as the editor mode is entered, may be called back anytime by using the S command. It will be especially useful for determining where the pointer (PTR) is located.

Remember the command P* that lists the entire file? A P without the * lists only the current line; P followed by a one or two-digit number results in a list of that number of lines beginning with the current line. You may move the line pointer with the commands U for up or N for next. These commands may be followed by an * which causes the line pointer to move to the first line or the last line, respectively. Following U or N by a one or two-digit number results in the line pointer moving up or down the number of lines indicated.

Using U to indicate up may result in the unfortunate error of thinking that its opposite must be D for down. D actually deletes a line, and may be used with the variations Dnn, to delete a specified number of lines, and D* to delete the entire file.

To insert new lines, enter I. The new line or lines will be inserted immediately after the line currently pointed to. The prompt ? is used to indicate the insert mode. To exit from insert mode, press Enter at the beginning of a new line. The command to delete the entire file, D*, automatically puts you into insert mode.

Two difficulties may be encountered in insert mode: I could find no direct way to insert a new line above the first line in the file. The problem can be solved by inserting the new line after the first line, retyping the original first line to follow the new one, and deleting the extra line.

The other problem occurs if you delete an entire file and then attempt to load a new program from tape. The delete command, D* automatically invokes the insert mode. The only way to return to the monitor is to Enter a program line. This annoyance can be avoided altogether; deletion of the existing file takes place automatically when a new program is loaded.

Identifier names may consist of any alphanumeric characters, but must begin with a letter. Keywords or reserved words, such as those used for commands, functions and operators, cannot be used as identifiers. Although only four characters are recognized, names may be longer, as in BASIC. For example, any words in this program which begin with the letters FIRS or SECO will be indistinguishable from the constants FIRST and SECOND.

The actual program steps are in lines five through eight, between Begin and End. (The period is required following End.) In Pascal, line indentation has no significance for the compiler. But indentation can help you visualize program structure. In Tiny Pascal, the right arrow causes a three-space tab for convenient indentation.

Lines five, six, seven and eight each contain a single Pascal statement. Statements are separated from each other by semicolons. Placing a semicolon after the last statement before End is optional. Although they may cause editing difficulties, multiple statement lines are allowed. No line can exceed 130 characters.

Lines five and six are assignment statements; notice the use of the symbol :=. Unlike BASIC, there are two separate symbols for equality and assignment in Pascal. They are = and :=, and cannot be used interchangeably. The assignment statement copies data rather than moving it.

The first Write statement (line seven) consists of seven elements separated by commas. Strings are enclosed by single quotation marks. First#, Second# and Sum# in the Write statement are instructions to print the values as decimal numbers. To output hexadecimal values, affix the character % to the variable name. The last instruction, 13, is the control code for carriage return/line feed.

Read Input

The program can be made more versatile with the use of READ statements for entering different values for First and Second. Delete the CONST declaration statement and rewrite the VAR declaration to include First and Second as variables. Move the pointer until it is at the beginning of the third line (BEGIN); enter I to insert new lines after line three. Enter the following four lines, then press Enter again to exit from the insert mode:

```
WRITE('THE FIRST NUMBER IS ');
READ(FIRST#);
WRITE(13,'THE SECOND NUMBER IS ');
READ(SECOND#);
```

Return to the monitor (Enter Q) to compile the new variation. Instead of typing C to compile, try C/-P. This monitor command compiles the source code without generating P-Code. This is a compiler dry run used to check for syntax errors before actually compiling the program. A third compiling command, C/-S, is used for long programs that require overwriting the source code to complete compilation. The C/-P

"A good way to learn about the idiosyncracies of the Pascal syntax is to make some intentional mistakes."

command should be used to debug these programs, since the C/-S command destroys the source code.

Compile and run the revised program. You may wish to experiment with different types of input. (Such as negative numbers or zeroes.)

As an exercise, add two lines to the program to multiply First and Second together and write "The product of 12 and 3 is 36." It will be necessary to add a new variable to the VAR declaration statement and to add an additional Write statement. As in BASIC, the multiplication symbol is *.

Integer Division in Pascal

Since Tiny Pascal supports only integers, division presents a special problem. In Pascal division of integers is indicated by the keyword DIV and the quotient is a truncated, not rounded, integer. Thus, 15 DIV 4 returns a quotient of 3. The slash symbol, used in Pascal for division of real numbers, cannot be used in Tiny Pascal. However, the MOD function will return the remainder of integer division. Here is a short program to illustrate how the DIV and the MOD functions work:

```
(*PASCAL INTEGER DIVISION*)
VAR FIRST, SECOND, QUO, REM:INTEGER;
BEGIN
 WRITE('THE FIRST NUMBER IS ');
 READ(FIRST#);
 WRITE(13,'THE SECOND NUMBER IS ');
 READ(SECOND#);
 QUO: = FIRST DIV SECOND;
 REM: = FIRST MOD SECOND;
 WRITE('THE QUOTIENT IS ',QUO#,13);
 WRITE('THE REMAINDER IS ',REM#)
END.
```

For output in fractional form, replace the last two Write statements with:

```
WRITE(FIRST#,' DIVIDED BY ',SECOND#,' EQUALS ',
QUO#,' AND ',REM#,'/',FIRST#)
```

Error Codes

If you have entered the suggested programs or experimented in other ways, you have most likely been introduced to the error codes. The Tiny Pascal manual explains the code messages. There are many error messages; and they are very specific. In practice, the messages don't always describe your mistake, and occasionally quite obvious syntax errors result in no error code at all. But for the most part, they will tell you where you went wrong.

A good way to learn about the idiosyncracies of the Pascal syntax is to make some intentional mistakes. If the variable declaration statement is retyped with an extra space after each comma, no error results. However, omitting spaces is less successful than adding extra ones. For example, if the space between the words VAR and FIRST is omitted, you'll get error number 18. If the eighth and ninth lines are retyped as follows, no error message is displayed.

```
QUO: = FIRSTDIVSECOND;
REM: = FIRSTMODSECOND;
```

However, the omitted spaces result in an execution error. FIRSTDIVSECOND and FIRSTMODSECOND are read by the compiler as single variables indistinguishable from the variable First.

Although extra spaces around punctuation are not necessary, they may be added as desired for program readability. Use caution when deleting spaces to make a program compact. Leave spaces around keywords, and use spaces any time ambiguity could result without them.

Omitting Begin results in error 18 (error in declaration part). In addition to being a delimiter of program statements, Begin also signals the End of the declaration statement. Omitting End results in error 14 (; expected). The compiler is looking for a delimiter to mark the end of the last program statement and cannot find it. Many errors which have no specific error message call error 14.

In Tiny Pascal, omitted words will not be implied. Although identifier names may be abbreviated to the first four letters, keywords must be typed in full.

Punctuation Errors

Be aware of punctuation, too. Omitting the period after a program's final End statement results in error 1000 (. missing). End statements are also used to mark the end of a loop or branch. An internal End statement is not followed by a period, but by a semicolon or no punctuation at all, depending on the context (see Table 1). The last program statement before an internal End statement does not require a semicolon, just as the last program statement before the final End does not.

Omitting # or % after a variable in a Read or Write statement does not cause an error message—but does cause execution errors. This kind of error is dangerous; you have no indication anything is wrong, unless the output is clearly unreasonable.

The moral of the punctuation story is

✓See List of Advertisers on page 306

"Since Pascal is executed so much faster than Level II BASIC, a larger number of loops must be used to achieve the desired delay."

this: Do not depend on error messages. If you are unsure of the correct punctuation, refer to the syntax diagrams in the Tiny Pascal manual.

Truth or Consequences

A Boolean statement (named after George Boole, the 19th century English mathematician) is a statement to be tested for truth. In Pascal, a true statement returns a one and a false statement returns a zero. Enter, compile and run the following program:

```
(*BOOLEAN EXPRESSIONS AND RELATIONAL
OPERATORS*)
VAR TRUE,FALSE:INTEGER;
BEGIN
  TRUE:=4=4;
  FALSE:=4=5;
  WRITE('A TRUE EXPRESSION HAS THE VALUE OF ',
  TRUE#,13);
  WRITE('A FALSE EXPRESSION HAS THE VALUE OF ',
  FALSE#)
END.
```

In the fourth and fifth lines, 4 = 4 and 4 = 5 are Boolean statements. These lines illustrate the difference between the assignment symbol, :,:=, and the equality symbol, =. The third statement, for example, could be interpreted as: If 4 equals 4, then assign a value of 1 to the variable: True.

There are five other relational operators which may be used within a Boolean statement. All six operators are illustrated in the following sample statements:

```
  X:=Y+6>Z+2
  XX:=Y+7*Y<TOTAL
  FALSE:=RIGHT=WRONG+WRONG
  TEST:=BALONEY<>STEAK
  WEEK:=FRIDAY>=MONDAY
  NEW:=MIN<=MAX
IF THEN ELSE
```

This construction is similar to its Level II BASIC counterpart. Try the following program:

```
(*COMBINATION LOCK*)
VAR COMBNUM:INTEGER;
BEGIN
  WRITE('ENTER THE FOUR DIGIT COMBINATION THAT
  WILL OPEN THE DOOR');
  READ(COMBNUM#);
  IF COMBNUM=2468 THEN WRITE('COME IN, THE
  DOOR IS OPEN!')
  ELSE WRITE('DUMMY, YOU ENTERED THE WRONG
  COMBINATION!')
END.
```

The argument of the IF statement is a Boolean statement which is evaluated by Pascal as a one or a zero. The lines containing If, Then and Else together are a single Pascal statement. A semicolon after the IF. . .THEN line results in the ubiquitous error 14. If the instructions following the IF. . .THEN statement require multiple statements, the following form is used:

```
IF COMBNUM=2468 THEN
  BEGIN
    WRITE ('COME IN, THE DOOR IS OPEN!');
    . . .(other statements to be executed follow). . .;
  END
ELSE
  BEGIN
    WRITE ('YOU DUMMY, YOU ENTERED THE WRONG
    COMBINATION!'');
    ...(other statements to be executed follow). . .;
  END
END
```

Begin and End (with no period following End) must be used to bracket the compound statements. Are you starting to see the importance of indenting program lines? The lines between each set of Begin and End statements, and those between each set of If and Else statements, should be indented. In a construction such as this one, proper indenting reveals the program logic and structure in a way that is seldom achieved in a BASIC program.

Else may be omitted in the single line form but not in the compound form. If Else is omitted, execution will fall through to the next line after the Then instruction. The word Then may not be omitted in either form.

Logical Operators

In addition to the relational operators, Pascal allows the use of the logical operators AND, OR and NOT. These may be used in IF. . .THEN statements and other constructions requiring Boolean statements. They are exact counterparts of the logical operators used in Level II BASIC. The following contain true statements and would all result in execution of the Write statements:

```
BEGIN
  IF(2=2)AND(3=3)THEN WRITE('BOTH STATEMENTS
  ARE TRUE.')
END.

BEGIN
  IF(2=2)OR(3=3)THEN WRITE('ONE OR BOTH OF
  THESE STATEMENTS ARE TRUE.')
END.

BEGIN
  IF(2=2)OR(3=4)THEN WRITE('ONE OR BOTH OF
  THESE STATEMENTS ARE TRUE.')
END.

BEGIN
  IF NOT(2=3)THEN WRITE('THE STATEMENT WITHIN
  PARENTHESES IS NOT TRUE.')
END.
```

Tiny Pascal does not require parentheses in the IF. . .THEN statements, but they are often necessary when logical operators are used because of the hierarchy of operations. Operations of the same level are performed from left to right, then operations of the next lower level are performed from left to right, etc. In Pascal, NOT is the highest level operator; *, AND, DIV and MOD share the next level; OR, + and − are the next lower order; and the relational operators are last. In the statements above, parentheses are needed to force evaluation of the equality of the expressions before the logical operations take place.

Loops: FOR-DO

The Pascal FOR-DO loop resembles the BASIC FOR-NEXT loop. The loop may be written as a single statement or combined statements. A one-line FOR-DO loop with no instruction following DO may be used to delay program execution:

```
FOR I:=1 TO 30000 DO;
```

The index variable, I, must be declared in the VAR statement before it may be used. Since Pascal is executed so much faster than Level II BASIC, a larger number of loops must be used to achieve the desired delay. The above loop took 27 seconds to execute.

The FOR-DO construction also allows compound statements as illustrated in the following program:

```
(*FACTORIAL PROGRAM*)
VAR FACTOR,NUMBER,COUNT:INTEGER;
BEGIN
  FACTOR:=1;
  WRITE('ENTER A NUMBER ');
  READ(NUMBER#);
  FOR COUNT:=1 TO NUMBER DO
    BEGIN
      WRITE('THINKING. . .',13);
      FACTOR:=COUNT*FACTOR
    END;
  WRITE(NUMBER#,' FACTORIAL EQUALS ',FACTOR#)
END.
```

The TRS-80 integer range of −32767 to 32767 seriously limits the usefulness of this program, as you will find if you try to enter

"The FOR-DO loop is used when repetition is not dependent upon a certain condition. If it is, Pascal offer two kinds of loops . . ."

any number larger than seven.

Note the semicolon after the End statement. The last statement before the loop End does not require a semicolon, nor does the last statement before the final End.

The index variable in a FOR-DO loop may be decremented using DOWNTO. (Increments or decrements of more than one, however, are not allowed.) Replace the seventh line with:

```
FOR COUNT: = NUMBER DOWNTO 2 DO
```

Conditional Looping

The FOR-DO loop is used when repetition is not dependent upon a certain condition. If it is, Pascal offers two kinds of loops, REPEAT UNTIL and WHILE DO. The difference between them is subtle, and in many cases either could be used. In the REPEAT UNTIL construction, the condition is tested after the statements in the loop are executed. In WHILE DO construction, the condition is tested first: If the condition is not present, execution of the instructions within the loop is avoided. Two program examples follow, one using REPEAT UNTIL, and the other using WHILE DO. The first converts a decimal number to its binary equivalent.

```
(*DECIMAL TO BINARY*)
VAR NUMBER, BINARY, QUOTIENT:INTEGER;
BEGIN
 WRITE('ENTER A NUMBER ');
 READ(NUMBER#);
 WRITE(13,'THE BINARY REPRESENTATION OF ',
 NUMBER#,' IS ');
 REPEAT
  QUOTIENT: = NUMBER DIV 2;
  BINARY: = NUMBER MOD 2;
  NUMBER: = QUOTIENT;
  WRITE(13,BINARY#)
 UNTIL QUOTIENT = 0;
 WRITE(13,'READ BINARY NUMBER FROM BOTTOM
 TO TOP')
END.
```

Repeat and Until form the boundaries of this loop, rather than Begin and End. Until is followed by a Boolean statement and execution of the loop continues until the statement is tested and found to be true.

In the WHILE-DO construction, the Boolean statement to be tested is placed between WHILE and DO. Begin and End (with a semicolon) are used as boundary statements. A program segment, used to compute a statistical formula containing the variable Y in the denominator, follows. Any time Y has the value of zero, execution of the formula statements is avoided and a substitute instruction is executed. The WHILE-DO loop avoids a division-by-zero error.

```
WHILE Y<>0 DO
 BEGIN
  ABSFX = (MI - MX)*P DIV SX*Y;
  WRITE(ABSFX#);
 END;
ELSE
 BEGIN
  ABSFX: = 0;
  WRITE('ABSFX = 0')
 END;
. . . continue with rest of program . . .
```

The Case for Multiple Branching

Pascal's Case statement is equivalent to BASIC's ON . . . GOTO. It has a variable (integer or expression) called the selector, which points to the statement to be executed. The example below shows a menu selection application of Case:

```
(*MENU SELECTOR*)
VAR CHOICE:INTEGER;
BEGIN
 WRITE('ENTER THE NUMBER OR THE ACTIVITY YOU
 HAVE SELECTED.');
 READ(CHOICE#);
 CASE CHOICE OF
  1:BEGIN
   . . . statements to be executed for choice number
   1 . . . ;
   . . . end each one but the last with a semicolon . . .
  END;
  2:BEGIN
   . . . statements to be executed for choice number
   2 . . . '
   . . . are you getting the idea now? . . .
  END;
  3:WRITE('YOU HAVE CHOSEN TO END THE
  PROGRAM.')
 END;
END.
```

The Case statement is matched by an End statement punctuated by a semicolon. Each branch has a constant, called a case label, which corresponds to a possible integer value of the Case selector variable. The branch statements may be single or multiple: If multiple, they are bounded by Begin and End (semicolon required).

It is possible to use more than one number for a single branch:

```
CASE CHOICE OF
 1,2,5:
 3:
 4,6:
END;
```

If a branch does not exist for a given value of the selector, execution passes to the statement after Case End. Tiny Pascal allows an Else statement to be used with

See List of Advertisers on page 306

"One last warning. . . if you have loaded the P-code of a program from tape, do not try to use the edit mode."

Case. This is useful if special instructions are needed when the selector variable has a value that does not correspond to the branch numbers.

The following program is a variation of the Menu Selector Program. The menu is offered repeatedly until option three is chosen. If the user inputs a value for Choice other than 1, 2 or 3, the message "Stick to the menu!" will be printed, followed by a return to the menu.

Take special care when entering this program; the Case statement is particularly finicky about semicolons.

```
(*MENU SELECTOR II*)
VAR CHOICE,I:INTEGER;
BEGIN
 I: = 1;
 REPEAT
  WRITE(13,'ENTER THE NUMBER OF THE ACTIVITY
   YOU WANT.');
  WRITE(13,'1—ACTIVITY FOR BEGINNERS');
  WRITE(13,'2—ACTIVITY FOR VIRTUOSOS');
  WRITE(13,'3—I DO NOT WANT ANY ACTIVITY',13);
  READ(CHOICE#);
  CASE CHOICE OF
   1:BEGIN
    WRITE(13,'YOU HAVE CHOSEN 1');
    WRITE(13,'THIS IS A GOOD CHOICE FOR
    BEGINNERS.')
   END;
  2:WRITE('YOU HAVE PICKED A TOUGH ONE. GOOD
  LUCK!');
  3:BEGIN
    WRITE('YOU HAVE CHOSEN TO END THE
    PROGRAM.');
    I: = 0
   END
  ELSE
   WRITE('STICK TO THE MENU!')
  END
 UNTIL I = 0
END.
```

Saving and Loading

You may save both source file and P-code for a program written in Tiny Pascal. The commands are "WS filename" and "WP filename", to save source and object files, respectively. Loading is done with the commands "LS filename" and "LP filename". File names may not exceed six characters in length. All commands to save and load are given from the monitor mode.

There are three warnings in the Tiny Pascal manual about loading and saving programs. When loading a program, you must enter the file name *exactly* as it was saved on tape. Warning number two is that there is no way to read the names of files on tape: If you forgot the file name you cannot retrieve the program.

The third warning in the user's manual is to use care not to get the source file and the object code mixed up when loading from tape. If you try to load the P-code, for example, with the command "LS filename", you will have to reload the entire system.

One last warning, this one my own. If you have loaded the P-code of a program from tape, do not try to use the edit mode. Leaving the monitor will result in the loss of your program, and it will have to be reloaded from tape.■

Construction	Statement Delimiters	
Entire Program	BEGIN	END.
IF THEN ELSE	BEGIN	END
FOR DO	BEGIN	END:
REPEAT UNTIL	REPEAT	UNTIL. . .;
WHILE DO	BEGIN	END;
CASE	CASE. . .	END
CASE BRANCHES	BEGIN	END; *

*Omit semicolon at end of last branch before Else or End.

Table 1. Summary of Multiple Statement Delimiters

Your best Model III peripheral buy is a modem.

Your best Model III modem buy is LYNX.

OK, you've familiarized yourself with your new TRS-80 Model III, and you're ready to expand.

You've got plenty of peripherals to choose from.

Disk drives, voice boxes, printers, joy sticks—the list is almost endless. And all may help you get more enjoyment out of the info you put in.

But there's the catch. Your Model III still relies solely on you for input. Without you to write increasingly complex programs—or pay cold cash to buy them—it's blind, deaf and dumb.

The real expansion will begin only when you give it ready access to the larger world of data communications. The world of free-access "bulletin board" resources. The world of electronic mail, instant news and financial reports and games from vast, affordable services like Source and CompuServe. The world of thousands of computer people just like you, in homes and businesses around the block and across the country.

It's a world you can tap through your telephone . . . but only if you're properly equipped.

You need a modem. And not just any modem.

You need LYNX.

The new LYNX is the ideal modem for your Model III. That's because it's the key element of a total direct-connect telephone communications package.

This package—including serial and computer bus interfaces, cables and terminal software—is available to you for a remarkable **SINGLE PRICE: $299.95.**

Compared to other modems, LYNX is superior. It doesn't make you buy adapter cables or circuit boards separately to bring it on line. It provides a full one-year factory warranty, instead of the "normal" 90 days.

It **includes** performance features—like **auto dial/answer,** programmable I/O porting, variable word length, parity and stop bits—that are either optional at extra cost or unavailable with other modems. It requires no tools to install.

It can be used with any TRS-80 communications programs; it's not restricted to its own special software. A full range of easy-to-find ST-80 cassette and disk software authored by Lance Micklus is available nationwide.

And best of all, when you compare the LYNX price with the total cost of bringing **any** other modem on line, it could save you **OVER $100.**

The new LYNX. It's the latest innovation from the people who made data communications affordable for TRS-80 Model I and Apple II users. It's the best first step you can take in expanding. It makes your Model III a whole new animal.

STILL AVAILABLE—LYNX for TRS-80 Model I. Price: $279.95.

ESI LYNX ✓278

123 LOCUST STREET LANCASTER, PENNSYLVANIA 17602
Phone 717/291-1116

TRS-80 is a trademark of the Radio Shack Division of Tandy Corporation.

Call one of these LYNX handlers today—TOLL FREE:

ADVENTURE INTERNATIONAL
Longwood, FL
800/327-7172

THE PROGRAM STORE
Washington, DC
800/424-2738

ADVANCED COMPUTER PRODUCTS
Irvine, CA
800/854-8241

STEVENS RADIO SHACK
Phoenixville, PA
800/345-6279

TSE HARDSIDE
Milford, NH
800/258-1790

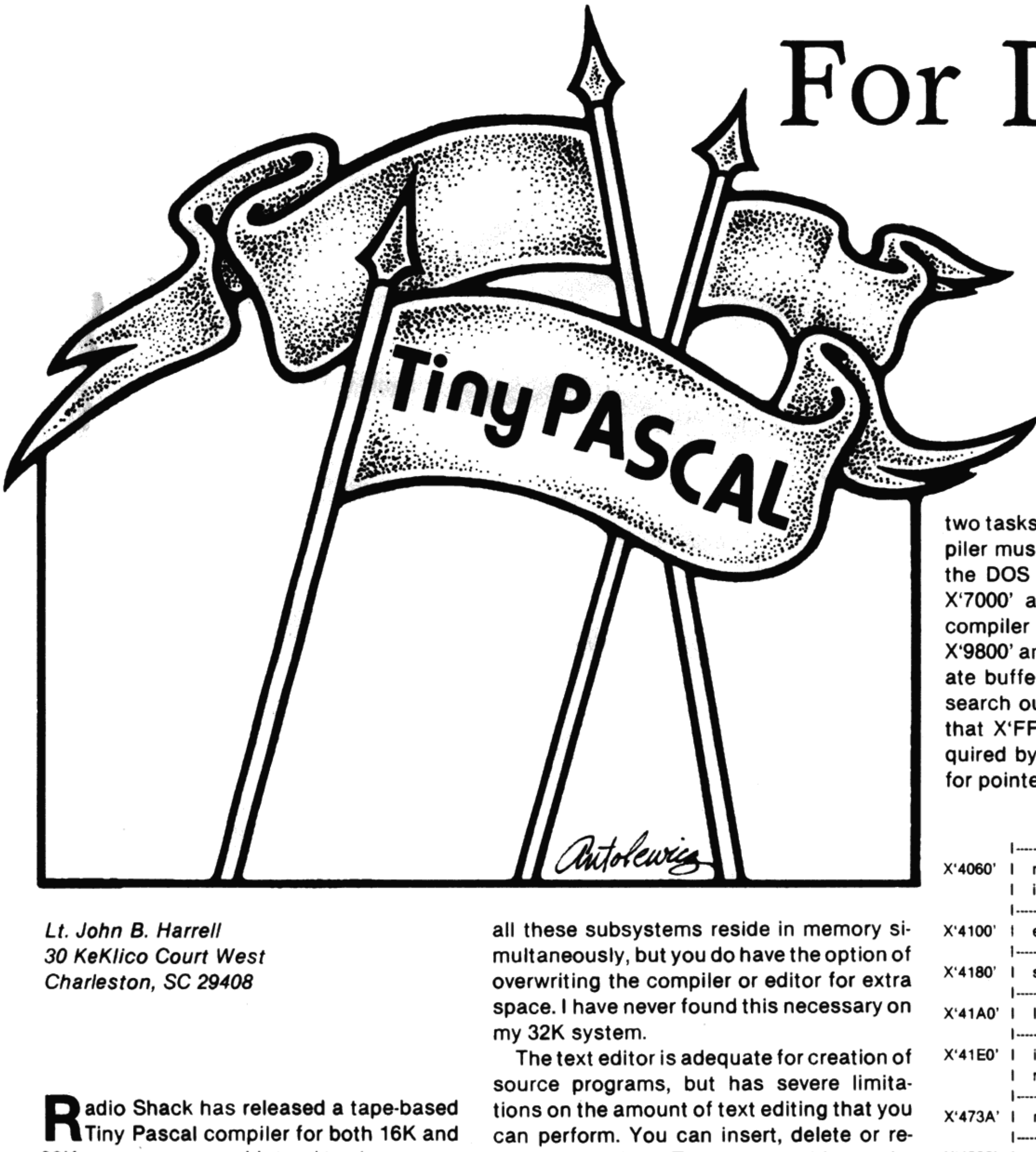
SIMUTEK
Tucson, AZ
800/528-1149

VISA

master charge

FCC Registration Number: A909KE-69488-DM-N

Modify Tiny Pascal For Disk



Lt. John B. Harrell
30 KeKlico Court West
Charleston, SC 29408

Radio Shack has released a tape-based Tiny Pascal compiler for both 16K and 32K+ system users. I intend to show users of 32K+ disk-based systems how to make this compiler run from the disk using disk input and output of source files. The article also shows that compiled code can be stored and loaded from the disk.

The Tiny Pascal system is a complete, self-contained operating system, consisting of a monitor, interpreter, compiler and editor. The monitor operates the entire system, and, additionally, saves and loads source and object programs to the cassette recorder. Output from the compiler is a compressed three-byte P-code. Normally, all these subsystems reside in memory simultaneously, but you do have the option of overwriting the compiler or editor for extra space. I have never found this necessary on my 32K system.

The text editor is adequate for creation of source programs, but has severe limitations on the amount of text editing that you can perform. You can insert, delete or replace entire lines. To overcome this restriction, I have modified the compiler to look for an ASCII text file that has been previously loaded into RAM storage, starting at address X'9800' to the end of memory. This is done by terminating the text file with an X'FF' byte as an end-of-file mark. (This is consistent with the end-of-file mark used by the compiler, but it would be easy to use Electric Pencil to create a file and then change the Pencil X'00' EOF to X'FF'.)

Modifying Tiny Pascal

To modify the compiler to run from disk, two tasks must be accomplished. The compiler must first be moved up to load above the DOS system requirements (I selected X'7000' as my load point). Secondly, the compiler must be modified to start at X'9800' and move text down to the appropriate buffer at X'73F0' (Fig. 1). It must then search out the end of text file (Remember that X'FF' byte?) and set the pointers required by the compiler system. See Fig. 2 for pointers that must be set.

X'4060'	reserved RAM for interpreter & monitor
X'4100'	entry points table
X'4180'	system control block
X'41A0'	I/O routines
X'41E0'	interpreter and run time routines
X'473A'	monitor
X'4990'	run time stack for editor or compiler (3-1/4 K)
X'5690'	editor P-code
X'5EA0'	compiler table
X'5FC0'	compiler P-code
X'73F0'	user memory for source and P-code

Fig. 1. Memory Map for the 32K+ Tiny Pascal System

"The Tiny Pascal system is a complete, self-contained operating system, consisting of a monitor/interpreter, compiler and editor."

To make the modification, type in Program Listing 1 using the Editor/Assembler, assemble it and write a system tape. You have now created the patch to the Tiny Pascal system, and all you must do to run it from the disk is load the Tiny Pascal system tape, 32K version (PAS32K), under normal Level II load procedures. To enter Level II, you must use the DOS BASIC2, or push reset while holding down the Break key. Then load the patch tape, execute it, and follow the instructions on the video screen. When it exits to the DOS reboot, dump the modified compiler system to disk using the address parameters displayed on the video.

Program Listings 2 and 3 are used to load a source file to memory and execute the compiler, and to save source files to the disk.

Listing 1 modifies the Tiny Pascal system in two parts. First, by moving the system from its resident area on loading (X'4D90' to X'73C6') to the area of RAM starting at address X'7000' and ending at X'9636'. And then it links the high RAM source code block move to the initialization phase of the compiler.

Lines 9–12 clear the screen and issue the prompt. Pressing any key will finish the patch. Lines 13–16 move Tiny Pascal to its new resident area. Lines 17–18 patch the source code block move into the compiler initialization phase. Lines 20–22 display the "done" message, wait for any key to be pressed and then reboot so the code can be loaded to the disk.

The rest of the patch code is ORGed to load in X'9637' following the moved compiler code. The initial phase of this code is identical to the power-up reset on the ROM chip, and is necessary because the compiler expects to be run in a Level II environment and not under the TRSDOS initialization of RAM areas X'4000' to X'40C0'. (This is lines 27–43 in the patch program). Lines 47–50 move the compiler from its disk load address to its normal load point at X'4D90'.

To clear the way for the source code to be

ADDRESS	FUNCTION
4180	Starting address of source code
4182	Ending address of source code
4184	Starting address of P-code
4186	Ending address of P-code
418C	Also contains starting address of source code
4196	Address of program currently in execution

All addresses are in hexadecimal

Fig. 2. Addresses Used to Patch the Tiny Pascal System

```
                 00001 ;*****************************************************
                 00002 ;*                                                   *
                 00003 ;*            TINY PASCAL DISK MCD                   *
                 00004 ;*                                                   *
                 00005 ;*            PRCGRAM LISTINC #1                     *
                 00006 ;*                                                   *
                 00007 ;*            WRITTEN BY JOHN B. HARRELL             *
                 00008 ;*                      12/05/80                     *
                 00009 ;*                                                   *
                 00010 ;*****************************************************
                 00011
B000             00012        ORG     0B000H
B000 3100B0      00013 PASMOD LD      SP,$
B003 CDC901      00014        CALL    CLS           ;CLEAR SCREEN
B006 219D96      00015        LD      HL,MS1
B009 CDA728      00016        CALL    OUTSTR        ;START PROMPT
B00C CD4900      00017        CALL    INKEY
B00F 21C673      00018        LD      HL,73C6H      ;END OF 32K PASCAL SYS
B012 113696      00019        LD      DE,9636H      ;WHERE IT WILL END
B015 013726      00020        LD      BC,2637H      ;BYTE COUNT
B018 EDB8        00021        LDDR                  ;MOVE IT OUT OF DOS
B01A 21D5BF      00022        LD      HL,MOVEIT
B01D 220C70      00023        LD      (700CH),HL    ;PATCH ADDR IN PAS32K
B020 210297      00024        LD      HL,MS2
B023 CDA728      00025        CALL    OUTSTR        ;END AND DONE
B026 CD4900      00026        CALL    INKEY
B029 C30000      00027        JP      0             ;REBOOT DOS
                 00028

                 00029 ;      CODE PATCH FOR PAS32K
                 00030
9637             00031        ORG     9637H         ;PATCH ADDRESS
9637 F3          00032 PATCH  DI
9638 AF          00033        XOR     A
9639 21D206      00034 PATCH1 LD      HL,06D2H
963C 110040      00035        LD      DE,4000H
963F 013600      00036        LD      BC,36H
9642 EDB0        00037        LDIR                  ;RE-WRITE LEVEL-II
9644 3D          00038        DEC     A             ;VECTOR AREA
9645 3D          00039        DEC     A             ;JUST LIKE ON A
9646 20F1        00040        JR      NZ,PATCH1     ;POWER-UP RESET
9648 0627        00041        LD      B,27H
964A 12          00042 PATCH2 LD      (DE),A        ;ZERO 39 BYTES
964B 13          00043        INC     DE
964C 10FC        00044        DJNZ    PATCH2
964E 118040      00045        LD      DE,4080H
9651 21F718      00046        LD      HL,18F7H
9654 012700      00047        LD      BC,27H
9657 EDB0        00048        LDIR
                 00049
                 00050 ;      END OF POWER-UP ROUTINE

                 00051
9659 210070      00052        LD      HL,7000H
965C 11904D      00053        LD      DE,4D90H
965F 013726      00054        LD      BC,2637H
9662 EDB0        00055        LDIR
                 00056                              ;MOVE PAS32K BACK
                 00057 ;      MOVE NEXT BLOCK TO HIGH CORE
                 00058
9664 217296      00059        LD      HL,START
9667 11D5BF      00060        LD      DE,0C000H-END+START-1
966A 012B00      00061        LD      BC,END-START+1
966D EDB0        00062        LDIR
                 00063

                 00064 ;      EXECUTE COMPILER
                 00065
966F C3904D      00066        JP      4D90H
                 00067
                 00068 ;      COMPILER WILL LINK TO THE ROUTINE IN HIGH
                 00069 ;      MEMORY TO MOVE SOURCE PROGRAM DOWN TO BUFFER
                 00070
9672 210098      00071 START  LD      HL,9800H      ;START OF BASIC BUFFER
9675 11F073      00072        LD      DE,73F0H      ;START OF PASCAL BUFFER
9678 010028      00073        LD      BC,2800H      ;BUFFER LENGTH
967B EDB0        00074        LDIR
```

Program continues

"Normally, all these subsystems reside. . . simultaneously, but you do have the option of overwriting. . . I have never found this necessary."

placed in its proper buffer, a small segment of code in lines 54-57 moves the source code loader to high memory where it will be out of the way. The last step is to execute the compiler, which will, in turn, move the system segments around and execute the source code loader prior to displaying the user prompt.

The last part of the patch program is the segment from line 66 to line 81. This portion of code moves the source code file from its load address (X'9800') to the respective buffer address (X'73F0'—Fig. 1) using a block move. Pointers are set to the starting address of the source code, and a block compare is executed to find the X'FF' end of file byte in the source code. On exit from the block compare, register HL contains the end of source code + 1, the start of the compiled P-code. Pointers are set for this and for the end of source code, and the compiler is executed. Reference Fig. 2 for pointers to set.

"I have found Tiny Pascal. . . an exciting, . . . powerful language"

The following changes to Listing 1 will enhance the operation of a 48K system and allow larger files to be used. Change the byte count in line 68 from X'2800' to X'6800'. Change the byte count in line 87 to read 0FFD5H vice 0BFD5H for the label MOVEIT. Change the address part of line 55 from 0C000H-. . . to 0-. . . .

Loading and Saving Source Files

Program Listings 2 and 3 are the mechanism through which source files are loaded and saved using disks. Enter both programs and save them under appropriate names. Pascal programs can now be loaded and saved.

Listing 2 loads a Pascal file from disk and calls Tiny Pascal. Line 1 protects memory above X'9800' and clears string space for the disk file to be loaded to a string array. You are requested to enter a filespec for the Pascal source file. If no file name is entered, the buffer is immediately terminated by the X'FF' end-of-file byte, and the compiler is called.

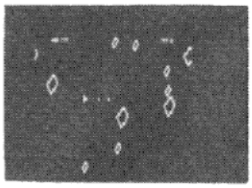
If the file is successfully opened, it is read line by line into a string array, terminat-

```
967D 3EFF      00075          LD      A,0FFH          ;TERMINATOR CHAR
967F 21F073    00076          LD      HL,73F0H
9682 010028    00077          LD      BC,2800H
9685 228041    00078          LD      (4180H),HL      ;PROGRAM START
9688 228C41    00079          LD      (418CH),HL      ;SAME
968B EDB1      00080          CPIR                    ;FIND TERMINATOR
968D 228441    00081          LD      (4184H),HL      ;START OF P-CODE
9690 228641    00082          LD      (4186H),HL      ;END OF P-CODE
9693 229641    00083          LD      (4196H),HL      ;ADDR OF CURRENT PGM
9696 2B        00084          DEC     HL              ;POINT TO END OF SOURCE
9697 228241    00085          LD      (4182H),HL      ;END OF SOURCE
969A C33A47    00086          JP      473AH           ;EXECUTE TINY PASCAL
969C           00087 END      EQU     $-1
               00088
01C9           00089 CLS      EQU     01C9H           ;CLEAR SCREEN
0049           00090 INKEY    EQU     0049H           ;WAIT FOR KEY
28A7           00091 OUTSTR   EQU     28A7H           ;OUTPUT STRING
BFD5           00092 MOVEIT   EQU     0BFD5H          ;SOURCE MOVER
               00093
969D 0D        00094 MS1      DEFB    13
969E 0D        00095          DEFB    13
969F 49        00096          DEFM    'IF YOU HAVE LOADED TINY PASCAL <PAS32K>'
     46 20 59 4F 55 20 48 41
     56 45 20 4C 4F 41 44 45
     44 20 54 49 4E 59 20 50
     41 53 43 41 4C 20 3C 50
     41 53 33 32 4B 3E
96C6 0D        00097          DEFB    13
96C7 56        00098          DEFM    'VERSION, PRESS ANY KEY---
     45 52 53 49 4F 4E 2C 20
     50 52 45 53 53 20 41 4E
     59 20 4B 45 59 2D 2D 2D
96E0 0D        00099          DEFB    13
96E1 4F        00100          DEFM    'OTHERWISE, HIT RESET AND LOAD IT'
     54 48 45 52 57 49 53 45
     2C 20 48 49 54 20 52 45
     53 45 54 20 41 4E 44 20
     4C 4F 41 44 20 49 54
9701 00        00101          DEFB    0
9702 0D0D      00102 MS2      DEFW    0D0DH
9704 49        00103          DEFM    'I HAVE FINISHED THE MODIFICATION.'
     20 48 41 56 45 20 46 49
     4E 49 53 48 45 44 20 54
     48 45 20 4D 4F 44 49 46
     49 43 41 54 49 4F 4E 2E
9725 0D        00104          DEFB    13
9726 50        00105          DEFM    'PRESS ANY KEY TO RE-BOOT DOS AND USE'
     52 45 53 53 20 41 4E 59
     20 4B 45 59 20 54 4F 20
     52 45 20 42 4F 4F 54 20
     44 4F 53 20 41 4E 44 20
     55 53 45
974A 0D        00106          DEFB    13
974B 54        00107          DEFM    'THE DOS <DUMP> COMMAND TO PUT IT ON DISK'
     48 45 20 44 4F 53 20 3C
     44 55 4D 50 3E 20 43 4F
     4D 4D 41 4E 44 20 54 4F
     20 50 55 54 20 49 54 20
     4F 4E 20 44 49 53 4B
9773 0D        00108          DEFB    13
9774 55        00109          DEFM    'UNDER THE NAME OF <PAS32K/CMD> WITH'
     4E 44 45 52 20 54 48 45
     20 4E 41 4D 45 20 4F 46
     20 3C 50 41 53 33 32 4B
     2F 43 4D 44 3E 20 57 49
     54 48
9797 0D        00110          DEFB    13
9798 54        00111          DEFM    'THE FOLLOWING PARAMETERS FOR THE DUMP'
     48 45 20 46 4F 4C 4C 4F
     57 49 4E 47 20 50 41 52
     41 4D 45 54 45 52 53 20
     46 4F 52 20 54 48 45 20
     44 55 4D 50
97BD 0D        00112          DEFB    13
97BE 53        00113          DEFM    'START=7000H, END=969CH, TRA=9637H'
     54 41 52 54 3D 37 30 30
     30 48 2C 20 45 4E 44 3D
     39 36 39 43 48 2C 20 54
     52 41 3D 39 36 33 37 48
97DF 00        00114          DEFB    0
               00115
8000           00116          END     PASMOD
00000 TOTAL ERRORS
```

Games from BIG FIVE will turn your computer into a

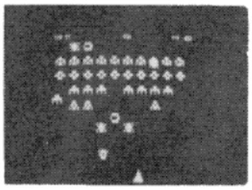
TRS-80 HOME ARCADE

SUPER NOVA©



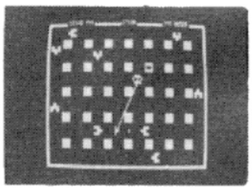
If you and your TRS-80 have longed for a fast-paced arcade-type game that is truly a challenge, then **SUPER NOVA** is what you've been waiting for. In this two player machine-language game, large asteroids float ominously around the screen. Suddenly your ship appears and you must destroy the asteroids before they destroy you! (But watch out because big asteroids break apart into little ones.) The controls that your ship will respond to are thrust, rotate, hyperspace, and fire. All right! You've done it! You've cleared away all the asteroids! But what is that saucer with the laser doing? Quick! You must destroy him fast because that guy's accurate! *As reviewed in May 1981* Byte *magazine.*

GALAXY INVASION©



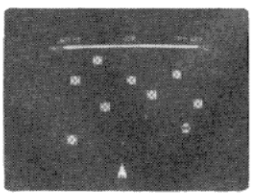
The sound of the klaxon is calling you! Cruel and crafty invaders have been spotted in battle formation warping toward Earth at an incredible speed. Suddenly, your ship materializes just below the huge flock of invaders. Quickly and skillfully you shift right and left as you carefully fire your lasers at them. But watch out! A few are breaking out of the convoy and flying straight at you! As the whine of their engines gets louder, you place your finger on the fire button knowing all too well that this shot must connect—or your mission will be permanently over! With sound effects!

ATTACK FORCE©



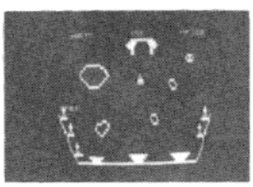
Your TRS-80 screen has been transformed into a maze-like playfield for this game. As your ship appears on the bottom of the screen, eight alien ramships appear on the top. All of them are traveling at flank speed directly at you! Quickly and boldly you move toward them and fire missiles to destroy them. But the more aliens you destroy, the faster the remaining ones become. If you get too good you must endure the wrath of the keeper of the mazefield: the menacing "Flagship". You must destroy him fast because, as you will find out, that guy's accurate! With sound effects!

COSMIC FIGHTER©

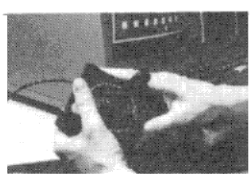


With thousands of stars whizzing by you, your **SPACE DESTROYER** ship comes out of hyperspace directly under a convoy of aliens. Almost effortlessly, you skillfully destroy every last one. But before you can congratulate yourself, another set appears. These seem to be slightly more intelligent than the first set. Quickly you eliminate all of them, too. But your fuel supply is rapidly diminishing. You must still destroy two more sets before you can dock with your space station. All right! The space station is now on your scanners! Oh no! Intruders have overtaken the station! You must skillfully fire your neutron lasers to eliminate the intruders from the station before your engines run out of fuel and explode! With sound!

METEOR MISSION II©



As you look down on your space viewer you can see the stranded astronauts that are crying out for you to rescue them. But first you must maneuver your shuttle down through the asteroids & meteors before you can reach them. Great! You've got one! But now can you get back to the space station to save your fellow shipmate or will you crash and kill both of you? You can fire your lasers to destroy the asteroids, but watch out, because there could be an alien FLAGSHIP lurking behind! Includes sound effects!



NEW JOYSTICKS!

For $39.95 it's now possible to have the famous ATARI joystick interfaced with your Model 1. All of our tapes are now completely compatible with the joystick. Packaged with complete instructions, you can even use it with your own programs! If your old tapes do not say "Joystick Version" on them and you wish to exchange them for new Joystick versions, enclose $2 and your old tape. (Call or write for info on Mod 3 joysticks.)

BIG FIVE SOFTWARE

P.O. Box 9078-185 · Van Nuys, CA 91409 · (213) 782-6861 357

Prices per game: Level 2, 16K Cassette Mod 1/Mod 3—$15.95
Level 2, 32K Diskette Mod 1/Mod 3—$19.95
10% discount for 2 items, 15% for 3 or more (excludes upgrades).
Please add $1.50 per order for postage & handling, Calif. residents add 6% sales tax.
Outside USA please add $3.00 per order for postage & handling.
We accept checks, money orders, and MC/Visa orders ($2.00 extra for COD).

All games ©1980 by Bill Hogue & Jeff Konyu.
Programs are written in machine language for high quality graphics.
Disk versions are self-booting and compatible with Mod 1 and Mod 3 disk systems.
High scores are automatically saved after each game on disk versions.
TRS-80 is a trademark of Tandy Corp.
Dealers: All games now available in full color packaging, please inquire.

"The text editor is adequate for creation of source programs, but has severe limitations on the amount of text editing that you can perform."

```
1 POKE &H40B1,&HFF:POKE &H40B2,&H97:CLEAR9900:CLS
2 DEFINT A-Z:AD=&H9800:DIM A$(500)
3 PRINT TAB(14)"** TINY * PASCAL ** FILE LOADER":PRINT
4 LINEINPUT"ENTER FILESPEC FOR SOURCE FILE: ";FS$
5 IF FS$="" THEN 19
6 ON ERROR GOTO 22
7 OPEN "I",1,FS$:ON ERROR GOTO 0
8 PRINT"LOADING *** "FS$" *** FROM DISK
9 PRINT:LN=0
10 LINEINPUT#1, L$
11 IF LEFT$(L$,1)<>CHR$(255) THEN LN=LN+1:A$(LN)=L$:GOTO 10
12 CLOSE
13 FOR I=1 TO LN
14 PRINT A$(I)
15 FOR J=1 TO LEN(A$(I))
16 POKE AD,ASC(MID$(A$(I),J,1))
17 AD=AD+1:NEXT:POKE AD,13:AD=AD+1
18 NEXT
19 POKE AD,255:POKE AD+1,255
20 CLEAR 50
21 CMD"PAS32K"
22 CMD"E":RESUME 4
```

Program Listing 2

```
1 POKE &H40B1,&HEF:POKE &H40B2,&H73:CLEAR 500:CLS:DEFINT A-Z
2 PRINT TAB(16)"** TINY * PASCAL ** FILE SAVER"
3 PRINT
4 LINE INPUT"ENTER FILESPEC FOR PASCAL FILE: ";FS$
5 IF FS$="" THEN 4
6 ON ERROR GOTO 15
7 OPEN"O",1,FS$:ON ERROR GOTO 0
8 AD=&H73F0
9 A$=""
10 J=PEEK(AD):IF J=255 THEN 14
11 AD=AD+1:IF J=13 THEN 13
12 A$=A$+CHR$(J):GOTO 10
13 PRINT#1,A$:PRINT A$:GOTO 9
14 PRINT#1,CHR$(255);CHR$(255):CLOSE:POKE&H40B1,&HFF:
 POKE&H40B2,&HBF:CLEAR50:END
15 CMD"E":RESUME 4
```

Program Listing 3

Breakout Program

Footnote to IF MEM lines following PROC PADDLE; these lines should read as follows:

IF MEM (KEYBD) = 32 THEN (USE← TO MOVE PADDLE LEFT*)*

IF MEM (KEYBD) = 64 THEN (USE→ TO MOVE PADDLE RIGHT*)*

```
CONST CURSOR=%4020;   VIDEO=%3C00;   KEYBD=%3840:

VAR     A,B,SPEED,XPOS,YPOS,PPOS,XDIR,YDIR,SCORE,SPVAR:INTEGER;
        K,STOP,BEST,NB,I,FLAG,TEMP,NBP:INTEGER;
FUNC RAND(N);
VAR TEMP:INTEGER;
BEGIN
   TEMP:=(A+B) AND %007F;   A:=B;   B:=TEMP;
   RAND:=((N*TEMP) DIV 128)+1   END;
PROC PTC(LINE,POS);
VAR TEMP:INTEGER;
BEGIN MEMW(CURSOR):=VIDEO+64*LINE+POS END;
```

Program continues

"Hitting the ball. . .causes varying amounts of backspin. . .losing the ball will get you a raspberry over the audio amplifier."

ing on the X'FF' byte. After closing the file, it is listed to the video and copied byte by byte into the buffer beginning at X'9800'. When this is finished, the compiler is called.

NEWDOS users may use Program Listings 2 and 3 as is. TRSDOS users will have to substitute <CMD"I","PAS32K"> in line 21 for <CMD"PAS32K">.

To save a Pascal source file, press reset when you have completed all operations requiring Tiny Pascal. After reboot, run the BASIC file saver. The text buffer is still intact in RAM beginning at X'73F0', and memory size will be set automatically to protect the area above X'73EF'. Enter a filespec to begin copying the file to disk and video. The disk file is ended by X'FFFF0D', and memory size is returned to full memory.

A Demonstration Pascal Program

Breakout (Program Listing 4) is an arcade game written in Tiny Pascal, and based totally on a program written in MMSFORTH by A. Shaeffer, published in *Byte* magazine, August, 1980, in an article by A. Richard and Jill Miller. This game is perfectly structured for Pascal and is a good example of the relative speed difference between Pascal and BASIC.

Pascal is a one-pass compiler requiring all variables, procedures, etc. to be defined prior to their use. The program, therefore, is a top-down design, bottom-up structure. A study of the MMSFORTH version shows quite a few primitives defined that are useful for the TRS-80.

The rules of Breakout are simple: Select the speed you want and the number of balls. As the ball hits more blocks in the wall, it moves higher and higher up the screen, gathering speed. The paddle is controlled by the right and left arrow keys.

If you are fortunate enough to chip away the entire wall, the screen will fill again, letting you play all of your balls. If you have sound hooked to your cassette port, you will hear a beep every time the ball collides with any white object on the screen. Hitting the ball with various portions of the paddle causes varying amounts of backspin to be put on the ball. Losing the ball will get you a raspberry over the audio amplifier.

The Game in Detail

The first function, RAND, returns a random number which is in the range of 1 to N. The method used is a Fibonacci series generator. This generator is seeded to start from the same initial value each time, and could be changed to randomize itself. In this particular function, N should be limited

```
PROC LINE(NUMBER);
BEGIN PTC(NUMBER,0);   WRITE(30)   END;

PROC BOP;
VAR I:INTEGER;
BEGIN
   FOR I:=1 TO 10 DO BEGIN OUTP(255,1); OUTP(255,2) END END;

PROC FILL(START,COUNT,CHAR);
VAR I:INTEGER;
BEGIN FOR I:= START TO START+COUNT-1 DO MEM(I):=CHAR END;

PROC PCLR;
BEGIN FILL(16320+PPOS,8,32) END;

PROC PSET;
BEGIN FILL(16320+PPOS,8,176) END;

FUNC MIN(A,B);
BEGIN IF A>B THEN MIN:=B ELSE MIN:=A END;

FUNC MAX(A,B);
BEGIN IF A>B THEN MAX:=A ELSE MAX:=B END;

PROC PADDLE;
BEGIN
   IF MEM(KEYBD)=32 THEN (* USE ']' TO MOVE PADDLE LEFT *)
      BEGIN PCLR;   PPOS:=MAX(2,PPOS-1);   PSET   END;
   IF MEM(KEYBD)=64 THEN (* USE '↑' TO MOVE PADDLE RIGHT *)
      BEGIN PCLR;   PPOS:=MIN(54,PPOS+1);   PSET   END   END;

PROC DSET(X,Y);
BEGIN PLOT(X+X,Y,1);   PLOT(X+X+1,Y,1)   END;

PROC DCLR(X,Y);
BEGIN PLOT(X+X,Y,0);   PLOT(X+X+1,Y,0)   END;

FUNC DTEST(X,Y);
BEGIN
   IF POINT(X+X,Y) AND POINT(X+X+1,Y) THEN DTEST:=1
   ELSE DTEST:=0   END;
PROC XCHK;
BEGIN
   IF XPOS<2 THEN
      BEGIN XDIR:=-XDIR;   XPOS:=2;   BOP   END;
   IF XPOS>61 THEN
      BEGIN XDIR:=-XDIR;   XPOS:=61;   BOP   END   END;

PROC YCHK;
BEGIN
   IF YPOS<5 THEN
      BEGIN YDIR:=1;   YPOS:=5;   SPVAR:=1;   BOP   END;
   IF YPOS<23 THEN SPVAR:=MIN(SPVAR,4);
   IF YPOS<19 THEN SPVAR:=MIN(SPVAR,3);
   IF YPOS<15 THEN SPVAR:=MIN(SPVAR,2)   END;

PROC PCHK;
VAR TEMP:INTEGER;
BEGIN
   FLAG:=0;
   IF YPOS>=47 THEN BEGIN
      YPOS:=46;   TEMP:=XPOS-PPOS;
      IF (TEMP>=0) AND (TEMP<8) THEN BEGIN
         YDIR:=-1;   BOP;
         CASE TEMP OF
            0: XDIR:=-2;   4: XDIR:=1;
            1: XDIR:=-1;   5: XDIR:=1;
            2: XDIR:=-1;   6: XDIR:=1;
            3: XDIR:=-1;   7: XDIR:=2   END   END
      ELSE FLAG:=1   END   END;

PROC INIT;
VAR I:INTEGER;
```

Program continues

". . . it is . . . a nice experimental compiler system. Even the casual experimenter can get his feet wet in this language."

to a value less than 128 for best results.

Procedures PTC and LINE are screen-control procedures. PTC sets the screen cursor to the value corresponding to the line number and character position it receives as parameters. LINE has one parameter—the line number. The cursor is set to the line number and that line is cleared.

BOP is the sound generator. Calling BOP causes the two low order bits of the cassette port to toggle, generating sound. A single call to BOP is used when the ball hits something, and a multiple call produces the raspberry noise when a ball is lost.

FILL is used to fill a block of screen RAM with a number from the selected character code. This procedure is used by PSET and PCLR to turn the paddle on the screen on and off.

PADDLE scans keyboard memory-mapped row X'3840' to determine whether the right and left arrow keys are being pressed. When a key is depressed, the paddle is cleared (PCLR), the paddle position is incremented or decremented while checking for screen limits, and the paddle is then set (PSET).

DSET, DCLR and DTEST are used to perform double-width graphics using the built-in functions of PLOT and POINT. A double graphic point defined by (X,Y) is set at the TRS-80 screen coordinates of (2*X,Y) and (2*X+1,Y).

XCHK, YCHK, and PCHK check the ball to see if it remained on-screen. XCHK reverses the X-direction and causes a BOP. YCHK checks the ball position and sets its speed according to its height inside the wall. If the ball hits the top screen border, it is reflected down at maximum speed with a BOP. PCHK checks to see if the ball should have or did hit the paddle. A flag is set to signify loss of the ball. If the ball hits the paddle, you get a BOP and the CASE statement is used to select the correct backspin.

CLR clears one block, adds up your score, gives it a BOP, and reflects the ball. CHKBALL increments the ball position and checks to see if the ball has hit anything. BALL uses CHKBALL to move the ball after clearing it, and resets the ball if it was not lost. CHKGAME checks if the wall is entirely gone. (The modulus of the score with 1800 will equal zero when the wall is gone.)

I have found Tiny Pascal to be an exciting, exceptionally powerful language. I realize much of the language isn't included in this very limited subset, but it is still a nice experimental compiler system. The price allows even the casual experimenter to "get his feet wet" in this language. ■

```
BEGIN
   WRITE(28,31,15);    (* CLEAR SCREEN, HOME CURSOR *)
   PTC(3,22);    WRITE('< B R E A K O U T >');
   LINE(10);    WRITE('SPEED (1-10, 1 IS FASTEST) ');
   READ(SPEED#);    SPEED := MIN(MAX(SPEED,10),1);
   LINE(12);    WRITE('NUMBER OF BALLS (1-50) ');
   READ(NB#);    NB:=MIN(50,MAX(1,NB));
   WRITE(28,31,15);
   FOR I:=0 TO 63 DO BEGIN DSET(I,3);    DSET(I,4)    END;
   FOR I:=3 TO 47 DO BEGIN
      DSET(0,I);    DSET(1,I);    DSET(62,I);    DSET(63,I)    END;
   FILL(15616,320,191);    SCORE:=0;    LINE(0);
   WRITE('BREAKOUT                      SCORE: 0      BEST: ');
   WRITE(BEST#);    PTC(0,54);    WRITE('BALL:')    END;

PROC CLR;
VAR I,TEMP:INTEGER;
BEGIN
   TEMP:=((XPOS-2) AND 124)+2;
   FOR I:=TEMP TO TEMP+3 DO DCLR(I,YPOS);
   SCORE := SCORE + ABS(YPOS-27);
   PTC(0,34);    WRITE(SCORE#);    BOP;
   YDIR := -YDIR       END;

PROC CHKBALL;
BEGIN
   YPOS := YPOS + YDIR;    XPOS := XPOS + XDIR;
   XCHK;    YCHK;    PCHK;
   IF DTEST(XPOS,YPOS) THEN CLR       END;

PROC BALL;
BEGIN
   DCLR(XPOS,YPOS);    CHKBALL;
   IF NOT FLAG THEN DSET(XPOS,YPOS)       END;

PROC CHKGAME;    (* CHECKS THE STATUS OF THE WALL AND GIVES
                    YOU A SURPRISE IF ALL GONE *)
BEGIN    IF (SCORE MOD 1800)=0 THEN FILL(15616,320,191) END;

PROC DELAY;
VAR I:INTEGER;
BEGIN FOR I:= 0 TO SPVAR*SPEED DO (* NOTHING *) END;


BEGIN (* MAIN ROUTINE OF 'BREAKOUT' *)

STOP:=0;    BEST:=0;
A:=55;    B:=89;    (* SEED RANDOM NUMBER GENERATOR *)

REPEAT
   PPOS:=28;    SPVAR:=0;    INIT;    PSET;
   FOR NBP:=1 TO NB DO BEGIN (* PLAY THE GAME *)
      FOR K:=1 TO (200 DIV SPEED) DO BEGIN DELAY; PADDLE END;
      PTC(0,61);    WRITE(NBP#);
      SPVAR:=5;    (*SLOW IT WAY DOWN UNTIL BLOCK IS HIT*)
      IF RAND(2)=1 THEN XDIR:=1 ELSE XDIR:=-1;
      YDIR:=1;    YPOS:=29;
      XPOS:=RAND(58)+2;
      REPEAT
         FOR K:=1 TO 3 DO PADDLE;
         BALL;
         CHKGAME;
         DELAY
      UNTIL FLAG;
      FOR K:=1 TO 12 DO BOP
   END (* OF THIS GAME, TEST FOR MORE *);
   BEST := MAX(BEST,SCORE);
   PTC(8,18);    WRITE(' RUN GAME AGAIN? ');
   REPEAT K:=INKEY UNTIL (K='Y') OR (K='N');
   IF K='Y' THEN STOP:=0 ELSE STOP:=1
UNTIL STOP

END (* OF BREAKOUT *).
```

TEXAS COMPUTER SYSTEMS

Offers Lowest Prices on

Color Computer

4K Level I $329
16K Level I $439
16K Extended Basic $489

With TCS Memory:
16K Level I $369
16K Extended Basic $449

Expansion Interfaces

Zero K Interface $254
16K Interface $359
32K Interface $469

With TCS Memory:
16K Interface $318
32K Interface $388

Pocket Computer & Interface

$ Call toll free

We have all Radio Shack computers and accessories, and a number of other brand computer items. Call us for your specific needs and get a copy of our catalog.

Epson MX-80

$ Call

Letter quality matrix similar to Line Printer IV & Centronics printer but has full software control of 40, 80, 66 or 132 columns. 80 cps bidirectional tractor feed, disposable printhead. $300 less than nearest competitive printer. Lists $645. Our price includes cable

SOFTWARE

MACPLUS $49.95

An Editor & Macro Assembler for Model III

We supply patches for the Macro Assembler to make it fully operative on the Model III. Our new editor includes virtually all the features of Radio Shack's SCRIPSIT plus these features: direct entry of graphics; auto tab for entering Assembly language instructions; imbedding of commands for justification, centering, margin control, change print types within text, etc. Also included on the disk are: **FORM/CMD** - Prints formated copies of your document to your specifications on your smart or dumb printer.

MODIFY/CMD — A utility which allows assembly language program modification from patches defined in an ASCII patch file.

MAC/BAS — A Basic program containing several utilities to aid in the preparation of source files.

Those described above and seven other complete programs make this an excellent value. Please specify Model I or III system.

Note: You must have, or purchase the Model I Microsoft Macro Assembler Package (Cat. #26-2202) from Radio Shack. If you don't already have this item, we are offering it at a special sale price in combination with the software package described above. Call for information and our low sale price.

ZAP III $19.95

A super zap for the Model III giving full sector and track access to any part of the TRSDOS diskettes. Jumps by tracks or sectors throughout the disk. Automatically goes to directory sectors of any disk with just one keystroke. Instant comparing of the same sectors on different disks. Full menus, lists, and descriptions of commands are available at any point. Just press clear and all commands and their descriptions are listed. Copy disk sectors, modify sectors, toggle between hex and ascii input of characters as you modify.

This disk also includes **COPYIT/CMD**, a utility program allowing the purchaser to make more than two back-up copies of SCRIPTSIT or VISICALC for private use. Good insurance for your valuable programs.

Special feature: the documentation includes a complete map of the location of all System Files on the Model III TRSDOS diskettes. Various patches also included to get rid of errors in TRSDOS 1.1 and 1.2 such as the mysterious EOF set to 0 on coping of files. This is a must purchase at $19.95.

MACPLUS-2 $39.95

A Macro Assembler Utility Kit.

Five programs for one low price:
RUN/CMD — A macro command processor.
SCREEN/CMD — Save video screen images to disk.
RELOCATE/CMD — Allows dynamic relocation of program.
PICMAC/CMD - Creates a screen of information that can be converted to an assembly language module and then used to print the screen.
PICBAS/CMD - Does the same but converts to a BASIC Program Module. Creates screens, menus, prompts, graphics, fast. It's like getting SCRIPTSIT for half price with other items and patches thrown in free.

Specify Model I or III.

EPSON MX-80 PRINTER DISKETTE

A complete diskette with 6 complete files for use with your Epson MX-80 Printer . . . ready to use.

M/CMD - Send all of the Epson Printer commands to the printer directly from DOS or as part of an auto booting system. Set up the printer from DOS to the typestyle you want; then run the program.

BIGLETT/BAS - Produce large graphic letters on your printer and screen. Dump the letters to printer or to disk. Allows 3 different size letters to the printer. Program is menu driven. You may also construct large letter titles for MERGEing into your basic program.

EPSON/SUB - Merge this small module into any Basic program and then issue any of the printer commands with a 2 letter command with letters representing the purpose of the command. You need not refer back and forth to documentation. Extremely easy to use and makes using the printer much easier than ever before.

JKL PATCH - This patches NEWDOS 2.1 and NEWDOS 80 so that the JKL function will print the TRS80 graphics while the printer remains in the standard mode rather than the TRS80 mode. This allows all of the features of the printer to be used but you can dump the screen to the printer at any time with all graphics.

DEMO/BAS - A super tutorial program with hundreds of REMs and hundreds of screen prompts that teach you how easy it is to create miracles with your Epson MX80 printer.

ARTICLE - A complete article about the Epson MX80 with all sorts of tips, concepts, etc. for getting the most out of your printer. The information for creating scheduling forms is a must for many of you.

DEFSTATE/SUB - A series of one word printer commands that can be merged into any Basic Programs that allow various automatic functions to occur. Print titles with graphic borders around them with one word. Center any text with one command, etc.

Special Price: All seven of the above modules on one diskette or on tape (less JKL) for your Model I or III for only **$29.95**.

Or, take a $10 discount if you buy this item with a purchase of the Epson MX-80.

The DOS Disk Library of Programs

A library of 5 programs for NEWDOS80, NEWDOS 2.1, TRSDOS (Model I & III)

DIRMAINT — A super directory-maintenance program. Purge, copy, rename, re-format, etc., quickly with minimum typing. Super graphic displays for easy use.

FILEMAINT — Creates master directory of all files on your library of diskettes. Master file is handled on disk, not on memory, for maximum speed. Access any file in 2 seconds. 48K supports 200 diskettes, 32K about 100. Takes less than hour to run 100 diskettes thru system.

CHAINBLD/BAS — A super chaining file creation editor & assembler for NEWDOS80. Very fast, easy to use. Reads and edits existing files.

HOMEMAIL — Unique home mailing list that stores data by modifying program itself. Fast & efficient for home user with less than 100 names.

GASMILE — Maintains excellent records of auto gas use & complete reports on MPG, etc. Also provides car maintenance alerts.

Model I or III Disk **$39.95** for all 5 programs

Printers

Daisy Wheel II
$1695

Line Printer IV $825
Line Printer V $649
Line Printer VI $1015
Line Printer VII $350

TEXAS COMPUTER SYSTEMS

Offers Lowest Prices on

TRS-80 COMPUTERS

Model II 64K $3349

An excellent computer for your business needs. Easy expandability & compatibility. No formal operator training needed. All accessories available—disk expansions, printers, software, at our low discount prices. Our fast Dallas air freight service can assure most deliveries within seven days after payment is received.

Free with purchase:
1 box of 10 double density diskettes. A $69 value.

SUGART DISK EXPANSION for the MODEL II

Uses the same reliable drive units found in the Model II CPU itself. Allows matched drives for reliable operations. Fast access time under CP/M or TRSDOS. Fully compatable with all existing software. Ready lights show disk activity. Silver/black paint matches Model II. The best drives available for less.

1 Drive, Single Cabinet $849
1 Drive, Triple Cabinet $949
2 Drives, Triple Cabinet $1424
3 Drives, Triple Cabinet $1899
Extra Drives $475

CORVUS HARD DISKS for the TRS-80 $Call

MICROSOFT

BASIC-80 - Extended disk Basic, long variable names, chaining, variable length records, ANSI compatable, interpretative version. **$294 w/manual. $30 manual only.**

BASIC COMPILER - Same Basic language as BASIC-80 except compiles Basic program into a directly executable, relocatable machine language file. Operates 3 to 10 times faster than regular Basic. Includes Macro-80 Assembler. Specify CPM or TRSDOS. **$325 w/manual. $30 manual only.**

FORTRAN-80 - Supports ANSI 66 plus many extensions including MACRO-80. Has library with manager, relocatable object compiler, and linking loader. **$400 w/manual. $30 manual only.**

COBOL-80 - Versitility gives Level 1 ANSI and most of Level 2. Allows complete sequential, relative and indexed file support with variable file names. STRING, UNSTRING, COMPUTE, VARYING/UNTIL, EXTEND, CALL, COPY, SEARCH, 3 dimensional arrays, compound and abbreviated conditions, nested IF. Interactive screen-handling extensions. Compatable assemler, linking loader, and relocatable library manager. **$575 w/manual. $30 manual only.**

Special—With TCS Memory

★ **Model III 48K 2 Disks $1895**

Uses proven MPI drives modified for flippy operation and Percom controller. Limited 90 day warranty. Call for details.

Model III 16K $849
Model III 32K $979
Model III 48K $1089

With TCS Memory:
Model III 32K $909
Model III 48K $969

Model III 32K 2 Disk RS232 $2100
Model III 48K 2 Disk RS232 $2230
Model III 32K 1 Disk $1729
Model III 48K 1 Disk $1849

- Payment: Money Order, Cashier's Check, Certified Check. Personal checks take 3 wks. VISA, MC, add 3%
- Prices subject to change any time.
- No tax out-of-state. Texans add 5%.
- Delivery subject to availability.
- Shipping extra, quoted by phone.

TEXAS COMPUTER SYSTEMS

✓25

Box 951, Brady Texas 76825

For fast, efficient service, we can air freight from Dallas to major a/p near you. Call for information.

Toll Free Number 800-351-1473

Texas Residents 915-597-0673

Pros and cons of this unconventional, stack oriented language.

A First Look at FORTH

John Krutch
P.O. Box 761
Crescent City, CA 95531

FORTH is an interesting and highly unconventional computer language that seems to be gaining a great deal of use. Its creator, Charles H. Moore, initially used it to control telescope equipment at Kitt Peak National Observatory in Arizona. Moore went on to found FORTH, Inc., which develops application programs and puts the language in new computers. FORTH has been used on the 6800, 6809, 8080, Z-80, 1802, 6502, and other microcomputers.

System Structure

A FORTH system includes a compiler and an interpreter. The compiler translates FORTH source code into intermediate code, which is a series of subroutine calls. The interpreter executes the intermediate code.

The subroutines of the intermediate code form a "dictionary" that occupies a major portion of the FORTH system. Each dictionary entry, or subroutine, is named by a word. FORTH programming builds sequences of words. The following are examples of words supplied with the system:

```
+
(
!
@
DUP
;CODE
CONSTANT
```

FORTH's nicest feature is that it allows you to create new words for new functions. New words are defined from predefined words. Once defined, a new word is added to the dictionary, and treated like a system-supplied word. New words are available for immediate execution or can be used to define other words.

FORTH also supports structured programming. In fact, for every control structure in Pascal, FORTH contains an equivalent structure. For example, there is an If...Else...Then structure which is very similar to Pascal's If...Then...Else. The Ncase...Casend construct is close to Pascal's Case...Of. Do...Loop is equivalent to Pascal's For...To...Do. While...Perform...Pend (Begin...While...Repeat or Begin...If...While in some versions of FORTH) is equivalent to Pascal's While...Do. And Begin...End is equivalent to the Pascal Repeat...Until. There is no GOTO in the language, nor does there seem to be a need for one.

The Stack

Programming in FORTH depends on a last-in, first-out stack, called the "parameter stack." Any number given to the system is placed on top of the parameter stack. The stack grows toward low memory.

If you type: 3 <enter>, where enter indicates a carriage return, 3 will be put on top of the stack. This process is illustrated in Fig. 1.

If you now type: 7 <enter>, 7 is placed on top of the stack (Fig. 2).

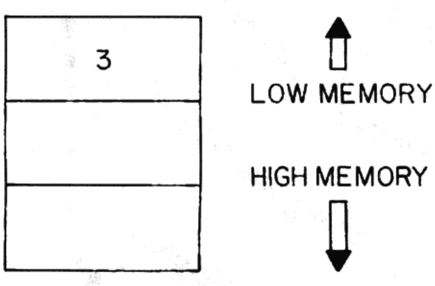


Fig. 1

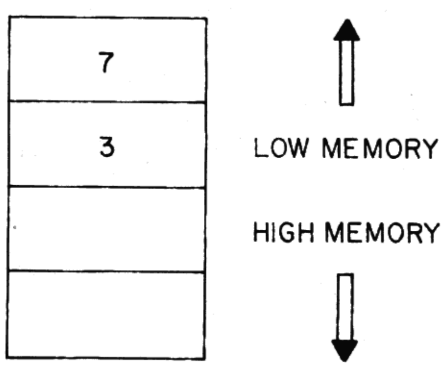


Fig. 2

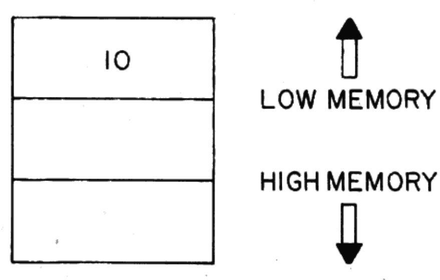


Fig. 3

"FORTH is an interesting and highly conventional computer language that seems to be gaining a great deal of use."

The FORTH word, +, takes the top two numbers on the stack, adds them together, and replaces them with the new value. If you now type: + <enter>, 10 is at the top of the stack, while 7 and 3 disappear (see Fig. 3).

At this point if you enter: <enter>, the top item of the stack is printed on the video. The stack is now empty.

Typing in items one at a time, followed by a carriage return, is not the only way to execute FORTH words. Entering the following sequence all at once would have the same effects: 3 7 + .

Let's examine more FORTH words. ECHO takes the number off the top of the stack and prints the ASCII character it represents; it performs exactly the same operation as the BASIC statement Print CHR$. If you enter: 65 ECHO, FORTH returns A, since 65 is the ASCII code for A. If you type: 48 56 90 ECHO ECHO ECHO <enter>, then Z-80 is printed.

Move copies a block of memory from one location to another. You specify how many bytes are to be copied, where the copying is to begin, and which block of memory locations will receive the copied bytes. The address where copying begins must be in the third-from-top position of the stack. The address of the first memory location to receive the copied bytes must be second from top. The number of bytes to be copied must be on top of the stack.

The TRS-80's memory-mapped video provides a way to watch Move working. If there is something besides empty space on the top line of the display, then: 15360 15808 Move <enter> will cause the entire top line of the display to be copied to a line near the middle of the display. (15360 is the top line's beginning address in video memory; 15808 is the middle line's beginning address in video memory; 64 is the number of bytes in one line and also the number of bytes to be moved.)

Decimal to Hex

FORTH switches easily from decimal to hexadecimal to octal arithmetic. The system starts up in decimal; the word Hex switches it to hexadecimal, and Decimal returns the system to base 10. Here's an example:

```
HEX 21 3C00 400 FILL DECIMAL <enter>
```

Hex puts the system in hexadecimal mode. The numbers 21,3C00, and 400 are put on the stack with 400 on top. Fill takes the third-from-top number (which must be one byte) and fills memory locations beginning with the second-from-top address with this byte. The number of locations filled is determined by the number on the top of the stack. The third-from-top number is hexadecimal 21, which is an ASCII exclamation point. The second-from-top number is hexadecimal 3C00, which is the first location of the video RAM. The top-of-stack number is hexadecimal 400 (decimal 1024). The word Fill causes the screen to fill with exclamation points. Decimal puts the system back in decimal mode.

Want a different radix, say, base 2? It's simple. The current radix is stored in the variable Base. The FORTH word C! (pronounced C-store) can be used to store a new radix in Base.

Suppose you want to see what the decimal number 1745 looks like in binary. Type: 1745 2 BASE C! . <enter>. 1745 is put on the stack, and 2 is put on top of it. Base puts the address of variable Base on the stack. C! puts the second-from-top number into the variable whose address is on the top of the stack; Variable Base now contains the radix 2. C! also removes from the stack both the address and the number that was put into the variable at that address, so 1745 is now on the top of the stack. The word "." prints the top-of-stack number. Since the system is now operating in base 2, the binary equivalent of 1745 is printed: 11011010001. The system will remain in base 2 until you type: DECIMAL <enter>.

New Words

To define a new word, use a colon to begin the definition and a semicolon to end it. For instance, to define the word Square, which takes the top-of-stack number and returns its square, you could do this:

```
: SQUARE DUP * . ; <enter>.
```

Square is the name of the new word. DUP causes the top-of-stack item to be duplicated; * multiplies the two top stack items together and places the result on top of the stack. The period prints the top-of-stack item on the display. The line is compiled instantly when you press the carriage return, which means that the word Square and its definition are placed in the dictionary and may be used like any other FORTH word.

If you enter 14 Square, the number 14 will be put on top of the stack, and the newly-defined word Square takes over and 196 is returned. Square always requires the number which will be squared to be on the top of the stack.

Creating a Source Program

In FORTH, a source program is stored as a series of contiguous screens or blocks. Each block is 1024 bytes (1K) long. This format is perfectly suited to the TRS-80 Models I and III, whose display can hold exactly 1024 bytes of character information, 16 rows by 64 columns. A FORTH block fills the TRS-80 display.

The Program Listing is a FORTH program which occupies one block. Line 0 is a comment line; anything within parentheses is ignored by the compiler. The program inputs a string from the keyboard; when enter is pressed, the string is printed on the screen backward.

Two procedures, GetCharacters and PopandPrint, are called by the main program and result in backward strings. GetCharacters marks the bottom of the stack by putting a negative number on it, then inputs whatever characters are typed and puts their ASCII codes on the stack. PopandPrint pops each code off the stack and prints it,

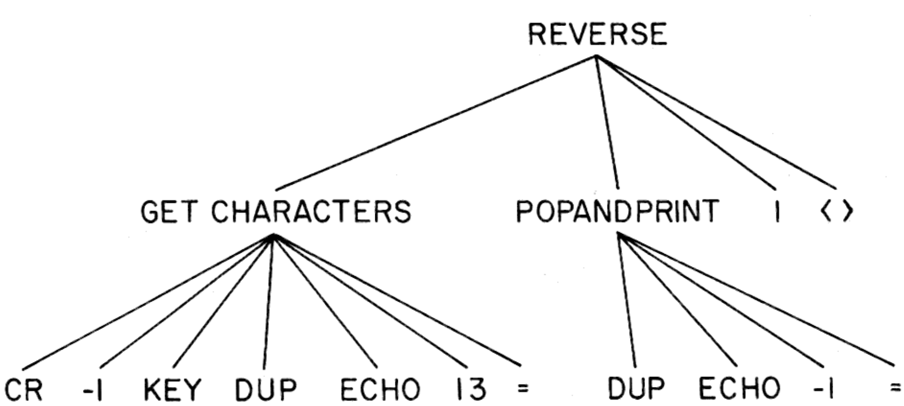


Fig. 4

"Though basic FORTH is fairly large... when you finish writing and compiling... you can strip away portions you don't need."

stopping when the end-of-stack marker (−1) is reached.

Since we're dealing with a LIFO (Last In, First Out) stack, the character codes are printed in reverse order from that in which they were typed. This is the reason for backward strings. The word Reverse causes GetCharacters and PopandPrint to be continually executed.

Once the block has been loaded and compiled, all you need do is type Reverse to begin execution. Reverse is defined in terms of GetCharacters and PopandPrint, which in turn are defined in terms of other words (see Fig. 4).

FORTH, Pro and Con

FORTH has a number of things going for it. Each implementation of FORTH is closely tailored to the hardware on which it's implemented. Programmers have nearly as much control over the computer as they would with assembly language. FORTH's modular, block-by-block programming style makes debugging easy, since each word can be tested and debugged as it is written. Because FORTH is implemented by a technique called "threaded code," it runs 10–20 times faster than an equivalent interpreted BASIC program.

Though basic FORTH is fairly large (mine occupies more than 10K), when you finish writing and compiling your program you can strip away portions you don't need, leaving perhaps 1K in memory at run-time.

On the other hand, there are disadvantages. Almost every individual operation in FORTH requires some sort of stack manipulation: A number is put on top of the stack, taken off the top, duplicated, swapped with the number above or below it, etc. This extensive manipulation of the stack leads to what is perhaps FORTH's greatest defect.

FORTH source programs (which are usually much longer and more involved than the Program Listing included here) are hard to read, and can be hard to write. There are so many stack manipulations during the course of a FORTH program that it's a difficult and puzzling task to keep track of them all.

There are more versions of FORTH for the TRS-80 available than any other language except BASIC. I use the system supplied by Miller Microcomputer Services of Natick, MA. Their FORTH is complete and well thought out, contains many extensions, and comes with an interactive 8080 assembler, among other useful items. Both tape and disk-based versions are available.

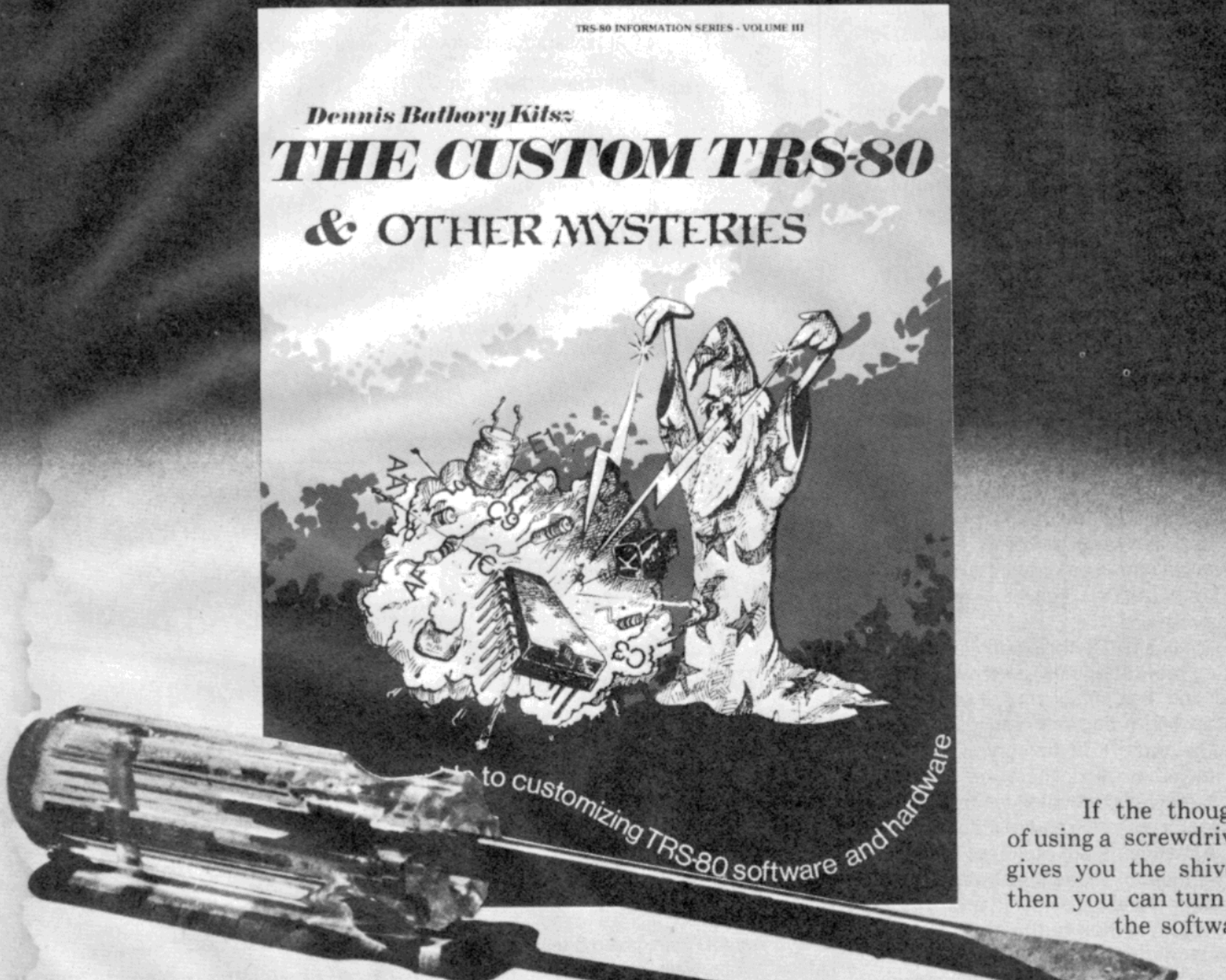
Sirius Systems, Programma International, and the Software Farm are some of the other suppliers of FORTH for the TRS-80. I want to leave you with one final warning: Good FORTH tutorial manuals are almost nonexistent at the present time.■

```
( REVERSE: ACCEPTS STRING FROM KEYBOARD AND INVERTS IT )

: GETCHARACTERS  CR  -1
   BEGIN
     KEY DUP DUP ECHO 13 =
   END ;

: POPANDPRINT
   BEGIN
     DUP ECHO  -1 =
   END ;

: REVERSE
   BEGIN
     GETCHARACTERS POPANDPRINT 1  1 <>
   END;
```

Program Listing

✓See List of Advertisers on page 306

TUNE-UP YOUR TRS-80



If the thought of using a screwdriver gives you the shivers then you can turn to the software section.

Ever wanted to do things to your TRS-80 that Radio Shack said couldn't be done? How about upper/lower case, reverse video, high-resolution graphics, a high-speed clock, audible keystrokes, an extra keyboard, and a real-time clock? Still not enough? How about using an 8-track as a mass storage device, making music, controlling a synthesiser, monitoring your data bus, and individual reverse characters? All these hardware modifications, plus lots more, are in **The Custom TRS-80** and Other Mysteries, by **Dennis Bathory Kitsz** – the latest book from **IJG Computer Services**.

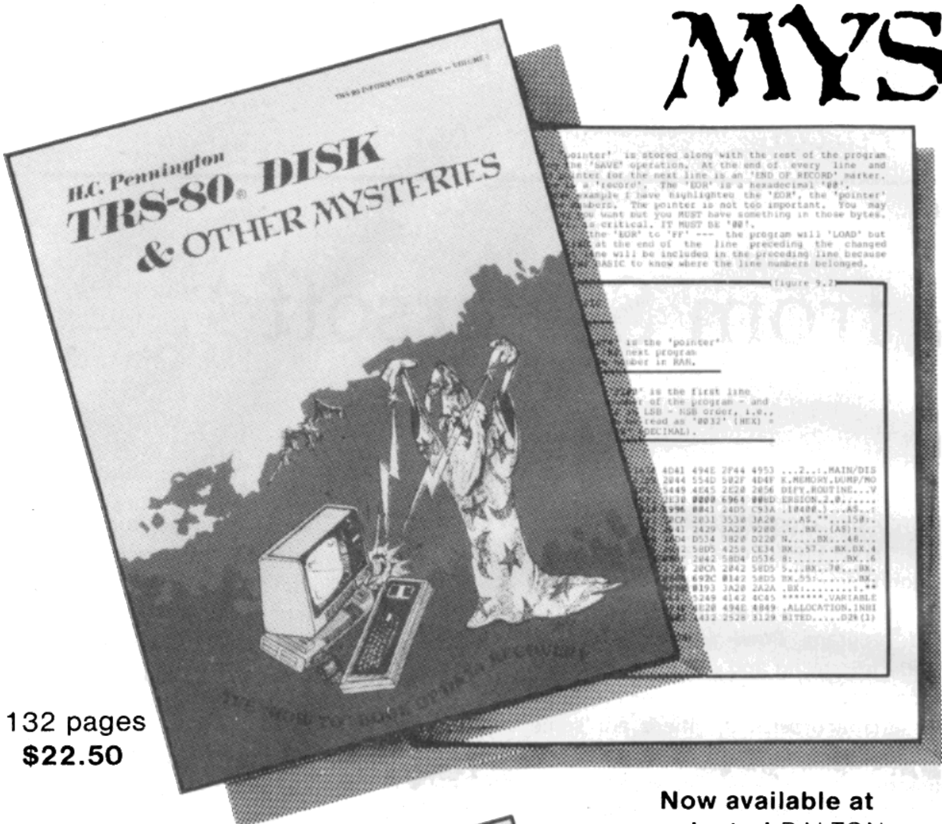
In this you learn how to make BASIC programs auto-execute, reset the memory size, patch into the interpreter, test memory with machine-language, pack program lines with machine code, and generate sound effects.

The Custom TRS-80 and Other Mysteries is more than 300 pages of practical information, and tested software, for **$29.95**. Order your copy now, and start turning your TRS-80 into a five-hundred-dollar supercomputer! Pick one up at your local **IJG** dealer or use the coupon on the opposite page.

IJG

'.. this is not only a worthwhile book but a great book. My advice is to get it and USE it!' - **William Barden Jr.**

OTHER MYSTERIES

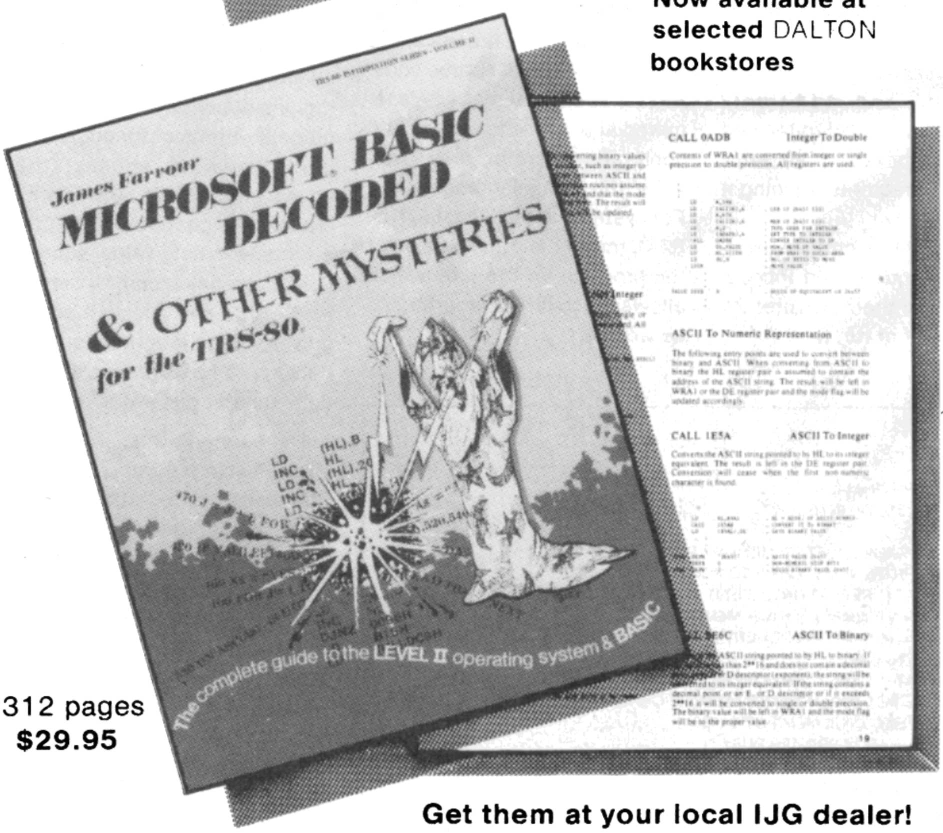


132 pages
$22.50

TRS-80 Disk and Other Mysteries is the definitive **fixit** book for disk users. Writen by Harvard Pennington it has more than 130 pages of easy to read, entertaining and immensely useful information - find out how to recover disk files, the layout of information on the disk, memory maps, problem solutions . . . the list goes on!

Many readers have saved days of work by recovering files that had been unreadable. Now in its fourth updated printing, **TRS-80 Disk** and Other Mysteries has been getting rave reviews in several magazines. Yours for only **$22.50** (plus $1.50 shipping, CA residents please add $1.35 sales tax).

Now available at selected DALTON **bookstores**



312 pages
$29.95

Microsoft BASIC Decoded and Other Mysteries is the complete guide to your Level II ROMs. With over seven thousand lines of comments and 6 additional chapters packed with information, this is easily the biggest, and best, book about the Level II ROMs available.

Exploit the full power of Microsoft BASIC, with the aid of more than 300 pages of tested examples, understandable explanations and detailed comments. Now available in a revised second printing, only **$29.95** (plus $2.00 shipping, CA residents add $1.80 sales tax).

Get them at your local IJG dealer!

'It has twelve thousand one-liners in it, and every one is great!'
- **Dennis Kitsz.**

Phone orders (714) 946-5805

IJG Computer Services, ✓37
1260 West Foothill Blvd.,
Upland, CA 91786

Please send me a copy of **TRS-80 Disk** and Other Mysteries for $22.50 plus $1.50 shipping. ☐

Please send me a copy of **Microsoft BASIC Decoded** and Other Mysteries for $29.95 plus $2.00 shipping. ☐

Please send me a copy of **The Custom TRS-80** and Other Mysteries for $29.95 plus $2.00 shipping. ☐

OK, send me all three **IJG** books for $82.40 plus $5.00 shipping. ☐

Overseas add $8.00 per book airmail.

Name ....

Address ....

City ....

State .... Zip ....

Charge my:

☐ MasterCard Interbank Code ....

☐ Visa Expiration Date ....

Card No. ....

☐ Check enclosed for ....

☐ Ship COD ($2.00 extra)

Signature ....

TRS-80 is a trademark of Tandy, Microsoft is trademark of Microsoft.

A useful tool for learning Pascal.

Tiny Pascal from Supersoft

'Tiny' Pascal
Supersoft
Champaign, IL
$40

Curtis H. Kyle
10821 Lamkins St.
Sun Valley, CA 91352

One day while I was thumbing through a magazine, I saw an ad for a 'Tiny' Pascal. I'd been interested in Pascal for some time. All the compilers I'd seen required a disk drive, or two, and at least 32K of memory. This Pascal required 16K, a cassette-equipped machine, and I sent for it.

The first thing I wanted to do, of course, was load the tape. I was disappointed to discover the program was only recorded once. Fortunately, the tape loaded.

The manual is not a primer on Pascal, so I bought one of the references recommended and began learning this new language. The syntax is simple, and before long I had written my first program. When I decided to list the program on my printer, I got my second big disappointment. There was no provision for printing lists, or outputting to a line printer under program control. I decided to write some programs that would provide these capabilities. (Program Listings 1 and 2.)

The system consists of four major parts: monitor, editor, compiler, and run time interpreter.

The monitor is in charge of overall system control. From the monitor you can enter the editor to create or modify source code, compile source code, and run the compiled program. Provisions are made for saving and loading source code or P-code using the cassette recorder.

The editor is adequate but doesn't compare with the Level II BASIC editor. Provisions are made for listing part or all of a program. Lines may be inserted or deleted and additional characters appended to the end of a line. Correcting an error within a line requires retyping it completely.

The compiler generates an intermediate code called P-code. The P-code is then executed by an interpreter at run time.

This leaves the programmer with about 4½K of memory for source and P-code. Normally the source code is entered and when it's compiled, the P-code immediately follows the source code in memory.

However, for large programs there are some options available. The programmer may choose to have the P-code replace the source code as it is compiled, making it possible to have a full 4½K of source code. Also, if additional memory is required at run time for arrays and such, the operator may overwrite the compiler and editor. For systems with at least 32K of RAM, there is another version of the program on the tape which allows much larger programs to be written.

```
*)
(* LIST *)
CONST PNTR = 14312;
VAR MP, TS, TA: INTEGER;
BEGIN
  MP: = %498E; (*SET UP MEMORY POINTER *)
  MEM(PNTR): = 10; (* CR/LF TO PRINTER *)
  REPEAT (* REPEAT UNTIL FIRST PERIOD ENCOUNTERED *)
    TS: = MEM(MP); (* FETCH MEMORY CHARACTER *)
    MP: = MP + 1 (* INCREMENT MEMORY POINTER *)
  UNTIL TS = 46;
  REPEAT (* REPEAT UNTIL PERIOD ENCOUNTERED *)
    TS: = MEM(MP); (* FETCH MEMORY CHARACTER *)
    REPEAT UNTIL MEM(PNTR)<128;
      (* WAIT IF PRINTER BUSY *)
    IF TS = 9 THEN FOR TA: = 1 TO 3 DO MEM(PNTR): = 32;
      (* INSERT TAB *)
    MEM(PNTR): = TS; MP: = MP + 1
      (* OUTPUT TO PRINTER AND INCREMENT MEMORY POINTER *)
    UNTIL TS = 46;
    MEM(PNTR): = 10 (* OUTPUT CR/LF TO CLEAR PRINT BUFFER *)
END.
```

Program Listing 1

What It Can Do

'Tiny' Pascal only supports integer variables. Lack of real (floating point) and character (string) variables limits its usefulness. All major control structures of Pascal are supported, including Begin...End, Repeat...Until, While...Do, If...Then...Else, For...Do, Case, Procedures, and Functions. Statements provide Read, Write, and integer arithmetic in decimal or hexidecimal, including one-dimensional arrays. Additional intrinsic functions include memory access (the equivalent of PEEK and POKE), machine language calls, I/O port access, absolute value, square, INKEY, graphics control similar to BASIC, and block memory moves. Game programmers will be sorry to hear there is no random number generator.

The listings are examples of 'Tiny' Pascal programs and provide some needed utilities. Program Listing 1 allows a program to be printed on the line printer. This program should be inserted before the program you want printed out. The compiler will only compile the List program and when executed, only the second program will be printed. Comments, of course, may be eliminated. They are enclosed by (* *). Using this technique, a program about 4K in length may be printed.

Program Listing 2 is a general purpose screen print routine. The parameters passed to it are the first line to be printed (1 to 16) and the number of lines to be printed. The main program listed is a test program for the screen print procedure. The printer output when full screen is selected is shown in Fig. 1.

While Level II BASIC is still more practical for any application, 'Tiny' Pascal offers a "shoestring" approach to learning this structured language.■

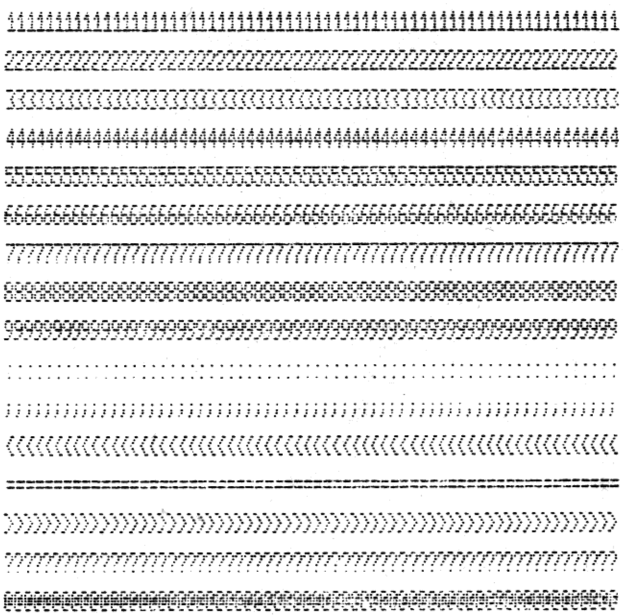


Figure 1

```
(* SCREEN PRINT *)
VAR STLN, NMLN, SADD, CHAR, CCNT: INTEGER;
PROC SPNT(STLN, NMLN);
CONST PNTR = 14312;
VAR FADD, LADD, CADD, TS: INTEGER;
BEGIN
FADD: = STLN*64 + 15296; LADD: = NMLN*64 + FADD - 1;
  (* COMPUTE FIRST AND LAST SCREEN ADDRESS *)
MEM(PNTR): = 10; (* OUTPUT CR/LF TO PRINTER *)
FOR CADD: = FADD TO LADD DO (* SET UP LOOP *)
BEGIN
  TS: = MEM(CADD); (* FETCH CURRENT SCREEN ADDRESS CHAR *)
  IF CADD MOD 64 = 0 THEN MEM(PNTR): = 10;
    (* CR/LF IF END OF LINE *)
  REPEAT UNTIL MEM(PNTR)<128; (* CHECK FOR BUSY *)
  MEM(PNTR): = TS (* OUTPUT CHARACTER TO PRINTER *)
END;
BEGIN (* MAIN PROGRAM TO TEST SCREEN PRINT PROCEDURE *)
  WRITE(28,31); (* CLEAR SCREEN *)
  WRITE('FIRST LINE'); READ(STLN#); WRITE(10,13);
    (* INPUT FIRST SCREEN LINE TO PRINT *)
  WRITE('NUMBER OF LINES'); READ(NMLN#); WRITE(28,31);
    (* INPUT NUMBER OF LINES TO PRINT *)
  CHAR: = 49; (* ASCII OF CHARACTER TO SCREEN *)
  SADD: = 15360; (* FIRST SCREEN ADDRESS *)
  REPEAT (* REPEAT UNTIL SCREEN FULL *)
    FOR CCNT: = 1 TO 64 DO
    BEGIN
      MEM(SADD): = CHAR;
      SADD: = SADD + 1
    END;
    CHAR: = CHAR + 1
  UNTIL SADD> = 16383;
  SPNT(STLN,NMLN) (* CALL SCREEN PRINT PROCEDURE *)
END.
```

Program Listing 2

Rose©

You've Got
TOTAL ACCESS©

(specializing in TRS80 *)

TO YOUR COMPUTER HARDWARE & SOFTWARE NEEDS. CALL ROSE TODAY!

I've Got

★ Disk Drives

TA400 (40-T) $289
TA 800 (80-T) $419
TA400 Flippy $319
TA800 Flippy $439
All with silver case and power supply

★ CABLES

2-Drive $23.95
4-Drive $33.95
Extender Cable $14.95

★ BARE DRIVES

TA400B $259
TA800B $389
TA400B Flippy $289
TA800B Flippy $409

★ MODEL III DRIVE KIT

Includes DD disk controller, two-drive power supply, mounting towers, all connectors, cables & hardware and instructions.
DO it yourself & save $395

★ OPERATING SYSTEMS

TRSDOS 2.3 Disk & Mannual ... $17.95
LDOS $129.95
NEWDOS/80 .... $129.95
DOSPLUS 3.3, 3.3D $89.95

★ IRON

Disk Drive Power Supply, Single $37
Disk Drive Case (silver) and Base $19
Memorex Diskettes (bx of 10) $25

90 day warranty on drives. Add $5.00 freight per drive in Cont. US. UPS COD charge $1.40. There is also a 15 day FREE TRIAL on TA drives. If not completely satisfied I'll refund your money (less shipping). I'll take exception to improper use or mishandeling.

★ USERS GROUPS

When your club makes a group buy, be sure to call me for a price.
Rose

TRS80 * COMPUTERS

★ COMPUTERS

26-1061 Mod III, LI, 4K $595
26-1062 Mod III, LII, 16K $850
26-1063 Mod III, 32K/2 Disk/RS232 .... $2150
26-4002 Mod II, 64K $3300
26-3001 Color Computer, 4K ... $330
26-3002 Color Computer, 16K .. $510
26-3501 Pocket Computer $212

★ DISK EXPANSION, MOD II

26-4160 One Disk $977
26-4162 Three Disk $1998
LOBO drive units also available.

★ PERIPHERALS

26-1140 ØK Exp. Interface $255
26-1141 16K Exp. Interface $305
26-1142 32K Exp. Interface $355
26-1172 D.C. Modem II $135
26-1206 CTR-80 Recorder $51
26-3503 P.C. Interface $42

★ PRINTERS

26-1165 Line Printer V $1581
26-1166 Line Printer VI $986
26-1158 Daisy Wheel II $1694*
* LIMITED AVAILABILITY

The complete line of Radio Shack computer products is available through TA with full RS warranty. Call me for price and delivery. Just because you don't see it don't mean we ain't got it!
Rose

★ SOFTWARE

WE have Adventure, Big 5, Med Systems, SBSG, Snapp, Radio Shack and much more. Call or write. Do it!

★ PRINTERS

EPSON MX-80 .... $479
EPSON MX-80 F/T $589
EPSON MX-100 .... Call
STARTWRITER $1799
45 CPS Daisy Wheel by C. Itoh
Parallel cables for any of the above $29.95

ROSE'S SPECIAL OF THE MONTH
EPSON MX-70
$369.00

Parallel cables $29.95
You'll hate yourself if you don't buy a dozen!

REAL TIME CLOCK CALENDER

T-Timer© $89.95

Clock continues to run no matter what mode you are in or when system is down with battery back-up*. Not affected by disk I/O. Plugs into screen printer port-has extender.
Sec, min, hr; day of week;
Date, mo, yr.
Mod. I only.

ORDER NOW! TOLL FREE 800-527-3582

Write or call Rose TOLL FREE at 1-800-527-3582(Texas residents call 214-234-1770). Please use the toll free lines for orders and literature requests only. Technical help or service use the Texas line. You can pay by VISA or MASTERCARD, you can send check or money order (allow a couple of weeks for personal checks to clear) or order COD (we ship COD's cash, certified check or money order only). Rose will take American money in just about any form. Add freight (UPS where possible) on all orders under $1000. If you buy $1000 or more Rose eats the freight! Texas residents cough up 5% sales tax. Allow 2-4 weeks for delivery. Order today - I need the money!

TOTAL ACCESS©
P.O. BOX 3002
RICHARDSON, TX 75080

214-234-1770 ✓6

* TRS80 & Radio Shack are trademarks of Tandy Corp.
© Copyright 1981 TOTAL ACCESS

TOACC/23

This one has a de facto seal of approval from Ft. Worth.

The Last CLOAD Fix

Walter L. Stanley
P.O. Box 15033
Las Cruces, NM 88001

The critical volume setting required for an error-free CLOAD has been of concern to most TRS-80 owners from the time they first installed Level II ROMs. As reported in the *Radio Shack Microcomputer Newsletter,* (May, 1979), they have developed a fix which they will install free of charge in all TRS-80s, *provided the seal is unbroken.* Harold Smith reported in the first issue of *80 Microcomputing* (80 Input) that the fix, which is on a 1½" × 1½" circuit board, is effective and well worth having installed. Other sources have likewise praised the modification as almost completely curing the CLOAD problem of volume sensitivity.

Alas, there is a substantial minority of TRS-80 owners who long ago opted to get inside the computer and change things around to improve and understand their machine. We did this with full knowledge that we gave up the option, forever, of having Radio Shack work on our TRS-80s. What are we to do to improve CLOAD?

We can plunk down $30 to $50 for an outboard cassette interface device to clean up the cassette signal, or we can continue to experience that sinking feeling when a C replaces the asterisk at the end of a five-minute program load.

I was delighted to have the opportunity to see inside a TRS-80 which has the CLOAD fix installed. (I understand that the modification is labelled the X2X mod by Radio Shack). Naturally, I documented it by tracing wires and foil cuts, and then spent a number of hours trying to understand it. I'd like to share what I found with you, since Radio Shack installs this mod at no charge, and I don't believe they would object to an owner building and installing it himself. The circuit can be built for less than $3 and an hour of time.

Fig. 1 is a combined circuit diagram and logic function diagram. Referring to the schematic in the *TRS-80 Microcomputer Technical Reference Handbook* is helpful, but not essential. The original cassette input circuit takes the recorder signal (CASSIN) through all four sections of a 3900 Norton amplifier (Z4 in the technical manual) where it is filtered, amplified, inverted, squared up, and desensitized to noise. The output of the final section of Z4 will be called CASSIN* since it is inverted. (To follow Radio Shack's notation, all active low signals are followed by an asterisk.) CASSIN* is normally high, and goes low in response to any signal from the

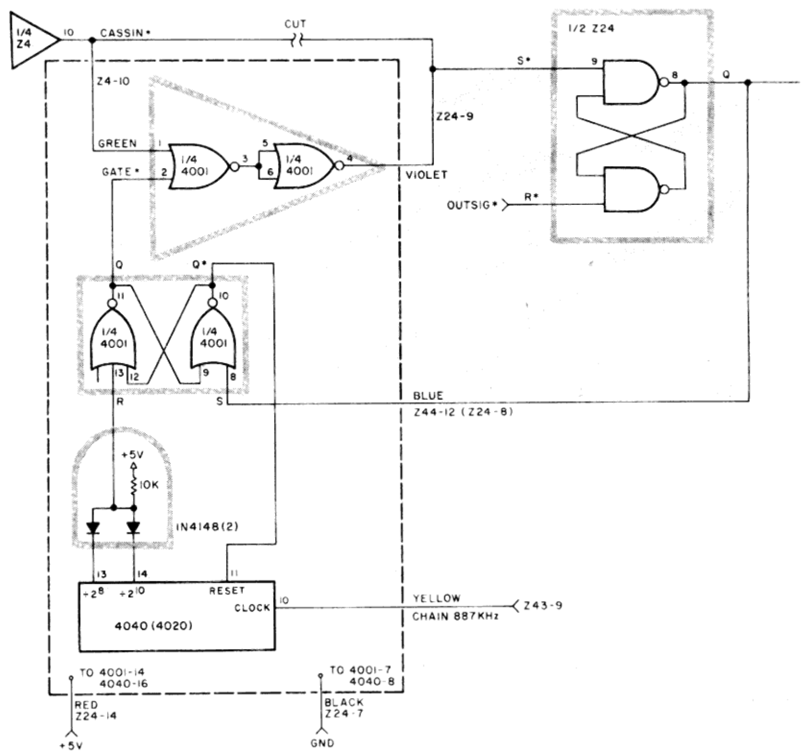


Fig. 1. This is the schematic for the CLOAD fix, with its location in the TRS-80 electronics. The dotted lines enclose all new components, which mount on a single 1½ inch × 1½ inch piece of perf board. The blue overprinting shows the grouping of gates functionally. An asterisk after a signal's label denotes active low logic.

recorder, be it a clock pulse, a data pulse, or noise that isn't filtered out. The recording format consists of clock pulses at two millisecond intervals and data windows halfway between clock pulses. A pulse will actually be present during this window if the data bit is a 1, but no pulse will occur if the data bit is a 0. It is possible, therefore, to see a legitimate pulse on the CASSIN* line as often as every millisecond.

What happens during a CLOAD is this: The cassette recorder is turned on by outputting data to port 255_{10}. Any output instruction to port 255 causes a signal line called OUTSIG* to go from high to low for the duration of the instruction (a matter of microseconds). Among other things, OUTSIG* is connected to pin 13 of Z24, an input to one of two NAND gates configured as an R-S flip flop. Pin 13 is the reset input of this flip flop, and since a negative going pulse causes the activity, we call that input R*.

With the recorder now running, eventually a pulse on CASSIN* will reach pin 9 of Z24 which is the set input (labelled S*) of that flip-flop. This pulse causes the Q output, pin 8 of Z24, to go high. The cassette input software immediately detects this high and executes an output instruction to port 255. This brings OUTSIG* low which resets the flip-flop and returns Q to a low state. The software is, meanwhile, in a carefully timed delay routine, and shortly after one millisecond has passed, it again tests whether Q is high or low. If it is high, another output instruction causes OUTSIG* to go low and resets the flip-flop. If Q is low, the software waits patiently for the appropriate time to test Q again.

The only flaw in the system is that any pulse that gets through the four sections of Z4 will set the flip-flop, which will stay set until the software tests its state. A noise pulse will always be seen as a 1 data bit, hence, the volume setting is critical. If it is too low, valid data pulses are missed; too high, too much noise is seen as data. And yet, there is only a tiny time interval during which it is possible for a valid pulse to be on the tape.

The solution exhibited by most of the outboard cassette improvers I have seen is to provide a noise-free signal to the cassette input of the computer. Essentially, this is done by a more sophisticated pulse shaping circuit than Radio Shack uses (Z4) which has the effect of presenting a digital rather than an analog signal to the computer. For those of you who do a lot of tape duplication, I heartily recommend that approach. For those of us not into tape duplication, it seems to be an approach that transforms a cheap tape recorder into an expensive one.

An alternate answer is to make Z24 insensitive to noise for most of the inter-pulse time; that is, to turn off the S* input to Z24. This could be done with software by delaying the output instruction to port 255, thus delaying the reset of Z24 until just before the next valid pulse is expected. In fact the new Level II two-chip ROM set does just that. Unfortunately, a 32K PROM is very expensive, and that is what contains the cassette load software. The Radio Shack CLOAD modification provides a cheap hardware alternative to an expensive reprogramming of the ROM software. Note from Fig. 1 that the direct path between pin 10 of Z4 and the S* input (pin 9) of Z24 is cut. Instead, the output from Z4 is routine to Z24 via a gated buffer made from two sections of a CD4001 quad NOR gate.

The buffer works as follows: So long as the input labelled GATE* is low, CASSIN* at pin 1 will be inverted at pin 3. The second NOR gate is merely an inverter, so the output signal finally presented to S* of Z24 is indeed CASSIN*. But if GATE* is made high, then pin 3 of the NOR gate will be low regardless of the state of CASSIN*, and Z24 sees only a high. The balance of the circuit controls GATE* so that after Z24 is set by a pulse, no further pulse can pass to Z24 until about 0.72 milliseconds later.

In order to accomplish that, the remaining two NOR gates are configured as an R-S flip-flop. Since the flip-flop is made of NOR gates, positive pulses set or reset its state. The set input, labelled S, is at pin 8 and is connected to the Q output of Z24. Take my word for it; pin 13 of the flip-flop, which is the reset input, is low at the start of a CLOAD, so Q* (pin 10) is high, which means that pin 11 (the Q output) is low, and so GATE* is low, and the gated buffer is free to pass CASSIN* right through to S* of Z24. Meanwhile, the Reset input of the CD4040 counter is held high since it, too, is connected to Q*, and this forces pins 13 and 14 of the counter low. The two diodes and the pull-up resistor form an AND gate, so the R input of the flip-flop is low.

Execute CLOAD, and wait for the first pulse. When it comes, it will pass right through the gated buffer and reach S* of Z24, triggering that flip-flop and causing Q to go high. This high triggers the set input of the NOR gate flip-flop, and so Q* goes low and Q goes high. Q high shuts the gate to any further pulses from CASSIN*, so that any pulses on CASSIN* (which must be noise) get sent to the bit bucket. Simultaneously, since Q* is low, Reset of the CD4040 is low, and the counter starts doing its thing at 887 KHz. About this time OUTSIG* goes low because the software saw that Q of Z24 was high. Q of Z24 goes low in response, and nothing else happens for awhile.

How long is awhile, you ask? Well, the CD4040 is happily counting pulses, and after 128 of them have been counted, pin 13 goes high (this is the divide by 2^8 output). Since the diode configuration is an AND gate, nothing happens. After 512 pulses have been counted, pin 14 goes high, but pin 13 has gone back low, so still nothing happens. But 128 pulses later, both pins 13 and 14 find themselves high, and the 10K resistor brings pin 13 of the CD4001 high. That resets the flip-flop, which lets CASSIN* get through to Z24 once again. "Awhile" turns out to be 1/887,000 times 640, or about 0.72 milliseconds, give or take a few microseconds.

These five components keep possible noise on CASSIN out of the system for nearly 75 percent

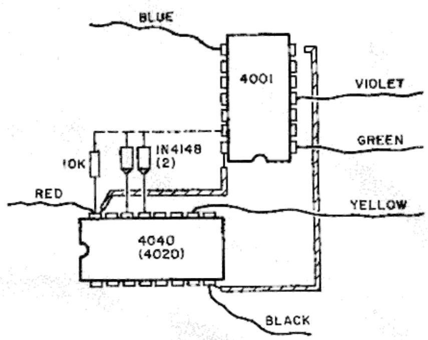


Fig. 2. Parts placement on the perf board is designed to permit use of component leads for much point to point wiring. In particular, note the placement of the diodes and resistor to allow their leads a straight shot to the IC pins to which they are soldered. Aside from the flying leads to the main circuit board, only +5V and ground wiring is on the top of the board.

✓See List of Advertisers on page 306

of the time between legitimate pulses. There is still a time interval when a noise pulse could trigger Z24 erroneously, but one does have to allow for variations in cassette recorder speed. Keeping noise out for 75 percent of the time gives a whopping improvement in reliability, especially when you consider that even if a noise pulse triggers the Z24 flip-flop, the next data pulse would have been a 1 about 50 percent of the time anyway. All this means that you can tolerate more noise output from Z4, which allows you to crank up the volume or your recorder by a pretty wide margin. This, in turn, means you are not as likely to lose legitimate data pulses. It doesn't seem, however, that this fix would do anything to help CLOAD at lower than recommended volume settings.

Building the circuit is very simple. Five components don't justify even a thought of a printed circuit board, so chop yourself off a 1½ inch × 1½ inch piece of perf board with holes on 1/10 inch centers. Layout of parts is not critical, but the arrangement shown in Fig. 2 allows some wiring with component leads. Contrary to many cautions, I have not found CMOS integrated circuits that fragile, and if you exercise reasonable care to avoid generating static electricity and solder quickly, you won't need sockets. Fig. 3 is a pictorial representation of the wiring on the underside of the board. (The only hooker in this project may be finding a CD4040. If you have one on hand, use it, but don't try to buy one at Radio Shack—it's not in their catalog.) Instead, pick up a CD4020. This is also a ripple counter, and thankfully, the pinout is identical to the CD4040 for the pins we use.

Time now to open the computer. If you are building this circuit you've done that before and don't need detailed instructions. Swing out the keyboard, there's no need to remove the main circuit board. With the keyboard unit positioned normally—that is, with the tape, video, and power jacks at the upper right hand corner—you will see a set of four rows of ICs from the foil side of the main circuit board. Z4 is the rightmost IC in the top row, and Z24 is directly below Z4 in the second row. Pin 1 of all ICs is the top right pin. Carefully cut the trace leading from Z4 pin 10 to Z24 pin 9. Solder the red +5V lead from the perf board to pin 14 of Z24, and the black ground lead to pin 7 of Z24. Any other convenient power take off points may be substituted. Solder the green wire to Z4 pin 10.

Locate Z43 and Z44 as follows: Directly below Z24, in the third row of ICs is Z42; immediately to the left is Z43, and to the left of that is Z44. Now solder the blue wire to Z44 pin

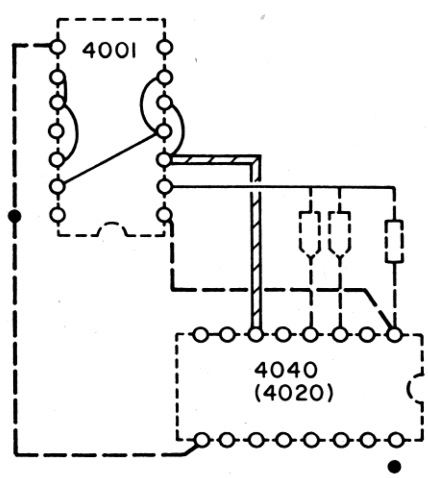


Fig. 3. The underside of the board shows the simple wiring required. Bend the resistor lead at a right angle to the resistor body and solder it directly to pin 13 of the 4001, then spot solder the diode leads to it. The connection from pin 12 of the 4001 to pin 11 of the 4020 (4040) should be insulated.

See List of Advertisers on page 306

10, or alternatively to Z24 pin 8. Solder the violet wire to Z24 pin 9, and finally solder the yellow wire to Z43 pin 9. Swing the keyboard back in place, and using double sided foam tape, attach the perf board to the foil side of the main PC board at a convenient location. If your computer has as many mods as mine, this last step may be the most difficult! (Incidentally, the foam tape sold by Radio Shack is a little thin, and I recommend using a double thickness.)

It probably took you longer to read this article than it did to build up and install the modification. Close up the computer, crank it up, and CLOAD away! I have been CLOADing wth the volume on my recorders cranked up to the stops with super results. The insensitivity to volume setting is great, and our friends at Tandy really did a good job with this modification.

Warning

The circuit I have described is the exact circuit installed by Radio Shack (except for the substitution of a CD4020 for the CD4040). Whether you build and install it, or have Radio Shack do if for you, any tapes CSAVEd with any of the CPU clock speedup circuits installed and with the CPU operating at the higher speed become impossible to load. Here's why: When the Z80 is being driven by the faster 2.66 MHz clock, the software timing of the cassette input routines sets up an expected clock pulse interval of 1.33 milliseconds insead of 2.0 milliseconds. Therefore, a data pulse would be expected at 0.67 msec, but the CLOAD fix locks out CASSIN* for 0.72 msec. The software literally never sees a data pulse! Why doesn't the clock also speed up that 887 KHz we used to feed the counter in our modification? Because that frequency is part of the video timing (it's called CHAIN in the *Technical Reference Handbook*), and fortunately, every inventor of a clock speedup circuit knows you don't fool around with video timing. So, do your CLOADing at normal clock speed, and execute your programs at the faster speed after they are safely in memory.

Alternatively, a modification can be made to the clock speedup circuit described in *80 Microcomputing* (Feb., 1980, see "Faster, Faster" by Dennis Kitsz). I have not yet built the speedup circuit, and so haven't tested it, but it should work. Mr. Kitsz uses two sections of a 74LS367 which he calls ZSPEED to gate either the fast or regular clock to the Z80. Break the connection of the diode at pin 13 of the CD4020 on the CLOAD modification and run a lead from pin 13 to pin 4 of ZSPEED. Run a second lead from pin 5 of ZSPEED to the cathode end of the diode you just disconnected. Pins 4 and 5 of the 74LS367 are an unused buffer whose tristate output state follows the normal speed clock control line.

With the normal speed clock active, the path from pin 13 of the CD4020 and the diode is made, and the CLOAD fix operates as previously described. However, when the fast clock is selected, the pin 5 output of the 74LS367 is tri-stated. Therefore, the Reset input of the NOR gate flip-flop will be triggered when pin 14 of the CD4020 goes high, after only 512 clock pulses. This works out to a time delay of 0.58 milliseconds, and although a little close, should still allow a pulse on CASSIN* every 0.67 milliseconds to go through the system.

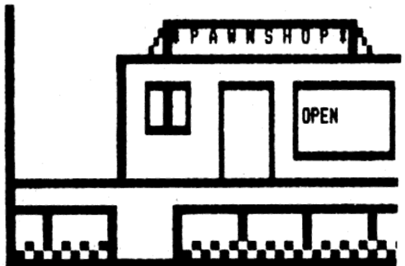
Most of us who have opened our computers have done so more than once, and have made several modifications. Therefore, I can't guarantee this compatibility of this modification with your system, or predict what impact it may have in connection with other mods you may have made. You must assume responsibility for any mods you make, including responsibility for their compatibility in combination in your TRS-80. But it sure is nice to know about some of the fixes that come out of Fort Worth, and I hope all who find out about one will take the trouble to share the information. Anyone know what the fix to the computer and expansion interface which corrects disk rebooting looks like?■

INTERPRO
presents

TRS-80* ILLUSTRATED ADVENTURE

The Atlantian Odyssey with Graphics

```
OBVIOUS EXITS-
NORTH
```




```
I AM ON A DOCK IN HAWII. I CAN RECOGNIZE :
PAWNSHOP. SAILBOAT.
I HAVE :THE SCUBA GEAR, WHICH I AM WEARING.
* CRYSYAL PYRAMID *. * MEDALLION *.
KNAPSACK CONTAINING: FLASHLIGHT. SPEARGUN.
WHAT DO I DO NOW?
```

MACHINE LANGUAGE

32 Graphic Locations · 150 Word Vocabulary
Exciting Adventure

MODEL I 48K DISK $29.95
WITHOUT GRAPHICS
16K CASSETTE... $14.95

DEALERS INVITED

*TM TANDY CORP.

TRS-80* 16 K-up UTILITY
ULTRA-MON
The Intelligent Monitor

The most powerful tool available to the novice or professional machine language programmer.

- ROM/DOS Independent
- Lineprinting disassembler
- Self relocating
- Single step through RAM or ROM with each instruction individually disassembled.

INTERPRETIVE EXECUTION allows you to execute your program or the ROM with Ultra-mon in complete control.

- DEBUG* type display with P.C. register disassembled.
- Breakpoints can be set in RAM or ROM.

.......AND MORE

* EXCELLENT REVIEWS IN APRIL 80 *
* MICROCOMPUTING AND APRIL *
* SOFTSIDE *

MODEL I, 16K CASSETTE
(Disk Loadable) $24.95

24 PAGE DOCUMENTATION WITH STEP-BY-STEP INSTRUCTIONS AND SIMPLE DEMONSTRATIONS

— *DEALERS INVITED* —

*TM TANDY CORP.

TO ORDER CALL or WRITE
INTERPRO
P.O. BOX 4211 ✓485
MANCHESTER, NH 03108
M-C **(603) 669-0477** VISA

TRS-80 (TM) MODEL III AND MODEL I SOFTWARE PACKAGES.

Can 30 user proven high-quality TRS-80 programs cost a mere $24.95?

We have a lot to gain by almost losing our shirts. Because we're a young, growing software company. We're BASICS AND BEYOND. And in order to build, we need your business.

MICROCOSM I. Thirty programs with everything from our incredible "Atlantis" game that'll challenge your intellect, to "Flowering Houseplants" for valuable reference.

MICROCOSM II. Twenty programs that require skill, the desire for intense concentration as in "Decipher." And practical guides like "Stain Removal."

Both packages, for 16K Model I or Model III, on quality tape cassettes, with hardcover storage case and detailed handbook, just $24.95 each.

Our prices will never be lower. Send today.

BOX 10 • AMAWALK, N.Y. 10501 (914) 962-2355 ✓49

BASICS & BEYOND, inc.

No charge for postage or handling. N.Y. residents add 5% sales tax. TRS-80 is a trademark of Radio Shack Division of Tandy Corp.

A MAJOR BREAKTHRU FOR

TRS 80™*

Electronic Engineers
Electronic Hobbyists
Electronic Students
Ham Operators

A General DC-AC (steady state) Analysis of Any Circuit

Will Analyze and Compute:

- Node, Branch, Element, Voltages
- Node, Branch, Element, Currents
- Branch Power Dissipation
- Magnitude and phase values and complete frequency response with graphic display
- Modify any element in circuit for desired results

A complete operational manual supplied comparable to I.B.M.'s E.C.A.P.™ Program
*A.C. Analysis Program *$149.95*
*D.C. Analysis Program *$89.95*

To Order Write:
G & L Software Enterp.
2304 N. 1st. Street
Upland, CA 91786

VISA · master charge

Do Not Send Cash in the Mail

TRS 80* is a trademark of the Tandy Corp., E.C.A.P. is a trademark of International Business Machines, Inc. ✓39

✓See List of Advertisers on page 306

Fastload

FOR TRS-80* MODEL I USERS ONLY

16 Times
Normal Speed

*TRS-80 is a trademark of Tandy Corp.

- **High speed load TRS-80* Level II cassettes**
- **Input 15K byte Level II program in 15 seconds**
- **Search BASIC or SYSTEM programs by name**

Unlike other high speed tape input devices, FASTLOAD uses standard format cassettes. Therefore, there is no need to re-record on other media. At 8000 baud, FASTLOAD is faster than disk for short programs. FASTLOAD reads tapes at the fast-forward speed of the CTR-41 cassette recorder. The recorder can also be used for CSAVE at the normal speed.

FASTLOAD connects to the 40 pin I/O or to the Expansion box. The control program does not use computer memory because it is in a built-in PROM. Other valuable features are keyboard debounce program, automatic key repeat routine and key-beep via cassette speaker. Price is $188.00 for FASTLOAD and $95.00 for the modified CTR-41 recorder.

✓422

Personal Micro Computers Inc.

475 Ellis Street, Mountain View, CA 94043 (415) 968-1604

A computerist's view of how it all began, seen in 20/20 hindsight.

Getting Involved

Robert A. Batty, WØKSG
2657 S. Bonanza Ave.
Tucson, AZ 85730

I've never liked Radio Shack because my few visits to their stores usually result in leaving empty-handed. I can never find the right parts!

I've been an amateur radio enthusiast for years, observing the Tandy Corporation's disregard of the ham's needs for parts and publications. I've avoided Tandy as best I can.

But, after reading two and one-half years of *Kilobaud Microcomputing* and other microcomputing magazines and watching the performance ratings of the TRS-80 climb, I bought the Radio Shack product.

Spending $500 in an enemy camp is not pleasant, but I did it when Tandy reduced their $600 TRS-80 by $100.

Justifying the Purchase

There were more important factors than the low price of the TRS-80 in my decision to buy. I started with a minimum configuration. Later I could upgrade to a system with disk drives, printer, and more memory, and even used other manufacturer's equipment.

Second, the TRS-80 qualifies as good word processing system. I felt uneasy about spending in excess of $2000 for a text editing, non-dot-matrix printer subsystem for the TRS-80. Instead, the Anderson-Jacobson, as recommended by Allan Domuret (*Kilobaud Microcomputing*, June, 1977) was in the $1000 range, I found.

Third, Tandy had published the manual, *TRS-80 Microcomputing Technical Reference Handbook*, complete with schematics. It convinced me that do-it-yourself maintenance of their product was possible.

Installation and Checkout

Unpacking and cabling the cassette recorder, video display, and keyboard was uneventful. I was impressed with the quality of the shipping package. (The octopus of exposed cables upset me, however—there had to be a better way.) The checkout procedures verified that I had indeed purchased a working system. The accompanying games, Blackjack and Backgammon, indicated further that my TRS-80 produced the desired results.

Dr. Lien's *Basic Computer Language*, which is included in the purchase price of the TRS-80, Level I system, is a fun course in learning Level I BASIC programming. If supplemented with another Lien text, *The Basic Handbook*, you will be able to adapt most published programs to fit in the 4K of memory in Level I.

Outgrowing the System

I soon found out (shortly before the 90-day warranty expired) that the Level I language was inadequate. An increase in memory was essential. I decided I could install the extra memory expansion modules in the keyboard unit myself. Consequently, Radio Shack's installation of the Level II came next. (Radio Shack can determine if an owner has tampered with a computer by covering an access screw with glyptol, a sealing wax that is destroyed by removing the screw. Destroying the glyptol will not only void the warranty, but there are rumors that the repair centers will charge a premium for servicing tampered-with keyboards.)

The upgrade to Level II at the repair center took less than a week. I checked it functionally by stepping through most of the test programs in *The Basic Handbook*. Success again, I thought, but troublesome times were ahead.

Disappointment After Disappointment

The Level II change included a manual entitled *Level II Basic Reference Manual*, disappointment number one upon graduating to Level II. Poorly written, containing none of the educational assistance taken for granted in the Level I manual, an inadequate table of contents and no index, I had the distinct feeling of being ripped off.

Fortunately, "TRS-80 Level II Reference Manual Index" by Sherman P. Wantz appeared in the February 1979 *Kilobaud Microcomputing*. This article allowed me to make a real reference manual out of Radio Shack's, even providing a page renumbering scheme.

Another shortcoming was soon to come—keybounce. This problem had been widely publicized in the personal computer magazines, but I hadn't experienced it until Level II was installed. This is possibly because I diligently kept the keyboard covered when not in use. Keybounce is the double entry of a character on a single keystroke. This problem is fixed by periodically removing the keypads of the faulty characters and cleaning the contacts. I found that Radio Shack's program, KBFIX, which is supplied with the Level II change, satisfactorily overcame the problem. The short

KBFIX program must be loaded each time the system is powered up.

My biggest disappointment, still unresolved, is the cassette loading problem. The Level II change increased the data rate during program loading and saving. This caused the volume control on the recorder to become sensitive. I must praise Radio Shack, who tried to overcome the problem by returning my equipment more than once to the repair center, resulting in some improvement. My program loading abort rate remains at about 10 percent however, which I consider too high. I have attained this rate only by using Peripheral People's "Data Dubber", constantly cleaning the tape path of the recorder, and using Memorex MRX3 oxide tapes, which have produced better results than all the other brands of tapes I tried. Fortunately, after saving programs, the CLOAD? command ensures that programs saved were actually recorded without error.

A minor concern was the failing recorder motor start relay (located in the keyboard). The relay contacts would weld while loading a program from tape. Instead of stopping when READY appeared on the screen, the recorder would proceed to the end of the tape. I found this could be minimized by hitting Enter before activating the recorder on a load or save command. The recorder play switch then takes the arc during the motor start operation.

Convinced now that my Level II required no further warranty maintenance, I ordered the 16K memory modules and jumpers. Installation in the keyboard was straightforward. Care must be taken, especially when flexing the cable joining the two boards. I was advised to use caution when handling the modules as they are extremely sensitive to static electricity.

After re-assembling the keyboard and re-installing the system cables, I was pleased to note the correct response to the PRINT MEM command. More rewarding, I was finally able to key in and run all those programs labelled Level II, 16K.

For the first time, I was thoroughly enjoying my TRS-80, doing all the things my Level I, 4K system couldn't support. Most of the programs published in the microcomputer magazines ran. Converting other systems' languages to the Microsoft BASIC of the TRS-80 was challenging, and especially rewarding when they ran without bugs.

But, in modifying and debugging programs, I found it extremely difficult to follow a program listing on the video screen, scrolling back and forth as the GOSUB and RETURN instructions demanded. Another problem. The only way to overcome this shortcoming was to attach a printer to the system. Inasmuch as word processing was ultimately a requirement for my system, I considered making the purchase.

A Selectric word processing printer such as the Anderson-Jacobson mentioned earlier was out of the question due to my lack of funds. A local Radio Shack flier arrived in the mail, offering the TRS-80 Quick Printer at quite a discount off their $499 catalog price.

This was the time to see if a discount Radio Shack would consider a 10 percent discount off the flier price. I phoned and they agreed. As shown in Table 1, I became a printer owner at a fairly reasonable cost.

Hardware	
Level I 4K	$529.00
Level II modification	106.00
16K upgrade	90.00
Quickprinter, cables, paper	382.00
Data Dubber	50.00
Software	
Blank tapes	40.00
Microsoft TYPING TUTOR	17.00
Discovery Bay WIN21	29.00
Lien The Basic Handbook	16.00
Lien Learning Level II	17.00
Radio Shack TRS-80 Microcomputer Technical Reference Handbook	11.00

Table 1

Printer Arrives

The printer arrived in five days and again I must commend the outstanding packaging done by Tandy.

A roll of aluminum-finish paper came with the printer (I hadn't expected this and had ordered an additional three). After installing the cable from the printer to my keyboard, I was ready to go.

Three print sizes were available and 150 lines per minute was the output speed. I was pleased and proceeded to print some of the programs I wanted to modify. My only gripe now was the uppercase/lowercase confusion factor—to print lowercase, the shift key must be depressed when entering data. This deficiency is not the fault of the printer, however. Using the aluminum paper is also not as disadvantageous as I thought; a felt-tipped pen does an adequate job making corrections on it.

I bought Peripheral People's Data Dubber not only to improve the tape loading problem, but to copy the machine language tapes I needed for backup. Not included in the expenditure table are the magazine subscriptions or necessities like tape head cleaner and cassette storage cases.

Would I Do It Again?

Absolutely. And in the same sequence: Level I, 4K, Level II, the 16K modification, and finally the printer.

Based on my pleasant experience with the printer, I would recommend mail-ordering the entire system. Warranty maintenance presents no problem. The only requirement is a sales receipt showing date of purchase.

Why not Level II initially? Three answers: (1) The Level I instruction manual is a rewarding way to get started. (2) Level II is more appreciated when it follows Level I. (3) Good programming habits are formed when memory is in short supply (4K).

Additionally, how would one know about the program loading deficiency if one hadn't started with Level I, where it doesn't exist? ■

✓See List of Advertisers on page 306

Yes, You Can Increase Your Programming Productivity

SNAPP II EXTENDED BASIC

SNAPP II EXTENDED BASIC A family of enhancements to the Model II BASIC interpreter. Part of the package originated with the best of APPARAT, INC.'s thoughts in implementing NEWDOS BASIC. The system is written entirely in machine language for SUPER FAST execution. The extensions are fully integrated into Model II BASIC, and require NO user Memory, and NO user disk space. The package is made up of the following six modules.

XBASIC—Six single key stroke commands to list the first, last, previous, next or current program line, or to edit the current line. Includes quick way to recover BASIC program following a NEW or system or accidental re-boot. Ten single character abbreviations for frequently used commands: AUTO, CLS, DELETE, EDIT, KILL, LIST, MERGE, NEW, LLIST, and SYSTEM.

XREF—A powerful cross-reference facility with output to display and/or printer. Trace a variable through the code. Determine easily if a variable is in use.

XDUMP—Permits the programmer to display and/or print the value of any or all program variables. Identifies the variable type for all variables. Each element of any array is listed separately.

XRENUM—An enhanced program line renumbering facility which allows specification of an upper limit of the block of lines to be renumbered, supports relocation of renumbered blocks of code, and supports duplication of blocks of code.

XFIND—A cross reference facility for key words and character strings, also includes global replacement of keywords.

XCOMPRESS—Compress your BASIC programs to an absolute minimum. Removes extraneous information; merge lines; even deletes statements which could not be executed. Typically saves 30-40% space even for programs without our REM statements! Also results in 7-10% improvement in execution speed.

ENTIRE PACKAGE FOR MODEL II $200
ENTIRE PACKAGE FOR MODEL III $125

AUTOMAP

Save time creating a formated screen with our Extended Basic Mapping Support system. The programmers task of communicating and displaying information with the user operator is completely automated with simple SEND and RECEIVE statement commands. Many different types of information can be sent and received from the operators with the use of a single command. Many hours of programming time are saved and cost efficiency is truly realized using Automap.

MODEL II $75
MODEL III $60

SUPERSNAPP X

You can sort fast using your present facilities, or you can do it faster with Racets' superb facilities, or you can use the fastest: SUPERSNAPP X. The heart of SUPERSNAPP X is a SUPER FAST in-memory sort routine that has been benchmarked against everything on the market and beats them all . . . hands down.

SUPERSNAPP X is the most important component of SNAPP X EXTENDED BUILTIN FUNCTIONS which is a much needed set of additions to the Model II BASIC interpreter that will put time saving power at your fingertips. Let's compare (using random data) SUPERSNAPP X and Racet's GSF SORT for speed:

SORT	SUPERSNAPP X	RACET GSF
10,000 integers	39 seconds	59 seconds
5,000 Singles	22 seconds	34 seconds
2,000 Strings	10 seconds	15 seconds

SUPERSNAPP X is guaranteed to be the FASTEST in memory SORT on the market or your money back. With it you also get these EXTENDED BUILTIN FUNCTIONS: PEEK, PEEKW, POKE, POKEW, XDATS, XTIMS, ETIMS, FILES, AND THE SPECIAL SCMD (SNAPP___COMMAND). PLUS: open "E" Set SCROLL PROTECTION, ERASE all ARRAYS in one command. Specify size and Blink rate of CURSOR. LONG ERROR MESSAGES. Read from Video Screen Read.
PEEK complete strings from memory. POKE complete strings to memory, convert upper case to lower case and vice-versa, turn complete screen off and on at will, extract largest or smallest values from user supplied list of numbers.
We consider this next function as probably the most powerful addition ever made to Microsoft BASIC. **PRINT USING INTO A STRING**. The ability to arrange data into a string variable should perk your imagination.

Model II $100.00
Model III $75.00

SPOOLER Model I, Model II and Model III

Our workhorse! Unlike the one supplied with TRSDOS 2.0, ours requires no special knowledge or training on the part of the operator. Additionally, ours performs much better. On the Tandy SPOOLER, every time a disk is accessed, the printer stops dead! This package is available for Model I, in the TRSDOS/NEWDOS 80 versions, or for the Model II or Model III. Greatly enhances system performance when running typical business applications. Many applications have been benchmarked to run nearly TWICE AS FAST with the SPOOLER installed. Installs in minutes and no changes are required to your programs. The Model II version requires NO user memory. If purchasing more than one system, call for price quote. $100

SNAPP SCREEN

Another dependable product from Snapp, Inc. Using rigid plastic, Snapp, Inc. designed the ultimate green screen at the lowest price. The Snapp Screen installs in a Snapp with the use of pressure sensitive tape. Reduce user fatigue with the Snapp Screen. $10

DOUBLETAKE 3741

This is not a football play but the way to play ball fast in converting IBM 3741 and similar formated diskettes to Radio Shack formated disks or vice versa. Fast is the name of the game. $200

AUTOFILE

Another big time saver from Snapp, Inc. Autofile offers all programmers much needed time saving tools to use when working with direct files. Eliminate tedious commands such as LSET's, MKIS's, MKDS's, MKSS's, CVD's, CVI's and CVS's. Even the FIELD statement has been eliminated. You won't have to guess as to where the FIELDed variable is. The ASC and CHRs function references will be performed automatically.

MODEL II $50
MODEL III $40

MASTER/SLAVE

This software package was designed to support the transferring of files from one Model II to another, via direct connection or modem/phone line connection. ALL kinds of files, and baud rates up to 9600 are fully supported. Transfer files in either direction, even with the SLAVE Model II UNATTENDED! $150

ULTRA PPD

This the ultimate Proportional Printer Driver that does the job the others do not. Add to the Electric Pencil and your print will look like its copy has been typeset. No word processor should be without this enhancement. Now available for the DWII and the LPIV. $100

ITOII

A helping hand when converting BASIC programs from the Model I to the Model II. Automatically adjusts PRINT @, and PRINT USING to compensate for differences in the language. Advises you where adjustments are necessary for PEEK, POKE, etc. $25

XSCREEN

Supports the copying of the full video screen to the printer. Can be invoked by the operator with a keystroke, or from your program with a USR call. Requires NO user memory $25

XPRINT

Print neatly formated hard copy listings of BASIC programs from disk. Programs may be ASCII or compressed. Quick and easy group selection allows you to print many listing with one command. $35

HOSTII/TERMII

Allows remote control of a Model II from another Model II, or any ASCII terminal. Our Host system, unlike the one supplied with TRSDOS 2.0, supports accurate screen positioning on the Term station. Without this feature, formated displays appear on the screen looking like randomly placed garbage. Requires NO user memory! This system is designed to provide software support to our customer locations without ever leaving the office. Custom versions are now available for most nationally distributed terminals as a $25.00 option. Call for details. $50

PL/B
The most powerful logical components have been selected from the worlds most powerful language to give you the most unbeatable program development tool to improve your programming productivity! Snapp, Inc. has merged the most useful components of BASIC with the power of PL/1. An unbeatable structured programming facility resulted that can increase programmers productivity up to 50%. Unlike competitive products, which are slow rigid preprocessors' representing a primitive attempt to bring Structured Programming concepts to the Micro-Based user, PL/B is today's fourth generation answer to a fourth generation problem! PL/B can give you triple the power of the earlier preprocessors at triple the speed. PL/B is fully integrated into the BASIC interpreter. Two modes of operation are available. Transparent (hidden) or Compile and save mode. Increase your productivity and save time with PL/B. $200.00

BREAKOUT
With Breakout you can learn the internal workings of TRSDOS 2.0 using any standard disassembler. Find your way out of the maze and save programming time by eliminating the unneeded steps in TRSDOS 2.0. This is a must product for all advanced TRS80 programmers. $200

DOSFIX
A collection of patches to TRSDOS and BASIC to enhance their usability and function. Includes our well known BREAK7E patches and facilities to disable and verify detect which will increase average disk speed by 30%. Free with any Model II Software Package. Purchased separately. $10

3M SCOTCH DISKETTES
Double density certified 8" Floppies for the Model II. Better quality is not available at any price. Ten diskettes to a box.

Quantity (boxes)	Price Per/Box
1	$34
5	$33
10	$32

Authorized Distributor
Information Processing Products 3M

TERMS OF SALE
Shipments normally made within one day of receipt of your order. Customer normally pays shipping cost, except pre-paid orders including software, in which case shipping cost is absorbed by Snapp, Inc. C.O.D. orders will have standard C.O.D. fee added to shipping charge. Net 10 days will be granted to governmental agencies, educational institutions and well rated business firms. Please include purchase order number. Ohio residents add 5½% sales tax or exemption certificate.

OUR GUARANTEE:
If your diskette arrives damaged, we will replace it without charge. If you ever accidentally damage it, we will replace it for a $10 handling charge. For a period of one year, we will provide you with any enhancements or updates for a $10 handling charge. For a period of one year, if errors are discovered in the programs, they will be corrected without charge. In the event we cannot correct an error, you may return the program material for a refund.

TRS-80 and TRSDOS are trademarks of the Radio Shack division of Tandy Corporation. NEWDOS and NEWDOS/80 are trademarks of Apparat, Inc. RACET GSF is a trademark of Racet Computes.

SNAPPWARE

Call our toll free number: 1-800-543-4628
Ohio residents call collect: (513) 891-4496

SNAPP, INC.
3719 Mantell
Cincinnati, Ohio 45236

Well, glitch my DOS. . . a transient suppressor of permanent value!

Lightning Strikes Twice

Paul C. Fowler, Jr.
2103 Charlton Lane
Radford, VA 24141

Dennis Murray
Compu-Tech
1005 Chestnut Dr.
Christiansburg, VA 24073

One rainy day, a friend was working on his computer when lightning struck across the street. For a brief moment the computer was working on itself. The lightning only destroyed one inexpensive IC, but in his excitement to find out what happened, my friend shorted out several other ICs. The whole problem might have been avoided with a good power line transient suppressor.

Power line transients (often called glitches) can have a wide range of effects on your computer, from no effect at all to complete destruction of the system. Typical glitches cause the program you've been working on for three hours to get confused or totally and unrecoverably wiped out. A power line transient is a momentary excursion of the power line voltage amplitude large enough to inject into your system's logic. Transients have many sources besides lightning; the most common household sources of glitches is probably an induction motor like those found on the compressors of re-

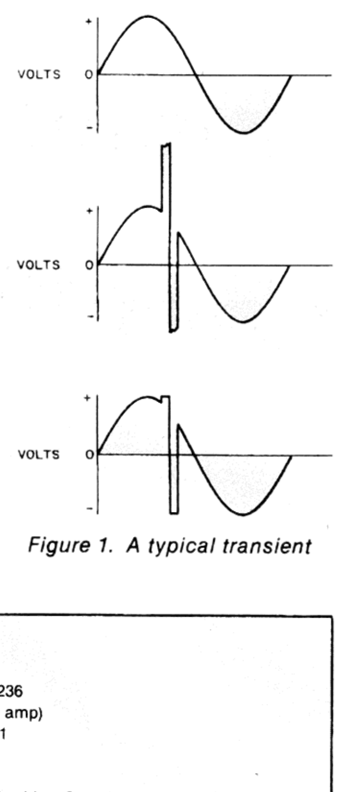


Figure 1. A typical transient

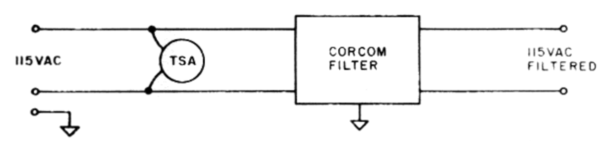


Figure 2. Modular Corcom filter in circuit

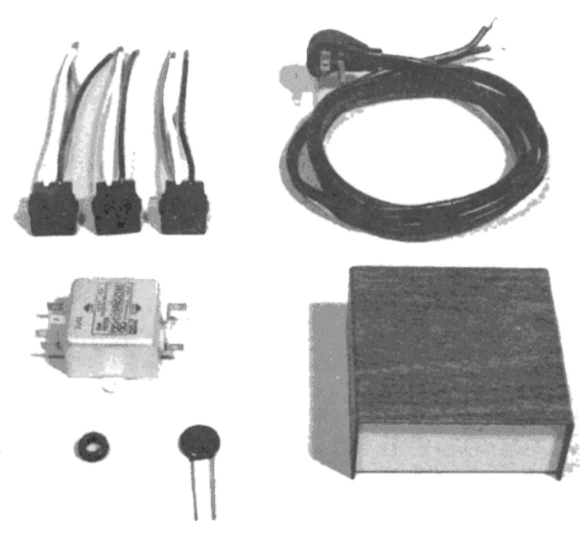


Photo 1. Pictured are all parts necessary to build the line filter. From left to right, top to bottom: chassis mount receptables (three-wire grounding type), line cord, RFI filter, metal enclosure, fuse holder, rubber grommet, ZNR transient surge suppressor.

Parts List

One AC line cord w/ground, Jameco 17236
One CORCOM 10k1 power line filter (10 amp)
One transient surge absorber, ZNR-K201
One metal housing, TEN-TEC TW-24
Three AC sockets, WABER #3015
One rubber Grommet or Strain Relief for Line Cord Radio Shack 278-1636 or 64-3025
Two 6-32 × 3/8 Flat Head Screws with Lock Washers and Nuts or two pop rivets
1 Ft. #18 Gauge Stranded Hook-up wire

frigerators, freezers, and air conditioners.

Fig. 1 shows a typical transient. We can avoid most and minimize all transient problems with a simple and inexpensive powerline filter. We can give our suppressor extra muscle by adding a transient surge absorber (TSA). The surge absorber acts like two back-to-back zener diodes and clips the transient as it attempts to rise above a predetermined level. If the filter doesn't get it, the transient surge absorber will.

Building the Circuit

The circuit is extremely simple because it employs a modular Corcom filter (Fig. 2). The most difficult part of this project is cutting square holes for the 115 volt receptacles (Fig. 3). I cut mine with a Mototool emery wheel. I then dressed the edges with a small flat file. Next, drill the 9/16" and 7/16" holes as shown for the fuse holder and powerline cord. Finally, drill two 5/32" holes for the power filter.

Mount all parts as shown in the photos, remembering to mount the line side of the Corcom filter toward the powercord insert hole.

After all parts are securely tightened, start your wiring as shown in Fig. 4. A note on soldering here may help the novice. First, clean and "tin" your soldering iron (do not use a soldering gun). Simply run the iron's tip over a moist sponge until the crusty stuff is gone and then apply a small amount of solder to the tip to make it shiny. On badly corroded irons, it may be necessary to first dress the tip with a

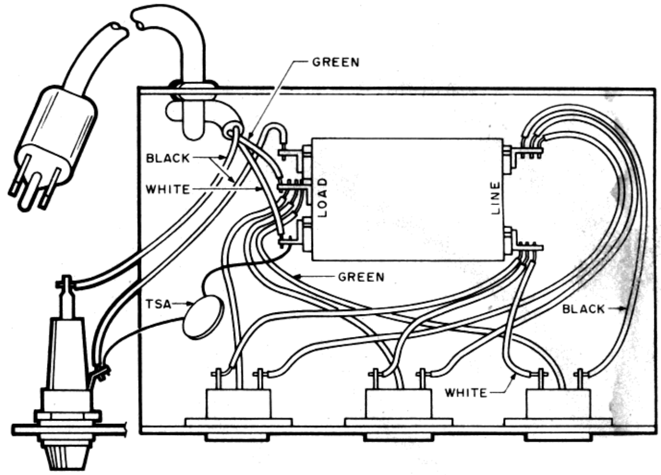


Figure 4. Wiring

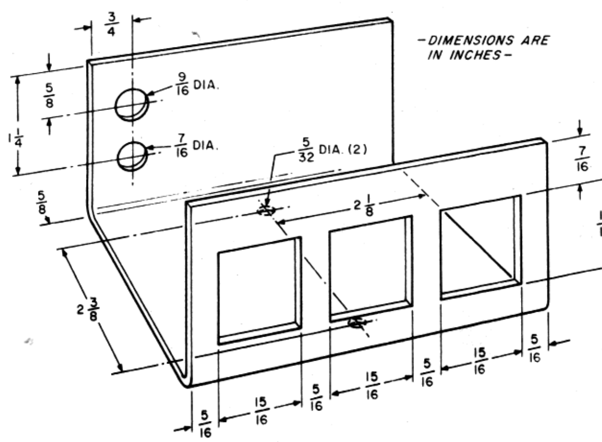


Figure 3. 115 volt receptacles

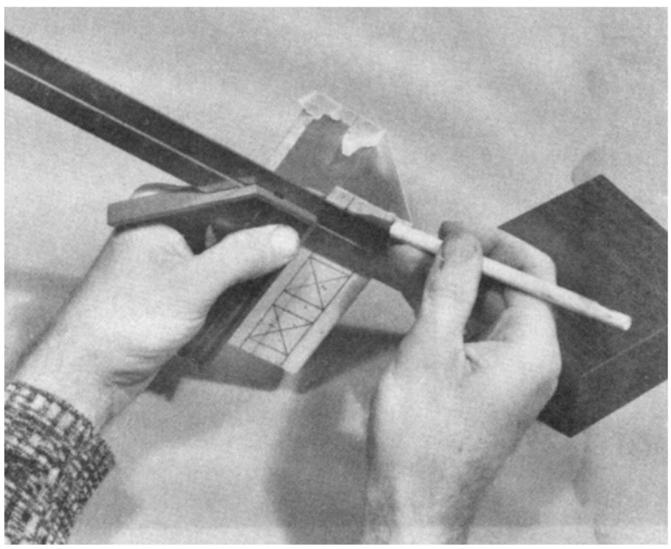


Photo 3. Using a ballpoint pen or soft lead pencil, mark locations and outline all holes to be cut.

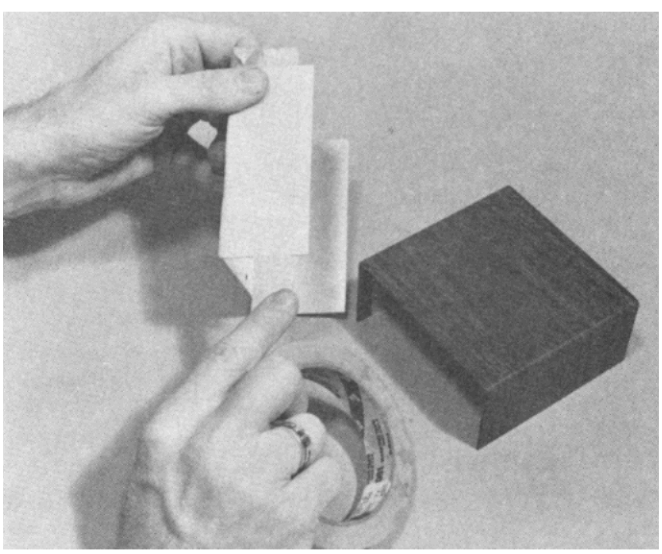


Photo 2. Apply masking tape to the front and back of the enclosure to protect its finish while marking and cutting all necessary holes.

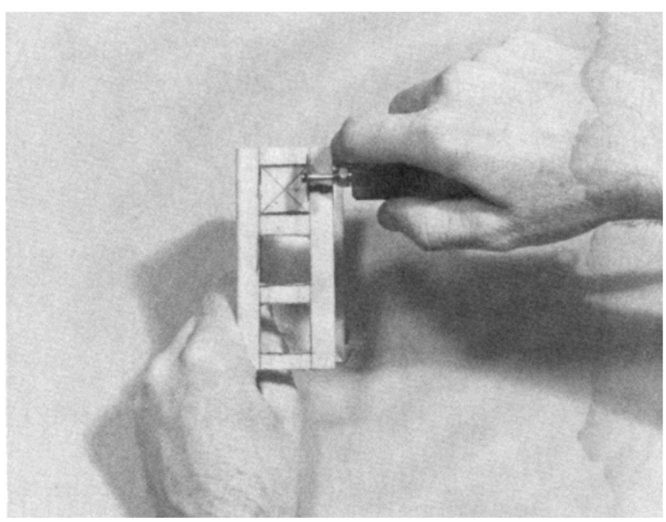


Photo 4. 15/16" square holes are required for the receptacles. These are most easily cut using a nibbler tool or dremel mototool with a cutting wheel.

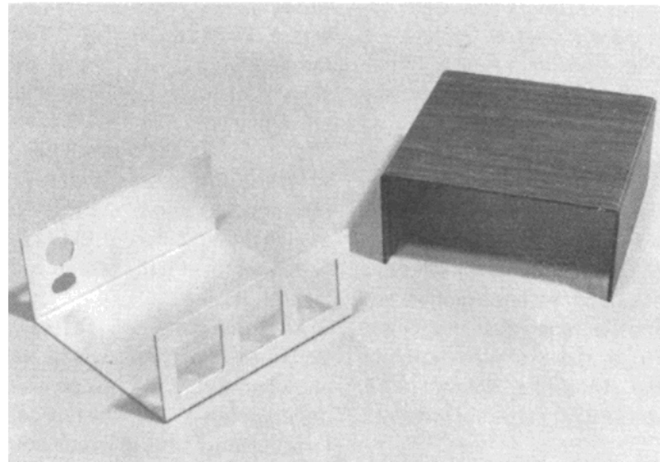


Photo 5. Remove remaining masking tape and wash reservoir with alcohol. Your completed enclosure should look like this.

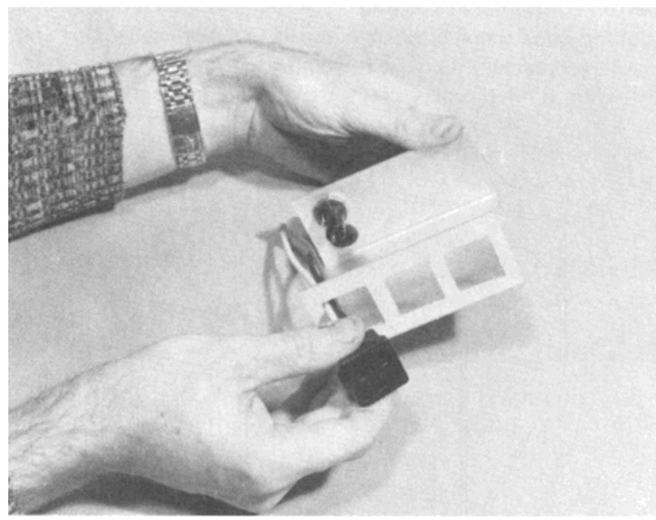


Photo 6. Install rubber grommet, line cord, fuse holder, and receptacles with grounding plug, facing downward to allow adequate clearance for most popular 110V plug styles.

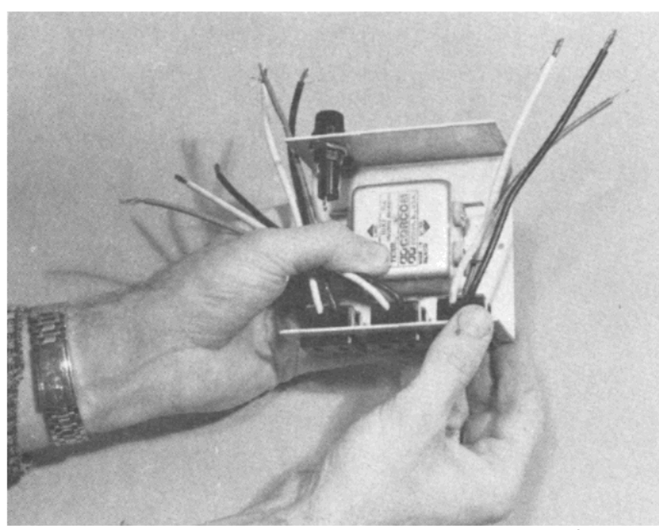


Photo 7. Position the RFI filter to provide adequate clearance between components. Mark location, drill holes, and mount filter. Due to their low profile, use of pop rivets is recommended to minimize the possibility of scratching the surface the filter is sitting on.

See List of Advertisers on page 306

file. Once your iron is ready, the secret to soldering is to heat the work piece and then let the work piece melt the solder.

Cut and strip all the #18 gauge hook-up wires to length and apply a bit of solder to each bare end. This is called tinning the wire, making it much easier to solder the connection. Be sure to put a knot in the power code, or apply some other form of strain relief, or you will eventually jerk some of your connections loose.

After making all connections, inspect your work, looking for cold solder joints or misconnections. When you are sure everything is correct, put the cover on the box and you're ready to plug the system into the wall.

A powerline filter will not guarantee protection from all transients. A direct hit of a lightning bolt, for example, will probably fry your refrigerator, washer, and dryer as well as your computer and the filter itself. However, given any distance, the filter will reduce problems by a substantial margin. It's a case of an ounce of prevention being worth a pound of cure.■

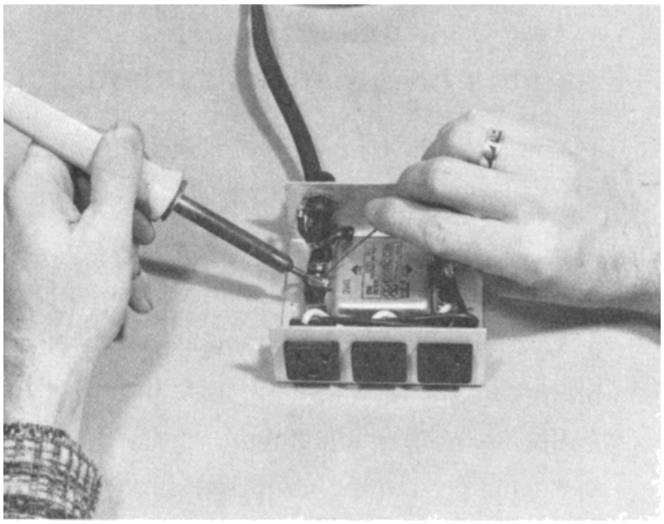


Photo 8. Cut receptacle wires to length and solder in place. Ensure proper polarity between the line cord and the receptacles. This can be most easily accomplished by plugging the line cord's plug into one of the receptacles and checking for continuity between one side of the RFI filter and one side of the line cord. Each side of the line cord should be soldered to the side of the RFI filter that shows continuity for that wire.

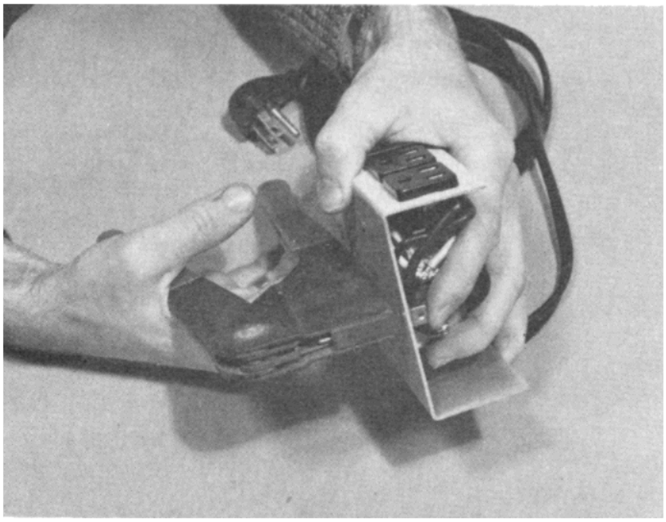


Photo 9. Fasten brackets to chassis. These will be used to hold the top cover in place. *(Photos continue on next page.)*

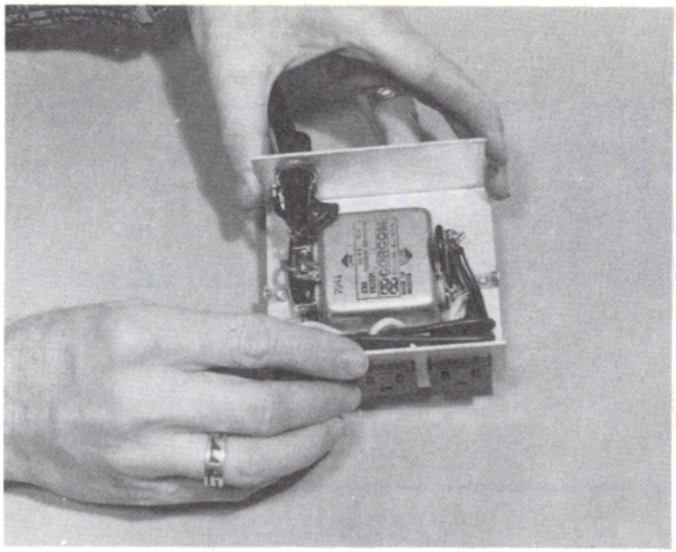


Photo 10. Your finished chassis should look like this.

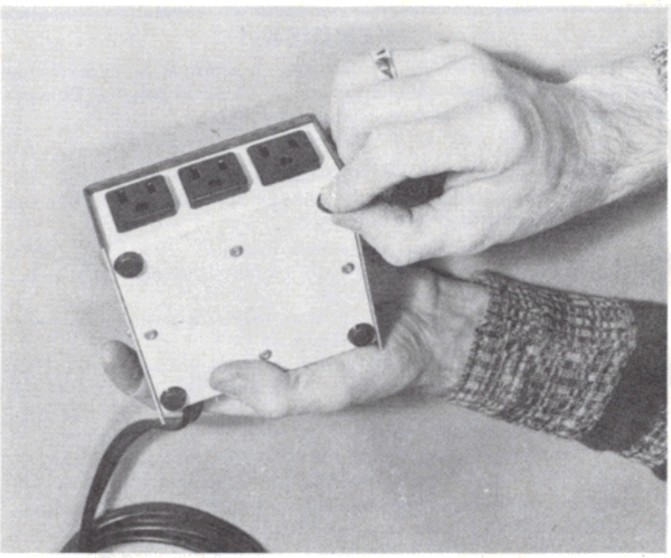


Photo 11. Fasten top cover to chassis and install rubber feet.

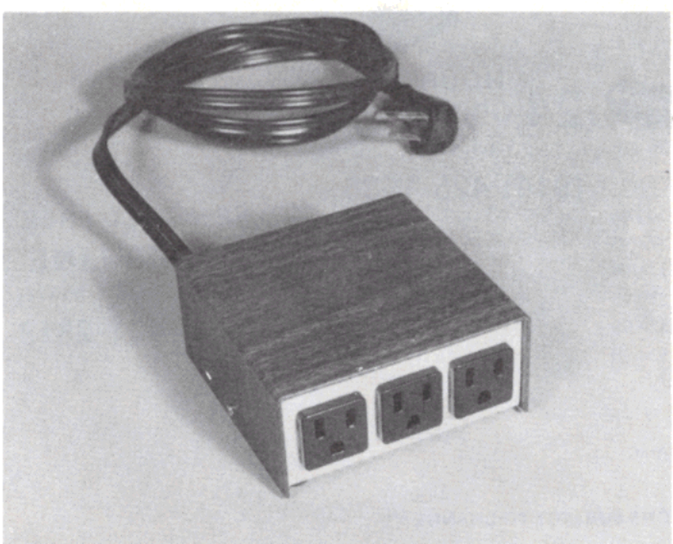


Photo 12. Install fuse in fuseholder and you're ready!

✓See List of Advertisers on page 306

✓See List of Advertisers on page 306

Replace that worn-out Ready message with something more regal.

Never Ready

Ron Balewski
412 E. Ridge Street
Nanticoke PA 18634

Have you ever wished you could change your computer's tired, worn-out ready message into something with a little more life? If so, this program is for you! It will enable your computer to display anything you want in place of its built-in ready. Unfortunately, there is one slight problem. The new message that you choose will not be displayed after a CLOAD. That will be the only time you'll have to suffer with the old message. I'll explain why a little later.

How It Works

The concept for this program is really quite simple. According to the TBUG manual, the basic warm-start re-entry point is 1A19H. After doing some disassembly from that point on, I found a DOS vector call at 1A1CH. This is a call to 41ACH, which contains nothing but a RETurn. Moving along in the disassembly, I finally found the call which displays the ready message.

With this in mind, let me now tell you exactly what I'm doing in the program. I plug a JumP to the starting address of my patch into DOS vector 41ACH so that control will pass to my patch (see Program Listing 1) during a restart. Lines 10 and 20 of the listing are two unknown calls that I encountered during the disassembly. They must be put here because I'm going to pass them up in ROM. Lines 30 and 40 display the new ready message. Line 50 pops the return address to oblivion in order to balance the stack (I will JumP back, not RETurn). Line 60 jumps back to ROM at the instruction immediately following the one which displays ready. Line 70 reserves memory to hold your new message. As you can see, I skip over the built-in prompt almost every time that it's displayed.

In order to make the initialization program a bit easier to use (and a lot easier to write) I wrote it in BASIC (see Program Listing 2). Lines 10 and 20 ask you for your new message. Lines 30 and 40 poke the patch into high memory. Lines 50 thru 90 poke your new message, letter by letter, into the storage area allocated after the program. Notice that 64 bytes are allocated, while you are limited to a 62 character message. That's because the last two bytes must be 0DH (carriage return, line feed), 00H (end of message indicator). Line 100 takes care of this. Lines 110 and 120 put the patch address into the DOS vector address. Line 130 clears the screen so you get the full impact of your new message the very first time it's displayed!

```
10  CALL   01F8H      ; ROM CALL
20  CALL   20F9H      ; ROM CALL
30  LD     HL,7FC0H   ; POINT TO NEW MESSAGE
40  CALL   28E1H      ; PRINT MESSAGE
50  POP    HL         ; BALANCE STACK
60  JP     1A2BH      ; JUMP BACK TO ROM
70  DEFS   64         ; SPACE FOR MESSAGE
```

Program Listing 1.

```
1 REM ***** N E W R E A D Y *****
2 REM BY RON BALEWSKI
10 CLS:PRINT@276,"N E W R E A D Y":PRINT:PRINT"YOUR MES
    SAGE (UP TO 62 CHARACTERS)":INPUTA$
20 IFLEN(A$)>62THENGOTO10
30 FORK=32688TO32703:READD:POKEK,D:NEXTK
40 DATA205,248,1,205,249,32,33,192,127,205,167,40,225,1
    95,43,26
50 K=32704
60 FORJ=1TOLEN(A$)
70 POKEK,ASC(MID$(A$,J,1))
80 K=K+1
90 NEXTJ
100 POKEK,13:POKEK+1,0
110 K=16812
120 POKEK,195:POKEK+1,176:POKEK+2,127
130 CLS
140 END
```

Program Listing 2.

The Problem

As I said earlier, this patch unfortunately has no effect after a CLOAD. This is because CLOAD is the one function which does not re-enter BASIC at the usual place. Instead, the ready message is printed inside of the CLOAD subroutine and control is then passed to a later point in the restart procedure, totally bypassing this patch. Considering how infrequently CLOAD is used (one CLOAD and your program is in memory), I don't think it's worth the memory you'd use to patch the message into the CLOAD subroutine. But if you're curious and would like to try, let me briefly explain how I would attempt to do it.

The Fix

The end of CLOAD is a JumP to location 1AE8H. As it turns out, there is a DOS vector at 1AECH calling 41B5H. I think you could patch a routine here which would move the cursor up a line, print the new message, and return, thereby overwriting the ready. I haven't tried this, but it's a possibility. One method that I did try was to insert the BASIC line:

```
125 K = 16821:POKE K,195:
POKE K + 1,176: POKE K + 2,127
```

This line causes the above mentioned vector to jump to the new ready patch, thereby printing the new message and jumping to the beginning of the restart procedure. This method seems to work to an extent. I have the standard ready on one line with the new message on the line below it.

Yes, Your Majesty

In order to use the program, CLOAD and RUN and tell it what your new message will be. If you want a comma in the message string, be sure to enclose the entire message in double quotes when you type it in. This is because the program uses an input statement to accept your new message.

The program, as written, will sit at the very top of a 16K machine. It's not relocatable because of the message pointer. To move this program somewhere else you'll have to change the values of the pokes as well as the eighth and/or ninth numbers in the data statement to show the starting location of the new message. It shouldn't be too difficult. For example, if you wanted to use this program at the top of a 4K machine, you would change the FOR statement in line 30 to:

```
FOR K = 20400T020415
```

You would also change the 127 in line 40 to 79 and the 32704 in line 50 to 20416. Everything else is the same. Oh yes, be sure to set the memory size accordingly (32688 as the program now stands) or things won't function right.

I think you'll find the new ready patch a very amusing program. I've received some very puzzled looks when someone sees my computer displaying "Yes, Your Majesty." ■

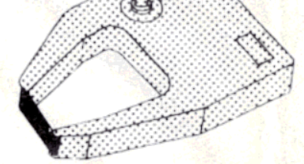
DISCOUNT PRINTER RIBBONS

40% OFF!! OR MORE!

Brand New, Top Quality, Exact Replacement Ribbons & Cartridges. These Ribbons Produce **Super Jet** Black Impressions and Ultra Reliable Print Life. They Are Delivered to Your Door Promptly for Much Less Than Most Retail Stores

★SPECIAL! BUY 10 and GET ONE FREE!

YOUR PRINTER	PACK SIZE	RETAIL LIST**	YOUR WHOLESALE PRICE	SIZE	COMMENTS	CAT. ORDER #
RADIO SHACK LPIII, LPV	one/pk	13.95/cart	8.95/Reload rib. only *(8.95 ea)*	.500'' x 45'	Nylon Incl Instr	R-T3
RADIO SHACK LPII, LPIV	3/pk	18.95/3 pk	11.95/3 pk *(3.98 ea)*	.563'' x 45'	Nylon Jet Blk	C-700
CENTRONICS. MODS 700 to 703	3/pk	18.95/3 pk	11.95/3 pk *(3.98 ea)*	.563'' x 45'	Nylon Jet Blk	C-700
CENTRONICS 100, 101A, 102, 103, 300, 301, 306, 308, 330, 358, 398, 500, 501, 503, 508, 588, 620, 820.	3/pk	26.33/3 pk	17.55/3 pk *(5.85 ea)*	1'' x 108'	Nylon Jet Blk 5 mil High Speed	C-100
CENTRONICS 704-705	1/pk	16.95 ea	13.95/Giant Cart *(13.95 ea)*	5/16'' x 210'	Giant Cart	C-7045
IBM -"SILVER DOLLAR" Sys. 34, Sys. 32 MDLA, Series IMDL4974, 5256, 3287, 3770, 3771-3774, 4974, 5100, 5103, 5110, 5228, 5256, 5320MDLA	5/pk	5.80 ea	14.90/5 pk *(2.98 ea)*	9/16'' x 30'	Nylon Jet Blk	R-300
IBM - HARMONICA ½''. SERIES I. MOD 4973/II, 3200, 3289, MOD 2.	3/pk	9.42 ea	20.85/3 pk *(6.95 ea)*	1/2'' x 108'	Nylon Jet Blk	C-350
TELETYPE MOD 33, 28, 35, 37, 38, 88.	10/pk	2.40 ea	13.90/10 pk *(1.39 ea)*	1/2'' x 36'	Nylon Jet Blk	R-450
DIABLO HYTYPE II (M/S BLK) HI YIELD. FITS 70 PRINTERS!	1/pk	9.31 ea	6.87 ea *(6.87 ea)*	5/16'' x "High Yield"	300,000 plus imp.	C-511
QUME (FITS 80 PRINTER MODS)	3/pk	18.00/3 pk	13.95/3 pk *(4.65 ea)*	1/4'' x 310'	Multistrike Film	C-525
WANG M/S. 5541W, WC, 5581, WD, 6581W, 2281W	1/pk	6.85 ea	5.95 ea *(5.95 ea)*	5/16'' x 393'	Multistrike Film	C-550
DEC ½ x 40YD.	3/pk	17.77/3 pk	12.95/3 pk *(4.32 ea)*	1/2'' x 120'	Double Spools	R-600
DEC ½ x 60YD.	3/pk	20.12/3 pk	14.25/3 pk *(4.75 ea)*	1/2'' x 180'	Double Spools	R-644
NEC SPINWRITER	4/pk	23.40/3 cart	23.60/4 pk rb. reload *(5.90 ea)*	1/2'' x 51'	Nylon/Ex Lng Life	R-400



TERMS:

MINIMUM PURCHASE - **$20**

PAYMENT BY: C.O.D.(UPS), CHECK, MASTER CARD, OR VISA CHARGE CARD.

VOLUME DISCOUNTS:

20 - 50 PACKS **10%**

51 - 100 PACKS **15%**

*UNDER $20, ADD $5 HANDLING.

**APPROX. RETAIL. PRICE VARIES.

ANCIE LABORATORIES ✓461

9204 Baltimore Blvd
College Park, MD 20740

301-345-6000 (Wash. D.C. Local)
301-792-2060 (Balto. MD Local)
800-638-0987 (National)

NAME__________

ADDRESS__________

CITY__________ STATE______ ZIP______

QTY | CAT.# | AMT.

TOTAL ______

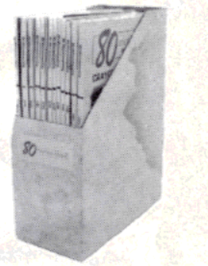
☐ Check Enclosed
☐ C.O.D.
☐ VISA
☐ MASTER CHARGE
ACCT. #
EXP. DATE
MIN. ORDER $20
PRICES SUBJECT TO CHANGE

ANCIE Laboratories
9204 Baltimore Blvd.
College Park, MD 20740

301-345-6000 (Wash. D.C. Local)
301-792-2060 (Balto. MD Local)
800-638-0987 (National)

While they last . . .

the complete year of 1980 80 Microcomputing (in its own shelf box . . . a $2.00 value—Free) for only $25.00.*



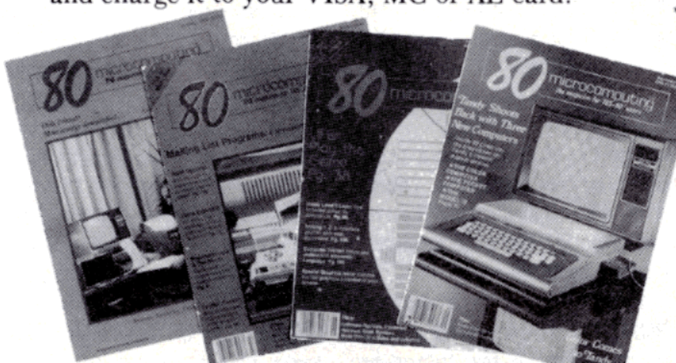
Did you miss the 2,256 pages of useful information published for your TRS-80** in *80 Microcomputing* last year? Now is the perfect time to catch up for only $25.00.*

In the first year of *80 Microcomputing* here were 335 articles on your TRS-80—2½ times more than any other computing magazine. Also in 1980 there were new product reviews, news and columns and best of all hundreds of dollars worth of useable programs.

So to unlock the key to your TRS-80 and double its value. . . send in today for the complete year of 1980. *80 Microcomputing* (in its own shelf box) for only $25.00.*

We have a limited number of these complete sets so send in the card today or call our toll free #800-258-5473 and charge it to your VISA, MC or AE card.

**plus shipping and handling ($3.00) **TRS-80 is a Trademark of Tandy*



Yes, I want a year of *80 Microcomputing* for $25.00 plus $3.00 for shipping and handling. Plus I'll receive a shelf box to store them in.

☐ Check enclosed for $______

Name __________

Address __________

City __________ State______ Zip______

Please allow 4 to 6 weeks for delivery

80 Microcomputing • 80 Pine Street • Peterborough, N.H. 03458

Toll free ordering number 800-258-5473

80BI

✓See List of Advertisers on page 306

More power to your TRS-80.

Enhance Your Level II BASIC

Gil Spencer (VK2JK)
Box 300
Spit Junction
NSW 2088, Australia

Every Level II owner worth his salt has by now bought a monitor program like RSM2, Monbug or T-Bug. Any of these fine programs make it tempting to dig around in the Level II ROM to see how it works.

The first thing that you acquire from your searching about is a profound respect for the ROM authors. It's obvious that every byte does plenty of work.

Discoveries

Sooner or later, while cruising through ROM land in your monitor, a lot a familiar road signs will flash past. Pull over to the side of the road near &1660 and have a closer look. Well, for goodness sake! There are all those familiar BASIC words like Run, List and Stop. It must be some sort of look-up table.

And look! There are some other English words hiding in the jumble. There's Merge... and Put...and Open. Wonder what they're there for? Get back into the monitor and let's head for RAM.

As we move through reserved RAM, we pass sign posts all pointing to the same place. There must be 20 or 30 in all, and each one says JP &012D. They're all bunched together near &4170. It must be a popular spot: Let's jump to 8012D and see why it's such an attraction.

Just a hick town, this &012D. A jump there results in this message to the screen—L3 Error. The Level II Manual says this about L3 Error: "Disk Basic only: An attempt was made to use a statement, function or command which is available only when the TRS-80 Mini Disk is connected via the Expansion Interface."

Gosh, I guess you need about $1000 to visit all those exotic L3 destinations. Well, I probably didn't want to go there anyway!

One day a stranger appeared on the shelves at the local Tandy store. No black jacket this! It's a manual with a rich brown jacket and it's called *TRSDOS & Disk Basic Reference Manual* (Catalog No. 26-2104).

A brief reading reveals that all those strange English words in the Level II ROM near &1660 are Disk Basic words available only to those rich enough to afford all that extra gear.

Digging for Answers

The Level II ROM is a pretty complex can of worms. There's a lot to learn about how it all fits together. If I live long enough it may all become clear.

With the use of a monitor, ROM can be unraveled a small bit at a time. A few things are now obvious:

- The Level II BASIC interpreter can recognize Disk BASIC words.
- While the interpreter recognizes the words it cannot implement them. It's just not quite smart enough. You might say it knows the words...but not the music!
- Most Disk BASIC words result in a jump to &012D via the links near &4170. Thus, an L3 Error results.

A bit more effort and most the mystery is solved. Table 1 shows what goes where. Each of these links is a jump: there are 28 of them. Twenty-four are accessed by a Disk BASIC word. The other four are reached from a particular point in ROM.

We've now established that RAM, from &4152 to &41A5, is reserved for Disk BASIC jump links.

Let's look a little further. There's also a pattern in RAM from &41A6 to &41E4. Dividing this area into three-byte lumps, we notice that each lump starts with a &C9 (hex for the op code RETurn).

This type of three-byte lump looks like it's a link for a Call from somewhere in ROM. The

```
10 DEF FNRAD(R) = (3.14159/180)*R
.
.
.
100 X = COS (FNRAD(W))
```

Degree to Radian Conversion Functions

Fig. 1

(Note: In the text, numbers prefixed with & are in hexadecimal notation)

Call has simply been short-circuited with a RETurn. The three-byte lump could be replaced with JP nn (which is three bytes long). In that way we could link new code into a particular spot in ROM. As long as the new code ends with a RETurn, we'll go right back to the calling routine in ROM.

An investigation of the Call links ultimately gives us Table 2.

Disk BASIC Word Links

RAM ADDRESS	LVL II CONTENTS	LINK FOR	JUMP FROM
&4152	JP &012D	CVI	T*
&4155	"	FN	&2524
&4158	"	CVS	T
&415B	"	DEF	T
&415E	"	CVD	T
&4161	"	EOF	T
&4164	"	LOC	T
&4167	"	LOF	T
&416A	"	MKI$	T
&416D	"	MKS$	T
&4170	"	MKD$	T
&4173	"	CMD	T
&4176	"	TIME$	&2510
&4179	"	OPEN	T
&417C	"	FIELD	T
&417F	"	GET	T
&4182	"	PUT	T
&4185	"	CLOSE	T
&4188	"	LOAD	T
&418B	"	MERGE	T
&418E	"	NAME	T
&4191	"	KILL	T
&4194	"	&	&24C8
&4197	"	LSET	T
&419A	"	RSET	T
&419D	"	INSTR	&2506
&41A0	"	SAVE	T
&41A3	"	LINE	T

JP &021D implements 'L3 ERROR'

*T = JUMP from BASIC word look-up table. Can be reached from Keyboard or from Program.

Table 1

Other Disk BASIC Links

RAM ADDRESS	LVL II CONTENTS	LINK WITH	CALLED FROM
&41A6	RET nn	ERROR	&19EC
&41A9	"	USR	&27FE
&41AC	"	READY	&1A1C
&41AF	"	KB INPUT	&0368
&41B2	"	BASIC INPUT	&1AA1
&41B5	"	" "	&1AEC
&41B8	"	" "	&1AF2
&41BB	"	NEW/END	&1B8C and &1DB0
&41BE	"	VDU SELECT	&2174
&41C1	"	OUTPUT TO DEVICE	&032C
&41C4	"	KB SCAN	&0358
&41C7	"	RUN	&1EA6
&41CA	"	PRINT	&206F
&41CD	"	—	&20C6
&41D0	"	CRLF	&2103
&41D3	"	TAB	&2108 and &2141
&41D6	"	COMMAND "INPUT"	&219E
&41D9	"	—	&2AEC
&41DC	"	READ	&222D
&14DF	"	READ/LIST	&2278 and &2B44
&41E2	RET nn	SYSTEM	&02B2

RET nn = 'Dead End' with room for JUMP or CALL instruction

Table 2

✓See List of Advertisers on page 306

Of the 21 links, 18 are each called from only one place in ROM. Three are called from two places. The column in Table 2 headed Link with is a bit suspect. That's because I haven't yet positively figured out what ROM is up to when the calls are given. When you work them out for certain, let me know!

So What?

A fair question. In my case, a study of these link tables started me on the road to a machine language program called Level II and 1/2. The source and object codes (called Twohaf) are reproduced in Program Listings 1 and 2. This short program (not much over 1K bytes) gives access to selected Disk BASIC commands. It also offers the ability to quickly renumber BASIC lines as you go along in a quite sophisticated way.

Finally, Twohaf provides a facility for appending one BASIC program to the end of another—concatenating BASIC programs.

The source of Twohaf was written using Tandy's EDTASM (version 1.2) on a 16K machine. Because of RAM limitations it had to be written in two separate parts. When the object code from both parts is put together, we have Level II and 1/2.

Like most of us, I have looked at many enhancement programs for the Level II TRS-80. Many are very elaborate and represent a lot of work. The trouble is that they take up too much room in RAM while providing a lot of functions that I personally never use. Furthermore, each enhancement is equipped with a multi-page manual that keeps getting misplaced.

It finally came time to make a list of things that would be useful enhancements for Level II for my applications.

The list was short:

1. Enable DEF FN (DEFine FuNction)
2. Enable DEFUSR (DEFine USeR)
3. Enable hexadecimal notation.
4. Enable BASIC line renumbering.
5. Enable appending one BASIC program to another.

All these functions are around as parts of various enhancements on the market. No one utility that I have found offers them all.

DEF FN (DEFine FuNction)

This is perhaps the most useful Disk BASIC command for my kind of programming. It permits the programmer to define a numeric or string function. After definition, the function can be called one or many times in a program. The Disk BASIC manual (referenced earlier) explains DEF FN.

A sample of the use of DEF FN is shown in Fig. 1. Note that the variable used to define a function need not match the variable used when the function is called.

Remember that a function definition is not limited to one variable. It's also useful to realize that one function can be used within the definition of an-

Program Listing 1

```
              ; < TWOHAF >     PART 1                    V 1.2    800401
              ; < TWOHAF >     PART 1                    V 1.2    800401
              ; A TRS-80 LEVEL II BASIC ENHANCEMENT PROGRAM
              ;
              ;                GIL SPENCER (VK2JK)
              ;                BOX 300 / SPIT JUNCTION, NSW  2088
              ;                AUSTRALIA
              ;
              ; THIS PROGRAM ENABLES THESE NEW BASIC FUNCTIONS -
              ;       1.  DEF FN       -DEFINE FUNCTION
              ;       2.  DEFUSR       -DEFINE USER
              ;       3.  &            -PREFIX FOR HEX NOTATION
              ;       4.  LINE         -LINE RENUMBERING COMMAND
              ;       5.  MERGE        -SET CONCATENATE PARTITION
              ;       6.  RSET         -RESET "         PARTITIONS
              ;
              ; FOR DETAILS, SEE TANDY TRSDOS & DISK BASIC MANUAL
              ;
              ; THIS PROGRAM IS LOADED BEGINNING AT -
42E9                  ORG     42E9H    ; (THE USUAL START OF BASIC)
              ;
              ; NOTE - ACTIVE PART IS PROTECTED BY RELOCATING THE
              ;        USUAL START OF BASIC HIGHER IN MEMORY.
              ;        THE INITIALIZATION ROUTINE IS OVERWRITTEN
              ;        AFTER EXIT FROM TWOHAF.
              ;
42E9 3AAF40   AS5     LD      A,(40AFH)
42EC 3D               DEC     A
42ED 3D               DEC     A
42EE 3D               DEC     A
42EF B7               OR      A
42F0 37               SCF
42F1 CA5643           JP      Z,AS6
42F4 4E               LD      C,(HL)
42F5 23               INC     HL
42F6 46               LD      B,(HL)
42F7 C5               PUSH    BC
42F8 FA1343           JP      M,AS7
42FB 23               INC     HL
42FC 4E               LD      C,(HL)
42FD 23               INC     HL
42FE 46               LD      B,(HL)
42FF C5               PUSH    BC
4300 E21343           JP      PO,AS7
4303 23               INC     HL
4304 DA0A43           JP      C,AS8
4307 211D41           LD      HL,411DH
430A 4E       AS8     LD      C,(HL)
430B 23               INC     HL
430C 46               LD      B,(HL)
430D 23               INC     HL
430E C5               PUSH    BC
430F 4E               LD      C,(HL)
4310 23               INC     HL
4311 46               LD      B,(HL)
4312 C5               PUSH    BC
4313 DA6043   AS7     JP      C,AS9
4316 CF       AS1     RST     8
4317 BE               CP      (HL)
4318 3E80             LD      A,80H
431A 32DC40           LD      (40DCH),A
431D B6               OR      (HL)
431E 47               LD      B,A
431F C31226           JP      2612H
4322 CD1643   ASC     CALL    AS1
4325 3AAF40           LD      A,(40AFH)
4328 B7               OR      A
4329 F5               PUSH    AF
432A 22F340           LD      (40F3H),HL
432D EB               EX      DE,HL
432E 7E               LD      A,(HL)
432F 23               INC     HL
4330 66               LD      H,(HL)
4331 6F               LD      L,A
4332 B4               OR      H
4333 CA4E44           JP      Z,AS2
4336 7E               LD      A,(HL)
4337 FE28             CP      28H
4339 C28943           JP      NZ,AS3+1
433C D7               RST     16
433D 22D840           LD      (40D8H),HL
4340 1802             JR      AS4
```

Program continues

✓See List of Advertisers on page 306

```
4342 CF       00910 AS10  RST   8
4343 2C       00920       INC   L
4344 0E04     00930 AS4   LD    C,04
4346 CD6319   00940       CALL  1963H
4349 3E80     00950       LD    A,80H
434B 32DC40   00960       LD    (40DCH),A
434E CD0D26   00970       CALL  260DH
4351 EB       00980       EX    DE,HL
4352 37       00990       SCF
4353 C3E942   01000 AS11  JP    AS5
4356 D22324   01010 AS6   JP    NC,2423H
4359 D5       01020       PUSH  DE
435A EB       01030       EX    DE,HL
435B CD8828   01040       CALL  2888H
435E D1       01050       POP   DE
435F AF       01060       XOR   A
4360 E5       01070 AS9   PUSH  HL
4361 F5       01080       PUSH  AF
4362 EB       01090       EX    DE,HL
4363 7E       01100       LD    A,(HL)
4364 FE29     01110       CP    29H
4366 20DA     01120       JR    NZ,AS10
4368 2AF340   01130       LD    HL,(40F3H)
436B CF       01140       RST   8
436C 28E5     01150       JR    Z,AS11
436E 2AD840   01160       LD    HL,(40D8H)
4371 CD0D26   01170 AS13  CALL  260DH
4374 E3       01180       EX    (SP),HL
4375 CD2B1F   01190       CALL  1F2BH
4378 7E       01200       LD    A,(HL)
4379 FE29     01210       CP    29H
437B 2807     01220       JR    Z,AS12
437D CF       01230       RST   8
437E 2C       01240       INC   L
437F E3       01250       EX    (SP),HL
4380 CF       01260       RST   8
4381 2C       01270       INC   L
4382 18ED     01280       JR    AS13
4384 D7       01290 AS12  RST   16
4385 E3       01300       EX    (SP),HL
4386 CF       01310       RST   8
4387 29       01320       ADD   HL,HL
4388 3ED5     01330 AS3   LD    A,0D5H
438A CF       01340       RST   8
438B D5       01350       PUSH  DE
438C CD3723   01360       CALL  2337H
438F 2B       01370       DEC   HL
4390 D7       01380       RST   16
4391 C29719   01390       JP    NZ,1997H
4394 E7       01400       RST   32
4395 282F     01410       JR    Z,AT1
4397 D1       01420 AT6   POP   DE
4398 F1       01430 AT4   POP   AF
4399 283F     01440       JR    Z,AT2
439B 301C     01450       JR    NC,AT3
439D E1       01460       POP   HL
439E C1       01470       POP   BC
439F 70       01480       LD    (HL),B
43A0 2B       01490       DEC   HL
43A1 71       01500       LD    (HL),C
43A2 FA9843   01510       JP    M,AT4
43A5 2B       01520       DEC   HL
43A6 C1       01530       POP   BC
43A7 70       01540       LD    (HL),B
43A8 2B       01550       DEC   HL
43A9 71       01560       LD    (HL),C
43AA E29843   01570       JP    PO,AT4
43AD 2B       01580       DEC   HL
43AE C1       01590       POP   BC
43AF 70       01600       LD    (HL),B
43B0 2B       01610       DEC   HL
43B1 71       01620       LD    (HL),C
43B2 2B       01630       DEC   HL
43B3 C1       01640       POP   BC
43B4 70       01650       LD    (HL),B
43B5 2B       01660       DEC   HL
43B6 71       01670       LD    (HL),C
43B7 18DF     01680       JR    AT4
43B9 D5       01690 AT3   PUSH  DE
43BA F5       01700       PUSH  AF
43BB E7       01710       RST   32
43BC 11D340   01720       LD    DE,40D3H
43BF CC8828   01730       CALL  Z,2888H
43C2 F1       01740       POP   AF
43C3 C31A28   01750       JP    281AH
43C6 2AB340   01760 AT1   LD    HL,(40B3H)
43C9 EB       01770       EX    DE,HL
43CA 2A2141   01780       LD    HL,(4121H)
43CD DF       01790       RST   24
43CE 3805     01800       JR    C,AT5
43D0 CD4328   01810       CALL  2843H
43D3 18C2     01820       JR    AT6
43D5 D1       01830 AT5   POP   DE
43D6 21D340   01840       LD    HL,40D3H
43D9 E5       01850       PUSH  HL
43DA CDF529   01860 AT2   CALL  29F5H
43DD 7E       01870       LD    A,(HL)
43DE 22B340   01880       LD    (40B3H),HL
43E1 E1       01890       POP   HL
43E2 77       01900       LD    (HL),A
```

Program continues

```
43E3 23                   INC    HL
43E4 71                   LD     (HL),C
43E5 23                   INC    HL
43E6 70                   LD     (HL),B
43E7 18AF                 JR     AT4
43E9 FEC1          DEF    CP     0C1H
43EB 2852                 JR     Z,DE1
43ED CD1643               CALL   AS1
43F0 CD2828               CALL   2828H
43F3 EB                   EX     DE,HL
43F4 73                   LD     (HL),E
43F5 23                   INC    HL
43F6 72                   LD     (HL),D
43F7 EB                   EX     DE,HL
43F8 7E                   LD     A,(HL)
43F9 FE28                 CP     28H
43FB C2051F               JP     NZ,1F05H
43FE D7                   RST    16
43FF CD0D26        DE2    CALL   260DH
4402 7E                   LD     A,(HL)
4403 FE29                 CP     29H
4405 CA051F               JP     Z,1F05H
4408 CF                   RST    8
4409 2C                   INC    L
440A C3FF43               JP     DE2
440D F1            USR    POP    AF
440E CD2D44               CALL   US1
4411 D5                   PUSH   DE
4412 CD2C25               CALL   252CH
4415 E3                   EX     (SP),HL
4416 4E                   LD     C,(HL)
4417 23                   INC    HL
4418 46                   LD     B,(HL)
4419 21E726               LD     HL,26E7H
441C E5                   PUSH   HL
441D C5                   PUSH   BC
441E 3AAF40               LD     A,(40AFH)
4421 F5                   PUSH   AF
4422 FE03                 CP     3
4424 CCDA29               CALL   Z,29DAH
4427 F1                   POP    AF
4428 EB                   EX     DE,HL
4429 212141               LD     HL,4121H
442C C9                   RET
442D D7            US1    RST    16
442E 010000               LD     BC,0
4431 3005                 JR     NC,US2
4433 D630                 SUB    30H
4435 17                   RLA
4436 4F                   LD     C,A
4437 D7                   RST    16
4438 EB            US2    EX     DE,HL
4439 211F4A               LD     HL,4A1FH
443C 09                   ADD    HL,BC
443D EB                   EX     DE,HL
443E C9                   RET
443F CD2D44        DE1    CALL   US1
4442 D5                   PUSH   DE
4443 CF                   RST    8
4444 D5                   PUSH   DE
4445 CD022B               CALL   2B02H
4448 E3                   EX     (SP),HL
4449 73                   LD     (HL),E
444A 23                   INC    HL
444B 72                   LD     (HL),D
444C E1                   POP    HL
444D C9                   RET
444E 1E2E          AS2    LD     E,2EH
4450 C3A219               JP     19A2H
4453 110000        HEX    LD     DE,0
4456 D7                   RST    16
4457 2B                   DEC    HL
4458 0605                 LD     B,5
445A 23            HX6    INC    HL
445B 7E                   LD     A,(HL)
445C CD3E1E               CALL   1E3EH
445F EB                   EX     DE,HL
4460 300A                 JR     NC,HX3
4462 FE3A                 CP     3AH
4464 3019                 JR     NC,HX4
4466 D630                 SUB    30H
4468 3815                 JR     C,HX4
446A 1806                 JR     HX5
446C FE47          HX3    CP     47H
446E 300F                 JR     NC,HX4
4470 D637                 SUB    37H
4472 29            HX5    ADD    HL,HL
4473 29                   ADD    HL,HL
4474 29                   ADD    HL,HL
4475 29                   ADD    HL,HL
4476 B5                   OR     L
4477 6F                   LD     L,A
4478 05                   DEC    B
4479 CAB207               JP     Z,07B2H
447C EB                   EX     DE,HL
447D 18DB                 JR     HX6
447F CD9A0A        HX4    CALL   0A9AH
4482 EB                   EX     DE,HL
4483 C9                   RET
4484 00            LAST   DEFB   0
0000                      END
00000 TOTAL ERRORS
```


✓See List of Advertisers on page 306

other.

DEFUSR (DEFine USeR)

Ten different USR routines (USR0 through USR9) can be defined. The purpose of DEFUSR is to tell the BASIC interpreter the start address of a machine language subroutine.

Values can be passed from BASIC to USR and vice versa. Consult both the Level II BASIC manual and the Disk BASIC manual for the complete story.

Hexadecimal Notation

Disk BASIC offers you a way to use both hexadecimal and octal notation in addition to decimal notation. This is a great help to the programmer. It avoids the errors made when code is available in hex (or octal) and needs to become part of a BASIC program (which requires decimal).

Level II and 1/2 skips the octal option. There just doesn't seem to be much demand for it, which simplifies matters. A number without a prefix is assumed to be decimal. A number with an ampersand (&) prefix is assumed to be hexadecimal. Thus, in Level II and 1/2 the interpretation of these two statements is identical: 10 DEFUSR3 = 27648, 10 DEFUSR3 = &6C00. Level II and 1/2 does not support hex notation in data or input statements.

Program Listing 2

```
                ; < TWOHAF >     PART 2                  V 1.2   800401
                ;
                ;                GIL SPENCER (VK2JK)
                ;                BOX 300 / SPIT JUNCTION, NSW  2088
                ;                AUSTRALIA
                ;
                ; THE FOLLOWING LABELS, DEFINED IN PART 1, ARE NEEDED IN
                ; PART 2 -
4322            ASC     DEFL     4322H
43E9            DEF     DEFL     43E9H
440D            USR     DEFL     440DH
4453            HEX     DEFL     4453H
4484            LAST    DEFL     4484H
                ;
4484                    ORG      LAST     ;CONNECT TO 1ST PART OF TWOHAF
                ;
4484 CD611B     LINE    CALL     1B61H
4487 011E1D             LD       BC,1D1EH
448A 2B                 DEC      HL
448B D7                 RST      16
448C C5                 PUSH     BC
448D 010A00             LD       BC,000AH
4490 C5                 PUSH     BC
4491 50                 LD       D,B
4492 58                 LD       E,B
4493 2826               JR       Z,NU1
4495 FE2C               CP       2CH
4497 2809               JR       Z,NU2
4499 D5                 PUSH     DE
449A CD4F1E             CALL     1E4FH
449D 42                 LD       B,D
449E 4B                 LD       C,E
449F D1                 POP      DE
44A0 2819               JR       Z,NU1
44A2 CF         NU2     RST      8
44A3 2C                 INC      L
44A4 CD4F1E             CALL     1E4FH
44A7 2812               JR       Z,NU1
44A9 F1                 POP      AF
44AA CF                 RST      8
44AB 2C                 INC      L
44AC D5                 PUSH     DE
44AD CD5A1E             CALL     1E5AH
44B0 C29719             JP       NZ,1997H
44B3 7A                 LD       A,D
44B4 B3                 OR       E
44B5 CA4A1E             JP       Z,1E4AH
44B8 EB                 EX       DE,HL
44B9 E3                 EX       (SP),HL
44BA EB                 EX       DE,HL
44BB C5         NU1     PUSH     BC
44BC EB                 EX       DE,HL
44BD 22E545             LD       (NU0),HL
44C0 EB                 EX       DE,HL
44C1 CD2C1B             CALL     1B2CH
44C4 D1                 POP      DE
44C5 D5                 PUSH     DE
44C6 C5                 PUSH     BC
44C7 CD2C1B             CALL     1B2CH
44CA 60                 LD       H,B
44CB 69                 LD       L,C
44CC D1                 POP      DE
44CD DF                 RST      24
44CE EB                 EX       DE,HL
44CF DA4A1E             JP       C,1E4AH
44D2 D1                 POP      DE
44D3 C1                 POP      BC
44D4 F1                 POP      AF
44D5 D5                 PUSH     DE
44D6 180E               JR       NU3
44D8 09         NU5     ADD      HL,BC
44D9 DA4A1E             JP       C,1E4AH
44DC EB                 EX       DE,HL
44DD E5                 PUSH     HL
44DE 21F9FF             LD       HL,0FFF9H
44E1 DF                 RST      24
44E2 E1                 POP      HL
44E3 DA4A1E             JP       C,1E4AH
44E6 D5         NU3     PUSH     DE
44E7 5E                 LD       E,(HL)
```

Program continues

BASIC Line Renumbering

This is certainly one facility that every programmer wishes he had at his finger tips. A line renumberer should be easy to use; should be fast; should fix all pointers (GOTO, GOSUB, etc.); and should be flexible (variable start points, variable increments, etc.).

Level II and 1/2 offers all these advantages. About the only thing it does not do is change any reference to line numbers that you may have buried in REM statments.

The Disk BASIC word that activates the line renumbering routine in Level II and 1/2 is Line. Used by itself, it assumes these default values:

- New first line number = 10
- Renumber *all* lines
- Increment = 10

The Line command can control all three of these parameters. For example, LINE 1000, 500, 20 means: new first line number = 1000; start renumbering with old line number 500; increment = 20.

Such control is extremely handy when you need to insert a large block of numbers in the middle of a BASIC program to accommodate a new routine.

Default values and specific values can be mixed in the Line command. For example – LINE, 100 means: new first line number = 10; renumber *all* lines; increment = 100.

Concatenating BASIC Programs

In EDP the word Merge means to shuffle or blend together. Append or Concatenate is a subset of Merge, meaning place next to each other.

As the word Append is not in the Disk BASIC vocabulary, I had to settle for Merge. Here's how you use it: You have a BASIC program called John in RAM. John's highest line number is

BYTEWRITER-1

The World's First Under $300 80 Column Dot Matrix Printer

PRICE BREAKTHROUGH FROM MICROTEK

Bytewriter-1, the $299 printer that has taken the industry by storm! A product made possible through cost-effective design, efficient manufacturing operations, a proven low-cost print mechanism and the simplest possible control electronics. We have done extensive testing with all three models of the TRS-80, the Apple II, and the Atari 400 and 800. That is why we can guarantee you won't have any interface problems with the Bytewriter-1.

TRS-80 is a trademark of Radio Shack, Div. of Tandy Corp.
Apple II is a trademark of Apple Computer, Inc.
Atari 400 & 800 are trademarks of Atari, Inc.
Bytewriter-1 is a trademark of Microtek, Inc.

Microtek, Inc. ~360
9514 Chesapeake Drive
San Diego, CA 92123
Tel. 714-278-0633
Outside Calif. call
toll free: 800-854-1081
TWX. 910-335-1269

MICROTEK inc.

BYTEWRITER-1 SPECIFICATIONS

Printing Technology:	7-wire bi-directional impact wire matrix
Printing Speed:	60 lines per minute (80 characters per second) continuous
Character Set:	96 character ASCII (upper and lower case)
Character Size:	10 characters per inch (80 columns per line) plus expanded printing
Paper:	Friction feed (synchronous), accepts single sheet and roll paper up to 9½ inches maximum width. Prints original plus 3 copies.

By the way, our replacement print-head costs less than $30 too!
Call or write today for more information.

990. You have another BASIC program called Les on cassette, and you want to add Les to the bottom end of John. In Level II and 1/2, simply enter the word Merge. The VDU will say OK. RAM is now apparently empty (like after New). CLOAD Les in the usual way. At this stage the line numbers of Les do not matter. After CLOAD it will appear as though only Les is in RAM. You can run it, change it, anything you like. You must renumber Les so that the lowest line number is higher than 990, which you can do quickly using the Level II and 1/2 Line command. The Line command will not affect John.

If you had a third program on cassette named Bruce that you want to add beneath Les, use the word Merge again. Then CLOAD Bruce and continue as with Les. Remember that Bruce must start at a higher line number than Les finished at.

When you've got all the pieces put together (and the only limit is how much available RAM you have), you're ready for the last command in the sequence, which is the Disk BASIC word RSET. When you enter RSET, the VDU will say OK. Now you've got one big BASIC program! Easy as that!

Details

You can add Level II and 1/2 to your arsenal in several ways.

- Use Listings 1 and 2. These listings have both the source code and the object code. If you have EDTASM I'd suggest you key the source into the EDTASM buffer. After all, you are sure to decide to improve it.
- If you don't have EDTASM, you can key in the object code using T-Bug or a similar utility. Use the two left-most columns of Listings 1 and 2. Be sure to get the right code in the right addresses.
- If you can't type (or don't want to) write to me and we can arrange either a trade or sale of a cassette version of the program.

After you have a copy of Level II and 1/2, it's easy to use. Load it into your Level II machine via System. The name is Twohaf. Hit the slant line command and everything's done. You're back in BASIC with extra functions at your fingertips.

To remind you that the enhancement is in there, a message will be printed before each ready prompt. This is nice when you've got as many versions around as I have!

The Level II and 1/2 code starts at the beginning of Level II free RAM (&42E9). Positioning it here means that you can almost forget about it. All Level II commands work as always. You can even set memory size for machine language routines in high memory.

You do lose about 1100 bytes of free RAM. Of course, if you find yourself with the memory size query again, you'll have to reload Twohaf, as the links are all set back to normal Level II by the bootstrap.

Knowing how the links in Tables 1 and 2 work will spark other ideas for you. It's nice to have some extra commands available that you can give from the keyboard.

For example, it is not at all unusual for me to have RSM2 loaded in high memory (&6C00 and higher) while jumping back and forth between BASIC and machine language. It's easy enough to get from RSM2 to BASIC—G 1A19 does that. Getting back to RSM2 used to be a hassle. I have to first type System, then slant line, followed by 27648 (or is it 26748?). That's now all changed. Only six bytes did it. The beginning of RSM2 is &6C00 and fortunately it's also the entry point. So six bytes were added just before &6C00.

```
6BFA  :  21 00 6C  :  LD HL,6C00
6BFD  :  22 80 41  :  LD (4180),HL
```

After the addition, RSM2 was rerecorded using: P 6BFA 7E00 6BFA. The result of this addition is to link RSM2 with the Disk BASIC word Get.

Now all that's needed from BASIC is to enter Get—and, presto, we're back in RSM2. The same approach can be used for your favorite monitor or other frequently needed machine language utility.

Now, if we could only speed up the baud rate of the cassette I/O, maybe we won't need disks after all . . . ■

```
44E8 7B                LD    A,E
44E9 23                INC   HL
44EA 56                LD    D,(HL)
44EB B2                OR    D
44EC EB                EX    DE,HL
44ED D1                POP   DE
44EE 2807              JR    Z,NU4
44F0 7E                LD    A,(HL)
44F1 23                INC   HL
44F2 B6                OR    (HL)
44F3 2B                DEC   HL
44F4 EB                EX    DE,HL
44F5 20E1              JR    NZ,NU5
44F7 C5          NU4   PUSH  BC
44F8 CD1846            CALL  NZ0
44FB CD2345            CALL  NU6+1
44FE 2AE545            LD    HL,(NU0)
4501 EB                EX    DE,HL
4502 CD2C1B            CALL  1B2CH
4505 60                LD    H,B
4506 69                LD    L,C
4507 C1                POP   BC
4508 D1                POP   DE
4509 D5          NU8   PUSH  DE
450A 5E                LD    E,(HL)
450B 7B                LD    A,E
450C 23                INC   HL
450D 56                LD    D,(HL)
450E B2                OR    D
450F 280D              JR    Z,NU7
4511 EB                EX    DE,HL
4512 E3                EX    (SP),HL
4513 EB                EX    DE,HL
4514 23                INC   HL
4515 73                LD    (HL),E
4516 23                INC   HL
4517 72                LD    (HL),D
4518 EB                EX    DE,HL
4519 09                ADD   HL,BC
451A EB                EX    DE,HL
451B E1                POP   HL
451C 18EB              JR    NU8
451E 01181A      NU7   LD    BC,1A18H
4521 C5                PUSH  BC
4522 FEF6        NU6   CP    0F6H
4524 AF                XOR   A
4525 32E745            LD    (FB),A
4528 2AA440            LD    HL,(40A4H)
452B 2B                DEC   HL
452C 23          NU9   INC   HL
452D 7E                LD    A,(HL)
452E 23                INC   HL
452F B6                OR    (HL)
4530 C8                RET   Z
4531 23                INC   HL
4532 5E                LD    E,(HL)
4533 23                INC   HL
4534 56                LD    D,(HL)
4535 D7          NW7   RST   16
4536 B7          NW3   OR    A
4537 28F3              JR    Z,NU9
4539 4F                LD    C,A
453A 3AE745            LD    A,(FB)
453D B7                OR    A
453E 79                LD    A,C
453F 2857              JR    Z,NW1
4541 FE9E              CP    9EH
4543 2025              JR    NZ,NW2
4545 D7                RST   16
4546 FE8D              CP    8DH
4548 20EC              JR    NZ,NW3
454A D7                RST   16
454B FE0E              CP    0EH
454D 20E7              JR    NZ,NW3
454F D5                PUSH  DE
4550 CDFE45            CALL  NW4
4553 7A                LD    A,D
4554 B3                OR    E
4555 201B              JR    NZ,NW5
4557 E5                PUSH  HL
4558 2AE845            LD    HL,(FC)
455B 2B                DEC   HL
455C 212000            LD    HL,20H
455F 2B                DEC   HL
4560 212000            LD    HL,20H
4563 2B                DEC   HL
4564 213000            LD    HL,30H
4567 E1                POP   HL
4568 182A              JR    NW6
456A FE0E        NW2   CP    0EH
456C 20C7              JR    NZ,NW7
456E D5                PUSH  DE
456F CDFE45            CALL  NW4
4572 E5          NW5   PUSH  HL
4573 CD2C1B            CALL  1B2CH
4576 0B                DEC   BC
4577 3E0D              LD    A,0DH
4579 384B              JR    C,NW8
457B CDF920            CALL  20F9H
457E 21D645            LD    HL,NW9
4581 D5                PUSH  DE
```

Program continues

```
4582 CDA728    Ø473Ø         CALL     28A7H
4585 E1        Ø474Ø         POP      HL
4586 CDAFØF    Ø475Ø         CALL     ØFAFH
4589 C1        Ø476Ø         POP      BC
458A CDØA46    Ø477Ø         CALL     NX1
458D E1        Ø478Ø         POP      HL
458E E5        Ø479Ø         PUSH     HL
458F C5        Ø48ØØ         PUSH     BC
459Ø CDA7ØF    Ø481Ø         CALL     ØFA7H
4593 E1        Ø482Ø NX6     POP      HL
4594 D1        Ø483Ø NW6     POP      DE
4595 2B        Ø484Ø         DEC      HL
4596 189D      Ø485Ø         JR       NW7
4598 FEØD      Ø486Ø NW1     CP       ØDH
459A 2Ø99      Ø487Ø         JR       NZ,NW7
459C D5        Ø488Ø         PUSH     DE
459D CDFE45    Ø489Ø         CALL     NW4
45AØ EB        Ø49ØØ         EX       DE,HL
45A1 23        Ø491Ø         INC      HL
45A2 23        Ø492Ø         INC      HL
45A3 23        Ø493Ø         INC      HL
45A4 4E        Ø494Ø         LD       C,(HL)
45A5 23        Ø495Ø         INC      HL
45A6 46        Ø496Ø         LD       B,(HL)
45A7 EB        Ø497Ø         EX       DE,HL
45A8 6Ø        Ø498Ø         LD       H,B
45A9 69        Ø499Ø         LD       L,C
45AA CDEB46    Ø5ØØØ         CALL     NX3
45AD EB        Ø5Ø1Ø         EX       DE,HL
45AE 13        Ø5Ø2Ø         INC      DE
45AF 2AE845    Ø5Ø3Ø         LD       HL,(FC)
45B2 2B        Ø5Ø4Ø         DEC      HL
45B3 2B        Ø5Ø5Ø         DEC      HL
45B4 2B        Ø5Ø6Ø         DEC      HL
45B5 ØEØ5      Ø5Ø7Ø         LD       C,5
45B7 1A        Ø5Ø8Ø NX5     LD       A,(DE)
45B8 B7        Ø5Ø9Ø         OR       A
45B9 283B      Ø51ØØ         JR       Z,NX4
45BB 77        Ø511Ø         LD       (HL),A
45BC 23        Ø512Ø         INC      HL
45BD 13        Ø513Ø         INC      DE
45BE ØD        Ø514Ø         DEC      C
45BF 2ØF6      Ø515Ø         JR       NZ,NX5
45C1 D1        Ø516Ø NX7     POP      DE
45C2 2B        Ø517Ø         DEC      HL
45C3 C33545    Ø518Ø         JP       NW7
45C6 219345    Ø519Ø NW8     LD       HL,NX6
45C9 E5        Ø52ØØ         PUSH     HL
45CA 2AE845    Ø521Ø         LD       HL,(FC)
45CD E5        Ø522Ø         PUSH     HL
45CE 2B        Ø523Ø         DEC      HL
45CF 7Ø        Ø524Ø         LD       (HL),B
45DØ 2B        Ø525Ø         DEC      HL
45D1 71        Ø526Ø         LD       (HL),C
45D2 2B        Ø527Ø         DEC      HL
45D3 77        Ø528Ø         LD       (HL),A
45D4 E1        Ø529Ø         POP      HL
45D5 C9        Ø53ØØ         RET
45D6 42        Ø531Ø NW9     DEFM     'BAD BRANCH TO '
45E4 ØØ        Ø532Ø         DEFB     Ø
ØØØ2           Ø533Ø NUØ     DEFS     2
ØØØ1           Ø534Ø FB      DEFS     1
ØØØ2           Ø535Ø FC      DEFS     2
45EA B7B6      Ø536Ø FJ      DEFW     ØB6B7H
45EC 9D9F      Ø537Ø         DEFW     9F9DH
45EE C28E      Ø538Ø NX2     DEFW     8EC2H
45FØ B4B5      Ø539Ø         DEFW     ØB5B4H
45F2 8DCA      Ø54ØØ         DEFW     ØCA8DH
45F4 9195      Ø541Ø         DEFW     9591H
45F6 362Ø      Ø542Ø NX4     LD       (HL),2ØH
45F8 23        Ø543Ø         INC      HL
45F9 ØD        Ø544Ø         DEC      C
45FA 2ØFA      Ø545Ø         JR       NZ,NX4
45FC 18C3      Ø546Ø         JR       NX7
45FE 23        Ø547Ø NW4     INC      HL
45FF 5E        Ø548Ø         LD       E,(HL)
46ØØ 23        Ø549Ø         INC      HL
46Ø1 56        Ø55ØØ         LD       D,(HL)
46Ø2 23        Ø551Ø         INC      HL
46Ø3 22E845    Ø552Ø         LD       (FC),HL
46Ø6 2B        Ø553Ø         DEC      HL
46Ø7 C3781D    Ø554Ø         JP       1D78H
46ØA D5        Ø555Ø NX1     PUSH     DE
46ØB C5        Ø556Ø         PUSH     BC
46ØC 3E2Ø      Ø557Ø         LD       A,2ØH
46ØE 1EØ5      Ø558Ø         LD       E,5
461Ø ØB        Ø559Ø NX8     DEC      BC
4611 Ø2        Ø56ØØ         LD       (BC),A
4612 1D        Ø561Ø         DEC      E
4613 2ØFB      Ø562Ø         JR       NZ,NX8
4615 C1        Ø563Ø         POP      BC
4616 D1        Ø564Ø         POP      DE
4617 C9        Ø565Ø         RET
4618 2AA44Ø    Ø566Ø NZØ     LD       HL,(4ØA4H)
461B 7E        Ø567Ø NZ2     LD       A,(HL)
461C 23        Ø568Ø         INC      HL
461D B6        Ø569Ø         OR       (HL)
461E 2838      Ø57ØØ         JR       Z,NZ1
462Ø 23        Ø571Ø         INC      HL
4621 23        Ø572Ø         INC      HL
```

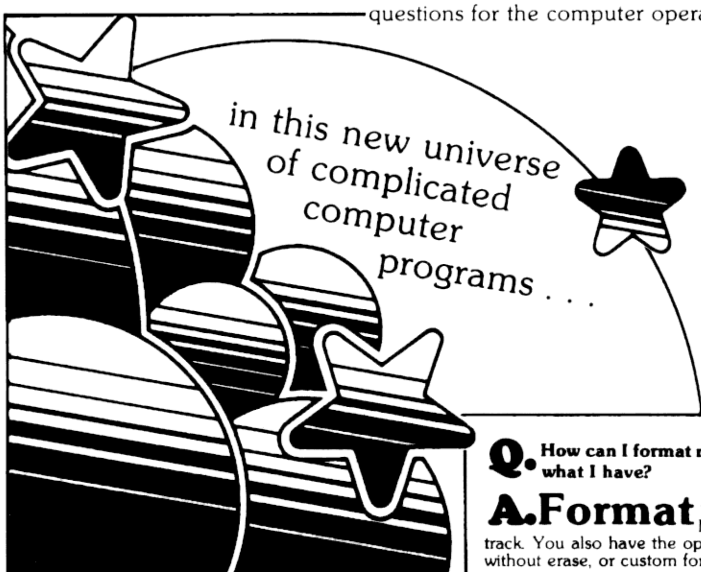
Program continues

✓See List of Advertisers on page 306

```
4622 23         05730       INC     HL
4623 7E         05740 NZ3   LD      A,(HL)
4624 B7         05750       OR      A
4625 23         05760       INC     HL
4626 28F3       05770       JR      Z,NZ2
4628 F22346     05780       JP      P,NZ3
462B 2B         05790       DEC     HL
462C 11EA45     05800       LD      DE,FJ
462F 0E0C       05810       LD      C,0CH
4631 1A         05820 NZ5   LD      A,(DE)
4632 BE         05830       CP      (HL)
4633 2845       05840       JR      Z,NZ4
4635 13         05850       INC     DE
4636 0D         05860       DEC     C
4637 20F8       05870       JR      NZ,NZ5
4639 7E         05880       LD      A,(HL)
463A FE89       05890       CP      89H
463C 23         05900       INC     HL
463D 20E4       05910       JR      NZ,NZ3
463F E5         05920       PUSH    HL
4640 2B         05930       DEC     HL
4641 D7         05940       RST     16
4642 FE23       05950       CP      23H
4644 200F       05960       JR      NZ,NZ6
4646 D7         05970       RST     16
4647 FEF3       05980       CP      0F3H
4649 200A       05990       JR      NZ,NZ6
464B D7         06000 NZ7   RST     16
464C 2807       06010       JR      Z,NZ6
464E FE2C       06020       CP      2CH
4650 20F9       06030       JR      NZ,NZ7
4652 F1         06040       POP     AF
4653 1825       06050       JR      NZ4
4655 E1         06060 NZ6   POP     HL
4656 18CB       06070       JR      NZ3
4658 2AA440     06080 NZ1   LD      HL,(40A4H)
465B EB         06090       EX      DE,HL
465C 62         06100 NA2   LD      H,D
465D 6B         06110       LD      L,E
465E 7E         06120       LD      A,(HL)
465F 23         06130       INC     HL
4660 B6         06140       OR      (HL)
4661 2867       06150       JR      Z,NZ8
4663 23         06160       INC     HL
4664 23         06170       INC     HL
4665 23         06180       INC     HL
4666 7E         06190 NA1   LD      A,(HL)
4667 23         06200       INC     HL
4668 FE0E       06210       CP      0EH
466A 2809       06220       JR      Z,NZ9
466C B7         06230       OR      A
466D 20F7       06240       JR      NZ,NA1
466F EB         06250       EX      DE,HL
4670 73         06260       LD      (HL),E
4671 23         06270       INC     HL
4672 72         06280       LD      (HL),D
4673 18E7       06290       JR      NA2
4675 23         06300 NZ9   INC     HL
4676 23         06310       INC     HL
4677 18ED       06320       JR      NA1
4679 2B         06330 NA7   DEC     HL
467A D7         06340 NZ4   RST     16
467B E5         06350       PUSH    HL
467C CD5A1E     06360       CALL    1E5AH
467F C1         06370       POP     BC
4680 7D         06380       LD      A,L
4681 91         06390       SUB     C
4682 2824       06400       JR      Z,NA3
4684 D5         06410       PUSH    DE
4685 5F         06420       LD      E,A
4686 3E02       06430       LD      A,2
4688 F5         06440       PUSH    AF
4689 3E05       06450       LD      A,5
468B 93         06460       SUB     E
468C 3007       06470       JR      NC,NA4
468E 2F         06480       CPL
468F 3C         06490       INC     A
4690 D1         06500       POP     DE
4691 C602       06510       ADD     A,2
4693 F5         06520       PUSH    AF
4694 AF         06530       XOR     A
4695 C5         06540 NA4   PUSH    BC
4696 C4D446     06550       CALL    NZ,NA5
4699 E1         06560       POP     HL
469A 360E       06570       LD      (HL),0EH
469C 23         06580       INC     HL
469D C1         06590       POP     BC
469E D1         06600       POP     DE
469F 73         06610       LD      (HL),E
46A0 23         06620       INC     HL
46A1 72         06630       LD      (HL),D
46A2 23         06640 NA6   INC     HL
46A3 3620       06650       LD      (HL),20H
46A5 10FB       06660       DJNZ    NA6
46A7 23         06670 NA8   INC     HL
46A8 2B         06680 NA3   DEC     HL
46A9 D7         06690       RST     16
46AA 3C         06700       INC     A
46AB 3D         06710       DEC     A
46AC 38CB       06720       JR      C,NA7
46AE 23         06730       INC     HL
```

Program continues

questions for the computer operator, concerned with the versatility of their TRS-80 programs.



How in heavens name, can SUPER UTILITY provide answers?

Super Utility is a powerful and sophisticated zapping program that allows you to go to the heart of the disk and read or modify data with ease, engaging simple one-key commands that threads through all of your logical decision choices. Super Utility, written by Kim Watt of Breeze Computing, Inc., is a stand alone program containing seven main menus, which are the answers to frustrated questions you have while struggling through your TRS-80 programs.

Q. Isn't there an easier way to examine and modify data?

A. Zap has an easy to read printout that reveals information in both HEX and ASCII and simultaneously moveable dual cursors. You can modify data using Hex, Decimal, ASCII or Binary input, and any changes are automatically updated on both sides of the readout. You can search through disk or file sectors, stopping anywhere to copy, compare and verify data on your disk.

Q. Isn't there an easier way to get rid of this data I don't want?

A. Purge enables you to clear a disk of unwanted data. Kill files by file-spec or have the computer list them one at a time for deletion.

Q. How can I format my disk without erasing what I have?

A. Format your disk and add tracks. Make a 35 track disk a 40 track. You also have the option of formatting with or without erase, or custom formatting your disk.

Q. This disk is protected. Isn't there some way I can copy it?

A. Disk Copy enables you to back up most TRS-80 readable disks, regardless of efforts to protect it. So, back up your original and back up your modified version, too, or . . .

A. Tape Copy if you wish to back up your tape.

Q. My disk won't boot. Now what do I do?

A. Disk Repair will recover killed files, if the file was accidently killed by this utility. Repair GAT table, HIT table and Boot. Read protect directory track and check directory.

Q. How can I get more access to my memory?

A. Memory allows you to move, test, compare, zero, exchange edit, or jump to, memory. Load memory to/from disk and input or output a byte to any port.

You will love the simplicity and freedom of modifying programs to suit your needs. Now, the only question left is . . .

Q. How do I order?

A. Send check or money order for only $49.95 plus $2.50 shipping and handling to:

✓76

Breeze Computing, Inc. P.O. Box 1013, Berkley, MI 48072

Foreign orders, please add $5.00 additional postage. Michigan residents add 4% sales tax.

Breeze Computing, Inc. will send every owner, upon registration of Super Utility, one back up copy. **$2.50.**

DISCOVER THE 6809 IN YOUR COLOR COMPUTER

Now you can explore the Radio Shack Color Computer's impressive potentials—as an inexpensive development system, a color peripheral, a process controller—ad infinitum. The Micro Works introduces these powerful software tools for utilizing the color computer at the assembly language level.

MONITOR TAPE: A cassette tape which allows you to:

- Examine or change memory using a formatted hex display
- Save areas of memory to cassette in binary (a "CSAVEM")
- Download/upload data or programs to a host system
- Move the video display page throughout RAM
- Send or receive RS-232 at up to 9600 baud
- Investigate and activate features of your computer, such as hi-res graphics or machine-language music
- Use your color computer as an intelligent peripheral for another computer, a color display or a 6809 program development tool

The monitor has 19 commands in all, and is relocatable and re-entrant.

80C Monitor Tape Price: $29.95

MONITOR ROM: The same program as the monitor tape, supplied on ROM. This allows BASIC to use the entire RAM space. And you don't need to re-load the monitor each time you use it. The ROM plugs into the Extended Basic ROM Socket or a modified ROMPACK.

80C Monitor ROM
Price: $39.95

C BUG IS HERE!!

INSIDE THE COLOR COMPUTER: This package is a disassembler which runs on the color computer and enables you to generate your own source listing of the BASIC interpreter ROM. Also included is a documentation package which gives useful ROM entry points, complete memory map, I/O hardware details and more. Disassembler features include cross-referencing of variables and labels; output code which can be re-assembled; output to an 80-column printer, small printer or screen; and a data table area specification which defaults to the table boundaries in the interpreter ROM. A 16K system is required for the use of this cassette.

80C Disassembler Price: $49.95

THE MICRO WORKS ✓109

Mastercharge and BankAmericard

P.O. BOX 1110 DEL MAR, CA 92014 714-942-2400

```
46AF CA1B46    06740          JP      Z,NZ2
46B2 2B        06750          DEC     HL
46B3 FE20      06760          CP      20H
46B5 28F0      06770          JR      Z,NA8
46B7 FE2C      06780          CP      2CH
46B9 28EC      06790          JR      Z,NA8
46BB FECE      06800          CP      0CEH
46BD 28E8      06810          JR      Z,NA8
46BF FED4      06820          CP      0D4H
46C1 3804      06830          JR      C,NA9
46C3 FED7      06840          CP      0D7H
46C5 38E0      06850          JR      C,NA8
46C7 C32346    06860 NA9      JP      NZ3
46CA 2AF940    06870 NZ8      LD      HL,(40F9H)
46CD 22FB40    06880          LD      (40FBH),HL
46D0 22FD40    06890          LD      (40FDH),HL
46D3 C9        06900          RET
46D4 E5        06910 NA5      PUSH    HL
46D5 2AF940    06920          LD      HL,(40F9H)
46D8 EB        06930          EX      DE,HL
46D9 2600      06940          LD      H,0
46DB 6F        06950          LD      L,A
46DC 19        06960          ADD     HL,DE
46DD 22F940    06970          LD      (40F9H),HL
46E0 44        06980          LD      B,H
46E1 4D        06990          LD      C,L
46E2 E1        07000          POP     HL
46E3 1A        07010 NB1      LD      A,(DE)
46E4 02        07020          LD      (BC),A
46E5 DF        07030          RST     24
46E6 C8        07040          RET     Z
46E7 1B        07050          DEC     DE
46E8 0B        07060          DEC     BC
46E9 18F8      07070          JR      NB1
46EB CD9A0A    07080 NX3      CALL    0A9AH
46EE AF        07090          XOR     A
46EF CD3410    07100          CALL    1034H
46F2 B6        07110          OR      (HL)
46F3 C3D90F    07120          JP      0FD9H
46F6 21FD46    07130 READY    LD      HL,GI1
46F9 CDA728    07140          CALL    28A7H   ;DISPLAY A STRING POINTED @ BY HL
46FC C9        07150          RET
46FD 0B        07160 GI1      DEFB    0BH
46FE 8785      07170          DEFW    8587H   ;GRAPHICS CHARACTER
4700 4C        07180          DEFM    'LEVEL II-1/2 (V 1.2) IS -'
4719 00        07190          DEFB    0       ;NUL= END OF STRING FOR CALL 28A7
471A ED5BF940  07200 MERGE    LD      DE,(40F9H)      ; ( )= END BASIC
471E 1B        07210          DEC     DE
471F D5        07220          PUSH    DE
4720 C1        07230          POP     BC
4721 1B        07240          DEC     DE
4722 1B        07250          DEC     DE
4723 1B        07260          DEC     DE
4724 1A        07270          LD      A,(DE)
4725 B7        07280          OR      A
4726 2005      07290          JR      NZ,ME2
4728 ED43F940  07300          LD      (40F9H),BC
472C 1B        07310          DEC     DE
472D 13        07320 ME2      INC     DE
472E 13        07330          INC     DE
472F ED53A440  07340          LD      (40A4H),DE
4733 214B47    07350          LD      HL,ME1          ; OK MESSAGE
4736 CDA728    07360          CALL    28A7H           ; DISPLAY A STRING
4739 C3191A    07370          JP      1A19H           ; JP TO BASIC
473C 215247    07380 RSET     LD      HL,TWOHAF
473F 22A440    07390          LD      (40A4H),HL      ; RESET 'BASIC BEGIN'
4742 214B47    07400          LD      HL,ME1
4745 CDA728    07410          CALL    28A7H
4748 C3191A    07420          JP      1A19H
474B 3C        07430 ME1      DEFM    '< OK >'
4751 00        07440          DEFB    0
               07450 ;
               07460 ; NOTE- PROGRAM FROM HERE DOWN IS LOST WHEN JP TO BASIC
               07470 ;       IS DONE.  IT IS ONLY NEEDED FOR INITIALIZATION.
               07480 ;
4752 212243    07490 TWOHAF   LD      HL,ASC
4755 225641    07500          LD      (4156H),HL
4758 21E943    07510          LD      HL,DEF
475B 225C41    07520          LD      (415CH),HL      ;LINK TO DISK BASIC 'DEF'
475E 215344    07530          LD      HL,HEX
4761 229541    07540          LD      (4195H),HL
4764 3EC3      07550          LD      A,0C3H          ;'JP' FOR USR & READY
4766 32A941    07560          LD      (41A9H),A
4769 210D44    07570          LD      HL,USR          ;LINK TO DISK BASIC 'USR'
476C 22AA41    07580          LD      (41AAH),HL
476F 218444    07590          LD      HL,LINE         ;LINK TO DISK B. 'LINE'
4772 22A441    07600          LD      (41A4H),HL
4775 32AC41    07610          LD      (41ACH),A
4778 21F646    07620          LD      HL,READY
477B 22AD41    07630          LD      (41ADH),HL      ;LINK WITH 'READY'
477E 211A47    07640          LD      HL,MERGE
4781 228C41    07650          LD      (418CH),HL      ;LINK WITH D.B. 'MERGE'
4784 213C47    07660          LD      HL,RSET
4787 229B41    07670          LD      (419BH),HL      ;LINK WITH D.B. 'RSET'
478A 215247    07680          LD      HL,TWOHAF
478D 22A440    07690          LD      (40A4H),HL      ;LOAD NEW 'BASIC BEGIN'
4790 CD491B    07700          CALL    1B49H           ;CALL LVL II 'NEW'
4793 C3191A    07710          JP      1A19H           ;JP TO BASIC
4752           07720          END     TWOHAF
00000 TOTAL ERRORS
```

A game for the refractionary.

The Level II Black Box

Morris Jones
533 Sutter St. #1206
San Francisco, CA 94102

When I tell you I have a Black Box for Level II BASIC, I'm not talking about a piece of hardware. Black Box is a game that tests your ability to find its contents.

Parker Brothers released Black Box last year as a game for two players. In their version, one player sets up the Black Box and the other plays detective. In this Level II program the computer sets up the Black Box and gives you the clues. You have the fun of digging into the box. Though the program does not keep score for two players, you can alternate at the keyboard and tally your own scores.

If you have played the Parker Brothers version you will find the computer a more devious opponent than a real person; people tend to use patterns, while BASIC is very random.

This Black Box is really an eight-unit by eight-unit square, much like a checker or chess board. The program hides three, four or five balls, or markers, in the box, and the object is to locate them by probing the box with imaginary rays. Only three conditions befall a ray: It may never leave the box; it may exit at a different location; or it may be reflected back out the same way it entered.

Imaginary Rays

A ray can enter the box in any of 32 places—eight squares on four sides. To launch a ray you must select one of the numbered locations. The ray advances one square at a time, until it leaves the box or hits a ball. If a ray hits a hidden ball, it will be absorbed. The program then marks the entry point with an H. See Fig. 1. During the actual game you cannot see the path the ray follows, but only where it comes out—if it does.

As the ray advances, if it sees a ball that is ahead one square and one square to the left or right, it will be deflected 90 degrees away from the ball and leave the box at another point. (In other words, if the ball is

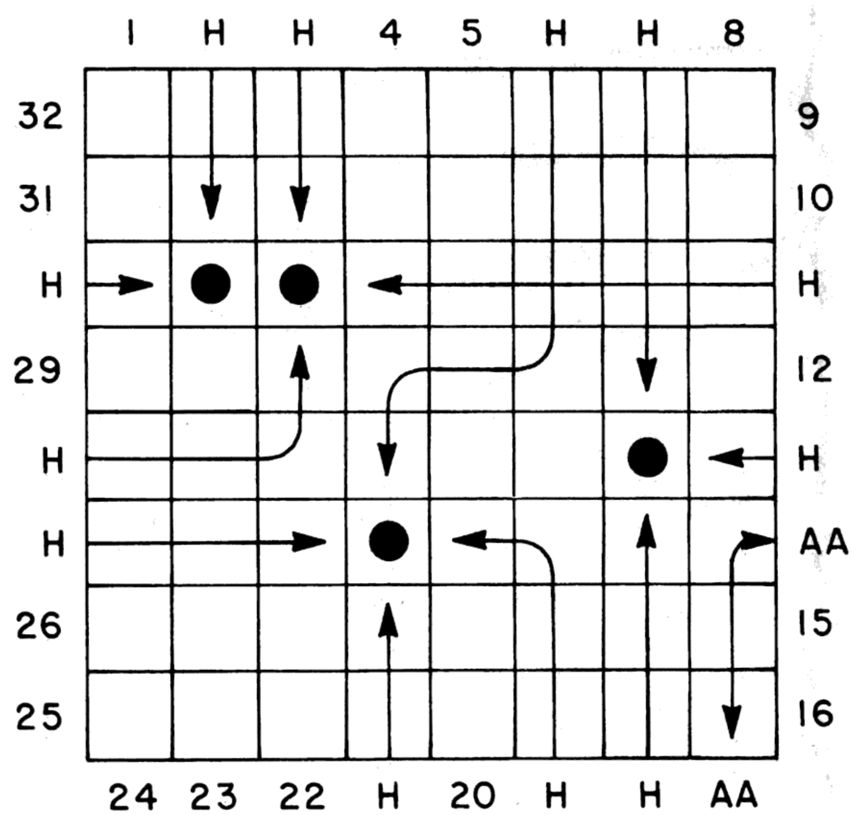


Fig. 1. Examples of hits: Two, three and seven are direct hits. Six, 19, and 28 are deflected hits. The program only marks the hit, not how the ray traveled. AA is an ordinary deflected ray for comparison.

"You will find the computer a more devious opponent than a real person."

ahead left, the ray will turn right.) The program then marks the entry and exit points with two double letters. The first deflection will be marked AA at both entry and exit, the second, BB, and so forth.

A ray can be deflected several times before it leaves the box, or it may be deflected to land on a hit, in which case the program simply marks H without telling you that the ray was first deflected. The ray can also travel straight through to the opposite side without being deflected at all. See Fig. 2 for examples of deflections. Fig. 1 also shows some deflected hits.

B(8,8)	hidden ball array
G(8,8)	guess array
N1(32)	array of edge marker strings
N2(16)	sets of double letters to use as markers
VI	video address
SC	score
C	"instructions in progress" flag
IM	index for N2, N2(IM) is next marker to use
X1	entry point for ray
X2	exit point for ray
E	direction vector

Table 1. Important Variables

Now consider the special case in which a ray finds a ball ahead to the left *and* right. The ray can be deflected neither left nor right, and, instead, is reflected back toward the entry square. The ray exits from the same point it entered, and the program marks the point with an R.

One other special case causes a reflection. If the entry point you choose is directly to the left or right of a ball hidden on the edge, the ray will never have a chance to enter the board and be deflected, thus it is marked as a reflection. Examples of this special case and other reflections are illustrated in Fig. 3.

Stop and Score

The initial instructions give animated descriptions of what happens to a ray when it enters the box. During the game launch as many rays as you like, place guess balls, or remove them as often as you need to before you stop and score.

When you place a guess ball, the computer puts a graphic marker on your guess square. This is useful as a reference while you check your board. If you find that the ball is in the wrong place, remove it and

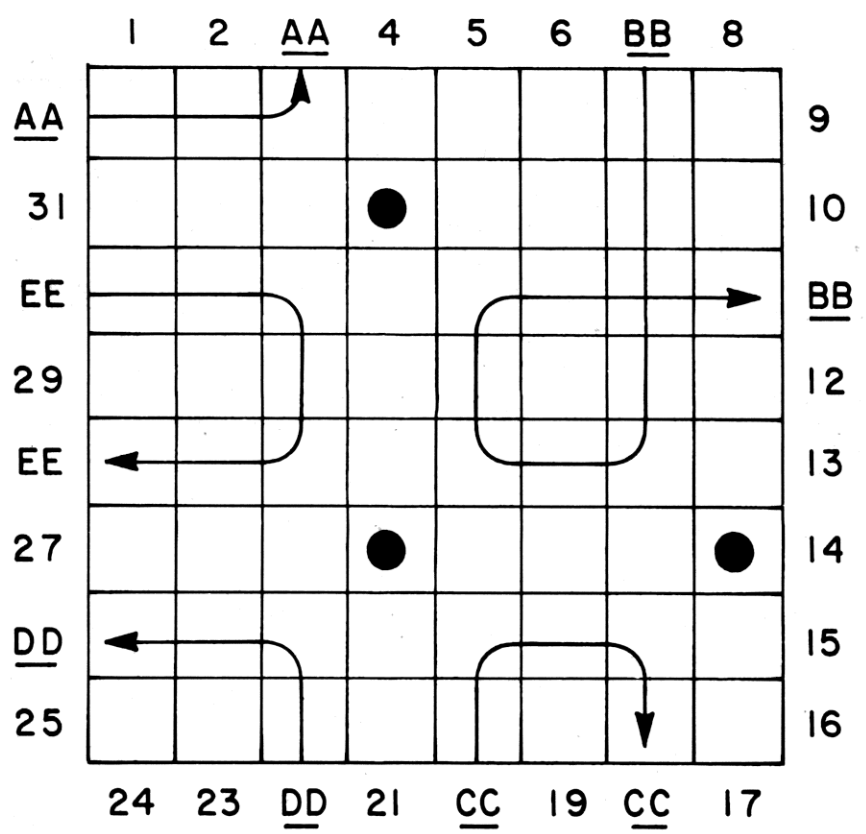


Fig. 2. Examples of deflected rays with markers at entry and exit. Notice that if you swap the entry point with the exit point, you achieve the same result—a deflection path will always produce one and only one pair of markers.

"Breaking the program during execution allows you to check for bugs, restart and draw the board again."

place it somewhere else. The program will not tell you whether a guess is right or wrong until you finally stop and score.

Your score is the number of lettered markers used. The object is to solve the Black Box using as few markers as possible. A hit or reflection scores one point, and a deflection scores two, but reveals more about the board. A wrong guess at the end of the game adds a penalty of five points.

A game with three balls is very simple. However, a game with five balls is considerably more difficult. When playing with five balls, it is possible to create an ambiguity, that is, one of the balls may be impossible to locate from the edges, though this rarely happens with random layouts.

The program takes full advantage of Level II features and is designed to be relatively crash proof. If the screen becomes filled with garbage, you can redraw the board. Breaking the program during execution allows you to check for bugs, restart and draw the board again. All of the input is done through INKEY$ routines and most responses do not require the Enter key.

Give Black Box a try next time you get tired of Swords and Sorcerers. It's a good solitaire game.

Program Notes

Black Box is one of my first programs. Though it's been refined and play-tested, I'm sure I still have things to learn about efficient programming. In particular, the ray movement logic (lines 2000–2340) seems too clumsy. The routine works beautifully, but if you find suggestions to reduce those If...Then statements, I would enjoy hearing from you.

For newer programmers, the many lines that state: IF E=1 IF B=2 IF VV<>8... etc., are not mistakes. The various conditions can be grouped together with "and" operators, but then BASIC will have to evaluate the entire expression before evaluating the If statement. I felt that the program would run a little faster if, after evaluating the first expression and finding it false, it need not evaluate the rest.

Here is a description of the major routines of the program:

Lines 10–110 initialize the program and set up the variables.

Lines 500–660 display the board for the first time and hide the balls. Line 610 is the entry point for each move.

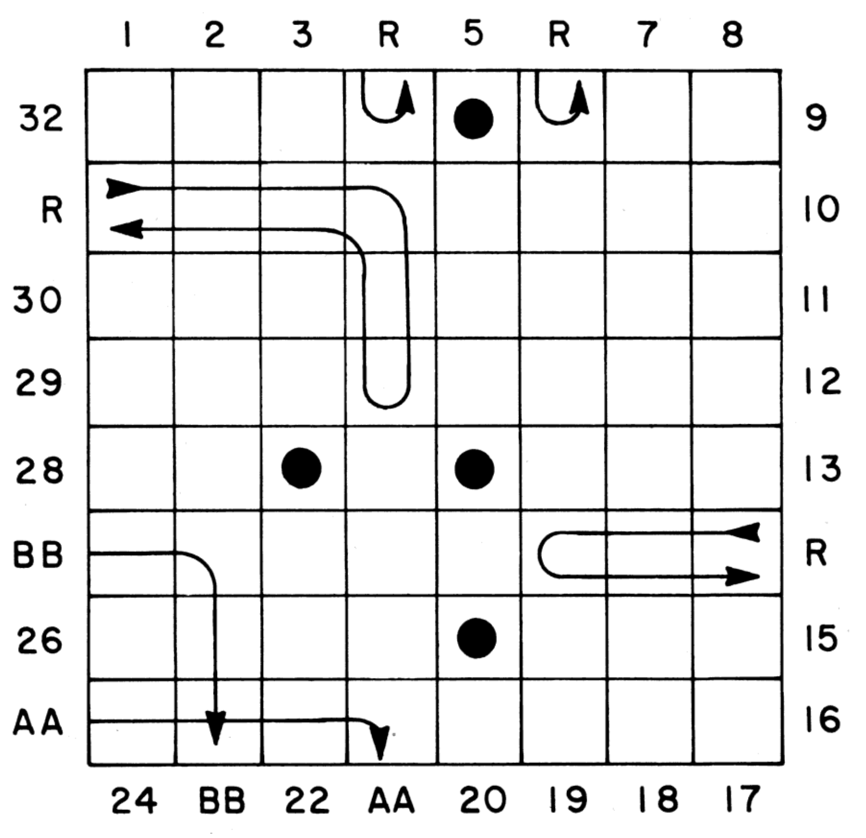


Fig. 3. Examples of reflections: Four and six reflect because of the ball on the edge (five would be a hit). Fourteen is a straightforward spell out reflection, and 31 is a deflected reflection. For comparison, AA and BB are ordinary deflected rays.

"The final display of the board shows where the program hid the balls."

Lines 1000–1260 make up the stop and score routine. The routine first checks to see if you have made the right number of guesses and returns to the menu if you haven't. It then checks the guess array against the ball array and scores discrepancies. Lines 1050–1210 show graphically where wrong guesses and correct locations are. The final display of the board shows where the program hid the balls.

Lines 2000–2340 provide the move logic for the rays. Variable C is a flag, telling if the program is in the instruction sequence. If so, the ray movement is displayed. This can be useful for debugging the ray movement logic.

Lines 3000–3040 place a guess in the guess array.

Lines 4000–4040 remove a guess from the guess array.

Lines 5000–5020 redraw the board.

Lines 6000–6080 are used by the routines at 3000 and 4000. The routine displays the coordinate markers, accepts the input, and sets the values for the array subscripts in order to place or remove a guess ball.

Lines 7000–7070 complete the ray movement logic by setting or changing the values for the edge markers and printing them around the box.

Lines 8000–8420 are the instructions. With variable C set equal to −1, the instructions go back through the program and display graphically what happens to rays as they go through the box using an example.■

Program Listing

```
10 CLS
20 CLEAR200
30 DIM N1$(32),B%(8,8),N2$(16),G%(8,8),N3$(16)
40 DEFSTRN,Y:DEFINTA-M,O-X,Z
50 VI=15360:SC=0:IM=1:W=0
55 FORI=1TO8:FORJ=1TO8:B(I,J)=0:G(I,J)=0:NEXTJ,I
60 FORI=1TO32:N1(I)=RIGHT$(STR$(I),2):NEXT
70 FORI=1TO7:N2(I)=STRING$(2,CHR$(64+I)):NEXT:FORI=8TO16:N2(I)=S
TRING$(2,CHR$(65+I)):NEXT
80 FORI=1TO8:N3(I)=STR$(I):NEXT:FORI=9TO16:N3(I)=CHR$(56+I)+" ":
NEXT
85 IFCTHEN110
90 CLS:PRINT:PRINT"BLACK BOX VERSION 2.0":PRINT"BY MORRIS JONES"
:PRINT:PRINT"DO YOU NEED INSTRUCTIONS? ";
95 Y=INKEY$:IFY=""THEN95
100 PRINTY;:FORI=1TO200:NEXT:IFY="Y"THENC=-1:GOTO8000:ELSEIFY<>"
N"PRINTCHR$(08);:GOTO95
110 C=0:CLS
500 FORVV=1TO8:FORHV=1TO8:D=VI+65+64*VV+3*HV:POKED,143:POKED+1,1
43:NEXTHV,VV
510 FORI=1TO8:PRINT@65+3*I,N1(I);:NEXT
520 FORI=9TO16:PRINT@156+64*(I-9),N1(I);:NEXT
530 FORI=17TO24:PRINT@665-3*(I-17),N1(I);:NEXT
540 FORI=25TO32:PRINT@577-64*(I-25),N1$(I);:NEXT
550 PRINT@99,"CHOOSE:";:PRINT@163,"DEGREE OF DIFFICULTY,";:PRINT
@227,"NUMBER OF BALLS (3-5)?";
560 H=VAL(INKEY$):IFH<3ORH>5THEN560ELSEPRINT@250,H;
570 FORI=1TOH
580 HV=RND(8):VV=RND(8)
590 IFB(HV,VV)THEN580ELSEB(HV,VV)=-1:NEXT
600 FORI=1TO700:NEXT
610 GOSUB8410
620 PRINT@99,"CHOOSE:";:PRINT@163,"1) STOP AND SCORE";:PRINT@227
,"2) LAUNCH RAY";:PRINT@291,"3) PLACE BALL GUESS";
630 PRINT@355,"4) REMOVE BALL GUESS";:PRINT@419,"5) REDRAW BOX";
:PRINT@773,'POINTS USED:";SC;
640 PRINT@803,STRING$(34," "):PRINT@803,"CHOICE?";
650 X0=VAL(INKEY$):IFX0<1ORX0>5THEN650ELSEPRINTX0;:FORI=1TO200:N
EXT
660 ONX0GOTO1000,2000,3000,4000,5000
1000 FORI=1TO8:FORJ=1TO8:IFG(I,J)THENW=W+1:NEXTJ,IELSENEXTJ,I
1010 IFW>HPRINT@803,"TOO MANY GUESSES";:W=0:FORI=1TO700:NEXT:GOT
O610
1020 IFW<HPRINT@803,"NOT ENOUGH GUESSES";:W=0:FORI=1TO700:NEXT:G
OTO610
1025 GOSUB8410:GI=-1
1030 FORI=1TO8:FORJ=1TO8
1040 IFG(I,J)ANDNOTB(I,J)THEND=VI+65+64*I+3*J:GI=0:ELSE1140
1050 PRINT@803,STRING$(29," ");:PRINT@803,"WRONG GUESS";
```

Program continues

★ ★

4MHZ, DOUBLE DENSITY, COLOR & B/W GRAPHICS . . THE LNW80 COMPUTER

COMPARE THE FEATURES AND PERFORMANCE

FEATURES	LNW80	PMC-80**	TRS-80* MODEL III
PROCESSOR	4.0 MHZ	1.8 MHZ	2.0 MHZ
LEVEL II BASIC INTERP.	YES	YES	LEVEL III BASIC
TRS80 MODEL 1 LEVEL II COMPATIBLE	YES	YES	NO
48K BYTES RAM	YES	YES	YES
CASSETTE BAUD RATE	500/1000	500	500/1500
FLOPPY DISK CONTROLLER	SINGLE/ DOUBLE	SINGLE	SINGLE/ DOUBLE
SERIAL RS232 PORT	YES	YES	YES
PRINTER PORT	YES	YES	YES
REAL TIME CLOCK	YES	YES	YES
24 X 80 CHARACTERS	YES	NO	NO
VIDEO MONITOR	YES	YES	YES
UPPER AND LOWER CASE	YES	OPTIONAL	YES
REVERSE VIDEO	YES	NO	NO
KEYBOARD	63 KEY	53 KEY	53 KEY
NUMERIC KEY PAD	YES	NO	YES
B/W GRAPHICS, 128 X 48	YES	YES	YES
HI-RESOLUTION B/W GRAPHICS, 480 X 192	YES	NO	NO
HI-RESOLUTION COLOR GRAPHICS (NTSC), 128 X 192 IN 8 COLORS	YES	NO	NO
HI-RESOLUTION COLOR GRAPHICS (RGB), 384 X 192 IN 8 COLORS	OPTIONAL	NO	NO
WARRANTY	6 MONTHS	90 DAYS	90 DAYS
TOTAL SYSTEM PRICE	$1,914.00	$1,840.00	$2,187.00
LESS MONITOR AND DISK DRIVE	$1,450.00	$1,375.00	---

When you've compared the features of an LNW80 Computer, you'll quickly understand why the LNW80 is the ultimate TRS80 software compatible system. LNW RESEARCH offers the most complete microcomputer system at an outstanding low price.
We back up our product with an unconventional 6 month warranty and a 10 days full refund policy, less shipping charges.

LNW80 Computer $1,450.00
LNW80 Computer w/B&W Monitor & one 5" Drive $1,914.00
All orders must be prepaid, CA residents please include 6% sales tax.
Contact us for shipping charges

* TRS80 Product of Tandy Corporation.
** PMC Product of Personal Microcomputer, Inc.

LNW80

- BARE PRINTED CIRCUIT BOARD & MANUAL $89.95

The LNW80 - A high-speed color computer totally compatible with the TRS-80*. The LNW80 gives you the edge in satisfying your computation needs in business, scientific and personal computation. With performance of 4 MHz, Z80A CPU, you'll achieve performance of over twice the processing speed of a TRS-80*. This means you'll get the performance that is comparable to the most expensive microcomputer with the compatibility to the world's most popular computer (TRS-80*) resulting in the widest software base.

FEATURES:
- TRS-80 Model 1 Level II Software Compatible
- High Resolution Graphics
  - RGB Output - 384 x 192 in 8 Colors
  - NTSC Video or RF MOD - 128 x 192 in 8 Colors
  - Black and White - 480 x 192
- 4 MHz CPU
- 500/1000 Baud Cassette
- Upper and Lower Case
- 16K Bytes RAM, 12K Bytes ROM
- Solder Masked and Silkscreened

LNW SYSTEM EXPANSION

- BARE PRINTED CIRCUIT BOARD AND MANUAL $69.95
  WITH GOLD CONNECTORS $84.95

The System Expansion will allow you to expand your LNW80, TRS-80*, or PMC-80** to a complete computer system that is still totally software compatible with the TRS-80* Model 1 Level II.

FEATURES:
- 32K Bytes Memory
- 5" Floppy Controller
- Serial RS232 20ma I/O
- Parallel Printer
- Real Time Clock
- Screen Printer Bus
- On Board Power Supply
- Solder Masked and Silkscreened

LNDoubler & DOS PLUS 3.3D

- Assembled and Tested W/DOS PLUS 3.3D. $175.00

Double-density disk storage for the LNW Research's "System Expansion" or the Tandy's "Expansion Interface". The LNDoubler™ is totally software compatible with any double density software generated for the Percom's Doubler***. The LNDoubler™ provides the following outstanding features.

- Store up to 350K bytes on a single 5" disk
- Single and double density data separation
- Precision write precompensation circuit
- Software switch between single and double density
- Easy plug in installation requiring no etch cuts, jumpers or soldering
- 35, 40, 77, 80 track 5" disk operation
- 120 day parts and labor Warranty

*** Doubler is a product of Percom Data Company, Inc.

DOS PLUS 3.3D

Micro Systems software's double density disk operating system. This operating system contains all the outstanding features of a well developed DOS, with ease in useability.

KEYBOARD

LNW80 KEYBOARD KIT $84.95

The Keyboard Kit contains a 63 key plus a 10 key, P.C. board, and remaining components.

CASE

LNW80 CASE $84.95

The streamline design of this metal case will house the LNW80, LWN System Expansion, LNW80 Keyboard, power supply and fan, LNDoubler™, or LNW Data Separator. This kit includes all the hardware to mount all of the above. Add $12.00 for shipping

PARTS AVAILABLE FROM LNW RESEARCH
- 4116 - 200ns RAM
  - 6 chip set $26.00
  - 8 chip set $33.50
  - 16 chip set $64.00
  - 24 chip set $94.00
  - 32 chip set $124.00

- LNW80 "Start up parts set" LNW80-1 . . . . $82.00
- LNW80 "Video parts set" LNW80-2 . . . . $31.00
- LNW80 Transformer LNW80-3 . . . . $18.00
- LNW80 Keyboard cable LNW80-4 . . . . $16.00
- 40 Pin computer to expansion cable $15.00
- System Expansion Transformer $19.00
- Floppy Controller (FD1771) and UART (TR1602) . . . $30.00

LNW RESEARCH
C O R P O R A T I O N
2620 WALNUT
TUSTIN CA. 92680 ✓53

ORDERS & INFO. NO. 714-544-5744
SERVICE NO. 714-641-8850

VISA MASTER CHARGE ACCEPTED

UNLESS NOTED ADD $3 FOR SHIPPING

●●

```
1060 FORL=1TO5
1070 POKED,143:POKED+1,143:FORK=1TO250:NEXT
1080 POKED,133:POKED+1,138:FORK=1TO250:NEXT
1090 NEXTL
1100 POKED,143:POKED+1,143:SC=SC+5
1110 PRINT@773,"POINTS USED:";SC;
1120 NEXTJ,I
1130 GOTO1220
1140 IFB(I,J)ANDNOTG(I,J)THEND=VI+65+64*I+3*JELSE1210
1150 PRINT@803,STRING$(29," ");:PRINT@803,"CORRECT LOCATION";
1160 FORL=1TO5
1170 POKED,133:POKED+1,138:FORK=1TO250:NEXT
1180 POKED,143:POKED+1,143:FORK=1TO250:NEXT
1190 NEXT
1200 POKED,133:POKED+1,138
1210 NEXTJ,I
1220 GOSUB8410:IFGIPRINT@739,"ALL CORRECT";
1230 PRINT@803,"PLAY AGAIN? ";
1240 Y0=INKEY$:IFY0=""THEN1240
1250 IFY0="N"THENPRINTY0:END
1255 IFY0<>"Y"THEN1240:ELSEPRINTY0;:FORI=1TO200:NEXT
1260 CLS:C=-1:GOTO50
2000 GOSUB8410:PRINT@99,"LAUNCH RAY";:PRINT@227,"0 RETURNS TO ME
NU";:PRINT@803,STRING$(28," "):PRINT@803,"CHOOSE VECTOR (01-32)?
 ";
2010 Y0=INKEY$:IFY0=""THEN2010ELSEIFASC(Y0)>47ANDASC(Y0)<58PRINT
Y0;:ELSE2010
2015 Y1=INKEY$:IFY1=""THEN2015ELSEIFY1=CHR$(08)PRINTY1;:GOTO2010
ELSEIFASC(Y1)=13THENY1="":GOTO2025:ELSEIFASC(Y1)>47ANDASC(Y1)<58
PRINTY1;ELSE2015
2020 Y=INKEY$:IFY=""THEN2020ELSEIFY=CHR$(08)PRINTY;:GOTO2015
2025 X1=VAL(Y0+Y1):IFX1=0THEN610ELSEIFX1>32THEN2000
2030 IFX1<9THENE=1:HV=1:VV=X1:GOTO2070
2040 IFX1<17ANDX1>8THENE=2:HV=X1-8:VV=8:GOTO2070
2050 IFX1<25ANDX1>16THENE=3:HV=8:VV=-X1+25:GOTO2070
2060 E=4:HV=-X1+33:VV=1
2070 IFB(HV,VV)THEN7010
2080 IFE=1ORE=3IFVV<>1IFB(HV,VV-1)THEN7000
2090 IFE=1ORE=3IFVV<>8IFB(HV,VV+1)THEN7000
2100 IFE=2ORE=4IFHV<>1IFB(HV-1,VV)THEN7000
2110 IFE=2ORE=4IFHV<>8IFB(HV+1,VV)THEN7000
2120 IFCTHEND=VI+65+64*HV+3*VV:POKED,133:POKED+1,138:FORI=1TO100
:NEXT:POKED,143:POKED+1,143
2130 IFE=1IFHV=8THEN2290ELSEIFB(HV+1,VV)THEN7010
2140 IFE=2IFVV=1THEN2290ELSEIFB(HV,VV-1)THEN7010
2150 IFE=3IFHV=1THEN2290ELSEIFB(HV-1,VV)THEN7010
2160 IFE=4IFVV=8THEN2290ELSEIFB(HV,VV+1)THEN7010
2170 IFE=1IFVV<>1IFB(HV+1,VV-1)THENE=4:GOTO2120
2180 IFE=1IFVV<>8IFB(HV+1,VV+1)THENE=2:GOTO2120
2190 IFE=2IFHV<>1IFB(HV-1,VV-1)THENE=1:GOTO2120
2200 IFE=2IFHV<>8IFB(HV+1,VV-1)THENE=3:GOTO2120
2210 IFE=3IFVV<>1IFB(HV-1,VV-1)THENE=4:GOTO2120
2220 IFE=3IFVV<>8IFB(HV-1,VV+1)THENE=2:GOTO2120
2230 IFE=4IFHV<>1IFB(HV-1,VV+1)THENE=1:GOTO2120
2240 IFE=4IFHV<>8IFB(HV+1,VV+1)THENE=3:GOTO2120
2250 IFE=1THENHV=HV+1:GOTO2120
2260 IFE=2THENVV=VV-1:GOTO2120
2270 IFE=3THENHV=HV-1:GOTO2120
2280 VV=VV+1:GOTO2120
2290 IFE=1THENX2=25-VV:GOTO2330
2300 IFE=2THENX2=33-HV:GOTO2330
2310 IFE=3THENX2=VV:GOTO2330
2320 X2=8+HV
2330 IFX2=X1THEN7000
2340 N1(X1)=N2(IM):N1(X2)=N2(IM):SC=SC+2:IM=IM+1:GOTO7020
3000 GOSUB8410
3010 PRINT@99,"PLACE A GUESS BALL";:PRINT@227,"0 RETURNS TO MENU
";:GOSUB6000
3020 G(HG,VG)=-1:D=VI+65+64*HG+3*VG
3030 POKED,133:POKED+1,138
3040 GOTO7020
4000 GOSUB8410
4010 PRINT@99,"REMOVE GUESS BALL";:PRINT@227,"0 RETURNS TO MENU"
;:GOSUB6000
4020 G(HG,VG)=0:D=VI+65+64*HG+3*VG
4030 POKED,143:POKED+1,143
4040 GOTO7020
5000 CLS:FORI=1TO8:FORJ=1TO8:D=VI+65+64*I+3*J:POKED,143:POKED+1,
143:NEXTJ,I
5010 FORI=1TO8:FORJ=1TO8:IFG(I,J)THEND=VI+65+64*I+3*J:POKED,133:
POKED+1,138:NEXTJ,I:ELSENEXTJ,I
5020 GOTO7020
6000 FORI=1TO8:PRINT@1+3*I,I;:PRINT@94+I*64," ";CHR$(64+I);:NEXT
6030 PRINT@803,"ENTER GUESS (EX: E6)? ";
6035 Y0=INKEY$:IFY0=""THEN6035ELSEIFY0="0"THENPRINTY0;:FORI=1TO2
```

Program continues

RS-232 Printer Driver Versatility

We have a versatile and flexible RS-232 printer driver that can be set up for a variety of printers. It uses the TRS-80* RS-232-C board. After loading, your LLIST and LPRINT statements will work in the normal manner.

Many parameters and options are available.

Driver Parameters:

- Form length. Number of lines per form (typically 66).
- Page size. Number of printed lines per page.
- Number of characters per line (carriage width).
- Number of nulls to insert after a carriage return.
- Number of nulls to insert after a form feed.

Driver Options:

- Top-of-form select (Form-feed or multiple line feeds).
- Long line disposition (truncate or fold).
- Use hardware handshake (yes/no).
- Process X-ON/X-OFF handshake (yes/no).
- Insert a line feed (L/F) after a carriage return (C/R) (yes/no).
- Convert lower case characters to upper case characters (yes/no).

Price: **$35.00** — includes cassette and manual.

Contact: Micro Systems Software
3235 Kifer Road, Suite 32
Santa Clara, California 95051
(408) 735-1650

✓493

*TRS-80 is a registered trademark of the Tandy Corp.

BUYER SELLER

ARE YOU PUZZLED BY THE USED COMPUTER MARKET?

Our unique, nationwide listing service puts buyers and sellers of used micro-equipment together.

Listings from $300 — $25,000
(Apples, TRS 80, IBM 5100's included)

YOU PAY ONLY FOR RESULTS!

We also offer for $6.75 the
USED MICRO-TRENDS REPORT
Data on:
Manufacturers • Models • Prices
Trends • Maintenance

call toll-free

USED COMPUTER EXCHANGE ✓494

(800) 327-9191 ext. 61

TOTAL DISKETTE BACKUP

DO NOT TAKE UNNECESSARY RISKS WITH YOUR SOFTWARE.

If your diskette software library is not completely backed up, mail the order below immediately! Or, if you are wasting diskettes by making backup copies of all your diskettes, your problem is solved! Using DUMPLOAD, the total contents of your diskettes can be safely dumped to tape. The hi-speed tape option allows six 35 or 40 track diskettes to fit on one C60 casette. This machine language utility will pay for itself the first time one of your valuable programs will not load.

- TRS-80 Model 1 16K - 48K
- TRSDOS or NEWDOS80 Compatible
- May be used to back-up TRSDOS, VTOS 4.0, NEWDOS, MICRODOS, or data disks. (Single Density)
- Tape verification routine included.
- Single drive owners are no longer required to keep switching diskettes to create a backup.
- Backup without having to remember Master passwords.

YES! RUSH ME MY COPY OF DUMPLOAD IMMEDIATELY

— $15.95 ON MY CAREFULLY PACKED TRSDOS OR NEWDOS80 DISKETTE ENCLOSED (priority service)

—$16.95 on Casette —$19.95 on Diskette

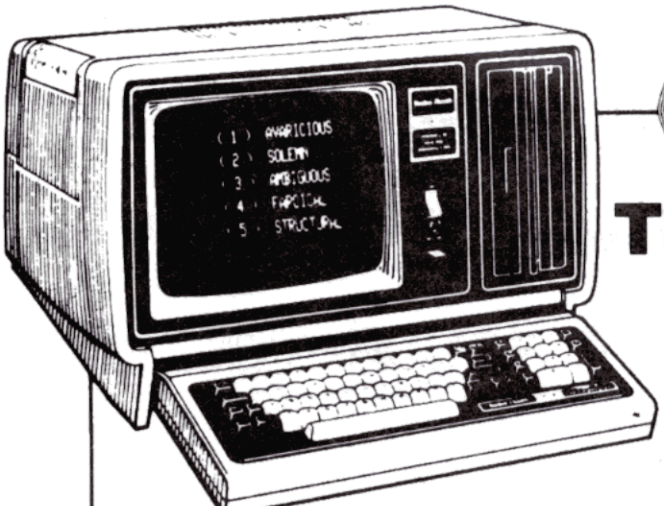
Name: ______

Address: ______

City: ______ State: ______

Zip: ______

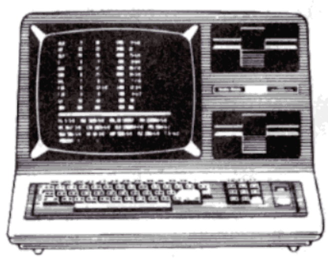
MAIL TO:
COMPLETE COMPUTER SERVICES ✓496
8188 HEATHER DRIVE
NEWBURGH, INDIANA 47630



Radio Shack® DEALER

TRS-80™ DISCOUNT

☐ NO OUT-OF-STATE TAX
☐ NO SHIPPING COSTS

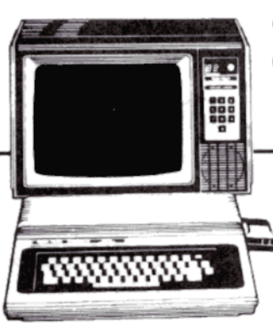


TRS-80 MODEL III
32K-2 DISKS
$2100

NEW PERSONAL COMPUTER . . . REAL-TIME CLOCK, SHARPER CRT IMAGES AND FASTER LOADING CASSETTES

TRS-80 MODEL II 64K $3500

PACKS ENOUGH DATA HANDLING POWER FOR MANY SMALL BUSINESSES.



TRS-80 COLOR COMPUTER OR VIDEO $339 each

A LOW COST, COLOR COMPUTER FOR PERSONAL BUSINESS OR ENTERTAINMENT

CERTIFIED CHECKS
CASHIERS CHECKS
OR CREDIT CARDS

MasterCard VISA

PERRY OIL & GAS INC. ✓496

137 NORTH MAIN STREET, PERRY, MICH. 48872

PHONE (517) 625-4161

WARRANTIES HONORED BY ALL RADIO SHACKS • *T.M. TANDY CORP.

✓See List of Advertisers on page 306

```
00:NEXT:GOTO6075
6040 IFASC(Y0)<65ORASC(Y0)>72THEN6035ELSEPRINTY0;
6050 Y1=INKEY$:IFY1=""THEN6050ELSEIFY1=CHR$(08)THENPRINTCHR$(08)
;:GOTO6035:ELSEIFVAL(Y1)<1ORVAL(Y1)>8THEN6050ELSEPRINTY1;
6060 Y=INKEY$:IFY=""THEN6060ELSEIFY=CHR$(08)THENPRINTCHR$(08);:G
OTO6050
6070 HG=ASC(Y0)-64:VG=VAL(Y1)
6075 FORI=1TO8:PRINT@1+3*I,"  ";:PRINT@94+I*64,"  ";:NEXT:IFY0="
0"THEN610
6080 RETURN
0"THEN610
6080 RETURN
7000 N1(X1)="R ":SC=SC+1:GOTO7020
7010 N1(X1)="H ":SC=SC+1
7020 FORI=1TO8:PRINT@65+3*I,N1(I);:NEXT
7030 FORI=9TO16:PRINT@156+64*(I-9),N1(I);:NEXT
7040 FORI=17TO24:PRINT@665-3*(I-17),N1(I);:NEXT
7050 FORI=25TO32:PRINT@577-64*(I-25),N1(I);:NEXT
7060 IFCTHEN8000
7070 GOTO610
8000 IFX1=2THEN8340ELSEIFX1=7THEN8310ELSEIFX1=14THEN8280ELSEIFX1
=15THEN8270ELSEIFX1=13THEN8260ELSEIFX1=30THEN8220ELSEIFX1=28THEN
8180ELSEIFX1=31THEN8140ELSEIFG(2,5)THEN8120ELSECLS:PRINT@89,"BLA
CK BOX"
8010 PRINT@195,"THE OBJECT OF THE GAME IS TO LOCATE THREE, FOUR
OR FIVE"
8020 PRINT"RANDOMLY HIDDEN BALLS IN AN EIGHT BY EIGHT FIELD.  yO
U WILL"
8030 PRINT"LOCATE THEM BY SENDING RAYS INTO THE FIELD FROM THE S
IDES."
8040 PRINT"A BALL WILL ABSORB A RAY THAT STRIKES IT DIRECTLY, OR
 DEFLECT"
8050 PRINT"A RAY THAT COMES WITHIN ONE SQUARE."
8060 PRINT"   FOLLOWING ARE SEVERAL EXAMPLES OF HOW RAYS WILL AC
T WHEN"
8070 PRINT"LAUNCHED INTO THE FIELD.  a RAY WILL EITHER BE A HIT,
 A"
8080 PRINT"REFLECTION, OR BE DEFLECTED TO LEAVE THE MATRIX AT AN
OTHER"
8090 PRINT"POINT.  PRESS ENTER EACH TIME FOR THE NEXT EXAMPLE."
8100 GOSUB8420
8110 CLS:B(2,5)=-1:B(4,5)=-1:B(6,8)=-1:G(2,5)=-1:G(4,5)=-1:G(6,8
)=-1:GOTO5000
8120 PRINT@99,"HERE IS A TYPICAL BALL";:PRINT@163,"ARRANGEMENT.
 A VECTOR SENT";:PRINT@227,"IN FROM 31 WOULD BE A HIT.";
8130 GOSUB8420:X1=31:GOTO2030
8140 GOSUB8420
8150 GOSUB8410
8160 PRINT@99,"A RAY FROM 28 WOULD";:PRINT@163,"DEFLECT TO 21, A
ND";:PRINT@227,"THE COMPUTER WOULD GIVE";:PRINT@291,"BOTH NUMBER
S A DOUBLE";:PRINT@355,"LETTER MARKER.";
8170 GOSUB8420:X1=28:GOTO2030
8180 GOSUB8420
8190 GOSUB8410
8200 PRINT@99,"A RAY FROM 30 WOULD BE";:PRINT@163,"REFLECTED BAC
K TO 30,";:PRINT@227,"AND MARKED WITH AN R.";
8210 GOSUB8420:X1=30:GOTO2030
8220 GOSUB8420
8230 GOSUB8410
8240 PRINT@99,"RAYS SENT IN FROM 13 AND";:PRINT@163,"15 WOULD AL
SO BE";:PRINT@227,"REFLECTIONS, AND MARKED";:PRINT@291,"WITH AN
R.";
8250 GOSUB8420:X1=13:GOTO2030
8260 X1=15:GOTO2030
8270 X1=14:GOTO2030
8280 GOSUB8420
8290 GOSUB8410:PRINT@99,"A RAY FROM 7 WOULD";:PRINT@163,"BE DEFL
ECTED TWICE,";:PRINT@227,"AND MARKED WITH A";:PRINT@291,"DIFFERE
NT DOUBLE LETTER.";:PRINT@355,"WATCH THIS ONE.";
8300 GOSUB8420:X1=7:GOTO2030
8310 GOSUB8420
8320 GOSUB8410:PRINT@99,"RAYS ENTERED AT 1, 2,";:PRINT@163,"OR 3
 WOULD FALL ALL THE";:PRINT@227,"WAY THROUGH.  HERE IS";:PRINT@2
91,"2, FOR EXAMPLE.";
8330 GOSUB8420:X1=2:GOTO2030
8340 GOSUB8420
8350 CLS:PRINT"   YOUR SCORE IS THE NUMBER OF MARKERS TOTAL THAT
 YOU USE"
8360 PRINT"TO LOCATE THE BALLS.  THE OBJECT IS TO HAVE THE LOWES
T TOTAL"
8370 PRINT"SCORE.":PRINT:PRINT"   YOU MAY PLACE BALL GUESSES AND
 REMOVE BALL GUESSES AT ANY"
8380 PRINT"TIME.  WHEN YOU THINK YOU HAVE A CORRECT LAYOUT, STOP
 AND"
8390 PRINT"SCORE.  AN INCORRECT BALL LOCATION COSTS FIVE POINTS.
"
8400 GOSUB8420:CLS:GOTO50
8410 FORI=1TO12:PRINT@35+I*64,STRING$(29," ");:NEXT:RETURN
8420 PRINT@803,"PRESS ENTER";:INPUTY:RETURN
```

An innovative word processing system for TRS-80* MOD I & MOD III

Lazy Writer®*

It is time to put your word processing program away and use a

Word Processing System

*Lazy Writer is the product of ABC Sales

Now accepting orders for MOD III version

©1980 by David Welsh

MOD I – $125
MOD III – $175
***Requires 2 drives for conversion only.**

LAZY WRITER Takes on Scripsit© by Radio Shack® and Electric Pencil©®

Has all the things that other word processing programs should have. Easy to use, written all in machine code. / It permits the inserting and deleting by characters, words, sentences, and paragraphs / Page scrolling up and down / Search ahead of the cursor or behind the cursor for any character / The cursor can be moved up, down, left and right / You can seek top of file and bottom of file / Block move of text, block delete of text/ Search and replace or search delete / Unlimited insert (to the limit of your machines memory) / Permits use with lower case /

Has things that other programs should have, but don't. Upper and lower case output to your printer (if your printer accepts lower case) without having your computer modified. ON UPPER CASE ONLY MACHINES: This program marks the capital letters so you can see which letters are CAPITALS and which are not. / Will change all upper characters text to lower case or all lower case to upper, A SINGLE COMMAND / Will capitalize the first letter of all sentences and all proper noun I's, WITH A SNGLE COMMAND / LOADS ANY ELECTRIC PENCIL / FILE, ASCII SAVED FILES, EDTASM FILES or BASIC PROGRAMS SAVED ASCII / Permits installing special control characters in your text for your printers special features, like double wide or condensed print / Definable screen length and definable print length to 255 characters wide / Screen editing that is not final till your command. This means that you can edit your file on the screen and if you don't like how it reads you can cancel and leave it the way it was / You can append files (which means that you can put one file to the end of another file) / No lost characters at the end of the line, even for the fastest typist / A directory of all your files is available to the user without leaving the program / Saving programs to disk easy enough for the non-computer user / To save memory, not all the program modules are in memory at one time but are called from the disk as needed / You can set tab positions like on a typewriter / 10 CUSTOM COMMAND KEYS for the experienced user there is a command file that permits many special functions that are all user defined (not enough space for better explanation in ad, send for complete overview) / Program has HELP file that is a short review of the commands that are available /

Standard Printer Module. This printer module is provided for the user as a standard feature. Optional special printer routines for custom printer will be available in the near future. In this original release, it has the following printer drivers and will support the following printing devices: RS232, TRS232 and PARALLEL printer ports. You have the following format commands: Justifies Text, Centers Text, Centers Title, Line Spacing, Line Length from 3-255 characters and Set Margins / Also send any ASCII code to any printer from the text / Save formatted text to the disk for spooling later / Information for customer to load his own special printer driver / Printing can be stopped and started by the user at any time and then restarted where you left off / You can print entire file or just print to bottom of the page /

Communication Package. RS232 COMMUNICATION TERMINAL PROGRAM permits you to communicate with other computers. Transfer files from one machine to another. Permits dumping memory across the phone lines. Receive files from other TRS-80's and "Shake Hands" with larger computers. This is the complete system called LAZY WRITER. There is no package written for the TRS-80* that is as comprehensive. This package is available for the TRS-80* MOD I, 32K or larger with at least a single disk drive. List price is from

$125.00

VISA · master charge

Dealer Inquiries Invited

SSM SOFT SECTOR MARKETING, INCORPORATED ✓516
6250 Middlebelt • Garden City, MI 48135
(313) 425-4020

C.O.D. - Certified Check, M.O. or Cash only. Sorry, no C.O.D. over $150.00! Most orders shipped next day. All orders must have shipping included. Please add 2% or $2.50, which ever is higher for shipping. Michigan residents, please add 4% tax. Add extra $1.50 for C.O.D. Personal checks take 3 weeks to clear. Send $1.00 for catalog - get $2.00 credit on next order.

NEW FEATURES in Lazy Writer
"The People Request, and David Welsh Delivers"

The system permits embedding ASCII commands into the text of the program. NOW you can do SUPERSCRIPT and SUBSCRIPT (if your printer can handle it). Underlining and boldface, printing of a single word in a paragraph, is now possible, at no extra cost.

A key that remembers the cursor position.

User definable special character. For use with printers that have printable characters that the TRS-80 keyboard does not normally support.

Margin control from within text. This means that you can change the margins of your printed text without stopping the printer routine and changing it.

Page offset with odd/even headers & footers. This means that you can print one page offset to the left of center and the next page offset to the right. This is very nice when you are writing a book.

Printing chaining feature. This permits having more than one file on disk and create one printed letter, contract, or book without having to reset the printer commands.

Mandatory space command. This is necessary when you are writing letters or papers that have certain words that are not to be broken-up. eg.:John P. Andhouser. This name can be made to be unbreakable to justify routines in the program.

Disk catalog. Now you can load your disk directory into memory and create a file of this information.

Now loads Machine Language extention programs that are writen for Lazy Writer.

An alternative to expensive Model II letter crunchers.

Everyman's Mod II Word Processor

Mike Kilroy, AC8V
6213 Concerto Ct.
Dayton, OH 45449

Many good word processors are available for the Model I from various magazines, but to put your Model II to work as a word processor requires a substantial outlay of cash. Instead of laying out many dollars for a Model II processor, I've modified an existing Model I program. The original program appeared in *80 Microcomputing's* May, 1980 issue, written by Delmer D. Hinrichs.

The program performs the following:

- •Creates or adds to text
- •Right justifies
- •Compiles
- •Sets formats for letters and the printer
- •Deletes blank lines automatically
- •Deletes specified line(s)
- •Edits on a per line basis
- •Displays all legal functions
- •Inserts line(s) at any point in the program
- •Deletes whole texts from memory or disk
- •Saves/loads any text to/from disk complete with its corresponding format
- •Moves whole blocks of text
- •Replaces one whole line with a new one
- •Provides a table of contents of all letters or text
- •Titles and pages, if wanted

What more could you want? The speed of this program in a standard 64K Model II is over 90 wpm. Could you keep up? The program has 17 commands, two of which have extensive subcommands.

Featured Commands

The Add command lets you add lines to text. An addition may be two lines or 100 lines long. Available in the add mode are the following subcommands:

Enter key	Ends line and carriage return.
Control J	Erases present line to start it over.
Down arrow	Ends line and page, adds end page marker (←).
Right arrow	Moves line all the way to the right.
Up arrow	Centers, ends, adds, "do not justify marker" (/) to present line.

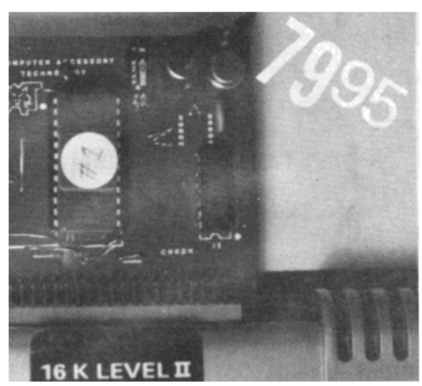
Program Listing

```
10 CLS:PRINTTAB(20)"BASIC WORD PROCESSOR"
30 DEFINTA-Z:CLEAR21000:NL=360:DIMA$(NL),X$(3),S(25),T(25)
40 B$=CHR$(23):C$=CHR$(143):F$="### ":N$="Y":PN$="N":P1$="Y"
50 S$=" ":H$=S$:LA=-1:P=1:FP=1:PL=23:LL=60:LM=10:U=32:G=10:H=6:V=27
60 ONERRORGOTO1645:L=LA:IT=0:R=0:A$="":PRINT:INPUT"COMMAND";A$:IFA$=""GOTO80
70 A=ASC(A$)-64:IFA>0THENONAGOTO90,480,510,760,790,1220,80,1320,1350,1390,1510,1
522,1580,80,80,1640,80,1750,1770,1520,80,1830,80,19
70
80 PRINT"** ENTRY ERROR **":GOTO60
90 CLS:D=0:N$="Y":IFLA<0THENL=0:GOTO130 'add
100 IF NL=LA+1THEN210ELSEIFL>FL+12THENB=L-12ELSEB=FL
110 FORI=BTOL:X=LEN(A$(I)):D=D+INT((X+4)/80-.01)
120 GOSUB1910:NEXTI:L=L+1
130 C=(L-FL+D)*80:IFC>1760THENPRINT:PRINT:C=1760
140 PRINT@C,USINGF$;L;:PRINTA$(L);:P=LEN(A$(L))+1:C=C+P+3:K=L+1
150 PRINT@C,C$;CHR$(02);:A$=INKEY$:PRINT@C,S$;:IFA$=""THEN150
160 GOSUB290:ONA-7GOTO360,410
170 IFA=13THENA$=S$:GOTO210ELSEIFA=10THEN380ELSEIFA=31THEN460
180 IFA=29THEN430ELSEIFA=30THEN330ELSEIFA=28THEN310
190 IFA=27THENIFLA<LTHENLA=L:GOTO60ELSE60
200 PRINT@C,A$;:A$(L)=A$(L)+A$:IFP<=LLTHENP=P+1:C=C+1:GOTO150
210 IFRTHEN60ELSEIFNL<=KTHENPRINT:PRINT"FILE FULL":LA=NL-1:GOTO60
220 IFLEN(A$(K))THENL=K:GOSUB1360
230 IFK>LATHENLA=K
240 IFA$=S$THEN280
250 FORM=LL+1TO2STEP-1:A$=MID$(A$(L),M,1):IFA$<>S$THENNEXTM:GOTO280
260 A$(K)=RIGHT$(A$(L),LL-M+1):A$(L)=LEFT$(A$(L),M-1)
270 PRINT@C-LL+M-1,B$;:L=K:GOTO130
280 A$(L)=LEFT$(A$(L),LL):L=K:GOTO130
290 A=ASC(A$):RETURN
310 IFP>LLTHEN210 'd
320 PRINT@C,CHR$(92);:A$(L)=A$(L)+CHR$(17):A$=S$:GOTO210
330 IFP>LLTHEN210 's-d
340 C=(L-FL+D)*80+4:IFC>1764THENC=1764
350 GOSUB1180:P=1:A$=S$:GOTO210
360 IFP=1THEN150 'l
370 C=C-1:PRINT@C,B$;:P=P-1:A$(L)=LEFT$(A$(L),P-1):GOTO150
380 IFP=1THEN150 's-l
390 A$(L)="":P=1:C=(L-FL+D)*80+4:IFC>1764THENC=1764
400 PRINT@C,B$;:GOTO150
410 IFP>LL-6THEN150 'r
420 A$(L)=A$(L)+STRING$(5,S$):C=C+5:P=P+5:GOTO150
430 IFP>LLTHEN210 's-r
440 C=(L-FL+D)*80+4:IFC>1764THENC=1764
450 GOSUB1200:P=1:A$=S$:GOTO210
460 IFP>LLTHEN210 'cl
470 PRINT@C,CHR$(93);:A$(L)=A$(L)+CHR$(20):A$=S$:GOTO210
```

Program continues

```
480 CLS:PRINT"DELETING BLANK LINES":FORJ=LATO0STEP-1 'blank
490 IFA$(J)=""THENFORI=JTOLA:A$(I)=A$(I+1):NEXTI:A$(LA)="":LA=LA-1
500 NEXTJ:IFRTHENRETURNELSE1830
510 INPUT"FIRST LINE TO COMPILE";F:IFF<0THENF=0 'compile
520 INPUT"LAST LINE TO COMPILE";Z:IFZ>LATHENZ=LA
530 IFF>=ZTHEN80ELSECLS:PRINT"COMPILING":FORL=FTOZ-1:K=L+1
540 X=LEN(A$(L)):X$="":IFX<2THEN620ELSEIFX<=LLGOTO600
550 FORI=XTO1STEP-1:A$=MID$(A$(L),I,1)
560 IFA$<>S$THENX$=A$+X$:NEXTI:GOTO600ELSEIFX$=""THENNEXTI
570 A=ASC(RIGHT$(X$,1)):IFA=33ORA=46ORA=58ORA=63THENX$=X$+" "
580 A$(L)=LEFT$(A$(L),I-1):IFLEN(A$(K))=0THENA$(K)=X$:GOTO540
590 A$(K)=X$+S$+A$(K):GOTO540
600 X=LEN(A$(L)):IFX<2THEN620ELSEFORI=XTO2STEP-1
610 IFRIGHT$(A$(L),1)=S$THENA$(L)=LEFT$(A$(L),I-1):NEXTI
620 NEXTL:FORL=FTOZ-1:K=L+1
630 X=LEN(A$(L)):Y=LEN(A$(K)):X$="":IFX=0ORY=0THEN750
640 A=ASC(RIGHT$(A$(L),1))
650 IFA=33ORA=46ORA=58ORA=63THENA$(L)=A$(L)+" ":X=X+1
660 FORI=1TOY:A$=MID$(A$(K),I,1)
670 IFA$<>S$THENX$=X$+A$:NEXTIELSEIFX$=""THENNEXTI
680 IFLL-X<ITHEN710
690 Y=Y-I:IFY<0THENY=0
700 A$(L)=A$(L)+S$+X$:A$(K)=RIGHT$(A$(K),Y):GOTO630
710 X=LEN(A$(L)):IFX<2THEN730ELSEFORI=XTO2STEP-1
720 IFRIGHT$(A$(L),1)=S$THENA$(L)=LEFT$(A$(L),I-1):NEXTI
730 IFY<2THEN750ELSEFORI=YTO2STEP-1
740 IFLEFT$(A$(K),1)=S$THENA$(K)=RIGHT$(A$(K),I-1):NEXTI
750 NEXTL:X=LEN(A$(Z)):GOTO900
760 INPUT"FIRST LINE TO DELETE";F:IFF<0THENF=0 'delete
770 INPUT"LAST LINE TO DELETE";Z:IFZ>LATHENZ=LA
780 IFF>ZTHEN80ELSEFORI=FTOZ:A$(I)="":NEXTI:GOTO1830
790 INPUT"EDIT LINE";L:IFL<0ORL>LAORA$(L)=""THEN80 'edit
800 C=4:P=1:X$(0)=A$(L):N$="Y"
810 CLS:I=L:GOSUB1910:N=1:Q$=""
820 GOSUB910:IFA>47ANDA<58THENQ$=Q$+A$:N=VAL(Q$):GOTO820
830 M=0:IFA=8THENY=-1:GOSUB940ELSEIFA=9THENY=5:GOSUB940ELSEIFA=UTHENY=1:GOSUB940
840 IFA=65THENA$(L)=X$(0):GOTO800 'a
850 IFLEN(A$(L))>=LLTHEN870
860 IFA=29THENGOSUB1200ELSEIFA=30THENGOSUB1180
870 IFA>66THENONA-66GOSUB960,1000,1960,1960,1960,1020,1030
880 IFA=83THENGOSUB1120ELSEIFA=88THENGOSUB1170ELSEIFA=76THENGOSUB800
890 IFM=1THENN=1:Q$="":GOTO820ELSEIFRTHENPRINT@400,;ELSE810
900 IFLL<XTHENPRINT"LINE";L;"HAS";X;"CHARACTERS":GOTO60ELSE1830
910 X$=MID$(A$(L),P,1)
920 PRINT@C,C$;CHR$(02);:A$=INKEY$:PRINT@C,X$;:IFA$=""THEN920
930 GOSUB290:X=LEN(A$(L)):IFA=13THENR=1:RETURNELSERETURN
940 M=1:FORI=1TON:P=P+Y:IFP>XTHENP=X:RETURN
950 IFP<1THENP=1:RETURNELSEC=C+Y:NEXTI:RETURN
960 Q=P:D=C:FORI=1TON:GOSUB910:IFRORA=27THENP=Q:C=D:RETURN 'c
970 PRINT@C,A$;:GOSUB1100:P=P+1:GOSUB1110:A$(L)=L$+A$+R$
980 A=U:C=C+1:IFP<=XTHENNEXTI
990 P=Q:C=D:RETURN
1000 IFP+N-1>XTHENN=X-P+1 'd
1010 GOSUB1100:Q=P:P=P+N:GOSUB1110:A$(L)=L$+R$:P=Q:RETURN
1020 GOSUB1100:A$(L)=L$+S$:PRINT@C,B$ 'h
1030 GOSUB910:IFRORA=27THENRETURN 'i
1040 IFA=28THENA$(L)=A$(L)+CHR$(17):R=1:RETURN
1050 IFA=31THENA$(L)=A$(L)+CHR$(20):R=1:RETURN
1060 PRINT@C,A$;:IFA=8THENY=-1:GOSUB940:GOTO1030
1070 IFA=9THENY=1:GOSUB940:GOTO1030ELSEIFP>XTHENX=P
1080 GOSUB1100:GOSUB1110:A$(L)=L$+A$+R$:PRINT@C,B$;A$+R$
1090 C=C+1:P=P+1:GOTO1030
1100 L$="":IFP<2THENRETURNELSEL$=LEFT$(A$(L),P-1):RETURN
1110 R$="":IFP>XTHENRETURNELSER$=RIGHT$(A$(L),X-P+1):RETURN
1120 GOSUB910:Q=P:D=C 's
1130 FORI=1TON:F=0:FORJ=Q+1TOX:D=D+1
1140 IFMID$(A$(L),J,1)=A$THENF=1:Q=J:J=X
1150 NEXTJ:NEXTI:IFFTHENP=Q:C=D
1160 A=U:RETURN
1170 A$(L)=A$(L)+S$:P=X+1:C=P+3:GOTO1030 'x
1180 A$(L)=STRING$((LL-LEN(A$(L)))/2,32)+A$(L)+CHR$(17) 's-d
1190 PRINT@C,B$;LEFT$(A$(L),LEN(A$(L))-1);CHR$(92);:RETURN
1200 A$(L)=STRING$(LL-LEN(A$(L))+1,32)+A$(L)+CHR$(13) 's-r
1210 PRINT@C,B$;A$(L);:RETURN
1220 CLS:PRINT"LINE LENGTH =";LL,:INPUT"NEW =";LL 'format
1230 PRINT"LINE SPACES =";S,:INPUT"NEW =";S
1240 PRINT"LINE NOS. = '";N$;"'",:INPUT"NEW (Y/N)";N$
1250 PRINT"FIRST LINE =";FL,:INPUT"NEW =";FL
1260 PRINT"LEFT MARGIN =";LM,:INPUT"NEW =";LM
1270 PRINT"PAGE LENGTH =";PL,:INPUT"NEW =";PL
1280 PRINT"PAGE NOS. = '";PN$;"'",:INPUT"NEW (Y/N)";PN$
1290 PRINT"FIRST PAGE =";FP,:INPUT"NEW =";FP
1300 PRINT"PAGE 1 NO. = '";P1$;"'",:INPUT"NEW (Y/N)";P1$
1305 PRINT"CHAR/INCH =";G,:INPUT"NEW (5,10,16.5) =";G
1307 PRINT"LINE/INCH =";H,:INPUT"NEW (6,8) =";H
1310 PRINT"HEADING = '";H$;"'    ",:INPUT"NEW =";H$:GOTO60
1320 CLS:PRINT"LEGAL COMMANDS ARE:":PRINT:PRINT 'help
1330 PRINT"A  ADD","B  BLANK","C  COMPILE","D  DELETE","E  EDIT","F  FORMAT","H
 HELP","I  INSERT","J  JUSTIFY","K  KILL","L  LOAD"
,"M  MOVE","P  PRINT","R  REPLACE","S  SAVE","V  VIDEO","X  EXIT","T  TABLE OF C
ONTENTS"
1340 PRINT:PRINT:PRINT"PRESS <ESC> KEY TO RETURN FROM A,E,I,R COMMAND MODE":GOTO
60
1350 INPUT"INSERT AT LINE";L:IFL<0ORL>LATHEN80 'insert
1360 IF NL=LA+1THENPRINT"FILE FULL":GOTO60ELSEIFRTHEN60
1370 FORI=LATOLSTEP-1:A$(I+1)=A$(I):NEXTI
1380 A$(L)="":LA=LA+1:L=L-1:IFITTHENRETURNELSEIT=1:GOTO90
1390 CLS:PRINT"JUSTIFYING":FORL=0TOLA:X=LEN(A$(L)) 'justify
1400 IFX<2THEN1500ELSEFORI=XTO2STEP-1:A=ASC(RIGHT$(A$(L),1))
1410 IFA=UTHENA$(L)=LEFT$(A$(L),I-1):X=X-1:NEXTI
1420 IFX>=LLORA=17ORA=20THEN1500ELSEJ=0:K=1:FORI=1TOX
1430 IFMID$(A$(L),I,1)<>S$THENK=0ELSEIFK=0THENK=1:S(J)=I:J=J+1
1440 NEXTI:IFJ=0THEN1500
1450 K=RND(J)-1:IFINT(J/2)=J/2ORJ=1THENN=1ELSEN=2
```

Program continues



✓See List of Advertisers on page 306

"Many good word processors are available for the Model I... but ...require a substantial outlay of cash."

Escape key	Ends adding session.
Left arrow	Ends, adds "do not justify" marker to present line.
Tab key	Moves cursor five spaces to right.
Backspace	Backspaces and erases last entry.

Though you can press Enter to end a line, the computer will automatically end it when you type the specified maximum number of characters per line. The computer shifts the whole word, if it extends beyond the maximum number of characters, to the following line.

The Blank command deletes all "no character" lines from the text. A space is a character. If you want a blank line in the text, say between your paragraphs, simply put one space on that line. This places a character (ASCII 32) on that line, and the Blank command will not touch it. As you will find out, this is a very useful command.

The Compile command will ask you the first and last line number that you wish to compile. It then shifts words from line to line, fitting the maximum number of words onto each line. Spaces at the beginning and end of all lines but the first are deleted. (It does not touch paragraph indentation on the first line.) For this reason Compile is best used on only one paragraph at a time. One space is inserted after each word, comma, and semicolon. Two spaces follow colons, periods, question marks and exclamation marks. Compiling only affects spaces after words and punctuation shifted between lines. For this reason, it should be used before the justifying command, or spaces added by justifying will be unaffected.

Use the Delete command to remove lines from the text. Delete leaves a blank line in place of the removed one. You can delete any number of consecutive lines at one time.

Now we get to the heart of the word processor: the Edit command. The edit mode works on one line at a time. Changes can be made with the following subcommands:

Tab key	Moves cursor five spaces to the right.
n(Backspace)	Left move of n spaces.
n(C)	Change next n characters to next n characters entered.
n(D)	Delete next n characters.
(X)	Extend the line.
(H)	Erase from present position in line to the end and enter extend mode.
(I)	Insert at the present position.
(L)	List the line and keep modifications.
(A)	Cancel previous changes and start over.
n(S)c	Search for the nth occurrence of character c.
Escape key	Quit H, I, X modes.
Right arrow	Moves line all the way right, ends edit.
Up arrow	Centers line and adds "do not justify"; ends editing.
Left arrow	Adds "do not justify"; ends editing.
Enter key	Saves all changes and ends editing.

The "n" above indicates the number of characters wanted. N is set at one if not entered. The left arrow key puts a "do not justify" character at the end of the line, and, as such, you must enter the extend mode first to add it to the line. Any editing changes before listing the line will be made permanent. Therefore, if you hit L, then decide you really didn't want to make a certain change, hitting A will not erase it. Most of the single character commands listed above are basically the same as the TRS-80 Model II editing commands.

The Format command allows you to set up the various format parameters of the letter and printer. They are self-explanatory:

Number characters/lines
Number spaces between lines
Line numbers (Y/N)
First line number
Left margin indentation amount
Number lines/page
Page numbers (Y/N)
First page number
Numbers on page one (Y/N)
Heading name
Printer characters/inch
Printer lines/inch

There is a Help command to assist you when you are inebriated or just plain too tired to recall the program's legal commands. It simply lists all the commands.

Insert is used to add those forgotten lines. After specifying the first line, the program goes into the add mode. All legal add mode subcommands are valid. Hitting the escape key will end the insert mode.

The Justify command right justifies the text. Any lines which don't have a "do not justify" or "end of page" marker will be right justified. Indentations at the beginning of lines are not touched. This command should be used after all compiling and editing is done.

Kill deletes the entire text on which you are working. It can also delete text that was previously stored on disk. A double check is

"A double check is provided to prevent killing something mistakenly."

```
1460 FORI=1TOLL-X:T(K)=T(K)+1:K=K+N:IFK)J-1THENK=K-J
1470 NEXTI:FORI=J-1TO0STEP-1:A$=STRING$(T(I),S$):T(I)=0
1480 A$(L)=LEFT$(A$(L),S(I))+A$+RIGHT$(A$(L),LEN(A$(L))-S(I))
1490 NEXTI
1500 NEXTL:GOTO1830
1510 CLS:INPUT"REALLY KILL (Y/N)";A$:IFA$()"Y"THEN60 'kill
1512 INPUT"KILL PRESENT LETTER (Y/N)";A$:IFA$="Y"THENRUN
1514 INPUT"KILL ANY LETTER ON DISC (Y/N)";A$:IFA$()"Y"THEN60
1516 INPUT"NAME OF LETTER ON DISC TO KILL";A$:KILLA$:GOTO60
1520 Z$="TABLE":GOTO1524
1522 INPUT"WHAT LETTER DO YOU WANT";Z$
1524 CLS:PRINT"LOADING FROM DISC: ";Z$
1525 OPEN"I",1,Z$
1530 INPUT#1,LA,LL,S,N$,FL,LM,PL,PN$,FP,P1$,H$
1540 FORI=0TOLASTEP4:INPUT#1,X$(0),X$(1),X$(2),X$(3)
1550 FORJ=0TO3:L=I+J:X=LEN(X$(J)):A$(L)="":IFX(1THEN1570
1560 FORK=1TOX:A$(L)=A$(L)+CHR$(ASC(MID$(X$(J),K,1))-128):NEXTK
1570 NEXTJ:NEXTI:CLOSE:GOTO60
1580 INPUT"FIRST LINE TO MOVE";F:IFF(0THENF=0 'move
1590 INPUT"LAST LINE TO MOVE";Z:IFZ)LATHENZ=LA
1600 IFF)ZTHEN80ELSEINPUT"FIRST NEW LINE";N:FORI=FTOZ
1610 IFLEN(A$(N))THENPRINT"LINE";N;"NOT EMPTY":GOTO60
1620 A$(N)=A$(I):A$(I)="":N=N+1:IFN)LATHENLA=N
1630 NEXTI:GOTO1830
1640 X=FP:M=FL:GOTO1661 'print
1645 IF ERR=5THENPRINT"YOU CAN'T DO THAT!!!!!":FORQ=0TO4000:NEXTQ:RESUME60
1647 IFERR=53THENPRINT"THAT FILE DOES NOT EXIST!!!":RESUME60
1650 IFERR()56THENONERRORGOTO0ELSEINPUT"PRINTER NOT READY.  ABORT (Y/N)";A$
1660 IFA$="Y"THENRESUME60ELSEINPUT"FIX PROBLEM AND PRESS ENTER";A$:RESUME1640
1661 IFG=5THENW=31ELSEIFG=10THENW=30ELSEIFG=16.5THENW=29
1662 IFH=6THENT=54ELSEIFH=8THENT=56
1665 INPUT"WANT TO ADJUST TO TOP OF PAGE (Y/N)";A$:IFA$="Y"THENSYSTEM"FORMS T"
1670 INPUT"WANT TO CHECK FORM, #'S, ETC. (Y/N)";Z$:IFZ$="Y"THEN1220ELSER=1:GOSUB
480:CLS:PRINT"PRINTING"
```

Program continues

provided to prevent killing something mistakenly. If you ask for a letter or text that is not stored on the disk, you will not get a "file not found" error; instead, you will be told "that letter does not exist" and the program will continue.

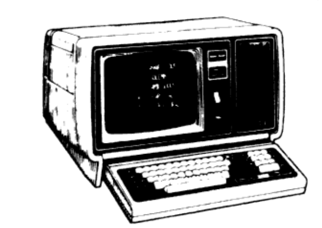
Like the Kill command, the Load command is protected against loading a non-existent text. It simply loads a specified text from the disk into the word processor.

Move is a very useful command which allows you to move as many lines as required from one position to another within the body of a letter. All necessary prompting is given to allow this command to be used.

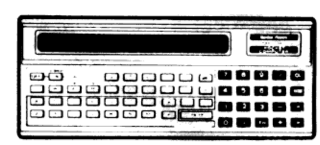
The Print command actually puts your work on paper. It asks first if you would like to adjust to the top of the page. It then asks if you would like to double-check the forms to be sure the line numbers are off. If you are printing more than one page, the program stops and asks you to press Enter when ready to print the next page (X). This allows you time to put in new paper before going on.

Replace is the command that replaces a

Here's a REAL PLUS

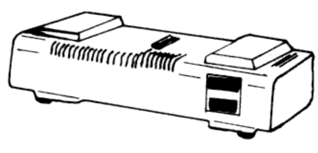


Model II 64K $3300

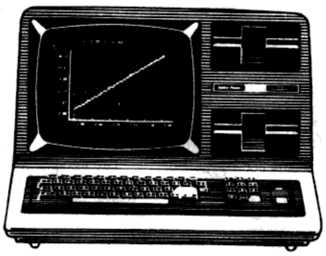


Pocket Computer $189

Color Computer 4K $310
w/16K Ext. Basic $489



32K Exp. Int. $362



Model III 16K $859

**Here are just a few of our fine offers . . .
call toll-free for full information.**

COMPUTERS

Model II 64K	$3300
Model III 4K LEV I	599
Model III 16K	859
Model III 32K	981.50
+ Model III 32K	915.50
Model III 48K	1104
+ Model III 48K	972
Model III 32K 2 Disc & RS232 c	2100
Color Computer 4K	310
Color Computer 16K	432.50
+ Color Computer 16K	366.50
Color Computer 16K w/extended basic	489
Pocket Computer	199
VIDEOTEX	310
APPLE 48K only	1279
ATARI 800 32K	789

+ Computer Plus New Equipment.
180 Day Extended Warranty

PERIPHERALS

Expansion Interface OK	$249
Expansion Interface 16K	359
+ Expansion Interface 16K	305.50
Expansion Interface 32K	469.95
+ Expansion Interface 32K	362
16K RAM N.E.C. 200 N.S. chips	39

MODEMS

Lynx Direct Connect	219
COMM 80 Interface	159.95
Chatterbox Interface	239
Telephone Interface II	169
R.S. Modem I D.C.	130
Atari 830 Modem	159
Atari 850 Interface	183

PRINTERS

Line Printer IV	849
Daisy Wheel II	1695
Line Printer VI	999
Epson MX80	499
Line Printer VII	315
Line Printer V	1610
Epson MX80 FT	599
Atari 825	
Pocket Computer Printer	130
Centronics 737	737

DISK DRIVES

R.S. Model III 1ST-Drive	712
TEAC 40 Track MI	319
R.S. 1 Drive Exp MII	999
R.S. 2 Drive Exp MII	1518
R.S. 3 Drive Exp MII	2040
Seimans 8″ MII	799
Atari 810 Disk	499

SOFTWARE

R.S. Software 10% off list	
Atari Software 10% off list	
Newdos + (40) track MI	88
Newdos 80 MIII	149
ST80III	149

ETC.

Verbatum 5″ Double Density	32
Verbatum 8″ Data Life	49.95
Ctr-80A recorder	52
C.C Joysticks	22

PLUS real back-up warrantees —

Factory warrantees on Apple and Atari equipment. Other equipment carries manufacturer's warranty or Computer Plus 180 day extended warranty. Combined warrantees carry Computer Plus 180 day warranty or original manufacturer's warranty.

DEALER INQUIRIES ARE INVITED

**Prices subject to change without notice.
Not responsible for typographical errors.**

TRS-80 is a registered trademark of Tandy Corp.

call TOLL FREE 1-800-343-8124

computer plus

Dept. D ✓130
245A Great Road
Littleton, MA 01460
617-486-3193

Write for your free catalog

"The Blank command deletes . . . lines from the text . . . As you will find out, this is . . . very useful."

whole line at a time. You specify what line you'd like to replace and then replace it.

You store a letter on disk with the Save command. Before you store it, you're prompted to save an updated version of the disk's table of contents.

Video displays the present text just as it will appear when printed. Many times, for instance after changing a parameter, you'll want to see the entire text.

Exit allows you to escape from the word processor program gracefully. The exact method used will be up to you. You may want to simply say end or stop on line 1980.

Table of contents was added as an afterthought and has proven very helpful. I added the table of contents to list all letters saved. That way you don't need to hit Break, System directory to find a letter. Entering T will get you the table of contents from the disk, which is just a letter named table stored on the disk.

I've set up this system with a Microline printer and Radio Shack Line Printer III. Undoubtedly other modifications to the print program may be necessary to accommodate other printers.■

```
1680 LPRINTCHR$(W),CHR$(V);CHR$(T);:IFPN$()"Y"OR(P1$="N"ANDX=1)THEN1700
1690 LPRINTTAB(LM)H$;TAB(LL+LM-7)"Page";USING"###";X:LPRINT
1700 FORP=MTOM+PL-1:IFP)LATHEN1740
1710 M=M+1:IFSTHENLPRINTSTRING$(S,10)
1720 LPRINTTAB(LM);:IFN$="Y"THENLPRINTUSINGF$;P;
1730 LPRINTA$(P):IFASC(RIGHT$(A$(P),1))()20THENNEXTP
1740 IFP)LATHEN60ELSESYSTEM"FORMS T":PRINT"PAGE";X+1;"READY, IF SO PRESS ENTER";
:INPUTE:X=X+1:GOTO1680
1750 INPUT"REPLACE LINE";L:IFL(0ORL)LATHEN80 'replace
1760 R=1:A$(L)="":L=L-1:GOTO90
1770 TC$="":INPUT"SAVE UPDATED TABLE OF CONTENTS (Y/N)";TC$:IFTC$="Y"THENZ$="TAB
LE":GOTO1773  ' save
1772 INPUT"PLEASE NAME THIS LETTER";Z$
1773 CLS:PRINT"SAVING PRESENT LETTER ON DISK AS: ";Z$
1775 OPEN"O",1,Z$
1780 PRINT#1,LA","LL","S","N$","FL","LM","PL","PN$","FP","P1$","H$:FORL=0TOLASTE
P4
1790 FORJ=0TO3:I=L+J:X=LEN(A$(I)):X$(J)="":IFX(1THEN1810
1800 FORK=1TOX:X$(J)=X$(J)+CHR$(ASC(MID$(A$(I),K,1))+128):NEXTK
1810 NEXTJ:PRINT#1,X$(0)","X$(1)","X$(2)","X$(3):NEXTL:CLOSE:GOTO60
1830 CLS:X=FP-1:FORM=FLTOLASTEPPL:X=X+1 'video
1840 IFP1$="N"ANDX=1THEN1860
1850 IFPN$="Y"THENPRINTH$;TAB(LL-7)"Page";USING"###";X:PRINT
1860 FORI=MTOM+PL-1:IFI)LATHEN1890
1870 IFSTHENPRINTSTRING$(S-1,10)
1880 GOSUB1910
1890 NEXTI:A$="":IFI(=LATHENINPUT"PRESS ENTER";A$:IFA$()""THENM=LA
1900 NEXTM:L=LA:GOTO60
1910 Y=LEN(A$(I)):IFYTHENA=ASC(RIGHT$(A$(I),1))ELSEA=0
1920 IFN$="Y"THENPRINTUSINGF$;I;
1930 PRINTA$(I);:IFA=17THENPRINTCHR$(92);
1940 IFA=20THENPRINTCHR$(93);ELSEIFA=UTHENPRINTCHR$(95);
1950 IFN$()"Y"ORY()80THENPRINT
1960 RETURN
1970 CLS:INPUT"REALLY EXIT (Y/N)";A$:IFA$()"Y"THEN60 'exit
1980 CLS:CLEAR500:PRINT"THE END":LOAD"INLAND",R
```


✓See List of Advertisers on page 306

GRAPHIC DIVERSIONS!

3-D Adventures

Our 3-D adventures combine the traditional word oriented adventure with full screen 3-D perspective graphics! Hallways, stairs, elevators, objects, and even characters in Asylum, are represented in 3-D graphics *as though you are actually there!*

". . . truly one of the best adventure games by any criteria. . . it borders on the unbelievable. Believe it. Buy it." M.J.M., *The Space Gamer*

"This has got to be the most infuriating, irritating, aggravating, frustrating, angering, spellbinding game on the market." D.M., *'80 Microcomputing*

ASYLUM places you in the residence home for Deathmaze survivors. To leave, you must deal with guards, fellow survivors, doctors, the infamous crazed carpenter, and much, much more.

TRS-80 Level II 16K or Model III 16K	$14.95
Model I DISK	$19.95

LABYRINTH places you in a huge maze of tunnels inhabited by gnomes, ghosts, witches, and an evil minotaur. You must find the weapons and treasures needed to destroy the minotaur before he destroys you!

TRS-80 Level II 16K or Model III 16K	$12.95
Model I DISK	$17.95

DEATHMAZE 5000 places you in a gigantic five-story building. There is only one goal. ESCAPE ALIVE! Monsters, dogs, vampires, and other vile horrors will plague your every step as you struggle to survive one of the most challenging adventures ever written.

TRS-80 Level II 16K or Model III 16K	$12.95
Model I DISK	$17.95

GRBASIC

GRBASIC extends Level II or DISK BASIC to include an easy to use graphics command set. A single BASIC command allows the user to draw a line between any two pixels on the screen in hundredths of a second! Coordinates can be chained to allow complex figures to be drawn by a single BASIC program line in less than a second!

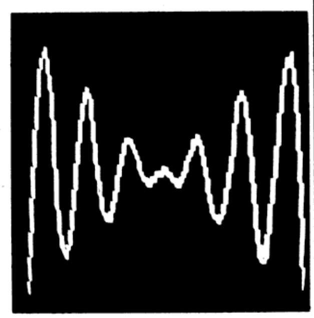
GRBASIC allows the definition of shapes. Once defined, a shape can be **rotated, scaled** up or down in size, drawn anywhere on the screen in less than a second, and can even be drawn totally or partially "off" the screen in extended space! And all with short, simple BASIC commands! Even multiple shapes are no problem!

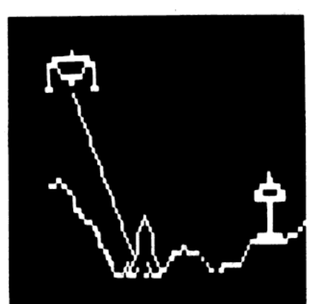
GRBASIC is not a string-packer or machine-language USR-called utility. GRBASIC is fully integrated into Level II and DISK BASIC. There is nothing on the market that offers its graphic features, except possibly the Apple II computer! Animation, scientific plotting, and professional data displays are now child's play!

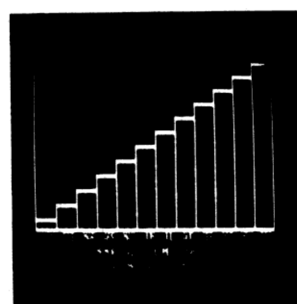
A new program, GRBASIC FUNCTION PLOTTER, allows the plotting of almost any function, including polar coordinate based figures and almost any wave form. Features include function definition and automatic screen scaling. REQUIRES GRBASIC!

GRBASIC cassette	$19.95
GRBASIC TRS-DOS disk	$24.95
GRBASIC NEWDOS80 disk	$24.95
GRBASIC FUNCTION PLOTTER	$14.95

Examples of graphics produced with GRBASIC







Med Systems Software
P.O. Box 2674-S Chapel Hill, NC 27514
(919) 933-1990 ✓128

MICRO-WORLD

JOURNEY TO THE CENTER OF THE TRS-80!

Micro-World is an adventure like no other. You are transformed into an electroid, and must travel through the circuits of your TRS-80! You will be required to solve many incredible problems, and in the process you will gain a knowledge of how a TRS-80 operates.

Micro-World is one of the most advanced pieces of machine language programming Med Systems has published. A special encoding scheme has allowed a 21K adventure to fit in 16K. Micro-World is verbose. Messages are frequent and fact filled. There are over 80 locations that must be explored.

Micro-World is an excellent educational simulation! It is supplied with a booklet containing a glossary and explanations of the electronic circuits inside the TRS-80. This does not in any way diminish the challenge! If anything, the challenge increases since you must gain a working knowledge of your computer to gain access to the final circuit!

Model I Level II 16K or Model III 16K cassette	$19.95
Model I disk	$24.95

The Playful Professor

The Playful Professor has been a constant best-seller since its introduction in 1980. This program is a mathematical tutorial that provides instruction in addition, subtraction, multiplication, and division, with or without fractions. Problems are presented in a game format that gives a step-by-step tutorial for incorrect answers.

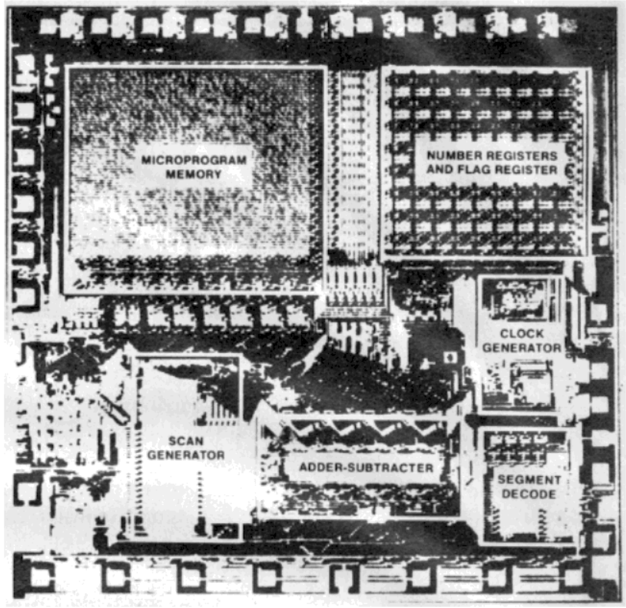
The Playful Professor places the user in a 60 room mansion haunted by an intelligent ghost who holds the key to the only door out. Options include 3 difficulty levels, choice of problem type, 1 or 2 players, pass option, and split difficulty levels to allow 7 year olds to play competitively with 27 year olds! Graphics are used extensively.

"I'm so impressed with The Playful Professor, Money Master, and Deathmaze. At the Mead School we use these three programs daily." R.J., Greenwich, Conn.

"My 7 year old daughter had a fun way to review her math . . . my 4 year old enjoys playing using the pass feature (what an experience it is to watch a 4 year old use a computer!)" S.C., Highland, Maryland

Model I or Model III Level II 16K	$12.95
Model I DISK	$16.95

Med Systems Software
P.O. Box 2674-S Chapel Hill, NC 27514
(919) 933-1990



SATISFACTION GUARANTEED!

If for any reason you are not satisfied with our products, return your order within 14 days for a prompt and cheerful refund.

ORDERING INFORMATION

Orders are processed within five working days. We pay all postage and handling within the U.S., Canada, and U.S. territories. European orders please include $3.00 for air post.

☐ Asylum	($14.95)	$ ______
☐ Deathmaze 500	($12.95)	$ ______
☐ Labyrinth	($12.95)	$ ______
☐ Adventures on Disk	(add $5.00)	$ ______
☐ GRBASIC Cassette	($19.95)	$ ______
☐ GRBASIC TRS-DOS Disk	($24.95)	$ ______
☐ GRBASIC NEWDOS 80 Disk	($24.95)	$ ______
☐ GRBASIC FUNCTION PLOTTER	($14.95)	$ ______
☐ Playful Professor	($12.95)	$ ______
☐ MICRO-WORLD	($19.95)	$ ______
☐ MICRO-WORLD Disk	($24.95)	$ ______
	Total	$ ______

Name ______________________
Street ______________________
City ____________ *State* ____________ *Zip* ______
Computer:
☐ TRS-80 LII 16K ☐ Model III 16K
☐ *Mastercard* ☐ *Visa* ☐ *check*
MC or VISA # ______________________
Expiration Date ______________________

✓128

Archbold's Mod I Speed-Up Kit

Model 1 Speed-Up
Archbold Electronics
Rancho Cordova, CA
$37.50

Richard C. McGarvey
221 Hirschfield Drive
Williamsville, NY 14221

Every computerist wants to do his computer work as fast as possible. Efficient programs are a good way to speed up your work. Compressing code or using faster code such as machine language also helps speed up your computer's operation. There are other tricks, but there is always the ultimate limitation of the clock speed of your computer. No matter how efficient your code or how fast your accessories, you can't beat the built-in limitation of a slow clock speed.

In the case of the TRS-80 Model I, that clock speed is based on the Z-80 CPU. The Z-80 is rated at two MHz but can function safely at about three MHz. Tandy, however, decided that for the best cassette I/O speed (and whatever other reasons) to limit the speed of the Model I to approximately 1.77 MHz. (I say approximately because slight differences between machines exist.) This is built-in slow down and not an insurmountable problem, since the computer is not operating at its maximum efficiency.

No Electronic Genius

I recently purchased a speed-up kit and installed it. I'm no electronic genius. In fact, I have trouble soldering, let alone accomplishing anything major. For that reason I have put off making any changes to the computer that require hardware modifications on a do-it-yourself basis. But I've had this computer for several years now and I get bolder as time goes on, so I decided to give it a shot. Here are the results of that attempt (written on my operating and high speed TRS-80).

The First Hurdle

When I order by mail I like to receive my order in a reasonable length of time. I usually supply a credit card number to avoid the delay of processing personal checks, but with the recent increase in interest rates I have vowed never again to charge. For this reason I went to the bank and purchased a money order so the speed-up kit could be sent without delay.

After four weeks I got nervous. After six weeks I wrote a letter to find out where my kit was. Eight weeks later to the day, my kit showed up in the mail. I don't know if there were any special problems, but eight weeks is a bit too long.

The Second Hurdle

The documentation leaves something to be desired. It includes 16 pages of instruction (the length is due to the different types of logic boards in the TRS-80). There is also a two page addendum which has numerous changes to the instructions. From experience I can tell you this: Read through the corrections and make them on the original instruction manual whether or not they apply to your board. This will help prevent disastrous errors later on. There is also a full-size photo of the computer's logic board that is marked for easy location of every connection and trace.

Program Names	Normal Speed	Minimum Speed-up
Busy Work	13 min. 27 sec.	5 min. 3 sec.
ANOVA	——— 41 sec.	——— 20 sec.
Descriptive Stats	——— 9 sec.	——— 7 sec.
For/Next Loops	1min. 19 sec.	——— 40 sec.

Table 1. Time Comparisons of Some Programs

The documentation is a little confusing. Read and understand it before you start. This is not the type of project you just jump into and then back up to correct mistakes. *Do it right the first time!* If you are not familiar with soldering techniques, printed circuit boards and ICs, you should consider having the kit installed by someone who knows what they are doing. It's not impossible for a novice to install the kit; I did it, but it was frustrating as hell. I could install the kit in about an hour now. At the time, it took me four hours of cursing and wondering.

The Kit

The Speed-up kit is composed of a small circuit board that is extremely well made. (See Photo 1 which shows the speed-up board as well as all of the accessory parts.) A twenty wire ribbon cable extends from the speed-up board, and these twenty wires are connected to various points on the computer logic board. In the kit box are a few extra parts: one diode, one LED, one capacitor, one length of wire, a connector/coupler and one length of very fine solder. You may or may not need these parts, depending on board type and whether or not you have an expansion interface and disk drives.

I'm sure you've seen the ads for the Archbold speed-up kit in *80 Microcomputing*. They are a little misleading. The ad doesn't mention that in order to pump up the speed of your computer to the maximum six MHz, you will need a Z-80B CPU to replace the one in your computer. The Z-80 is fine (in most cases) for the three MHz conversion, but you will most likely run into trouble if you go faster. If your Z-80 won't handle the 100 percent increase, up to a $27 investment in a Z-80A will fix the problem. On the other hand, the Z-80B is rated at six MHz which makes it *very* fast compared to the Z-80, and the cost is around $30. Also note that peculiarities between computers may give you

"So far I have found only one operation problem in the . kit modification: keyboard debounce."

better or worse results and you must remember that all of the above times are subject to those differences.

They also fail to mention that in order to make the Z-80B operative you need a delay line. You can purchase it from Archbold or from a local supply store—either way it runs about $20. Also, if you have an expansion interface, you need a delay line for that too. Now you are up to about a $55 increase over and above the price of the kit.

Installation

The installation is not too difficult, but I would like to give potential users a few tips.

1) Be careful soldering. Too much heat is a no-no. Also, too much solder can cause a run or even a thread to bridge between two points that are not supposed to be connected. That will not cause any permanent damage, but the computer won't work and those errors are a real pain to locate.

2) Be extremely careful when you fold the keyboard unit away from the logic board. The connector is sensitive and rough handling can cause an expensive break.

3) Be neat! Little bits of solder or wire in the unit may sit there for a long time. A jolt to the desk or keyboard, and the computer is suddenly down. If you take it to a repair shop they will dust it off and charge you an arm and both legs as well as your first born child.

4) Have all the necessary tools ready when you start to work. You need to be orderly when working with systems as complicated as printed circuit boards.

5) Remember that your computer uses very little power. For that reason there is very little excess power available. When you add accessories that use computer power, you are straining the limits of the computer's regulator. Don't make 300 modifications that all use computer power or the computer will crawl away and die. Use only those that are most important to you.

Does It Work?

The kit does work and you can power up in either high or low speed (a simple modification to the kit board) and then change it to the opposite speed with a simple OUT 254,1 (or 0, depending on which way you want to go). The LED that is included changes color to indicate what mode you are in. Red is normal, green is high speed and yellow is 50 percent slower than normal (50 percent below normal is switch controlled). Also, the board automatically reduces speed when cassette or disk I/O is performed, and then returns to high speed (if that was selected) when the I/O is complete. Note that if manual switching is used, cassette automatic down speed will no longer occur.

If manually switched into high speed, disk access will cause automatic slow down, but cassette access will require an OUT instruction (or manual switching) to slow the computer. This manual switching makes it possible to run 50 percent slower than normal or high speed. Switching can be done during program operation with no interference in the program's performance; however, the extra work for the minimum benefit seems to me to be a waste of time. The switch is not included in the kit but is readily available at your local supply store.

I tested the speed-up with several programs. See Table 1 for the results for comparison. You will note a substantial increase in operating time between the normal times and the speed-up times due to my 100 percent increase. Not all the speeds are twice as fast, but that is due to the type of operation the computer is doing. The longer programs show 100 percent or better increase in operating time.

Operational Problems

So far I have found only one operational problem in the speed-up kit modification: keyboard bounce. I have the old style Level II, so I need a debounce program. DOS as well as Scripsit have a debounce routine built in. Level II provides numerous debounce routines. The problem is that debounce is accomplished by introducing a time delay in the keyboard scan. That means that if a key double strikes, the second strike will occur before the delay is complete and the keyboard scan is reimplemented. Therefore, the second key strike (bounce) is ignored.

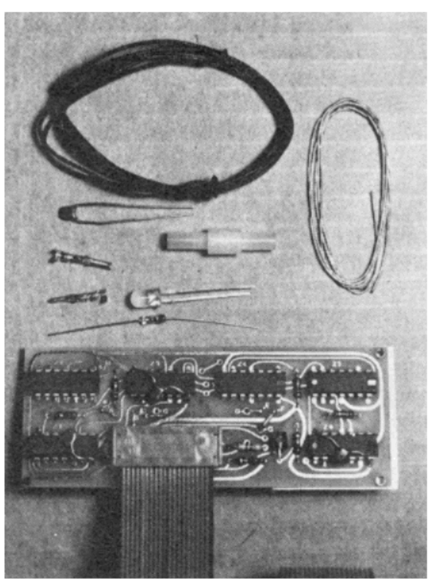


Photo 1. The Archbold Speed-up kit with all accessories.

✓See List of Advertisers on page 306

"The speed-up manual states that the Archbold speed-up is the best... the manual doesn't lie."

Well, when you speed up the operation you shorten the debounce delay. As a result, I have noticed an occasional keybounce. At my speed-up rate it is not bad, and cleaning the keys seems to have eliminated it altogether. I do see a problem for those who jump up to six MHz, because the delay will be so short as to be totally ineffectual. If you know your debounce program you can increase the delay and solve the problem. If not, you will be stuck! With no debounce and the high speed mod it is likely that the key bounce problem will be magnified. (My experience and testing verifies this.)

There is a bright side to this point that might interest users of Level II with the new ROM based debounce. I have heard complaints that this built-in debounce is too slow, and fast typists can easily outrun it. In that case the high speed will probably make the ROM based debounce a little easier to deal with.

Unexpected Goodies

I have a disk system and I very seldom use my cassette at all. There are, however, a few games and applications programs that are not disk compatible, so occasionally I run the cassette. After installing the Archbold kit, I wanted to see how it affected an Invaders game. Well, the game ran super fast and posed a real challenge, but I noticed something strange. The cassette loaded the first time with no error. That surprised me because that particular cassette usually took several attempts to load. In fact, I thought that there was a problem with the first program copy on that cassette because it always gave me a checksum error. I usually had to load the second copy and often that didn't work either.

This bit of good fortune started me thinking, so I pulled out some other difficult to load cassettes in both BASIC and System. They all loaded the first time! I won't guarantee that the same will happen in your case, but I suspect that when the speed-up kit is in the normal mode it actually runs a bit slower than the original set-up. For that reason the data transfer is less likely to miss bits and less checksum errors will occur. I may be off base with this conclusion, but I can say that all my cassettes load. Even ones that I had given up for dead now load fine. Also I have noted that BASIC and System tapes load at the same volume.

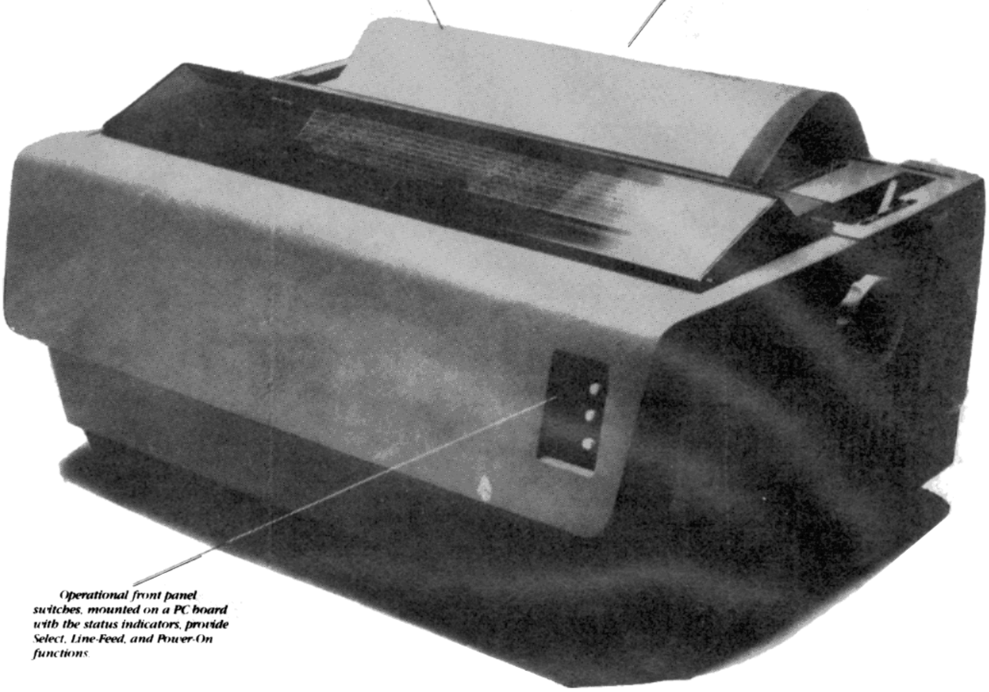
Conclusion

The speed-up manual states that the Archbold speed-up is the best available regardless of price. I haven't tried any of the others but I know people who have. Based on what they have said, the manual doesn't lie. I'm very satisfied with the increase speed of operation, and even though the installation was a bit tricky, it was well worth it. If you would like that extra speed, then buy Archbold—it works and it works right! ■

Note: In all fairness to Archbold Electronics I should point out that I ordered my kit from an old advertisement. The March Issue of 80 Micro *has a new ad which is more accurate and points out the Z-80A and B options. Also, the price is now up to $45.00.*

NEED A LETTER QUALITY PRINTER BUT DON'T WANT TO PAY THE PRICE?

Then American Business Computers Has Just The Printer For You!



The Starwriter Daisy Wheel Printer is designed to fill a distinct gap in the peripheral marketplace. A speed of 25 cps and its low price make it the ideal choice for today's systems designer interested in achieving superior price/performance ratios.

Incorporating the latest LSI technology, the Starwriter is built by one of the world's most respected computer peripheral manufacturers. The printer is furnished complete and ready-to-use, requiring no changes in hardware or software. Starwriter offers an industry standard parallel interface, as well as RS232-C interface. Total plug compatibility and a wide variety of interface matching capabilities help lower the system integration costs for OEMs and end-users.

Using a 96-character wheel, the Starwriter produces letter-quality printing on 3 sharp copies with up to 136 columns. Starwriter offers the highest degree of vertical and horizontal positioning, resulting in the most precise character placement in the industry. The easy-to-change character wheel also makes our printer a perfect choice for international applications. Compatible with sheet feeders, starwriter accommodates paper widths to 381 mm (15") and uses industry standard ribbon cartridges.

Along with a self-test capability and a programmable VFU (Vertical Format Unit), Starwriter furnishes the operator with all desirable status functions, commands and program selectable switches. Panel lamps indicate the current Paper, Select and Power status.

The easy plug-in compatibility of Starwriter and its outstanding print performance makes the Starwriter perfect for TRS-80*, Apple*, and S-100 users.

Starwriter is backed up by a one-year warranty (3 months on parts and labor, 9 succeeding months on parts) and is further supported by a strong, fully trained and technically proficient service organization. The Starwriter is available for immediate shipment in any quantity desired.

For more information, call American Business Computer, 118 So. Mill Street, Pryor, Oklahoma 74361; Telephone 918-825-4844.

*TM of Tandy Corp
*TM of Apple Computer Corp.

AMERICAN BUSINESS COMPUTERS ✓529

118 So. Mill Street
Pryor, Oklahoma 74361
Telephone 918-825-4844

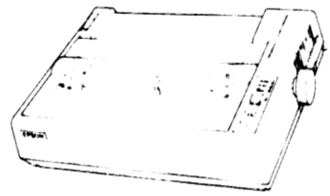
Dealer Inquiries Invited

BUSINESS COMPUTERS

master charge

*TM TANDY CORP.

THE MX-80 NOT ONLY DOES EVERYTHING, IT DOES EVERYTHING WELL.



This is the new Epson MX-80 dot matrix printer. It does just about everything you could ask a printer to do. Quickly. Quietly. Reliably. In fact, for OEM installations, the MX-80 may be the single best, all-round printer you can buy. But that's not the best reason to buy it.
The MX-80 prints bidirectionally at 80 CPS in a user-defined choice of 40, 80, 66 or 132 columns. And if that's not fast enough, its logical seeking function minimizes print head travel time. The MX-80 prints 96 ASCII, 64 graphic and eight international characters with a tack-sharp 9x9 matrix. For a long time. Epson printers are known for reliability and the MX-80 is no exception. But that's not the best reason to buy it either.
The print head has a life expectancy of up to 100 x 10^6 characters, and when it wears out, just throw it away. A new one costs less than $30 and the only tool you need to change it is attached to the end of your arm. The MX-80 is compact weighs only 12 lbs., and the whole unit, including the two stepper motors controlling carriage and paper feeding functions, is precisely controlled by an internal microprocessor. But even that isn't why you should specify the MX-80.
The best reason is this: because Epson makes more printers than anyone else in the world, we can afford to sell each one for a little less.

...Call For Unbelievably 'Low Price

525

OKIDATE MICROLINE 80 PRINTER - $479

The Microline is built on a rugged cast aluminum base to withstand the rigors of continuous business use. It is driven by two motors and will run all day with no duty cycle limitations. Microline printers use a dense 9 x 7 dot pattern to produce crisp, clean copies, first copy to last. The seven pins in the head are "fired" using energy stored in tension members. This technology permits the use of short, low mass pins made with an extremely hard alloy. The head produces less heat, thereby extending its life. 524

★ Check It Out! ★

EPSON MX-80FT

That's right — **MX-80FT.**

The **FT** on the end means Friction and Tractor. The Fantastic **MX-80** Printer is now available in a version which will accept letterhead-type paper **And** tractor-type paper. Call or write for our (as always) unbelievable low price.

527

AW...WHAT THE HECK

RAM Memory Chips for the TRS-80

It is the policy of American Business Computers to offer merchandise at the lowest price possible. Serveral months back we began selling RAM Memory Chips for the TRS-80 for $45.00 per set. Someone else sold chips for $44.00. We sold them for $38.00. They sold them for $37.95. So we say "AW ..... WHAT THE HECK!" Let's see the other guys beat this price.

$22

PER 16K SET

These chips are brand new "4116's". These 200 nanosecond chips are fully compatible with all TRS-80 produces. Instructions for insertion are included, however the dip shunts required for converting a 4K Model I to a 16K Model I are not included at this low price.

526

Epson.

This printer costs less than $450. just how much less we can't tell you - But if you will give us a call we think you will be pleasantly suprised.

This is the Epson MX-70. The lowest priced dot matrix printer you can buy. Now, that in itself should make it very attractive to a lot of people. But you ain't heard the half of it.

To begin with, the MX-70 has a lot more in common with our now-famous MX-80 than just the name. Like unequalled Epson reliability. And technological breakthroughs like the world's first disposable print head. But frankly, the MX-80 packs a lot more power than some people need. So Epson built the MX-70 to be a no-frills printer. At a no-frills price.

But the MX-70 is still a great little printer. They give you 80 CPS unidirectional printing. Top-of-form recognition. Programmable line feed and form lengths. Plain paper printing. An easy-to-read 5x7 matrix. Self test. And an adjustable tractor feed.

That's what you'd expect from a basic little printer. But here's something you wouldn't expect: the finest graphics package on the market today. Free.

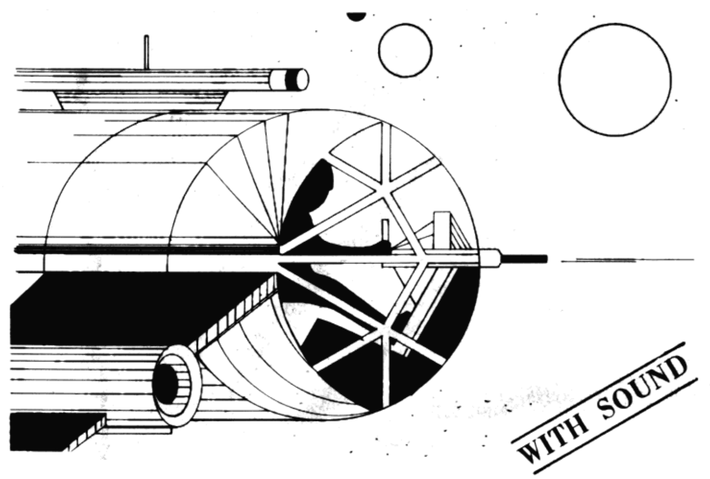
They call it GRAFTRAX II. And it means 480 dots across the page, resolution to 60 dots per inch, and a graphic image free of the jitter and overlap that plagues other printers. You get cleaner grays and finer point resolution.

So now you've got a choice. You want more power and extra functions, you buy the MX-80. You want a basic little printer that prints, and keeps on printing, you buy the MX-70. They're both at American Business Computers.

Call for unbelievably low price.

528

GO BOLDLY . . .
Where No TRS-80* Program Has Gone Before!



DANGER IN ORBIT

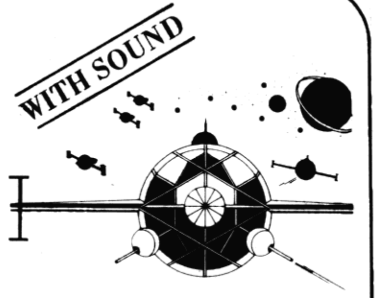
DATE: 28.02.2047
LOCATION: 270 million miles from Terra
MISSION: Maintaining Terra's Space Lanes

Briefing will follow:
1.1 Your mission is to destroy any asteroids in your sector and to prevent alien spacecraft from infiltrating the Terran Defense Network.
1.2 Your ship is armed with an anti-matter cannon. You can shoot large asteroids, but this turns them into many smaller asteroids, each capable of destroying your ship.
1.3 In addition, alien ships can make instantaneous hyperspace jumps into your area and start firing on your ship.
1.4 You'll need lightning reflexes and nerves of steel to survive Danger In Orbit. We have no use for non-survivors!

Danger In Orbit, a real-time, machine-language game, features variable levels of difficulty, superb high-speed graphics, sound effects and automatic score keeping. (T1 or T2)

Order No. 0237R
$14.95 Tape.
Order No. 0247RD
$19.95 Disk.

BALL TURRET GUNNER

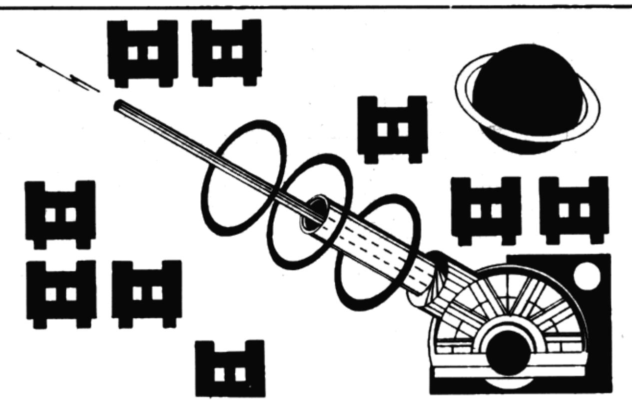


For years the Petro Resource Conglomerate has attacked our photon collection stations and strangled our deep-space trade routes. The PRC Exxonerator Class light fighters (code name: Gnat) have been their main weapon. Now you can strike back, by joining the Ball Turret Gunner Service.

Imagine yourself at the control console of an LW-1417 Stratoblazer (Type B Strategic Laser Weapon). Your Hindsight Director informs you that a Gnat fighter is coming in for an attack. You pivot your gigawatt laser turret until you can see the target on your monitor. The Range Indicator shows him coming in fast. The Targeting Computer studies his course and speed as your finger tenses over the firing key. You know you'll have only a fraction of a second in which to react. The Gnat fighter's evasive maneuvers cause him to dance in your sights. Suddenly, you see the FIRE Command and you react instinctively. Your laser beam lashes out and reduces the Gnat to an expanding ball of ionized gas. Mission accomplished!

Ball Turret Gunner, with your choice of multiple levels of difficulty, optional sound effects and superb graphics, is more than just a game. It's an adventure. Experience it! (T1)

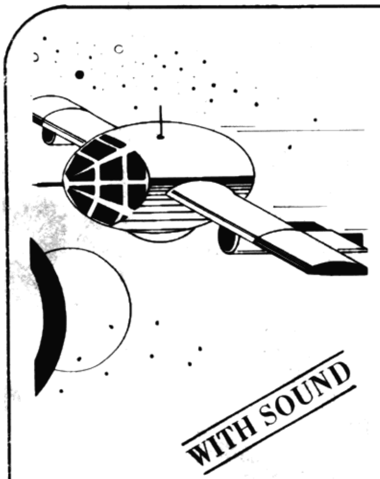
Order No. 0051R $9.95.



ALIEN ATTACK FORCE

The INVADERS are coming! Earth's defenses are dead except for your Laser base. Your assignment is to destroy the approaching INVADERS before they destroy Earth. Before Earth's sensors failed, they detected 550 armed invaders in space, speeding toward us in 10 attack formations of 55 in each group. The sensors detected four different types of attack craft: Large, Medium, Small, and short profile craft which is the most difficult to destroy. If you cannot stop these space attackers they will stop Earth..... for good. (T1)

Order No. 0240R $9.95.



COSMIC PATROL

WARNING: PLAYERS OF THIS GAME SHOULD BE PREPARED FOR A STATE OF REALISM HITHERTO UNAVAILABLE ON THE TRS-80

Skilled players soon master many difficult computer games, but COSMIC PATROL is in a world all its own. The challenge intensifies! Supporting graphics and sound (optional) make each encounter an exciting new experience. It all adds up to a Super 3-S package. . . skill, sight and sound.

Scenario: The Cosmic Patrol program puts you in the command chair of a small interstellar patrol craft. Your mission is to defend Terran space and prey on the Quelon supply ships which carry essential parts and lubricants for that implacably hostile robotic force. The drone freighters are fairly easy pickings for the accomplished starship pilot, but beware of the I-Fighter escorts. They're armed, fast and piloted by intelligent robots linked to battle computers. They *never* miss.

The Cosmic Patrol program is not just another search and destroy game. With its fast, real-time action, impressive sound option and superb graphics, this machine-language program is the best of its genre.

Don't keep putting quarter after quarter into arcade games or spending big bucks for video game cartridges. Get Cosmic Patrol from Instant Software—and get the best for less! (T1 or T2)

Order No. 0223R
$14.95 Tape.
Order No. 0224RD
$19.95 Disk.

TO ORDER

See Your Local Instant Software Dealer or Call Toll-Free 1-800-258-5473
(Orders Only)
In NH dial 1-603-924-7296

Instant Software™
PETERBOROUGH, N.H. 03458 ✓530

(T1) = TRS-80 Model 1, Level II, 16K RAM.
(T2) = TRS-80 Model 1, Level II 16K, expansion interface 16K + 1 disk drive.

Pony programs for the Pocket Computer.

Nag Analysis

Dave Crosby
300 Windsor Drive
Oak Brook, IL 60521

"The way to make a small fortune at the race track is start with a large fortune."
—Anonymous.

With this in mind, I offer you a program for the pocket computer to handicap the nags.

The program considers the following: days between races; distance; class; stretch, gain; call positions; jockey weight; speed rating; and earnings.

The computer takes certain facts into consideration in helping you to bet. If the horse has not raced in the last 15 days, it is eliminated and you go on to rate the next post position. If it has been 15 days or less, the horse is awarded 10 points. If the distance on today's race is the same as the last time the horse ran, another 10 points are awarded. Any stretch gain in the last three races gives the horse 10 points.

Call positions are totalled for the last three races and then entered. For example, if the horse was first five times (at any call) in three races, enter 5. The second call works the same way. The computer awards 10 points for a first call, five for a second. Ten points are awarded if the jockey's weight is the same for this race as the last. Enter the speed rating for the last three races and the computer will average them and award points accordingly. The last characteristic is earnings: enter earnings for the year and the number of starts the horse made.

The computer keeps a running total and gives a total for the post position you are working on. When all horses have been rated, you enter 0 for post position and the computer will rank the horses by score. You may want to remove program lines with a pause, to speed things up.

Bet on the horse with the highest rating. ■

Note: The basic idea for the handicap system used in this program was from the book *The Horseplayers Guide to Winning System* by Alec MacKenzie and Bert Randolph Sugar, published by Corwin Books.

```
5 'HANDYCAP PROGRAM FOR TRS-80 POCKET COMPUTER
10 CLEAR
20 INPUT "TODAYS DATE ";A
30 INPUT "DAYS IN PREVIOUS MONTH ";C
40 INPUT "TODAYS DISTANCE ";B:J=0:L=15
50 INPUT "POST # ";E:IF E=0THEN 465
55 IF E>15 THEN 50
60 D=0:IF E >J LET J=E
65 IF E<L LET L=E
70 PAUSE "DAYS BETWEEN RACES "
80 INPUT "LAST RACE DAY ";F:G=A
90 IF F>G LET G=G+C
100 IF (G-F)>15 PAUSE "ELIMINATED":A(E+27)=0:A(E+12)=E:GOTO 50
110 I=10:GOSUB 440
120 INPUT "DISTANCE LAST RACE ";F
130 IF F=B LET I=10:GOSUB440:GOTO 150
140 GOSUB 460
150 PAUSE "CLASS CHECK"
160 INPUT "TODAYS CLAIM" ;F
170 INPUT "LAST CLAIM ";G
180 I=INT ((G-F)/100+.5):GOSUB 440
190 INPUT "GAIN IN STRETCH? (Y/N)";K$
200 IF K$="Y"LET I=10:GOSUB 440:GOTO 220
210 GOSUB 460
220 PAUSE "RUNNING POSITION"
230 INPUT "1ST ANY CALL ";F
240 INPUT "2ND ANY CALL ";G
250 I=F*10+G*5:GOSUB440
260 PAUSE "ASSIGNED WEIGHT"
270 INPUT"WEIGHT THIS RACE ";F
280 INPUT "WEIGHT LAST RACE ";G
290 IF F=GLET I=10:GOSUB 440:GOTO310
300 GOSUB 460
310 PAUSE "LAST 3 SPEEDS "
320 INPUT "FIRST ";F
330 INPUT "SECOND ";G
340 INPUT "THIRD ";H
350 I=INT ((F+G+H)/3+.5):GOSUB 440
360 PAUSE "AVERAGE EARNINGS"
370 INPUT "YEARS EARNINGS ";F
380 INPUT "NUMBER OF STARTS ";G:IF G=0 LET I=0:GOTO 400
390 I=INT (F/G*.1):D=D+I
400 PAUSE "AWARD ";I
410 PRINT "SCORE PP# ";E;" IS ";D
420 A(E+12)=E:A(E+27)=D:GOTO 50
440 D=D+I:PAUSE "AWARD  ";I
450 PAUSE "SCORE NOW ";D:RETURN
460 PAUSE "NO POINTS AWARDED":RETURN
465 K$="":INPUT "SORT?  (Y/N)";K$:IFK$="N"THEN 530
470 FOR F=L2 TO    J:B=0: FOR G=L TO J-F
480 IF A(G+27)>=A(G+28)THEN 510
490 H=A(G+27):A(G+27)=A(G+28):A(G+28)=H
500 H=A(G+12):A(G+12)=A(G+13):A(G+13)=H:B=1
510 NEXT G:IF B=0 THEN 530
520 NEXT F
530 FOR F=L TO J: D=F +12 :E=F+27
540 IF A(E)<>0 THEN PRINT F;" PP# ";A(D);" POINTS=";A(E)
550 NEXT F:INPUT"REPEAT? (Y/N) ";K$:IF K$="Y" THEN 530
560 GOTO 40
```

Program Listing 1

1. Type R. to start program.
2. For Today's Date and Last Race Day, just enter the day, i.e., 10, 15, 20, etc. for "days in previous month".
3. For distance, enter 6 for six furlongs, 1 for one mile, 1.16 for 1 1/16, etc. The same holds for both distance inputs.
4. For the stretch gain question Y = yes, N = no.
5. Post call is for last three races.
6. When you get the total score for one post position, press Enter for the next post position.
7. When all post positions have been entered, enter 0 for P.P.# question.
8. If you wish horses ranked, answer Y to sort question.

Table 1. Program Notes

Generate assembly listings of machine code with this BASIC program.

A BASIC Disassembler

Frank Delfine
26 Oceanview Ave.
Farmingville, NY 11738

The disassembler presented here generates assembly listings of machine code residing in high memory (approximately 31300 or higher). The program also helps examine the Level II PROMS so we can take advantage of the TRS-80's BASIC routines.

The disassembler is written in BASIC and occupies just under 11K of RAM. Strings and other variable space required to run the program bring the total RAM requirements to about 13K, leaving about 2K of high RAM in a 16K system to load the machine code to be disassembled.

The program drives a line printer; if you have no printer, the LPRINT statements in program line 190 must be changed to PRINTs to send the listing to the screen. The number of lines per page in line 205 should then be changed from 50 to 15 so as not to scroll the listing off the screen before it can be read.

To make the program more flexible and less aggravating to use, some little extras have been included. The program lets you declare blocks of memory within your specified disassembly range as data so the program will not try to disassemble them. To make it easier to locate these data blocks within the program code, an auxiliary program is included (Program Listing 2) which displays any block of memory in ASCII on the CRT. An entire machine coded program can be scanned in a few minutes before disassembly. Data tables are usually easy to recognize when the data is presented in this format (i.e., data such as letter strings used for CRT prompts, sequential number strings, patterned data, etc.). The addresses of these tables can be written down and typed into the disassembler later.

If a table is missed by this method, it can usually be spotted in the disassembly listing as

Program Listing 1. DISASM

```
1 CLEAR(1ØØØ):DIM I$(255),BL(255):N=Ø:M=Ø:GOTO8ØØØ
1Ø CLS:C=Ø:PRINTTAB(15)"* * *  Z-8Ø DISASSEMBLER  * * *
   "
2Ø PRINT:INPUT"DUMP TITLE";F$
3Ø PRINT:INPUT"START ADR (IN DEC)";A
4Ø INPUT"END ADR (IN DEC)";B
42 PRINT:INPUT"DECLARE ANY DATA SEGMENTS";V$
43 IF V$="Y"GOSUB25ØØØ
5Ø PRINT:INPUT"TURN ON LP THEN 'ENTER'";B$
6Ø LPRINT" ":LPRINTTAB(2Ø)F$:LPRINT" "
79 Q=A
8Ø H1=INT(A/4Ø96):H2=INT((A-H1*4Ø96)/256)
9Ø L1=INT((A-H1*4Ø96-H2*256)/16)
1ØØ L2=A-H1*4Ø96-H2*256-L1*16
111 IF M<>Ø GOTO113
112 GOTO12Ø
113 IF A>=A(N)ANDA<=B(N)GOTO115
114 GOTO12Ø
115 D=PEEK(A):GOSUB5ØØ9Ø:FH$="D":FL$="A":GH$="T":GL$="A
    ":
T$=I$(D):I$(D)=""
116 IF A=B(N)THENN=N+1:M=M-1:GOTO13Ø
117 GOTO13Ø
12Ø D=PEEK(A):GOSUB6ØØ
13Ø V=D:GOSUB3ØØØØ
14Ø DH$=H$:DL$=L$
16Ø IF H1>=1Ø GOTO24Ø
165 H1$=STR$(H1):H1$=MID$(H1$,2,1)
17Ø IF H2>=1Ø GOTO25Ø
175 H2$=STR$(H2):H2$=MID$(H2$,2,1)
18Ø IF L1>=1Ø GOTO26Ø
182 L1$=STR$(L1):L1$=MID$(L1$,2,1)
185 IF L2>=1Ø GOTO27Ø
187 L2$=STR$(L2):L2$=MID$(L2$,2,1)
19Ø LPRINTQ;:LPRINTTAB(1Ø);:LPRINTUSING"%   %!!!!";" ",
     H1$,H2$,L1$,L2$;:
LPRINTUSING"%   %!!!!!!!!";" ",D
     H$,DL$,EH$,EL$,FH$,FL$,GH$,GL$;:
LPRINTTAB(3Ø)I$(D
     )
2ØØ A=A+1:C=C+1:IF A>=B+1 GOTO 4ØØ
2Ø5 IF C=5Ø GOTO5ØØ
21Ø IFGH$="T"THENI$(D)=T$
211 GOTO79
24Ø X=H1-1Ø+65:H1$=CHR$(X):GOTO17Ø
25Ø X=H2-1Ø+65:H2$=CHR$(X):GOTO18Ø
26Ø X=L1-1Ø+65:L1$=CHR$(X):GOTO185
27Ø X=L2-1Ø+65:L2$=CHR$(X):GOTO19Ø
4ØØ PRINT"* *  DUMP COMPLETE  * *
41Ø INPUT"AGAIN (Y/N)";B$
42Ø IFB$="Y"GOTO1Ø
425 GOTO53Ø
5ØØ PRINT:PRINT"* * * *  END OF PAGE  * * * *"
51Ø INPUT"RELOAD THEN 'ENTER'";B$:C=Ø:GOTO79
53Ø END
6ØØ IF D<64 OR D>127 GOTO74Ø
6Ø5 IFD=118THENRETURN
61Ø DH=INT(D/16):DL=D-DH*16
62Ø G=DLAND7:F=((DAND56)/8)
64Ø I$(D)="LD        "+J$(F)+","
```

Program continues

```
720 I$(D)=I$(D)+J$(G):GOSUB50090:RETURN
740 IFD<128ORD>191GOTO1000
750 DH=INT(D/16):DL=D-(DH*16)
760 G=DLAND7:F=((DAND120)/8)
780 IF F=0 I$(D)="ADD        A,"
790 IF F=1 I$(D)="ADC        A,"
800 IF F=2 I$(D)="SUB        "
810 IF F=3 I$(D)="SBC        A,"
820 IF F=4 I$(D)="AND        "
830 IF F=5 I$(D)="XOR        "
840 IF F=6 I$(D)="OR         "
850 IF F=7 I$(D)="CP         "
860 I$(D)=I$(D)+J$(G):GOSUB50090:RETURN
1000 IFBL(D)=1GOTO1030
1010 IFBL(D)=2GOTO1040
1020 IFBL(D)=3GOTO1060
1025 IFBL(D)=4GOTO15000
1026 IFD=221ORD=253GOTO26000
1030 GOSUB50090:RETURN
1040 A=A+1:Z=PEEK(A):GOTO2000
1060 A=A+1:Z=PEEK(A):A=A+1:Z1=PEEK(A)
2000 V=Z:GOSUB30000:EH$=H$:EL$=L$
2070 IF BL(D)=3 GOTO 3000
2080 GOSUB50100:GOTO4000
3000 V=Z1:GOSUB30000:FH$=H$:FL$=L$
4000 P1$=FH$+FL$+EH$+EL$:P2$=EH$+EL$:GH$="":GL$=""
4001 F=DAND7:G=DAND56:G=G/8
4002 IFF=6AND(DAND192)=0THENGOTO5010
4003 IFF=2AND(DAND192)=192THEN GOTO5030
4004 IFF=4 GOTO5050
4005 IFF=0AND(G<>2)THENGOTO5020
4008 I$(33)="LD        HL,"+P1$
4010 I$(34)="LD        ("+P1$+"),HL"
4030 I$(50)="LD        ("+P1$+"),A"
4040 I$(205)="CALL      "+P1$
4050 I$(195)="JP        "+P1$
4060 I$(58)="LD        A,("+P1$+")"
4070 I$(254)="CP        "+P2$
4110 I$(42)="LD        HL,("+P1$+")"
4123 I$(49)="LD        SP,"+P1$
4129 I$(17)="LD        DE,"+P1$
4130 I$(16)="DJNZ      "+P2$
4132 I$(1)="LD        BC,"+P1$
4135 I$(198)="ADD       A,"+P2$
4138 I$(206)="ADC       A,"+P2$
4140 I$(211)="OUT       "+P2$+",A"
4142 I$(214)="SUB       "+P2$
4144 I$(219)="IN        A,"+P2$
4146 I$(222)="SBC       A,"+P2$
4149 I$(230)="AND       "+P2$
4152 I$(238)="XOR       "+P2$
4155 I$(246)="OR        "+P2$
4158 RETURN
5010 I$(D)="LD        "+J$(G)+","+P2$:RETURN
5020 IFG=3THENI$(D)="JR        "+P2$:RETURN
5022 G=G-4:I$(D)="JR        "+S$(G)+","+P2$:RETURN
5030 I$(D)="JP        "+S$(G)+","+P1$:RETURN
5050 I$(D)="CALL      "+S$(G)+","+P1$:RETURN
8000 FORS=0TO63:BL(S)=1:NEXTS
8010 FORS=192TO255:BL(S)=1:NEXTS
8020 BL(118)=1:BL(6)=2:BL(14)=2:BL(16)=2:BL(22)=2:BL(24
)=2:
BL(30)=2:BL(32)=2:BL(38)=2:BL(40)=2:BL(46)=2:
     BL(48)=2:BL(54)=2:
BL(56)=2
8030 BL(62)=2:BL(198)=2:BL(206)=2:BL(211)=2:BL(214)=2:B
     L(219)=2
:BL(222)=2:BL(230)=2:BL(238)=2:BL(246)=2:
     BL(254)=2
8040 BL(1)=3:BL(17)=3:BL(33)=3:BL(34)=3:BL(42)=3:BL(49)
     =3:
BL(50)=3:BL(58)=3:BL(194)=3:BL(195)=3:BL(196)=
     3:BL(202)=3:
BL(204)=3:BL(205)=3:BL(210)=3:BL(212)
     =3:BL(218)=3:BL(220)=3
8050 BL(226)=3:BL(228)=3:BL(234)=3:BL(236)=3:BL(242)=3:

BL(244)=3:BL(250)=3:BL(252)=3:BL(203)=4:BL(237)=4
     :BL(221)=0
:BL(253)=0
8070 J$(0)="B":J$(1)="C":J$(2)="D":J$(3)="E":J$(4)="H":
     J$(5)="L"
8080 J$(6)="(HL)":J$(7)="A"
8090 S$(0)="NZ":S$(1)="Z":S$(2)="NC":S$(3)="C":S$(4)="P
     O":
S$(5)="PE":S$(6)="P":S$(7)="M"
8100 P$(0)="SBC       HL,":P$(1)="ADC       HL,"
8110 P$(2)="SBC       HL,":P$(3)="ADC       HL,"
8120 P$(4)="SBC       HL,":P$(5)="ADC       HL,"
8130 P$(7)="ADC       HL,":N$(0)="BC":N$(1)="BC":N$(2)=
     "DE"
8140 N$(3)="DE":N$(4)="HL":N$(5)="HL":N$(7)="SP"
8150 L$(0)="N":L$(1)="I":M$(0)="0":M$(2)="1":M$(3)="2"
8160 O$(0)="I,A":O$(1)="R,A":O$(2)="A,I":O$(3)="A,R"
8170 Q$(0)="LD":Q$(1)="CP":Q$(2)="IN":Q$(3)="OUT"
9000 I$(0)="NOP":I$(2)="LD        (BC),A":I$(3)="INC
     BC"
9001 I$(7)="RLC       A":I$(8)="EX        AF,AF'"
9002 I$(9)="ADD       HL,BC":I$(10)="LD        A,(BC)"
9003 I$(11)="DEC       BC":I$(15)="RRC       A"
9004 I$(18)="LD        (DE),A":I$(19)="INC       DE"
9005 I$(23)="RLA":I$(25)="ADD       HL,DE":I$(26)="LD
     A,(DE)"
9006 I$(27)="DEC       DE":I$(31)="RRA":I$(35)="INC
     HL"
9007 I$(39)="DAA":I$(41)="ADD       HL,HL":I$(43)="DEC
     HL"
9008 I$(47)="CPL":I$(51)="INC       SP":I$(55)="SCF"
9009 I$(57)="ADD       HL,SP":I$(59)="DEC       SP":I$(
     63)="CCF"
9010 I$(43)="DEC       HL"
9011 I$(192)="RET       NZ":I$(197)="PUSH      BC"
9012 I$(199)="RST       0":I$(200)="RET       Z":I$(207
     )="RST       8"
9013 I$(208)="RET       NC":I$(209)="POP       DE"
9014 I$(213)="PUSH      DE":I$(215)="RST       10H"
9015 I$(216)="RET       C":I$(223)="RST       18H"
9016 I$(224)="RET       PO":I$(227)="EX        (SP),HL"
9017 I$(231)="RST       20H":I$(232)="RET       PE"
9018 I$(233)="JP        (HL)":I$(235)="EX        DE,HL"
9019 I$(239)="RST       28H":I$(240)="RET       P"
9020 I$(201)="RET"
9021 I$(241)="POP       AF":I$(243)="DI":I$(245)="PUSH
     AF"
9022 I$(247)="RST       30H":I$(248)="RET       M":I$(2
     51)="EI"
9023 I$(249)="LD        SP,HL":I$(255)="RST       38H"
9030 I$(197)="PUSH      BC"
9040 I$(193)="POP       BC":I$(229)="PUSH      HL"
9050 I$(225)="POP       HL"
9060 I$(217)="EXX"
9070 I$(118)="HALT"
9080 L=0:FOR D=5TO45STEP8:I$(D)="DEC       "+J$(L):L=L+
     1:NEXT
9090 L=0:FORD=4TO44STEP8:I$(D)="INC       "+J$(L):L=L+1
     :NEXT
9100 I$(60)="INC       A":I$(61)="DEC       A":GOTO10
15000 IFD=203GOTO15040
15020 IFD=237GOTO40200
15040 A=A+1:Z=PEEK(A)
15050 EH=INT(Z/16):EL=Z-EH*16:G=EL AND 7
15060 F=INT((EL AND 8)/8):F=F+(EH*2):GOSUB15070
15066 GOTO15380
15070 IFF=0I$(D)="RLC       ":RETURN
15080 IFF=1I$(D)="RRC       ":RETURN
15090 IFF=2I$(D)="RL        ":RETURN
15100 IFF=3I$(D)="RR        ":RETURN
15110 IFF=4I$(D)="SLA       ":RETURN
15120 IFF=5I$(D)="SRA       ":RETURN
15130 IFF=7I$(D)="SRL       ":RETURN
15140 IFF>=8ANDF<=15GOTO15180
15150 IFF>=16ANDF<=23GOTO15200
15160 F=FAND7:F$=CHR$(F+48)
15170 I$(D)="SET       "+F$+",":RETURN
15180 F=FAND7:F$=CHR$(F+48)
15190 I$(D)="BIT       "+F$+",":RETURN
15200 F=FAND7:F$=CHR$(F+48)
15210 I$(D)="RES       "+F$+",":RETURN
15380 IF EH>=10 GOTO15430
15390 EH$=STR$(EH):EH$=MID$(EH$,2,1)
15400 IF EL>=10 GOTO15440
15410 EL$=STR$(EL):EL$=MID$(EL$,2,1)
15420 GOSUB50100:I$(D)=I$(D)+J$(G):RETURN
15430 X=EH+55:EH$=CHR$(X):GOTO15400
15440 X=EL+55:EL$=CHR$(X):GOTO15420
20000 IFDAND7=5GOTO20040
20010 IFDAND7=4GOTO20040
20020 GOTO1030
20040 L=DAND56
20050 G=L/8
20130 IFDAND7=5THENI$(D)="DEC       "+J$(G):GOTO20150
20140 I$(D)="INC       "+J$(G)
20150 GOSUB50090:RETURN
25000 N=1
25010 INPUT"START ADR=";A(N):INPUT"END ADR=";B(N)
25020 INPUT"ANOTHER SEGMENT";V$
25030 IF V$="Y"THEN N=N+1:GOTO25010
25040 M=N:N=1:RETURN
26000 IFD=221THENV$="IX":GOTO26004
26002 V$="IY"
26004 A=A+1:Z=PEEK(A)
26010 IF Z=203 GOTO27000
26020 IFZ>=70ANDZ<=190GOTO40000
26025 IFZ=33ORZ=34ORZ=42GOTO26180
26027 IFZ=52ORZ=53GOTO26280
26028 IFZ=54GOTO26350
26029 V=Z:GOSUB30000:EH$=H$:EL$=L$:GOSUB50100
26030 IFZ=9 I$(D)="ADD       "+V$+",BC"
26040 IFZ=25I$(D)="ADD       "+V$+",DE"
26050 IFZ=35I$(D)="INC       "+V$
26060 IFZ=41I$(D)="ADD       "+V$+","+V$
26070 IFZ=43I$(D)="DEC       "+V$
26080 IFZ=57I$(D)="ADD       "+V$+",SP"
26090 IFZ=225I$(D)="POP       "+V$
26100 IFZ=227I$(D)="EX        (SP),"+V$
26110 IFZ=229I$(D)="PUSH      "+V$
26120 IFZ=233I$(D)="JP        ("+V$+")"
26130 IFZ=249I$(D)="LD        SP,"+V$
```

Program continues

```
26150 RETURN
26180 V=Z:GOSUB30000
26190 EH$=H$:EL$=L$
26200 A=A+1:Z1=PEEK(A):V=Z1:GOSUB30000
26210 FH$=H$:FL$=L$:A=A+1
26220 Z1=PEEK(A):V=Z1:GOSUB30000
26230 GH$=H$:GL$=L$
26240 IFZ=33I$(D)="LD           "+V$+","+GH$+GL$+FH$+FL$
26250 IFZ=34I$(D)="LD           ("+GH$+GL$+FH$+FL$+"),"+V$
26260 IFZ=42I$(D)="LD           "+V$+",("+GH$+GL$+FH$+FL$+
      ")"
26270 RETURN
26280 V=Z:GOSUB30000:EH$=H$:EL$=L$
26300 A=A+1:Z1=PEEK(A):V=Z1:GOSUB30000
26310 FH$=H$:FL$=L$:GH$="":GL$=""
26320 IFZ=52I$(D)="INC          ("+V$+"+"+FH$+FL$+")"
26330 IFZ=53I$(D)="DEC          ("+V$+"+"+FH$+FL$+")"
26340 RETURN
26350 V=Z:GOSUB30000:EH$=H$:EL$=L$
26370 A=A+1:Z1=PEEK(A):V=Z1:GOSUB30000
26380 FH$=H$:FL$=L$:A=A+1
26390 Z1=PEEK(A):V=Z1:GOSUB30000
26400 GH$=H$:GL$=L$
26410 I$(D)="LD          ("+V$+"+"+FH$+FL$+"),"+GH$+GL$
26420 RETURN
27000 A=A+1:EH$=CHR$(67):EL$=CHR$(66)
27010 V=PEEK(A):GOSUB30000
27020 FH$=H$:FL$=L$
27050 A=A+1:Z2=PEEK(A):O=Z2AND248:O=O/8
27055 Q$="("+V$+"+"+FH$+FL$+")"
27060 F=O:GOSUB15070
27070 I$(D)=I$(D)+Q$
27080 V=Z2:GOSUB30000:GH$=H$:GL$=L$:RETURN
30000 H=INT(V/16):L=V-(H*16)
30010 IFH>=10 GOTO30060
30020 H$=STR$(H):H$=MID$(H$,2,1)
30030 IF L>=10GOTO30070
30040 L$=STR$(L):L$=MID$(L$,2,1):RETURN
30060 X1=H+55:H$=CHR$(X1):GOTO30030
30070 X1=L+55:L$=CHR$(X1):RETURN
40000 V=Z:GOSUB30000:EH$=H$:EL$=L$:GH$="":GL$=""
40020 A=A+1:Z1=PEEK(A)
40030 V=Z1:GOSUB30000:FH$=H$:FL$=L$
40050 IF Z=126THENI$(D)="LD          A,("+V$+"+"+FH$+FL$+
      ")":RETURN
40060 P=ZAND240
40070 IF P=112GOTO40140
40080 IF P>=128 GOTO40170
40090 P=ZAND56:P=P/8:GOSUB50000
40120 I$(D)="LD          "+G$+",("+V$+"+"+FH$+FL$+")":RET
      URN
40140 P=ZAND7:GOSUB50000
40150 I$(D)="LD          ("+V$+"+"+FH$+FL$+"),"+G$:RETURN
40170 P=ZAND56:P=P/8:GOSUB50000
40180 I$(D)=I$(D)+V$+"+"+FH$+FL$+")"
40190 RETURN
40200 A=A+1:Z=PEEK(A):V=Z:GOSUB30000:EH$=H$:EL$=L$
40210 IFZ=67ORZ=75ORZ=83ORZ=91ORZ=115ORZ=123THEN40450
40220 GOSUB50100:F=ZAND248:G=ZAND7
40230 IFF=160GOTO40410
40240 IFF=168GOTO40420
40250 IFF=176GOTO40430
40260 IFF=184GOTO40440
40270 F=ZAND56:F=F/8:G=ZAND7
40280 1FF=6I$(D)="SBC         HL,SP":RETURN
40290 IFG=0I$(D)="IN          "+J$(F)+",(C)":RETURN
40300 IFG=1I$(D)="OUT         (C),"+J$(F):RETURN
40310 IFG=2I$(D)=P$(F)+N$(F):RETURN
40320 IFG=4I$(D)="NEG":RETURN
40330 IFG=5I$(D)="RET"+L$(F):RETURN
40340 IFG=6I$(D)="IM "+M$(F):RETURN
40350 IFG<>7I$(D)="* * *":RETURN
40360 IFF<=3GOTO40400
40370 IFF=4I$(D)="RRD":RETURN
40380 IFF=5I$(D)="RLD":RETURN
40390 GOTO40350
40400 I$(D)="LD          "+O$(F):RETURN
40410 I$(D)=Q$(G)+"I":RETURN
40420 I$(D)=Q$(G)+"D":RETURN
40430 I$(D)=Q$(G)+"IR":RETURN
40440 I$(D)=Q$(G)+"DR":RETURN
40450 A=A+1:Z1=PEEK(A):V=Z1:GOSUB30000:FH$=H$:FL$=L$
40460 A=A+1:Z1=PEEK(A):V=Z1:GOSUB30000:GH$=H$:GL$=L$
40470 GG$=GH$+GL$+FH$+FL$
40480 IFZ=67I$(D)="LD           ("+GG$+"),BC":RETURN
40490 IFZ=75I$(D)="LD           BC,("+GG$+")":RETURN
40500 IFZ=83I$(D)="LD           ("+GG$+"),DE":RETURN
40510 IFZ=91I$(D)="LD           DE,("+GG$+")":RETURN
40520 IFZ=115I$(D)="LD           ("+GG$+"),SP":RETURN
40530 IFZ=123I$(D)="LD          SP,("+GG$+")":RETURN
50000 IF P=0 THEN G$="B" : I$(D)="ADD        A,("
50010 IF P=1 THEN G$="C" :I$(D)="ADC        A,("
50020 IF P=2 THEN G$="D":I$(D)="SUB        ("
50030 IF P=3 THEN G$="E":I$(D)="SBC        A,("
50040 IF P=4 THEN G$="H":I$(D)="AND        ("
50050 IF P=5 THEN G$="L":I$(D)="XOR        ("
50060 IF P=6 THEN G$="***":I$(D)="OR         ("
50070 IF P=7 THEN G$="A":I$(D)="CP         ("
50080 RETURN
50090 EH$="":EL$="":FH$="":FL$="":GH$="":GL$="":RETURN
50100 FH$="":FL$="":GH$="":GL$="":RETURN
```

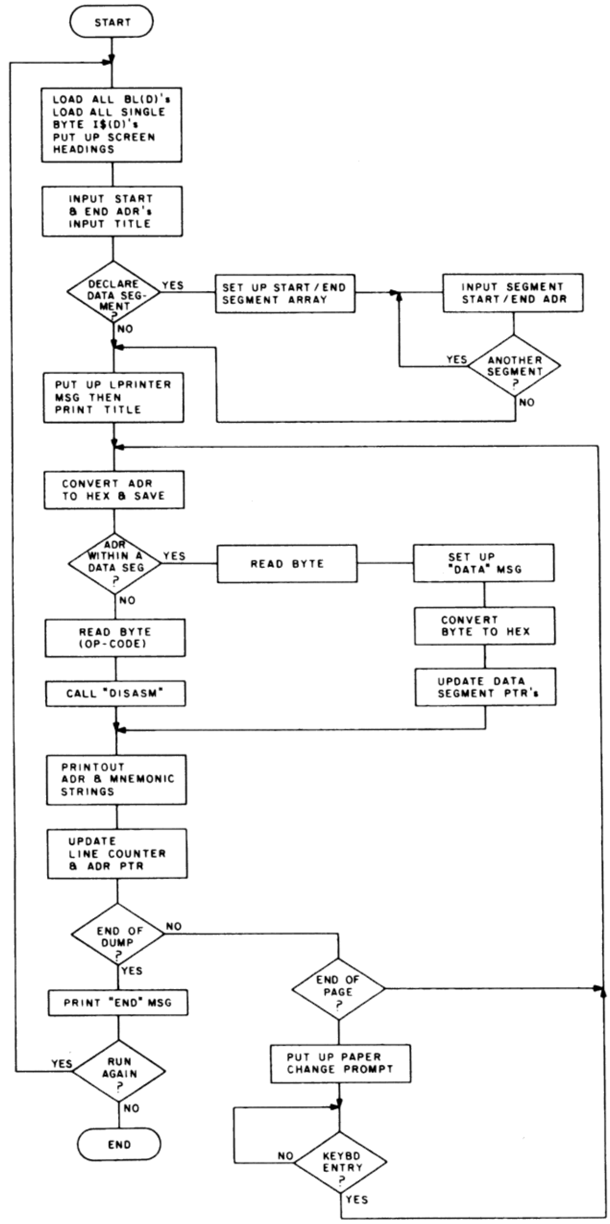


Fig. 1.

a run of instructions that don't make much sense in a particular section of the program. With practice, these things can be spotted quickly. To make printout paging easier, the program will stop after printing 50 lines and prompt you to reload the printer before continuing.

Once the machine code to be disassembled is loaded into memory, you may generate a listing by running the program and answering the following questions:

DUMP TITLE:?—You can specify a title for the first page.

START ADR(IN DEC):?—Specify the first address to disassemble (in decimal form).

END ADR(IN DEC):?—Specify the last address to disassemble (in decimal form).

DECLARE ANY DATA SEGMENTS?—Answer Y or N. If Y, the following message will appear.

START ADR = ?—Answer with the first address of the code you wish to declare as data.

END ADR = ?—Answer with the end address of the data segment

ANOTHER SEGMENT?—If you respond with Y the START/END ADR messages will repeat for another data segment. If you answer with N the following message will appear. (This is where the program would have gone had you entered N to the

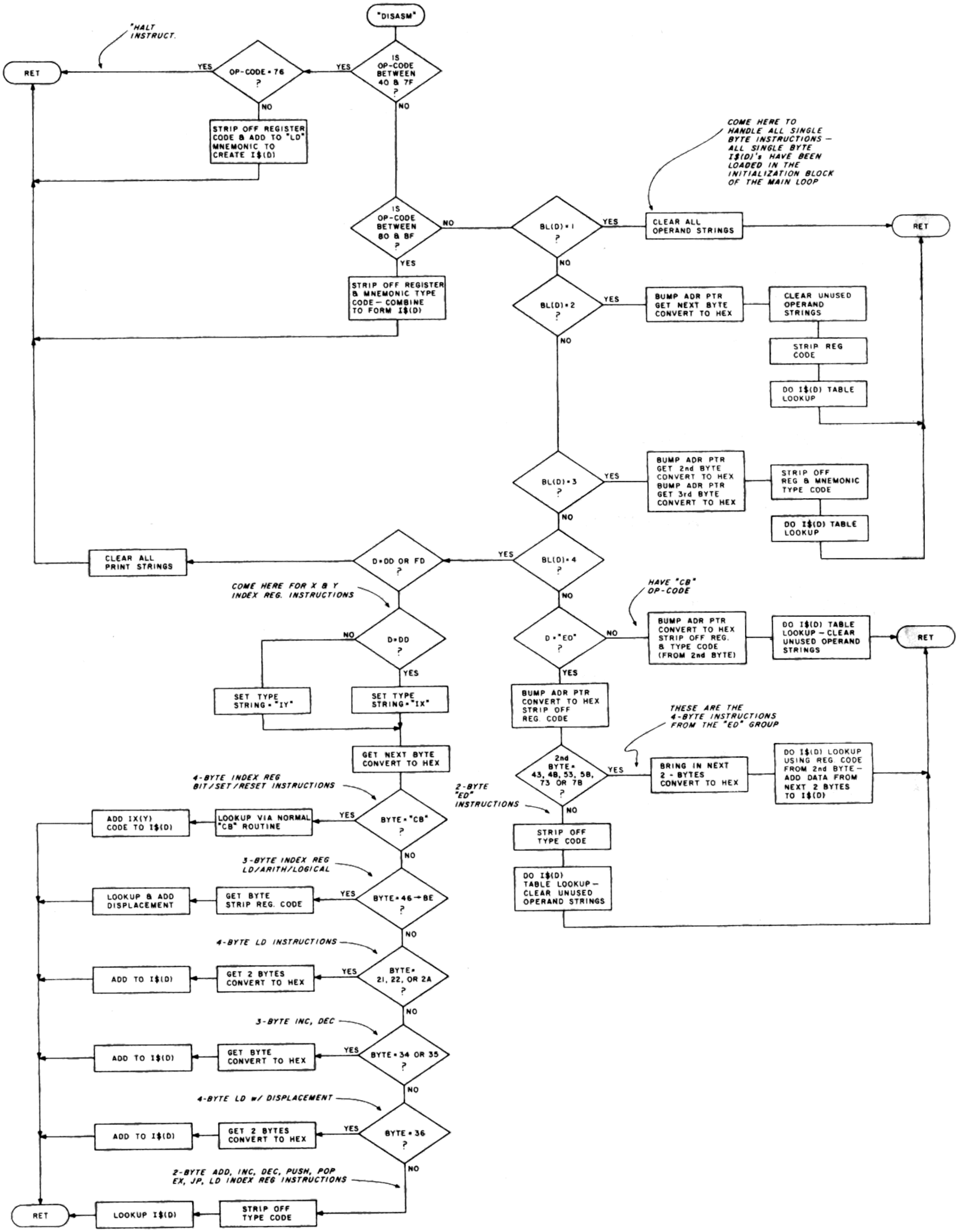

"DISASM"
IS OP-CODE BETWEEN 40 & 7F ?
OP-CODE = 76 ?
*HALT INSTRUCT.
RET
STRIP OFF REGISTER CODE & ADD TO "LD" MNEMONIC TO CREATE I$(D)
IS OP-CODE BETWEEN 80 & BF ?
STRIP OFF REGISTER & MNEMONIC TYPE CODE — COMBINE TO FORM I$(D)
BL(D) = 1 ?
COME HERE TO HANDLE ALL SINGLE BYTE INSTRUCTIONS — ALL SINGLE BYTE I$(D)'s HAVE BEEN LOADED IN THE INITIALIZATION BLOCK OF THE MAIN LOOP
CLEAR ALL OPERAND STRINGS
BL(D) = 2 ?
BUMP ADR PTR GET NEXT BYTE CONVERT TO HEX
CLEAR UNUSED OPERAND STRINGS
STRIP REG CODE
DO I$(D) TABLE LOOKUP
BL(D) = 3 ?
BUMP ADR PTR GET 2nd BYTE CONVERT TO HEX BUMP ADR PTR GET 3rd BYTE CONVERT TO HEX
STRIP OFF REG & MNEMONIC TYPE CODE
DO I$(D) TABLE LOOKUP
BL(D) = 4 ?
D = DD OR FD ?
CLEAR ALL PRINT STRINGS
COME HERE FOR X & Y INDEX REG. INSTRUCTIONS
D = DD ?
SET TYPE STRING = "IY"
SET TYPE STRING = "IX"
GET NEXT BYTE CONVERT TO HEX
4-BYTE INDEX REG BIT/SET/RESET INSTRUCTIONS
BYTE = "CB" ?
LOOKUP VIA NORMAL "CB" ROUTINE
ADD IX(Y) CODE TO I$(D)
3-BYTE INDEX REG LD/ARITH/LOGICAL
BYTE = 46 → BE ?
GET BYTE STRIP REG. CODE
LOOKUP & ADD DISPLACEMENT
4-BYTE LD INSTRUCTIONS
BYTE = 21, 22, OR 2A ?
GET 2 BYTES CONVERT TO HEX
ADD TO I$(D)
3-BYTE INC, DEC
BYTE = 34 OR 35 ?
GET BYTE CONVERT TO HEX
ADD TO I$(D)
4-BYTE LD w/ DISPLACEMENT
BYTE = 36 ?
GET 2 BYTES CONVERT TO HEX
ADD TO I$(D)
2-BYTE ADD, INC, DEC, PUSH, POP EX, JP, LD INDEX REG INSTRUCTIONS
STRIP OFF TYPE CODE
LOOKUP I$(D)
HAVE "CB" OP-CODE
D = "ED" ?
BUMP ADR PTR CONVERT TO HEX STRIP OFF REG. & TYPE CODE (FROM 2nd BYTE)
DO I$(D) TABLE LOOKUP — CLEAR UNUSED OPERAND STRINGS
BUMP ADR PTR CONVERT TO HEX STRIP OFF REG. CODE
THESE ARE THE 4-BYTE INSTRUCTIONS FROM THE "ED" GROUP
2nd BYTE = 43, 4B, 53, 5B, 73 OR 7B ?
BRING IN NEXT 2 - BYTES CONVERT TO HEX
DO I$(D) LOOKUP USING REG. CODE FROM 2nd BYTE — ADD DATA FROM NEXT 2 BYTES TO I$(D)
2-BYTE "ED" INSTRUCTIONS
STRIP OFF TYPE CODE
DO I$(D) TABLE LOOKUP — CLEAR UNUSED OPERAND STRINGS
YES
NO

Fig. 2.

DECLARE ANY DATA SEGMENTS question.)

TURN ON LP THEN 'ENTER'?—Be sure the printer is on, then press the Enter key. The listing will then be directed to the line printer.

To keep program length to a minimum, unnecessary spaces are deleted and many instructions are crammed onto a single line. This makes the program difficult to read, so take care when you type the code in. A bit more memory may be conserved if the number of spaces between the instruction mnemonic and the operand are reduced in all the instruction strings (i.e., all I$(D) strings). This will crowd the printout, but may be worth it if you need the extra space.

The addresses in the listing are in both hex and decimal to make it easier to go back and POKE different values into a location for experimentation or debugging of modified software. This program can be used in conjunction with EDTASM or a similar Z-80 assembler to relocate machine code programs.

Program Description

The program is divided into a main loop and a large subroutine called DISASM (see flow chart). The main loop sequences the address pointer through memory, prints the disassembled code and decides whether the pointer is within a declared data segment. DISASM does the actual disassembly.

The output statement is formatted as in Table 2.

DISASM determines values for EH$,EL$,FH$,FL$,GH$,GL$ and I$(D) while the main loop handles Q,HI$,H2$,LI$,L2$,DH$, DL$. All single byte instruction mnemonics are loaded into memory during the initialization of the main loop. At this time an array called BL(D) is also loaded. This array holds a value for each op-code (0–255) which is used by DISASM to route the program to the proper routine for disassembly. In most cases this value corresponds to the instruction byte length, but there are exceptions (mainly in the DD,ED and FD instructions which are variable in byte length).

A few smaller arrays (J$(0–7), S$(0–7), P$(0–7), N$(0–7), O$(0–3) and Q$(0–3)) are also loaded at this time and are included in order to make some of the other values of I$(D) more universal. They contain some common suffixes which are concatenated with the same I$(D) value to form a number of different instructions, thereby saving I$(D) string space.

The program is rather long and therefore does not leave much space for machine code programs over about 2K (in a 16K system). It resides in low RAM which is where a lot of Radio Shack's canned programs are written (i.e., EDTASM and T-BUG) so unless these programs are copied to higher memory they cannot be disassembled by this program.

Another point to consider which may or may not be objectionable is execution speed. Since the program is in BASIC and a lot of string operations are being performed, some instruc-

A	ADDRESS POINTER
B	END ADDRESS POINTER
C	LINE COUNTER
D	1ST DATA BYTE (CAN BE OP-CODE OR DATA)
F,G	USED TO STORE REGISTER MNEMONIC TYPE CODES
L,M,N	GENERAL COUNTER VARIABLE
P	USED TO HOLD REGISTER CODE
Q	TEMP STORAGE FOR DECIMAL VALUE OF ADR PTR
S	GENERAL COUNTER VARIABLE
X	TEMP STORAGE IN HEX CONVERSION ROUTINES
Z	USED FOR TEMP STORAGE OF 2ND DATA BYTE
H1,H2	HEX VERSION OF HIGH ORDER ADR BYTE
L1,L2	HEX VERSION OF LOW ORDER ADR BYTE
Z1	TEMP STORAGE OF 3RD DATA BYTE
Z2	TEMP STORAGE OF 4TH DATA BYTE
A(N)	START ADDRESS ARRAY FOR DATA SEGMENTS
B(N)	END ADDRESS ARRAY FOR DATA SEGMENTS
BL(D)	INSTRUCTION BYTE LENGTH ARRAY
B$	GENERAL PURPOSE STRING
F$	DUMP TITLE STRING
H$	HOLDS ASCII VERSION OF MSB IN HEX CONV ROUTINE
L$	HOLDS ASCII VERSION OF LSB IN HEX CONV ROUTINE
Q$,V$	GENERAL PURPOSE STRING
DH$	MSB OF 1ST BYTE (ASCII REPRESENTATION)
DL$	LSB OF 1ST BYTE (ASCII REPRESENTATION)
EH$	MSB OF 2ND BYTE (ASCII REPRESENTATION)
EL$	LSB OF 2ND BYTE (ASCII REPRESENTATION)
FH$	MSB OF 3RD BYTE (ASCII REPRESENTATION)
FL$	LSB OF 3RD BYTE (ASCII REPRESENTATION)
GH$	MSB OF 4TH BYTE (ASCII REPRESENTATION)
GL$	LSB OF 4TH BYTE (ASCII REPRESENTATION)
H1$	MSB OF HIGH ORDER ADDRESS (ASCII REPRESENTATION)
H2$	LSB OF HIGH ORDER ADDRESS (ASCII REPRESENTATION)
L1$	MSB OF LOW ORDER ADDRESS (ASCII REPRESENTATION)
L2$	LSB OF LOW ORDER ADDRESS (ASCII REPRESENTATION)
P1$	CONCATENATION OF FH$ + FL$ + EH$ + EL$
P2$	CONCATENATION OF EH$ + EL$
I$(D)	INSTRUCTION MNEMONIC STRING ARRAY
J$(N)	ARRAY HOLDING ASCII VERSION OF REGISTER CODES
L$(N)	HOLDS ASCII "N" AND "I"
M$(N)	HOLDS ASCII "0", "1" AND "2"
N$(N)	ARRAY HOLDING ASCII VERSION OF REGISTER PAIRS
O$(N)	HOLDS ASCII "I,A"/"R,A"/"A,I"/"A,R"
P$(N)	HOLDS ASCII MNEMONICS FOR "SBC HL"&"ADC HL"
Q$(N)	HOLDS ASCII "LD"/"CP"/"IN"/"OUT"
S$(N)	ARRAY HOLDING ASCII CODES FOR CONDITIONAL INSTRUCTIONS (I.E. "NZ", "Z", "NC", ETC.)

Table 1. Variables and Arrays

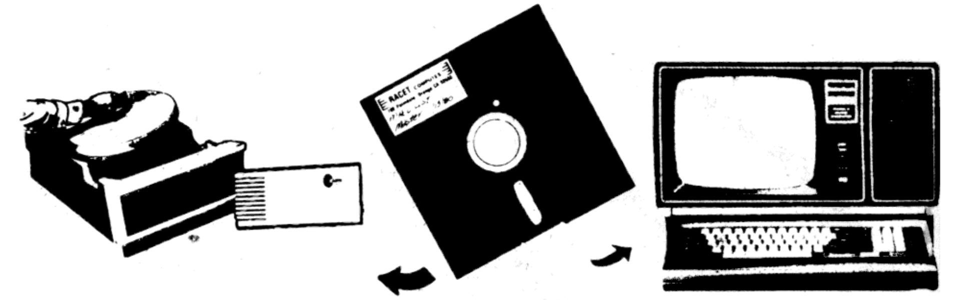
VARIABLE:	Q	HI,2$/LI,2$	DH,L$/EH,L$/FH,L$/GH,L$	I$(D)
DESCRIPT:	DEC ADR	4 CHR HEX ADR	1st-4th DATA BYTES (1st byte = OP-CODE)	MNEMONIC STRING

Table 2. DISASM Output Statement

RACET SORTS — RACET UTILITIES — RACET computes — RACET SORTS — RACET UTILITIES — RACET computes — RACET SORTS — RACET UTILITIES — RACET computes

HARD DISK MULTIPLEXOR

FOR TRS-80* Mod II



NOW YOU CAN HAVE THAT LARGE COMMON DATA BASE!!

- Allows up to 4 Mod II's to connect to a single controller — up to 4 hard disk drives per controller. Users may access the same file simultaneously (first-come first-served).
- Uses Cameo controller and standard 10-megabyte cartridge (hard) disk drives along with RACET Hard/Soft Disk System (HSD) software. Removable Disk Pack Backup!
- Access times 3 to 8 times faster than floppy. Mixed floppy/hard disk operation supported.
- Compatible with your existing TRSDOS programs — you need only change filenames! All BASIC statements are identical.
- A single file may be as large as one disk. Directory expandable to handle thousands of files.
- Includes special utilities — XCOPY for backup and copies, XPURGE for multiple deletions, DCS directory catalog system, and Hard Disk Superzap. FORMAT utility includes options for specifying sectors/gran, platters/drive, logical disk size, etc.

HARD DISK DRIVE AND CONTROLLER $5995 **RACET HSD Software $400** Call for multiuser pricing. Dealers call for OEM pricing.

★ ★ **NEW** ★ ★ **DISCAT** (32K 1-drive Min) **Mod I, III $50.00**
This comprehensive Diskette Cataloguing/Indexing utility allows the user to keep track of thousands of programs in a categorized library. Machine language program works with all TRSDOS and NEWDOS versions. Files include program names and extensions, program length, diskette numbers, front and back, and diskette free space.

★ ★ **NEW** ★ ★ **KFS-80** (1-drive 32K Min — Mod II 64K) **Mod I, III $100.00; Mod II $175.00**
The keyed file system provides keyed and sequential access to multiple files. Provides the programmer with a powerful disk handling facility for development of data base applications. Binary tree index system provides rapid access to file records.

★ ★ **NEW** ★ ★ **MAILLIST** (1-drive 32K Min — Mod II 64K) **Mod I, III $75.00; Mod II $150.00**
This ISAM-based maillist minimizes disk access times. Four keys — no separate sorting. Supports 9-digit zip code and 3-digit state code. Up to 30 attributes. Mask and query selection. Record access times under 4 seconds!!

★ ★ **NEW** ★ ★ **LPSPOOL** (32K 1-drive Min) **Mod I $75.00**
LPSPOOL — Add multi-tasking to permit concurrent printing while running your application program. The spooler and despooler obtain print jobs from queues maintained by the system as print files are generated. LPSPOOL supports both parallel and serial printers.

BASIC LINK FACILITY 'BLINK' (Mod I Min 32K 1-disk) **Mod I $25.00; Mod II $50.00; Mod III $30.00**
Link from one BASIC program to another saving all variables! The new program can be smaller or larger than the original program in memory. The chained program may either replace the original program, or can be **merged** by statement number. The statement number where the chained program execution is to begin may be specified!

INFINITE BASIC (Mod I & Mod III Tape or Disk) **Mod I $50.00; Mod III $60.00**
Extends Level II BASIC with complete MATRIX functions and 50 more string functions. Includes RACET machine language sorts! Sort 1000 elements in 9 seconds!! Select only functions you want to optimize memory usage.

INFINITE BUSINESS (Requires Infinite BASIC) **Mod I & III $30.00**
Complete printer pagination controls — auto headers, footers, page numbers. Packed decimal arithmetic — 127 digit accuracy +, −, *, /. Binary search of sorted and unsorted arrays. Hash codes.

COMPROC (Mod I & Mod III — Disk only) **Mod I $20.00; Mod III $30.00**
Command Processor. Auto your disk to perform any sequence of instructions that you can give from the keyboard. DIR, FREE, pause, wait for user input, BASIC, No. of FILES and MEM SIZE, RUN program, respond to input statements, BREAK, return to DOS, etc. Includes lowercase driver software, debounce and screenprint!

GSF (Mod I & III Tape or Disk — Specify Memory Size) **Mod I $25.00; Mod II $50.00; Mod III $30.00**
Generalized Subroutine Facilities. The STANDARD against which all other sorts are compared! Machine language — fast and powerful! Multi-key multi-variable and multi-key character string. Zero and move arrays. Mod II includes USR PEEKS and POKES. Includes sample programs.

DSM (Mod I Min 32K 2-drive system. Mod II 64K 1-drive. Mod III Min 32K 1-drive) **Mod I $75.00; Mod II $150.00; Mod III $90.00**
Disk Sort/Merge for RANDOM files. All machine language stand-alone package for sorting speed. Establish sort specification in simple BASIC command File. Execute from DOS. Only operator action to sort is to change diskettes when requested! Handles multiple diskette files! Super fast sort times — improved disk I/O times make this the fastest Disk Sort/Merge available on the TRS.

UTILITY PACKAGE (Mod II 64K) **$150.00**
Important enhancements to the Mod II. The file recovery capabilities alone will pay for the package in even one application! Fully documented in 124 page manual! XHIT, XGAT, XCOPY and SUPERZAP are used to reconstruct or recover data from bad diskettes! XCOPY provides multi-file copies, 'wild-card' mask select, absolute sector mode and other features. SUPERZAP allows examine/change any sector on diskette including track-0, and absolute disk backup/copy with I/O recovery. DCS builds consolidated directories from multiple diskettes into a single display or listing sorted by disk name or file name plus more. Change Disk ID with DISKID. XCREATE preallocates files and sets 'LOF' to end to speed disk accesses. DEBUGII adds single step, trace, subroutine calling, program looping, dynamic disassembly and more!!

BASIC CROSS REFERENCE UTILITY (Mod II 64K) **$50.00**
SEEK and FIND functions for Variables, Line Numbers, Strings, Keywords. 'All' options available for line numbers and variables. Load from BASIC — Call with 'CTRL'R. Output to screen or printer!

DEVELOPMENT PACKAGE (Mod II 64K) **$125.00**
Includes RACET machine language SUPERZAP, Apparat Disassembler, and Model II interface to the Microsoft 'Editor Assembler Plus' software package including uploading services and patches for Disk I/O. Purchase price includes complete copy of Editor Assembler + and documentation for Mod I. Assemble directly into memory, MACRO facility, save all or portions of source to disk, dynamic debug facility (ZBUG), extended editor commands.

CIRCLE READER REQUEST FOR FREE 24-PAGE CATALOG
*TRS-80 IS A TRADEMARK OF TANDY CORPORTION

CHECK, VISA, M/C, C.O.D., PURCHASE ORDER
TELEPHONE ORDERS ACCEPTED (714) 637-5016

RACET COMPUTES
1330 N. GLASSEL, SUITE 'M', ORANGE, CA 92665

✓41

RACET SORTS — RACET UTILITIES — RACET computes — RACET SORTS — RACET UTILITIES — RACET computes — RACET SORTS RACET SORTS — RACET UTILITIES —

✓See List of Advertisers on page 306

tions can take several seconds to be processed. These difficulties could be minimized were the program written in machine code. However, I'll leave this for another time. I have included a list of the variables used for those who would like to experiment or try to make the code more compact. If an 8080-only version of the program is all that is necessary it can be made considerably shorter. The program presented here will generate mnemonics for 697 instructions.■

```
10 CLS:C=0:INPUT"ENTER START ADDRESS(DEC)";A
20 INPUT"ENTER END ADDRESS(DEC)";B
30 CLS
40 Z=PEEK(A)
41 IF Z<32 GOTO50
42 IF Z>94 GOTO50
43 PRINTA,CHR$(Z)
44 GOTO70
50 A=A+1:IF A>=B GOTO100
55 GOTO40
60 PRINTA,CHR$(Z)
70 IF A>=B GOTO100
80 A=A+1:C=C+1:IF  C=15 GOTO120
90 GOTO40
100 PRINT"RUN ENDED....."
110 END
120 INPUT"PAGE FULL...HIT ANY KEY TO CONTINUE";S$
130 C=0:GOTO40
```

Program Listing 2. ASCII Display Program

```
0     0000   F3       DI
1     0001   AF       XOR    A
2     0002   C37406   JP     0674
5     0005   C30040   JP     4000
8     0008   C30040   JP     4000
11    000B   E1       POP    HL
12    000C   E9       JP     (HL)
13    000D   C39F06   JP     069F
16    0010   C30340   JP     4003
19    0013   C5       PUSH   BC
20    0014   0601     LD     B,01
22    0016   182E     JR     2E
24    0018   C30640   JP     4006
27    001B   C5       PUSH   BC
28    001C   0602     LD     B,02
30    001E   1826     JR     26
32    0020   C30940   JP     4009
35    0023   C5       PUSH   BC
36    0024   0604     LD     B,04
38    0026   181E     JR     1E
40    0028   C30C40   JP     400C
43    002B   111540   LD     DE,4015
46    002E   18E3     JR     E3
48    0030   C30F40   JP     400F
51    0033   111D40   LD     DE,401D
54    0036   18E3     JR     E3
56    0038   C31240   JP     4012
59    003B   112540   LD     DE,4025
62    003E   18DB     JR     DB
64    0040   C3D905   JP     05D9
67    0043   C9       RET
68    0044   00       NOP
69    0045   00       NOP
70    0046   C3C203   JP     03C2
73    0049   CD2B00   CALL   002B
76    004C   B7       OR     A
77    004D   C0       RET    NZ
78    004E   18F9     JR     F9
80    0050   0D       DEC    C
81    0051   0D       DEC    C
82    0052   1F       RRA
83    0053   1F       RRA
84    0054   01015B   LD     BC,5B01
87    0057   1B       DEC    DE
88    0058   0A       LD     A,(BC)
89    0059   1A       LD     A,(DE)
90    005A   08       EX     AF,AF'
91    005B   1809     JR     09
93    005D   19       ADD    HL,DE
94    005E   2020     JR     NZ,20
```

Program Listing 3. Sample Disassembly Listing

Get Level I printouts from Level II.

LList For Level I

Everett Ogden
16 Herber Ave.
Delmar, NY 12054

One of the many shortcomings of Level I is the difficulty of obtaining a printout of a program. Radio Shack has a screen printer, but the image leaves a lot to be desired. The method described here requires you have access to a Level II with a printer. That may not be an insurmountable obstacle.

If you belong to a computer club you probably know someone with that equipment. Schools that use the TRS-80 for training are likely to have both systems and, if you can't find a Level II and printer any other way, you may be able to talk your local Radio Shack store into letting you use theirs.

Using Level II's CONV, you can convert Level I programs to Level II, but that will expand abbreviated statements and change PRINT AT to PRINT @. It's easy, however, to get a true listing of the Level I version.

CONV checks a list of Level I statements to see which of them are used in the program. If the part of the program it is working on is not on the list, it leaves it intact and goes on. At the end of the list is a jump address if no match is found. If this address is moved to the head of the list, no match will ever be found and no conversions performed.

When you have found your Level II do the following:

- Load the Level I/Level II conversion program, but do not jump to it. Return to the BASIC monitor.
- Enter the following instructions:

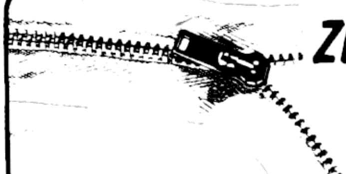
```
POKE 32468, 251
POKE 32469, 222
```

This puts the "no match" address at the head of the list.

- Now jump into CONV and follow the normal conversion procedure. The entry address is 31478.

When you list the program you will find that it is still in Level I format. You can now LLIST it. Don't try to run it, though, because Level II will hang up on the abbreviations and PRINT AT statements. Statements that were written out in full will have been properly converted to Level II tokens because that is done in another part of the program.

I discovered this idea while converting a program that used the abbreviation M. (MEM). It hung up on that line, and when I listed it I saw M. had not been converted. When I disassembled CONV, I found out why: Radio Shack left that word out! ■



✓See List of Advertisers on page 306

BUSINESS SOFTWARE

Program	Media	Machine	Price
Lazy Writer	D	Mod I	$125.00
Lazy Writer	D	Mod III	$175.00
Postman Data Handler	D	Mod I	$125.00
Graphix	D	Mod I or III	$49.95
Critpath	D	Mod I	$600.00
Critpath	D	Mod III	$750.00
Compleat Idiots Bookeeper	D	Mod I or III	$49.95

UTILITY SOFTWARE

Super Utility	D	Mod I	$49.95
Super Utility	D	Mod I	Call
Quick Fix	D	Mod I	$34.95
Chain Maker (for ND-80 only)	D	Mod I	$14.95
Tape Copy 1	D	PMC/Mod I	$14.95
Tape Copy 2	D	Mod I and III	$14.95
Hyres	D	Mod I	$9.95
Bug +	D	Mod I and III	$14.95
Make 80	D	Mod I	$14.95
Double Zap (ND-80)	D	Mod I	$49.95
Double Zap (LDOS)	D	Mod I	$39.95
Easy Term	D	Mod I or III	$59.95

GAMES

Space Colony	C	Mod I	$14.95
TRS Super Invaders	C	Mod I	$19.95
Super Vaders	C/D	Mod III	$19.95
Attack Force	C	Mod I or III	$15.95
Galaxy Invasion	C	Mod I or III	$15.95
Super Nova	C	Mod I or III	$15.95
Meteor Mission II	C	Mod I or III	$15.95
Cosmic Fighter	C	Mod I or III	$15.95
Symon Capture	C	Mod I or III	$12.95
Flippy 1.3	C	Mod I or III	$14.95

All the game prices are for the cassette version only. The disk version is extra. Call for price.

OPERATING SYSTEMS

NEWDOS/80	D	Mod I	$149.95
LDOS	D	Mod I	$139.95

BOOKS

Mod III Rom Commented	Book	Mod III	$22.50 + 2.50 Shipping

SPECIAL

If you purchase NEWDOS/80 receive CHAIN MAKER at now extra charge!
Only $149.00
plus $2.50 shipping and handling

Purchase LAZY WRITER & POSTMAN DATA HANDLER (Mod I version only)
for $250.00
RECEIVE POSTWRITER AT NO EXTRA CHARGE
This month only

SUPER SPECIAL OF THE MONTH

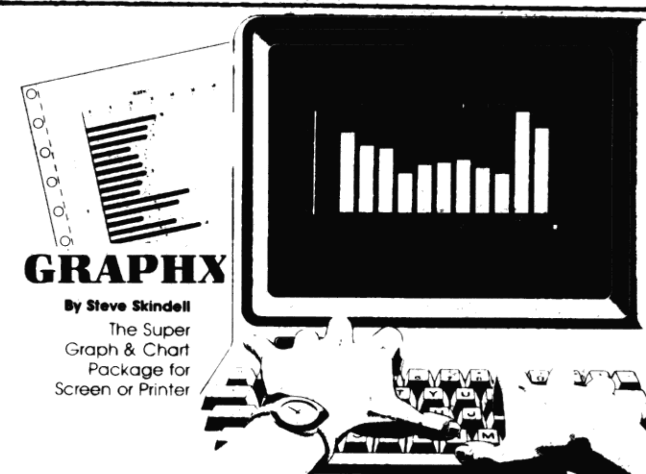
Purchase any business package from Soft Sector Marketing, Inc. this month and receive QUICK FIX for only **$25.00**
This is a $34.95 value.

MasterCard

SEND FOR OUR NEW FREE CATALOG!
Or catalog is included with all orders.

VISA

See last month's 80 Micro for more details on our software or send for overviews.



By Steve Skindell• 1981. This is a program that is for the person who does reports or requires some sort of plotted output to show gains or losses, or any type of output that needs graphs. This program puts to the screen or to a printer the plotted points in bar graph format and plots the mean, and the averages of input. This is a very important program for accountants, CPAs and the average businessman to evaluate, at a moment's glance where he is, was, or where he is going. Files saved to disk can be recalled at any time to be reexamined, modified, or just reprinted. An extra feature: if you have the Microline 80 printer, by Okidata or Epson MX-80, your output is in true graphics. Information is supplied for the user so he can modify this program for other printers. Comes complete and ready to run. Requires MOD I or MOD III. 48K disk. Printer optional (132 col.) Only **$49.95**

MX-80 PACKAGE SPECIAL!

MX-80* – New

You Pay Only **$599.00**
Save Up To **$162.95***
Hurry – Supply Limited!
And At No Extra Cost To You:
1 Box of Paper (2700 sheets 20# white) - 2 Extra Ribbons - 1 Extra Print Head and Shipping Included for all UPS in U.S.
***Compared to all items at list price!**
*for MX-80 f/t add $150.00
Sorry, no C.O.D. on this special. To receive this discount you must prepay.
If you don't require all these extra items, call for price.

HOW TO ORDER

SSM SOFT SECTOR MARKETING, INCORPORATED
6250 Middlebelt • Garden City, MI 48135 • 1 (313) 425-4020

Dealer Inquiries Invited. ✓515

C.O.D. - certified check, M.O. or cash only. Sorry, no C.O.D. over $150.00! Most orders shipped next day. All orders must have shipping included. Please add 2% or $2.50, whichever is higher, for shipping (unless otherwise stated). Out-of-country orders, please add $10.00 additional shipping and handling, balance returned upon shipment.

*TRS-80 is a product of Radio Shack, division of the Tandy Corporation.

See previous page for ordering info.

If the curve fits, snare it!

Regression And Correlation

C. Brian Honess
22 Shaftesbury Lane
Columbia, SC 29209

The techniques of regression and correlation are used for forecasting, predicting, determining the degree of relation between variables and how well some equation describes that relationship, etc. We'll develop a program to find the best fitting straight line in a set of plotted data values, the degree of correlation between the variables, and look at the problem of fitting a parabolic curve to the data.

I'll use two variables, considering the traditional problem of height and weight. We'll test a group of people to determine what the relation of height to weight is, and use the results to predict the weight of someone with a known height. Collecting data from ten people, we produce Table 1.

Use the variable X for the height, and the variable Y for the weight. Also, attach a subscript to each data value or pair of values, so that X_5 will correspond to the height of the fifth person, and the height and weight of the third person can be represented as X_3, Y_3.

The next step is to plot each pair of points. Since we want to determine a person's weight, given his or her height, we'll call Y the dependent variable, and plot it on the Y axis. The set of plotted points forms a scatter diagram.

Subject No.	Height (in)	Weight (lbs)
1	54	100
2	72	160
3	60	150
4	74	195
5	62	110
6	75	180
7	66	135
8	57	135
9	66	150
10	60	120

Table 1. Height-weight Chart

We don't need to plot a scatter diagram to find the best fitting straight line through the data points. The scatter diagram will be useful when determining if a straight line is the best curve to fit.

A straight line appears to fit the scatter diagram in Fig. 1 about as well as any other curve. I've drawn one in, guessing at where it would be. This freehand method won't be good enough for all applications, and we'll want to code a program to find the best fitting line.

The equation for a straight line is $Y = a + bX$, where a is the Y intercept (the place where the line crosses the Y axis), and b is the slope (the angle the line makes with the X axis).

The Y intercept is difficult to guess at, since our scatter diagram is not drawn so the origin is in correct perspective.

We'll guess that the Y intercept is about -90. The slope will be easier to estimate, since our "guess line" goes through two data points in the scatter diagram. One point has an X value of 54 and a Y value of 100, the other point has an X value of 66 and a Y value of 150. Therefore:

$$\text{slope} = \frac{\text{rise}}{\text{run}} = \frac{Y_2 - Y_1}{X_2 - X_1} = \frac{150 - 100}{66 - 54} = 4.16667$$

The method we'll use to find the best fitting line is called the method of least squares. We'll take the sum of the distances between each point and the line, squared, move the line around until the sum is minimized, and call this the best fitting line. The resulting line is also called a regression curve of Y on X, since Y is the dependent variable in this example.

Now code a BASIC program to calculate a and b. First determine the maximum number of data values so we can code the DIMension statement. We'll use 100 valid data points; if you have more, raise the vector sizes in the DIM statement. We'll use the trailer principle, so we'll have to DIMension an additional storage location to hold the trailer value.

```
10 DIM X(101), Y(101)
```

Key the data pairs into a program statement, and get into a loop to read the values. You may alter the program so that you can enter the X values all at once, then enter the Y values. I'm going to enter the data values as pairs (X_1, Y_1; X_2, Y_2; X_3, . . . , Y_n).

Inside the loop we'll determine whether the first number in

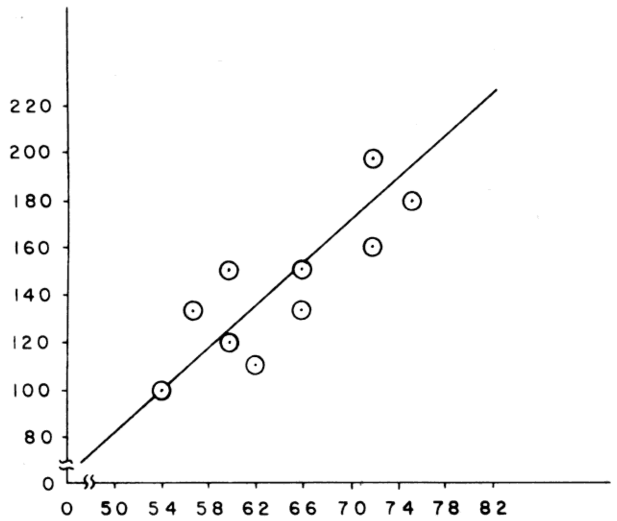


Fig. 1. Plot Diagram

a pair is negative. If it is, we've found the trailer, and can exit the loop. Subtract one from the loop index after exit. The result will be stored in variable N, and tells us how many pairs of data values we have.

Let's assume our data is always positive. If your data doesn't fit this assumption, change the value of the trailer so that it is "more negative" than your smallest data value. You may want to set up a permanent trailer value of some very large negative number.

```
20 FOR I=1 TO 101
30 READ X(I), Y(I)
40 IF X(I)<0 THEN 60
50 NEXT I
60 N=I-1
70 DATA 54,100,72,160,60,150,74,195
71 DATA 62,110,75,180,66,135,57,135
72 DATA 66,150,60,120,-1,-1
```

Data statements can go anywhere in the program, but there is one trick. You'll notice that I've used two trailer values; the READ statement in line 30 expects to read two values—if it doesn't, it won't go to line 40 to check for the trailer. The second trailer value could be any number.

The program, thus far, will have loaded the first 11 locations of the X and Y vectors with their data values, and returned a value of 10 for N. (The eleventh value in each of the two vectors will be the trailer value, a -1.)

Next Step

The next step is to calculate the values of a and b in the equations given earlier.

$$a = \frac{(\Sigma Y)(\Sigma X^2) - (\Sigma X)(\Sigma XY)}{N\Sigma X^2 - (\Sigma X)^2}$$

$$b = \frac{N\Sigma XY - (\Sigma X)(\Sigma Y)}{N\Sigma X^2 - (\Sigma X)^2}$$

Note: I've used some shortcut notation: Every time you see the Σ symbol, it means to sum all values of the corresponding variable. In other words:

ΣXY means: $\sum_{i=1}^{n} X_iY_i.$

The formulae indicate that we need to find four sums: the Y values, the X values, the XY values, and the X^2 values. The formulae don't show the sum of the Y^2 values, but we're going to find it anyway, to use later.

Name the variables and then code the summation process:

Sum of X values	SX
Sum of Y values	SY
Sum of X^2 values	X2
Sum of Y^2 values	Y2
Sum of XY values	XY

BASIC automatically puts a value of zero into each variable, so you can skip lines 80 through 84 if you wish.

```
80 SX=0
81 SY=0
82 X2=0
83 Y2=0
84 XY=0

90 FOR I=1 TO N
100 SX=SX+X(I)
110 SY=SY+Y(I)
120 X2=X2+X(I)*X(I)
130 Y2=Y2+Y(I)*Y(I)
140 XY=XY+X(I)*Y(I)
150 NEXT I
```

The denominators are the same in the equations for calculating a and b. We have to calculate this only once, and call the results D. We then continue to calculate a and b.

```
160 D=N*X2-SX*SX
170 A=(SY*X2-SX*XY)/D
180 B=(N*XY-SX*SY)/D
190 PRINT "Y-INTERCEPT=";A
200 PRINT "SLOPE       =";B
210 PRINT
220 PRINT "EQUATION OF
    BEST FITTING LINE:"
230 PRINT
240 PRINT "   Y=";A;"+";B;"* X"
250 PRINT
```

If you key in the program with the test data, you should get a Y intercept of -84.4519 and a slope of 3.52867. The equation is: $Y = -84.4519 + 3.52867\ X$. We weren't too far off with guesses of -90 and 4.16. The best fitting line is one that is rotated slightly clockwise from the guess line we drew.

Now that we know the equation of the best fitting line, we can use it to predict values of Y (weight) when given values of X (height). For example, assume someone is 5'10" tall. This is 70 inches: we'd enter the equation with an X value of 70.

$$\begin{aligned} Y &= -84.4519 + (3.52867 \cdot 70) \\ &= -84.4519 + 247.007 \\ &= 162.555 \text{ lbs.} \end{aligned}$$

We can get the computer to do this for us with a little more coding:

```
260 INPUT "WANT TO PREDICT Y VALUES
    ? (1=YES, 0=NO)"; K
270 IF K=0 THEN 320
280 INPUT "ENTER AN X VALUE "; XX
290 YY=A+B*XX
300 PRINT " PREDICTED Y IS: "; YY
310 GO TO 260
320
```

Note: I used XX and YY in this

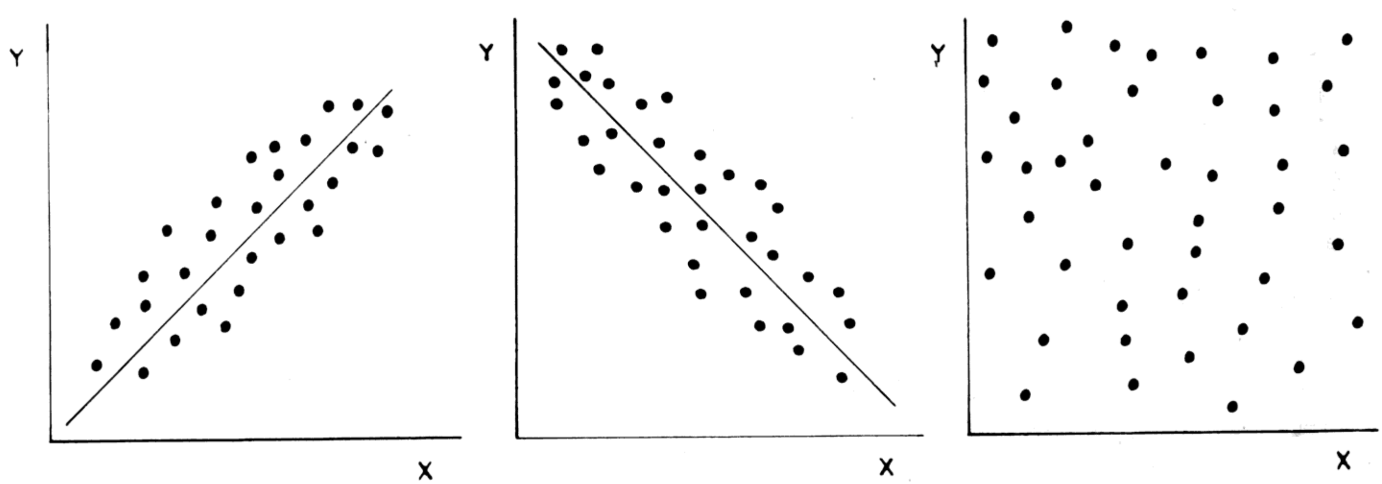


Fig. 2. Positive, Negative and Uncorrelated Variable Diagrams

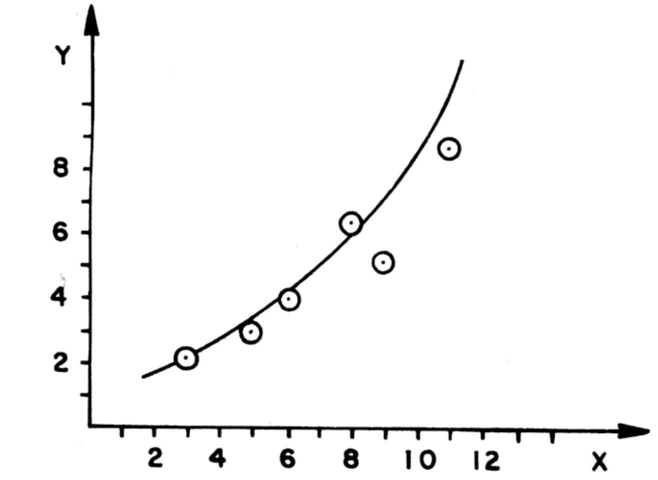


Fig. 3. Scatter Diagram of Quadratic Curve

segment, since X and Y have been used for other purposes in the program. In fact, they are each dimensioned variables, and have 101 locations each.

Correlation

In our example, the value of Y increased as the value of X increased. This is a positive, or direct, correlation. If the value of Y decreased as the value of X increased, it would be a negative, or inverse, correlation. If there didn't seem to be any relation between the two variables, this would suggest no correlation (uncorrelated). These cases are illustrated in Fig. 2.

We can go one step farther, and say that each of the first two examples has a linear correlation—there is a linear relationship between the variables, and a straight line would be the best choice for a regression equation.

A popular way to measure the degree of correlation is by calculating the coefficient of correlation r which can assume any value between -1 and $+1$. A value of -1 indicates a negative correlation in which all data values lie on the regression line. A value of $+1$ indicates a positive correlation in which all data values lie on the regression line. An r value of zero indicates an uncorrelated relationship. In our example we'd expect an r value somewhere between zero and $+1$, since there is a positive correlation between height and weight, but all values do not lie on the regression line.

$$r = \frac{N\Sigma XY - (\Sigma X)(\Sigma Y)}{\sqrt{(N\Sigma X^2 - (\Sigma X)^2)\cdot(N\Sigma Y^2 - (\Sigma Y)^2)}}$$

This won't be nearly as bad as it seems, because we've already found all the sums we need to substitute into the equation.

The thought of keying that equation into one line scares me; let's do it in pieces. Do the numerator first, call it NU; then do the two major terms in the denominator, calling them T1 and T2. Finally, we'll put it all together and find r.

```
320 NU=N*XY-SX*SY
330 T1=N*X2-SX*SX
340 T2=N*Y2-SY*SY
350 R=NU/SQR(T1*T2)
360 PRINT "CORRELATION COEFFICENT
    (R)=";R
```

When you run this with the example data, you'll find r = .856481, indicating a strong positive correlation.

The next program will do nonlinear regression. In other words, we'll be fitting curved lines to a set of data points.

Consider the following set of values:

X	Y
3	2
5	3
6	4
8	6
9	5
11	8

Once again, Y is the dependent variable. We'll again plot a scatter diagram, Fig. 3, to aid our visualization of the relation between X and Y. This time it isn't a straight line, but a parabola, or quadratic curve. The general equation for a parabola is: $Y = a + bX + cX^2$. We'll calculate the values of the coefficients a, b, and c.

The coefficients a, b, and c can be found by solving the following set of simultaneous equations:

$$\Sigma X = aN + b\Sigma X + c\Sigma X^2$$
$$\Sigma XY = a\Sigma X + b\Sigma X^2 + c\Sigma X^3$$
$$\Sigma X^2Y = a\Sigma X^2 + b\Sigma X^3 + c\Sigma X^4$$

This time we need seven sums; name them and start coding the program. DIMension the X and Y vectors as we did before, use the trailer principle to read in the values, and initialize each of the sums to zero, if you like.

Sum in formulae	BASIC Variable Name
ΣX	SX
ΣX^2	X2
ΣX^3	X3
ΣX^4	X4
Σ- XY	XY
ΣY	SY
Σ- X- ²Y	YX

```
10 DIM X(101), Y(101)
20 FOR I=1 TO 101
30 READ X(I), Y(I)
40 DATA 3,2,5,3,6,4,8,6,9,5,11,8,-1,-1
50 IF X(I)<0 THEN 70
60 NEXT I
70 N=I-1
80 SX=0
90 X2=0
100 X3=0
110 X4=0
120 XY=0
130 SY=0
140 YX=0
150 FOR I=1 TO N
160 SX=SX+X(I)
170 X2=X2+X(I)*X(I)
180 X3=X3+X(I)*X(I)*X(I)
190 X4=X4+X(I)*X(I)*X(I)*X(I)
200 XY=XY+X(I)*Y(I)
210 SY=SY+Y(I)
220 YX=YX+X(I)*X(I)*Y(I)
230 NEXT I
```

I added a temporary PRINT statement to the program, 235 PRINT SX;X2;X3;X4;XY;SY;YX, and got the following values:

42 336 2940 27300 226 28 1994

You can check your keying to this point.

If we substitute these values into the three simultaneous equations, we have:

$$28 = a6 + b42 + c336$$
$$226 = a42 + b336 + c2940$$
$$1994 = a336 + b2940 + c27300$$

Written in a more conventional form:

$$6a + 42b + 336c = 28$$
$$42a + 336b + 2940c = 226$$
$$336a + 2940b + 27300c = 1994$$

We'll now have to solve three simultaneous equations with three unknowns (a, b, c) using determinants. Determinants will seem complex at first, but the method will be easy to code, and will automatically produce answers.

Determinants

The determinant of a square matrix of order two is defined in Fig. 4. Find the product of the two numbers on the diagonal sloping down to the right, and then subtract the product of the two numbers on the other diagonal (Fig. 5). Try finding the solution to a determinant with some numerical values, as in Fig. 6. We have to solve determinants of order three, however, which is a matter of solving three determinants of order two, and multiplying (Fig. 7).

If we write down the coeffi-

$$\begin{vmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{vmatrix} = x_{11}x_{22} - x_{12}x_{21}$$

Fig. 4.

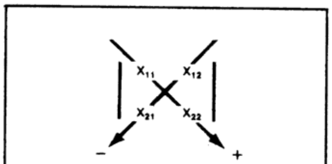


Fig. 5.

$$\begin{vmatrix} 5 & 8 \\ -2 & 4 \end{vmatrix} = (5\cdot4) - (8\cdot -2) = 20 - (-16) = 36$$

Fig. 6.

HIGH SPEED CASSETTE SYSTEM

Poor Man's Floppy

FOR TRS-80*

Now the widely acclaimed JPC Cassette System is available for your TRS-80* computer. The price is only $90.00

[Reprint of June 1980 Review, *80 Microcomputing*]

TC-8 Cassette System
JPC Products
Albuquerque, NM
Kit: $90
Assembled: $120

by Carl A. Kollar

I guess I don't have to tell any TRS-80 owners how frustrating the cassette system that comes with the computer can be. Even with the factory mod that's available, the annoyance of loading and checking programs becomes just barely tolerable.

If you're like me, after you've just plunked down a chunk of money for a Level II 16K machine, "you ain't got nuttin left" for even one disk drive at 500 bucks apiece. So you suffer.

A reasonable alternative is the Exatron Stringy Floppy (ESF). This will cost you about 250 bucks and totally eliminates your loading and saving problems, automatically and fast. I've had one of these for about six months and love it!

But, if the price is still too steep, have I got a device for you!

The Device

The February 1980 issue of *Microcomputing* had an ad that intrigued the hell out of me. It was a high-speed cassette system by JPC Products acclaimed as a "poor man's floppy." It made all sorts of seemingly ridiculous claims such as "loads five times faster," "stores 50,000 bytes on a 10-minute cassette," "less than one bad load in a million bytes with the volume control anywhere between one and eight."

All this for a measly [90] bucks? How could this be? A call to Albuquerque answered a few questions: Yes, it had its own power supply, and, it stored programs five times faster because it utilized higher density data. The computer outputs the information at a higher rate out of the rear keyboard connector.

The ad had even claimed anyone could build it even if you have never soldered before. JPC would make it work, if you couldn't—for free. I was sold. I placed my order, and it arrived about two months later (parts shortage).

I work in electronics, so I found the unit exceptionally easy to build. It took about an hour. The manual is superb. (That's better than great.) It was clear, concise and exact with no ambiguities. Important parts placements are stressed (polarity markings on electrolytics, bands on diodes, etc.).

JPC was right! With these instructions, you couldn't go wrong. The board quality is excellent. It is double-sided and parts locations are clearly marked on the component side of the board. There are no jumper wires to install. JPC utilizes PC traces and plated-through holes for connections to traces on the other side of the board.

Also, there are absolutely no adjustments or settings to bother with.

The documentation is a sheaf of 8½ × 11 papers stapled together. It is written in the nicest format I've seen in a while. Each command and/or subjects is covered on its own sheet in large type. All explanations are in easy to read English—not computerese.

Commands and Features

SAVE"filename": Saves your BASIC program on cassette.
LOAD: Reads the next BASIC program from the cassette.
LOAD"filename": Searches for and loads the specified file from cassette.
LOAD? and LOAD?"filename": Reads file from cassette, and compares contents to memory.
LOADN: Prints a list of all the programs on a cassette, until interrupted by the "break" key.
LOADN"filename": Same as above except the tape will stop at the end of the program named.
KILL: Removes the file manager program from memory so that the extra memory can be used by large programs.
RSET: Allows the operator to rewind and position the tape on tape recorders that have these functions tied to the motor control jack.
RUN"filename": TC-8 searches for a specified program and runs it immediately.
PUT"filename": Same as SAVE "filename", except it is for use with system tapes.
GET: Same as LOAD, except it is for use with system tapes.
GET"filename": Same as LOAD "filename", except it is for use with system tapes.
GET? and GET?"filename": Same as LOAD? and LOAD?"filename", except it is for use with system tapes.
GETN and GETN"filename": Same as LOADN and LOADN"filename", except it is for use with system tapes.
OPEN: Required before cassette input or output of a data file can be attempted.
CLOSE: Required to end a cassette data file.
PRINT#: Allows numerical or string data to be output to a cassette file.
INPUT#: Allows numerical or string data to be input from a cassette file.

I haven't counted them, so I don't know about the "one load in a million bytes" claim, but my son, Anthony (age 11), loaded about 30 of his programs from his Radio Shack format tape to a new TC-8 format tape. He's run them all and found no bad loads.

Unlike the standard tape system, you can position your tape anywhere before the program you want and not have to look for a blank spot between programs. The TC-8 patiently waits for the program you want and then starts loading without getting confused by the portion of the previous program you just fed it.

Try that on your regular cassette system; you'll wear out the reset button. ■

ORDER NOW

To order your TC-8 kit, send your check or money order for $90.00 plus $3.50 postage and handling to JPC PRODUCTS CO., 12021 Paisano Ct., Albuquerque, NM 87112 (New Mexico residents add 4% sales tax). Credit card orders accepted by phone or mail. Personal checks will delay shipment. We will otherwise immediately ship you the TC-8 kit, the cabinet, the ribbon cable, the power adapter, an instruction manual, and a cassette containing the software.

✓190

JPC PRODUCTS CO.
Phone (505) 294-4623
12021 Paisano Ct.
Albuquerque, N.M. 87112

✓See List of Advertisers on page 306

$$\begin{vmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{vmatrix} =$$

$$x_{11}\cdot\begin{vmatrix} x_{22} & x_{23} \\ x_{32} & x_{33} \end{vmatrix} - x_{12}\cdot\begin{vmatrix} x_{21} & x_{23} \\ x_{31} & x_{33} \end{vmatrix} + x_{13}\cdot\begin{vmatrix} x_{21} & x_{22} \\ x_{31} & x_{32} \end{vmatrix}$$

Fig. 7.

cients of a, b, and c from our three simultaneous equations in determinant form, we have Fig. 8.

I've made a column vector of the three values to the right of the equals signs in the equations, to the right of my 3 × 3 matrix, just for reference.

Here's the trick: the 3 × 3 matrix is going to become the denominator in three division operations, each of which will find a, b, or c. The three numerators will be formed by substituting the column vector to the right for each of the three columns of the 3 × 3 matrix, in turn. In other words, we calculate as shown in Fig. 9.

On the left are the values for the example we're working with, and on the right are the variable names for the general case. The six 3 × 3 matrices on the right are the ones we'll code a program to solve. Each of the six will be solved using the general method for a 3 × 3 determinant.

The denominators are all the same, so let's begin by writing the code to find the determinant of the denominator:

```
240 D1 = X2*X4 - X3*X3
250 D2 = SX*X4 - X3*X2
260 D3 = SX*X3 - X2*X2
270 D  = (N*D1) - (SX*D2) + (X2*D3)
```

If you want to check your progress, put in the temporary PRINT statement 275 PRINT D, and you should get a value of 63504.

Now write the code for calculating the other three determinant values, and solve for a, b, and c, using the same methods we used for the denominator.

```
280 A1 = X2*X4 - X3*X3
290 A2 = XY*X4 - X3*YX
300 A3 = XY*X3 - X2*YX
310 A = ((SY*A1) - (SX*A2) + (X2*A3))/D
320 B1 = XY*X4 - X3*YX
330 B2 = SX*X4 - X3*X2
340 B3 = SX*YX - XY*X2
350 B = ((N*B1) - (SY*B2) + (X2*B3))/D
360 C1 = X2*YX - XY*X3
370 C2 = SX*YX - XY*X2
380 C3 = SX*X3 - X2*X2
390 C = ((N*C1) - (SX*C2) + (SY*C3))/D
400 PRINT
410 PRINT "A = "; A
420 PRINT "B"; B
430 PRINT "C = "; C
```

$$a = \frac{\begin{vmatrix} 28 & 42 & 336 \\ 226 & 336 & 2940 \\ 1994 & 2940 & 27300 \end{vmatrix}}{\begin{vmatrix} 6 & 42 & 336 \\ 42 & 336 & 2940 \\ 336 & 2940 & 27300 \end{vmatrix}} = \frac{\begin{vmatrix} SY & SX & X2 \\ XY & X2 & X3 \\ YX & X3 & X4 \end{vmatrix}}{\begin{vmatrix} N & SX & X2 \\ SX & X2 & X3 \\ X2 & X3 & X4 \end{vmatrix}}$$

$$b = \frac{\begin{vmatrix} 6 & 28 & 336 \\ 42 & 226 & 2940 \\ 336 & 1994 & 27300 \end{vmatrix}}{\begin{vmatrix} 6 & 42 & 336 \\ 42 & 336 & 2940 \\ 336 & 2940 & 27300 \end{vmatrix}} = \frac{\begin{vmatrix} N & SY & X2 \\ SX & XY & X3 \\ X2 & YX & X4 \end{vmatrix}}{\begin{vmatrix} N & SX & X2 \\ SX & X2 & X3 \\ X2 & X3 & X4 \end{vmatrix}}$$

$$c = \frac{\begin{vmatrix} 6 & 42 & 28 \\ 42 & 336 & 226 \\ 336 & 2940 & 1994 \end{vmatrix}}{\begin{vmatrix} 6 & 42 & 336 \\ 42 & 336 & 2940 \\ 336 & 2940 & 27300 \end{vmatrix}} = \frac{\begin{vmatrix} N & SX & SY \\ SX & X2 & XY \\ X2 & X3 & YX \end{vmatrix}}{\begin{vmatrix} N & SX & X2 \\ SX & X2 & X3 \\ X2 & X3 & X4 \end{vmatrix}}$$

Fig. 9.

```
440 PRINT
450 PRINT "BEST FITTING PARABOLA:"
460 PRINT
470 PRINT "Y = "; A; " + "; B; "*X + "; C; "*
    X↑2"
480 PRINT
490 END
```

$$\begin{vmatrix} 6 & 42 & 336 \\ 42 & 336 & 2940 \\ 336 & 2940 & 27300 \end{vmatrix} \qquad \begin{vmatrix} 28 \\ 226 \\ 1994 \end{vmatrix}$$

Fig. 8.

Key this in, adding it to the first part of the program, run it, and you should get:

A = .666667
B = .380952
C = .0238095

The equation for the best fitting parabola, therefore, is:

$$Y = 0.666667 + 0.380952X + 0.0238095X^2$$

You can use this equation for making predictions of Y after keying in X values, just as we did with the linear regression model. Check lines 260 through 310 in the previous program. Lines 260, 270, and 280 should be the same. Line 290 would become: YY = A + B*XX + C*XX↑2 and lines 300 and 310 would be the same. Of course, you'd have to adjust the various line numbers to correspond to the later placement in the program. ■

It does just that!

The Variable Lister

John L. Webster
9606 Todd Mill
Huntsville, AL 35803

On several occasions, I have wished that my TRS-80 would list out the variables used in a program. This would help in documenting software and selecting variables during the development of a new program. After waiting a few months for someone else to publish an answer to my wish, I gave in and worked out my own solution. Surprisingly, it was easier than anticipated.

PEEKing at the storage area after entering a variety of variables led to the discovery that a numerical flag such as two, three, four, or eight is used to signal which type of variable follows. A flag two signifies an integer variable, a three indicates a string variable, a four indicates single precison (unspecified TRS-80 variables fall into this category by default), and an eight signals a double precision variable.

The next character in sequence is the ASCII value for the subscript or second portion of the variable name. The third character is the ASCII value of the first portion of the variable name.

The contents of addresses 40F9 and 40FA (16633 and 16634) give the starting address for the variable storage areas.

My Variable Lister program begins by PEEKing at 16633 and 16634 then goes to the indicated memory area and PEEKs and tests for the numerical flags two, three, four, or eight. I used odd-ball variable names (ZV through ZY) in the Variable Lister program so that it would not make use of variable names normally found. Line 10015 in the program tests for the first of these odd-balls, "ZV", and tells us when the searching of variables has reached the searching program's own storage area. At that point we branch to line 10070 and print out the results of the variables found.

Using Variable Lister requires that it be entered at the end of the program whose variables are to be searched out and listed. Since it is only an 18-line routine, this might be tolerable; however, it can also be merged with any program using smaller line numbers. As an experiment I merged the Variable Lister with Radio Shack's Backgammon demo program using the following procedure:

- Enter and CSAVE the Variable Lister on cassette.
- CLOAD the program whose variables are to be found and listed.
- Do a PRINT PEEK(16633),PEEK(16634). Write down the results.
- If PEEK(16633) was equal to or greater than two go to step six.
- POKE 16548,PEEK(16633) + 254:POKE 16549,PEEK(16634) – 1 go to step seven.
- POKE 16548,PEEK(16633) – 2:POKE 16549,PEEK(16634)
- CLOAD the Variable Lister.
- POKE 16548,233:POKE 16549,66
- Run the program. (This is an important step!)
- Break and then GOTO 10000.

I have observed that when the Variable Lister is appended to a target program, the target program must be run before the GOTO 10000. Otherwise, the variables are not found.

To those of you who have been waiting for someone else to publish a variable listing procedure, I hope that this simple program will satisfy your wish. It doesn't do the array variable but they are usually flagged by their DIM statement. ■

```
5000 REM *** INSTRUCTIONS ***
5010 CLS:PRINTTAB(10);"VARIABLE-LISTER INSTRUCTIONS"
5020 PRINT:PRINT"THIS PROGRAM MUST BE APPENDED TO THE TARGET PRO
GRAM
5030 PRINT"BY THE FOLLOWING METHOD:"
5040 PRINT"(1) CLOAD THE TARGET PROGRAM."
5050 PRINT"(2) PRINTPEEK(16633),PEEK(16634) WRITE DOWN RESULTS."

5060 PRINT"(3) IF PEEK(16633)=>2 THEN SKIP STEP 4 AND DO STEP 5.
"
5070 PRINT"(4) POKE16548,PEEK(16633)+254:POKE16549,PEEK(16634)-1
"
5075 PRINT"JUMP TO STEP 6."
5080 PRINT"(5) POKE16548,PEEK(16633)-2:POKE16549,PEEK(16634)
5090 PRINT"(6) CLOAD VARIABLE LISTER PROGRAM"
5100 PRINT"(7) POKE16548,233:POKE16549,66"
5110 PRINT"(8) RUN THE TARGET PROGRAM! ( THIS IS ESSENTIAL! )."
5120 PRINT"(9) 'BREAK' AND 'GOTO10000' TO EXECUTE LISTER."
5150 END
10000 REM *** VARIABLE LISTER ***
10005 ZV=PEEK(16633)+256*PEEK(16634)
10010 ZW=PEEK(ZV):ZX=PEEK(ZV+1):ZY=PEEK(ZV+2)
10015 IF ZY=90 AND ZX=86 THEN 10070
10020 LPRINT CHR$(ZY);CHR$(ZX);
10030 IF ZW=2 THEN 10080
10040 IF ZW=3 THEN 10090
10050 IF ZW=4 THEN 10100
10060 IF ZW=8 THEN 10110
10070 LPRINT:LPRINT"VARIABLE SEARCH ENDED":STOP
10080 LPRINT" % INTEGER":ZV=ZV+5:GOTO10010
10090 LPRINT"$ ";PEEK(ZV+3);" CHAR. LONG":ZV=ZV+6:GOTO10010
10100 LPRINT" ! SINGLE PRECISION":ZV=ZV+7:GOTO10010
10110 LPRINT" # DOUBLE PRECISION":ZV=ZV+11:GOTO10010
10200 PRINT"ARRAY VARIABLES ARE NOT INCLUDED":END
```

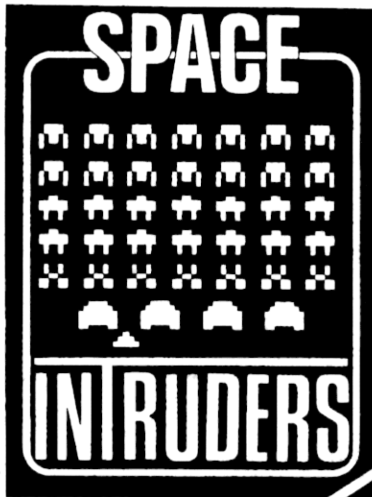
Program Listing

SPACE INTRUDERS

As battalions of evil invaders come out of the sky, you jump into your laser turrent and start defending the Earth from their unending attack.

This game is the best rendition yet of the most popular arcade game in history. Designed to emulate the Deluxe version, this game incorporates all the intrinsities of the original from the "SOS" of escaping intruders to the splitting invaders.

Even if you have one of the other versions on the market, you'll still consider this program a must for your collection.



SPACE INTRUDERS
by DOUG KENNEDY

010-0116 TAPE $19.95
TRS-80 Model I/III and PMC-80 16K & up

012-0116 DISK $20.95
Trs-80 Model I — 32K and up
Disk supplied on protected media

NEW

SCOTT ADAMS SEAL OF APPROVAL

TRS-80*

SOFTWARE AUTHORS!!

Join the ranks of the leading writers in the industry by becoming an **Adventure International** author! We are looking for games (arcade, "Adventure"-style, simulation, role-playing) and outstanding utility packages for all of the major computers. Only the best will do! If you have a program which you feel would fit into our current catalog, then we invite you to submit it for review. Send a machine-readable copy, a description of any hardware needed, complete documentation and loading instructions to us in care of the **Software Review Board.** (Due to the tremendous number of programs we receive monthly, we can only return a submission if you include a stamped, self-addressed envelope.)

SOFTWARE DEALERS!!

If you aren't carrying the **Adventure International** line of products, then you're missing out on the fastest growing and most asked for programs in the industry today. We have over 150 quality products, with more being added every month. We offer our dealers support through national advertising, professional packaging and an **established** market. If you would like more information on joining our dealer family, call our toll-free number (800) 327-7172 and ask for a dealer representative.

If you own a TRS-80 (Models I, II, II, Color and Pocket), Apple, Atari, or other desk-top computer, then we have the programs for you! Order from your favorite software dealer or call toll-free **(800) 327-7172** today to order or for your free catalog.

Adventure INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.

BOX 3435 • LONGWOOD, FL 32750
Call toll free for catalog (800) 327-7172

CATALOG & ORDERING
TOLL FREE HOT LINE (800) 327-7172

529

*TRS-80 is a trademark of the Tandy Corporation

A Model 33, a Multi-80 and a dash of software.

Recipe for Hard Copy

Larry Keith
3903 Marsha Drive
Columbus, OH 43207

Do you have a Level II TRS-80 and dream of hard copy capability? If you have $2000, your local Radio Shack store has an expansion interface and a line printer. However, $2000 is more than most of us can handle, but take heart—I have a recipe you'll like.

Take one used teletype machine, add one interface device, combine with a printer driver program, and season with a couple hours work. The result: Instant hard copy with minimum expenditure.

So you don't know anything about teletype machines and you've never seen a printer interface device that doesn't require an expansion interface. What does a driver program look like? I will describe each element of the system, provide sources for each and give you some first-hand advice on how to create the cheap and painless print system.

Ingredients

The teletype machine, or more correctly, the teleprinter set, is an electro-mechanical device originally designed for communications use, either in a land line or radio circuit. After ASCII (American Standard Code for Information Interchange) was developed, teleprinters using that code became popular. There are still a lot of non-ASCII machines floating around—stay away from them. We do not need the aggravation of adding an ASCII to baudot converter to our system.

There are several manufacturers of teleprinters: Kleinschmidt, Mite, Lorenz, ITT Creed and Teletype Corporation. Teletype Corporation of Skokie, IL, is the predominant manufacturer and your best bet for a used machine. I will confine my discussion to the Teletype brand.

During my research, I found references to Models 11, 12, 14, 15, 19, 20, 26, 28, 29, 31, 32, 33, 35, 37, 38, 40 and 43. Models 33, 35 or 43 use ASCII and are still in service throughout the country. Consequently, repair service is available and parts and supplies are easily obtained.

Let's look at one popular model, the Teletype Model 33. The Model 33 can be found in three distinct configurations: receive only (RO); keyboard send-receive (KSR); and automatic send-receive (ASR).

The RO set can only receive and print messages. The KSR set can originate messages as well as receive and print. The ASR set transmits, receives, prints and has the capability to automatically initiate, accept and control incoming messages.

There are also variations within the three configurations. Some machines are friction feed and some use sprocket form feed. You may find a unit with a tape punch and a tape reader. All use ASCII, print on 8½-inch paper, print 72 characters per line and 10 lines to the inch at 110 baud (100 wpm), and are ideal line printers for hobby computer applications.

These machines are advertised for around $300. Concentrate on the receive only (RO) machines; they cannot be used as terminals, so the price is usually lower. And don't overlook some of the oddball machines, as long as they will print the whole character set. Such machines are often ideal for this application. Hamfests are also a good place to find teletype equipment, since a lot of hams use them for radio-teletype applications.

Sources

The following are some sources to investigate. (I cannot personally recommend these companies since I have not done business with them.)

- Typetronics
  Box 8873
  Ft. Lauderdale, FL 33310

Photo 1.

- General Peripherals
  68 B Merrimac St.
  Danberry, CT 06810
- D. Lavers
  Suite 719, 5217 Morris St.
  Halifax, NS, Canada B3J 1B7
- Lawrence R. Pfleger, K9WJB
  2600 South 14th St.
  St. Cloud, MN 56301

You also need an interface from your TRS-80 to the teleprinter. Several magazine construction articles describe such an interface. (Dig out those old copies of *Kilobaud* and take a look.) There are also ready-made interface devices available. The following are some sources:

- Small System Software
  P.O. Box 366
  Newbury Park, CA 91320
- Hobby World
  19511 Business Center Drive
  Northridge, CA 91324
- Salvage 2
  1358 Byron Avenue
  Columbus, OH 43227

Finally, you must have a program to drive the teleprinter. Salvage 2 and Small System Software furnish printer-driver programs with their interface devices. Other driver programs have been published.

A Printer Subsystem

I use a Model 33 Teletype machine. I purchased a Multi-80 from Salvage 2. The Multi-80 ($49.95) is an interface device with some additional features. I also purchased the TRS-232 Formatter program from Small System Software ($14.95). The total cost of my printer subsystem was less than $275.

The Multi-80 is a multiple purpose TRS-80 accessory developed and produced by Salvage 2. Check the following features:

- A solid state Teletype interface that uses cassette output to feed the Teletype machine—no expansion interface required.
- A cassette relay protection circuit that uses internal TRS-80 cassette motor relay to switch to a heavy-duty relay in the Multi-80.
- A cassette control circuit that allows you to gain control of the cassette recorder by simply flipping a switch. Fast-forward, rewind, and all other cassette motor controls are available without unplugging the subminiature plug or resorting to software control.
- Audio output is available from a rear panel jack on the Multi-80 for monitoring the CLOAD and CSAVE functions and for using that output for sound effects in user programs.

The Multi-80 runs from 117 V ac. The unit comes with instructions, a printer-driver program on cassette, and a one-year limited guarantee.

The unit is packaged in a neat gray and black box that measures 3 × 5 × 6 inches. Connections are via jacks located on the rear panel. Two switch controls and an LED are mounted on the front panel.

Installation and Operation

Three interconnecting cables run from the Multi-80 to the cassette recorder, and one cable runs from the Multi-80 to the Teletype machine. Two cables must be terminated on each end with miniature plugs and one cable must be terminated on each end with a subminiature plug. I used Radio Shack 72-inch shielded cables. The Teletype to Multi-80 cable is Radio Shack

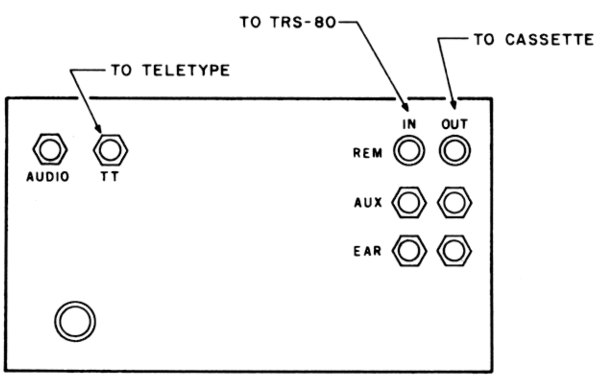


Fig. 1. Multi-80 Connections

Photo 2.

number 22 speaker wire.

Installation is simple. I removed the cassette plugs from the recorder and plugged them into the corresponding input jacks on the rear of the Multi-80. I connected the output jacks on the Multi-80 to the cassette recorder using the 72-inch cables. The jacks are clearly marked, and by connecting both ends of each cable before I started on the next one, I avoided getting my wires crossed. After connecting the Teletype to the Multi-80 via the output jack marked TT, I plugged the line cord into the last available wall outlet in my computer room (see Fig. 1).

Multi-80 operation is not complex. The switch marked PR and CA controls the printer/cassette signal. This switch can be left in the PR (printer) position except when CSAVEing a program. Then flip the switch to CA (cassette auxiliary) until the CSAVE is complete. The red LED on the panel lights up whenever the recorder relay is activated.

The TRS-232 Formatter

The TRS-232 Formatter program is a new advanced printer software package from Small System Software. This versatile BASIC program is exactly what you need to produce professional hardcopy. Features include:

- Printer paging allows you to set the number of lines for your paper and the number of lines that you wish printed on each page.
- Line length control causes

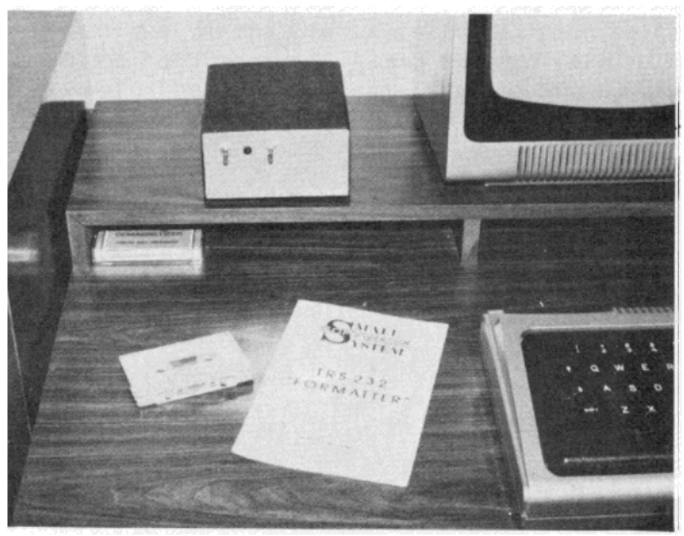


Photo 3.

MULTI-USER OASIS HAS THE FEATURES PROS DEMAND. READ WHY.

(THEN COMPARE.)

NOW AVAILABLE FOR TRS 80 MOD II

Computer experts (the pros) usually have big computer experience. That's why when they shop system software for Z80 micros, they look for the big system features they're used to. And that's why they like Multi-User OASIS. You will too.

DATA INTEGRITY: FILE & AUTOMATIC RECORD LOCKING

The biggest challenge for any multi-user system is co-ordinating requests from several users to change the same record at the same time.

Without proper co-ordination the confusion and problems of inaccurate or even destroyed data can be staggering.

OASIS File and Automatic Record Locking features solve these problems.

For example: normally all users can view a particular record at the same time. But, if that record is being updated by one user, automatic record locking will deny all other users access to the record until the up-date is completed. So records are always accurate, up-to-date and integrity is assured.

Pros demand file & automatic record locking. OASIS has it.

SYSTEM SECURITY: LOGON, PASSWORD & USER ACCOUNTING

Controlling who gets on your system and what they do once they're on it is the essence of system security.

Without this control, unauthorized users could access your programs and data and do what they like. A frightening prospect isn't it?

And multi-users can multiply the problem.

But with the Logon, Password and Privilege Level features of Multi-User OASIS, a system manager can specify for each user which programs and files may be accessed—and for what purpose.

Security is further enhanced by User Accounting—a feature that lets you keep a history of which user has been logged on, when and for how long.

Pros insist on these security features. OASIS has them.

EFFICIENCY: RE-ENTRANT BASIC

A multi-user system is often not even practical on computers limited to 64K memory.

OASIS Re-entrant BASIC makes it practical.

How?

Because all users use a single run-time BASIC module, to execute their compiled programs, less memory is needed. Even if you have more than 64K, your pay-off is cost saving and more efficient use of all the memory you have available—because it services more users.

Sound like a pro feature? It is. And OASIS has it.

AND LOTS MORE...

Multi-User OASIS supports as many as 16 terminals and can run in as little as 32K memory. Or, with bank switching, as much as 784K.

Multi-Tasking lets each user run more than one job at the same time.

And there's our BASIC—a compiler, interpreter and debugger all in one. An OASIS exclusive.

Still more: Editor; Hard & Floppy Disk Support; Keyed (ISAM), Direct & Sequential Files; Mail-Box; Scheduler; Spooler; all from OASIS.

Our documentation is recognized as some of the best, most extensive, in the industry. And, of course, there's plenty of application software.

Put it all together and it's easy to see why the real pros like OASIS. Join them. Send your order today.

OASIS

MAKES MICROS RUN LIKE MINIS

OASIS IS AVAILABLE FOR

SYSTEMS: Altos; Compucorp; Cromemco; Delta Products; Digital Group; Digital Microsystems; Dynabyte; Godbout; [illegible]; Index; Intersystems; North Star; Onyx; SD Systems; TRS 80 Mod II; Vector Graphic; Vorimex.

CONTROLLERS: Bell Controls; Cameo; Corvus; Konan; Micromation; Micropolis; Tarbell; [illegible]; [illegible]; X Comp.

Write for complete, free Application Software Directory.

PLEASE SEND ME:

Product	Price with Manual	Manual Only
OPERATING SYSTEM (Includes: EXEC Language; File Management; User Accounting; Device Drivers; Print Spooler; General Text Editor; etc.) SINGLE-USER	$150	$17.50
MULTI-USER	350	17.50
BASIC COMPILER/INTERPRETER/DEBUGGER	100	15.00
RE-ENTRANT BASIC COMPILER/INTERPRETER/DEBUGGER	150	15.00
DEVELOPMENT PACKAGE (Macro Assembler; Linkage Editor; Debugger)	150	25.00
TEXT EDITOR & SCRIPT PROCESSOR	150	15.00
DIAGNOSTIC & CONVERSION UTILITIES (Memory Test; Assembly Language; Converters; File Recovery; Disk Test; File Copy from other OS; etc.)	100	15.00
COMMUNICATIONS PACKAGE (Terminal Emulator; File Send & Receive)	100	15.00
PACKAGE PRICE (All of Above) SINGLE-USER	500	60.00
MULTI-USER	850	60.00
FILE SORT	100	15.00
COBOL-ANSI '74	750	35.00

Order OASIS from:

Phase One Systems, Inc.
7700 Edgewater Drive, Suite 830
Oakland, CA 94621

Telephone (415) 562-8085
TWX 910-366-7139

NAME ____________
STREET (NO BOX #) ____________
CITY ____________
STATE ________ ZIP ________
AMOUNT $ ________
(Attach system description; add $3 for shipping; California residents add sales tax)
☐ Check enclosed ☐ VISA
☐ UPS C.O.D. ☐ Mastercharge
Card Number ____________
Expiration Date ____________
Signature ____________

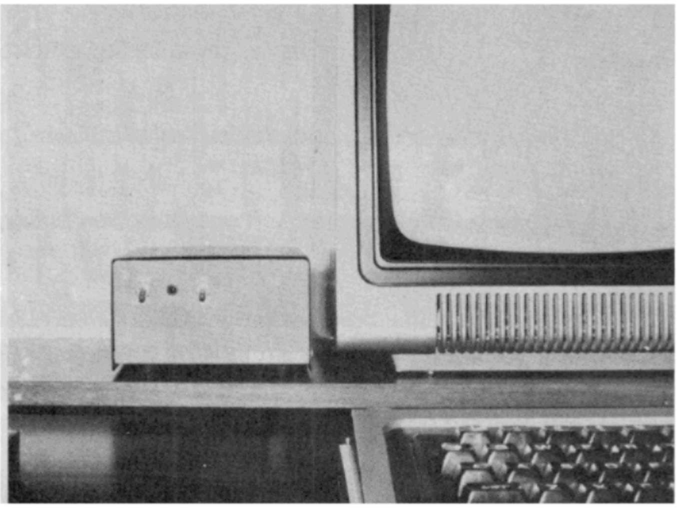


Photo 4.

the program to execute an automatic carriage return when a selected number of characters is reached.

● Smart line termination is the feature I like best. After a selected number of characters, which must be less than the line length control, is reached, the program looks for a space, colon, semi-colon or comma. If one is found, the program executes an early carriage return. This feature minimizes word division and enhances readability.

● Line indentation is an automatic feature. If a line is continued, the program advances five spaces before it continues printing.

● Print all screen output is an option that allows all Print statements to be directed to the printer. This option can be input via the keyboard or embedded in your program.

● Simultaneous screen display directs print to both the printer and the video display.

● Printer pause can be used to stop the printer at the end of a line by hitting the space bar. Each successive input of space bar will cause one more line to print. Hitting any other key continues normal printer operation.

● Keyboard debounce is included and loads with Formatter; it can be disabled.

● Multiple baud rates of 110, 134.5, 150, 300, 450, 600, 1200, 2400, 4800, or 9600 may be selected during program initialization. Page six of the instruction booklet supplies modification to allow for non-standard baud rates.

● Line feeds and form feeds

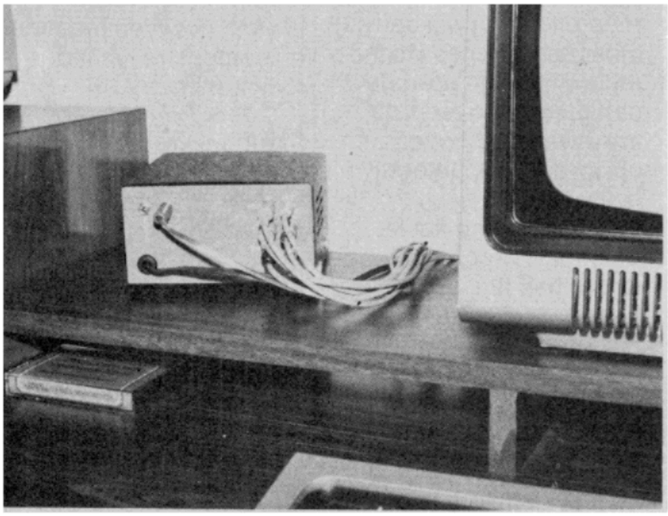


Photo 5.

are available; if your printer will not accept a form feed character, this program will use multiple line feeds to make up for that deficiency.

- Documentation is excellent. The 17-page instruction booklet includes complete instructions in clear, readable English for BASIC programs, assembly language listings and instructions for customizing the program.

The Formatter is a BASIC program that POKEs language instructions into a protected high memory area. The program requires 360 bytes at the top of memory. The instruction booklet suggests answering the memory size question with the value 32400.

Running the program for the first time, the following questions must be answered. Hitting the enter key in lieu of answering the questions assigns default values.

QUESTION	DEFAULT VALUE
ENTER INITIALIZATION MEMORY SIZE?	32400
SELECT BAUD RATE?	300
ADD LINE FEED AFTER CARRIAGE RETURN (Y/N)?	Y
DOES YOUR PRINTER RECOGNIZE FORM FEEDS (Y/N)?	Y
MAXIMUM ALLOWED PRINTER LINE LENGTH (25-250)	80
LINE LENGTH FOR EARLY LINE TERMINATION (20-250)?	70
NUMBER OF NULLS (0-120)?	0
TOTAL NUMBER OF LINES PER PAGE (0-120)?	66
NUMBER OF LINES TO PRINT ON EACH PAGE (0-120)?	58
ECHO PRINTER TO SCREEN (Y/N)?	Y
ARE YOU USING DISK BASIC (Y/N)?	Y
DO YOU WANT KEYBOARD DEBOUNCE (Y/N)?	Y

For my Model 33, I used the values shown below:

QUESTION	DEFAULT VALUE
ENTER INITIALIZATION MEMORY SIZE?	32400
SELECT BAUD RATE?	110
ADD LINE FEED AFTER CARRIAGE RETURN (Y/N)?	Y
DOES YOUR PRINTER RECOGNIZE FORM FEEDS (Y/N)?	N
MAXIMUM ALLOWED PRINTER LINE LENGTH (25-250)	72
LINE LENGTH FOR EARLY LINE TERMINATION (20-250)?	62
NUMBER OF NULLS (0-120)?	2
TOTAL NUMBER OF LINES PER PAGE (0-120)?	66
NUMBER OF LINES TO PRINT ON EACH PAGE (0-120)?	56
ECHO PRINTER TO SCREEN (Y/N)?	Y
ARE YOU USING DISK BASIC (Y/N)?	N
DO YOU WANT KEYBOARD DEBOUNCE (Y/N)?	Y

The answers describe the characteristics of my Model 33 teleprinter and my TRS-80 Level II 16K system. Setting the early line termination as 62 assures very few continued lines will have any divided words. Fifty-six lines of print gives me about 5/6 of an inch margin at the top and bottom of the page.

Twelve program lines within Formatter control the options. I customized the program for my printer, changing the variables, and deleting the input statements and all REM statements. My customized Formatter loads faster, since about half the original program lines were REM statements.

Three special keyboard commands are recognized by Formatter, and may also be used within a program. The commands are:

- LPRINT CHR$(1)— Sets the print all screen output option.
- LPRINT CHR$(2)— Clears the print all screen output option.
- LPRINT CHR$(3)— Reinitializes the Formatter line counter to prevent erroneous form feed after a partial page has been printed. (The line counter does not automatically clear when a partial page is printed.) If you are using the keyboard debounce routine, you can use the clear key to reset the line counter, also.

Day to day operation is a snap. I always need the debounce routine. I am in the middle of a major project, documenting all my programs via the printer. I will soon be able to throw away all those annotated code sheets and use only my notebook of printer-produced listings. I will soon have a paper index of all my programs, an index of my Level II manual, a real tape log, and an index of my computer magazines. None of these would be possible without my printer. ■

✓See List of Advertisers on page 306

THE PROGRAM STORE

✓17

SPACE FIGHTER

By Sparky Starks from Adventure Int.
As mercenary and galactic police officer, you must maintain the condition and control of all parts of your spacecraft. You sit at the controls while peering out of the digital spaceview port. Suddenly something appears on your screen: is it a Starpirate or a friendly merchant ship? You can't tell yet, and at this speed you may have only a fraction of a second to make an attack/no attack decision.

Speed and sound effects enhance the graphic action of this tactical game. Can you take the pressure?

Model I & III, 16K tape...$24.95
Model I disk version......$29.95

BASIC Programming Asst.

From Instant Software
What an aide to the writing and debugging of BASIC programs! With BPA you can list out your program variables, with a notation of variable type, whether each is an array or not, and a notation of line numbers where the variable occurs. What's more, BPA even tells you the line numbers where any variable changes value! You can also produce a cross-reference to all GOTOs, GOSUBs, and IF..THEN statements for easy visualization of program flow. Want to find that last INPUT statement? BPA will give all line numbers where selected BASIC keywords reside. Save time and frustration by using BPA when you program.

Tape for 16K, 32K & 48K...$14.95

PARSECTOR V

From Synergistic Solar
Using a unique split screen (with instructions for making a dividing hood), Parsector V lets two people participate in this exciting space strategy game (can also be played against the computer). Once a Parsector is conquered, it is yours -- until your opponent discovers it and tries to wrench it from you. Action sounds and graphics add to the thrill of this fast-moving game.

FLIP SIDE SPECIAL: Parsector 8 -- a tournament version with a full 64 parsectors for human opponents only (no computer)

Tape...$19.95

TRS-80 Level II 16K unless otherwise noted

MYCHESS

From Computer Services & Programma
Finally, a formidable chess opponent for even the most serious player: MYCHESS has attained the highest USCF rating ever attained by a microcomputer program. With MYCHESS, you choose from 9 levels of play (taking from 5 seconds to several days per move), you can select a printout of a game, and you can even save a game on disk for later completion.

32K disk (Mod I)...$34.95

IMAGE COMPUTER PRODUCTS

ACTION PACK I
Four machine language computer games for one low price! Includes:

Space Ace -- a search & destroy laser battle.
Shooting Gallery -- marksmanship game for up to 4 players.
Bomber Run -- bomb the ground targets from your plane or man the ground artillary to shoot the bomber down.
Air-Sea Battle -- choose the role of pilot or sea captain. Which will sink first, the plane or the ship?

All 4 on 16K tape...$19.95

STRATEGY PACK III
Three games designed to tax your reasoning, skill, and luck. Includes:

Wildcatting -- an oil field simulation requiring forethought & strategy.
Frame Up -- score points while trying to trap your opponent on the grid.
Recall -- "Concentration" type game with several challenging options.

All 3 on 16K tape...$19.95

PACKER

From Cottage Software
Packer gives tremendous control over the readability and efficiency of your BASIC programs. Specify "PACK" and the program will compress text into multiple statement lines. This really speeds up storage, load, and execution time. It can reduce the memory requirement by as much as 33% while saving disk or tape space, too.

Also included are four handy utilities: "MOVE" relocates program lines, "RENUMB" allows program renumbering, "SHORT" deletes unecessary words and REMarks, and "UNPACK" separates multistatement lines to ease editing. 16K, 32K & 48K tape...$29.95

Honestly...

...we are reluctant to advertise any hardware device. We suggest that you purchase such equipment from your local store, whenever possible.

However, most computer stores are ignoring TRS-80 owners, and they do not carry peripherals for TRS-80. Because of this situation, we have decided to occasionally advertise hardware for the TRS-80 that we feel is of exceptional merit.

ORCHESTRA-80

"Sometimes a product is marketed that is of such good quality and value that it soon becomes the standard in its class. Orchestra-80 is this kind of product."
80 Microcomputing, May 1981, p. 30.

If your idea of TRS-80 computer "music" is a series of raspy noises that sound like a cross between a kazoo and a buzz saw, you haven't experienced Orchestra-80. This brilliant hardware/software combination plugs right into your TRS-80 Model I (with or without expansion interface) and any amplifier to allow you to produce REAL music in up to four simultaneous voices.

From the moment you try out Orchestra-80 with the four sample compositions included, you'll be hooked. Using the "music operating system" along with the exceptionally complete documentation, you'll soon be programing your own music -- even if you have no previous musical knowledge or ability. You control the notes played by each voice, along with the tone quality, loudness, phrasing, tempo -- everything that goes into the orchestration of quality music. You can even redefine each "instrument" by specifying the number of harmonics and the relative balance of each one!

From funky rock to complex symphonic, Orchestra-80 takes you were you want to go musically for only $79.95 (D/A synthesizer, manual, and program for TRS-80 Model I, 16K tape, transferrable to 32K disk)

NOW IN STEREO, TOO!
Stereo Version: Orchestra-85 . . . $129.95

COMING SOON:
Orchestra-90 for TRS-80 Model III.

SUMMER MADNESS

In July and August, The heat of Washington's summer can drive you MAD! It worked on us, and you can take advantage of our madness with these offers designed to keep you inside and cool with your computer.

$14.95 Value—FREE!

With any purchase of $30 or more, we'll send your choice of one of the following programs -- FREE!

SUMMER MADNESS

Word Wars

By Hernhuter & Taylor from Acorn
WORD CHALLENGE depicts the length of a phrase. The phrases are either computer generated or typed in by your opponent. The player's challenge is to figure out the phrase, based on the letters guessed.

WORD GAME allows many levels of strategy as you try to guess the letters of a mystery word. Each letter has points based on its frequency of use in English language.

Tape...$14.95 -- FREE WITH $30 PURCHASE!

Battle of the Bulge

From Computer Simulations
For the serious war gamer, this World War II simulation offers incredible historical accuracy. Take command of 36 U.S. combat units and reenact this great 1944 battle. Would you have faired better than the actual result -- or worse?

Included are combat maps and charts, with notation of the topography, towns, enemy positions and type.

Tape...$14.95 -- FREE WITH $30 PURCHASE!

Death Dreadnaut

From Programmer's Guild
If your sense of humor is twisted, if your fantasies run toward the macabre, then this machine language adventure was made for you! The strange and disgusting room descriptions make it fun; grisly death lurking around each twist and turn make it amusing; a complex game structure makes it difficult. Spine-tingling suspense, horrifying surprises -- what more could you ask?

Tape...$14.95 -- FREE WITH $30 PURCHASE!

Visit Our Baltimore, MD Store: W. Bell Plaza—6600 Security Blvd

TO ORDER CALL TOLL FREE 800 424-2738 For information Call (202) 363-9797

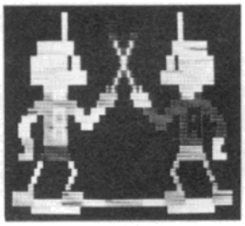
THE PROGRAM STORE
4200 Wisconsin Avenue NW, Dept. K1 Box 9609
Washington, D.C. 20016

MAIL ORDERS: Send check or M.O. for total purchase price, plus $1.00 postage & handling. D.C. residents, add 6% tax. Charge card customers: include all embossed information on card.

Get the most from your micro with software and accessories from one of the world's largest selections.

The Program Store

CALL TOLL FREE 800 424-2738



new

DUEL «N» DROIDS

By Leo Christopherson from Acorn
Your 'droid has already learned NIM, so now it's time to teach it how to wield a laser sword! Leo Christopherson, author of "Android NIM," "Dancing Demon" and other animations, has developed a new type of animation and high-quality sound in his latest work.

Your 'droid starts out as a lowly clown. You teach it how to use a laser sword by controlling its movements. After training it to be a "Grand Master," you enter the tournament against the program's skilled 'droid! Entertainment for all ages.

Protected Tape...$14.95
Protected Disk...$20.95

DEATH-MAZE 5000

from Med Systems
A new breed of adventuring! Venture through a graphically represented 3-D maze, with halls that could dead end -- or recede to infinity. Step through the doors or drop into the pits. Will you encounter monsters and mayhem, or will you be treated to useful objects and information? Will you ever get out alive?

You may never find your way out of Deathmaze 5000, but you'll keep trying!

16K TRS-80, 32K APPLE II...$12.95

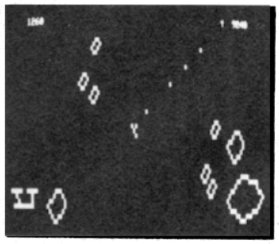
Unbelievable Realtime 3-D Graphics!

FLIGHT SIMULATION

From Sub-Logic
The wait is over! If 3-D graphics seem impossible on the low resolution TRS-80, you haven't seen this brilliant program. During FLIGHT SIMULATION, you instantly select instrument flight, radar, or a breathtaking pilot's-eye-view. But be sure to strap yourself in -- you're liable to get dizzy!

Once you put in some air time learning to fly your TRS-80, head for enemy territory and try to bomb the fuel depot and airstrip while fighting off five enemy warplanes. Good Luck!
Level I or II Tape...$25.00

TRS-80 Level II 16K unless otherwise noted



SUPER NOVA

By Bill Hague from Big Five
Asteroids surround your ship. You must shoot the asteroids, as well as any alien spaceships. Written in fast machine code, this game is GREAT!

You may encounter five different kinds of alien ships, including the very deadly flagship. You shoot from your ship's position, rotate it, use your thrusters to move -- if you are overwhelmed, you can even get away to hyperspace. Fast and exciting.

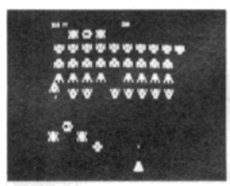
Tape....$14.95

SPACE WAR

By Device Oriented Games from Acorn
A two-player, real-time action game that lets each player control a spaceship with rotate, thrust, fire, and hyperspace. Five game options (including gravity) and three playing speeds. In fast machine language.

Tape...$9.95

GALAXY INVASION



By Hogue & Konyu from Big-Five
"The rage of the arcades" is now available for TRS-80! Exciting sound effects add to the action as the invaders swoop down to destroy your base. Even while you have your hands full battling the aliens, you have to watch out for the Flagship! Super graphics, super action, super fun!

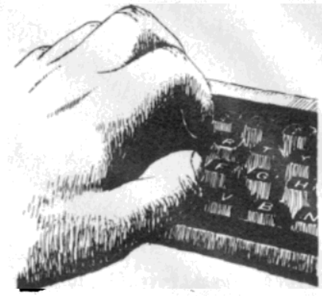
Level I or II, tape...$14.95

PINBALL

By John Allen from Acorn
Get your flipper fingers ready for action in this real-time, machine language game.

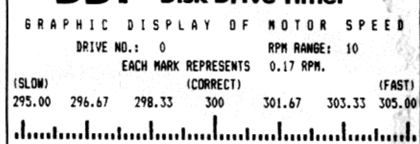
Lots of sound and flashing graphics make this fast action game so much like the real thing that you'll have to remind yourself not to shake your TRS-80. Choose from five playing speeds to match your skill. Can you beat your friends' scores? Will you avoid the dreaded "Bermuda Square?" Get PINBALL today and find out.
Protected tape...$14.95
Protected disk...$20.95

TYPING TUTOR



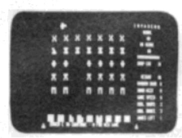
By Ainsworth & Baker from Microsoft
Speed up your programming and word processing with this excellent touch-typing instructional program. Divided into two sections, the program first teaches proper finger positioning. You practice keying various characters, the program adding new ones as you progress. In the practice paragraph section, you are evaluated for accuracy and rated in words per minute. The program continuously adjusts to your increasing skill, telling you which characters you miss and where you are slow. One of the most practical programs we know of for TRS-80.
$14.95

DDT Disk Drive Timer



from Disco-Tech
Analyze and adjust your disk drive motor speed with a real-time graphic display. Manual details use for Radio Shack, Shugart, MPI, Pertec and Vista drives, and DDT can be used with any drive. All you need is DDT, two screwdrivers and five minutes.

Disk.....$29.95



INVADERS FROM SPACE

by Carl Miller from Acorn
A fast machine language approach to this classic (and addictive) space game. As you play, the aliens drop bombs, move from side to side, and try to overrun your bases. Hold them off -- and score -- by shooting them down. But, just as you think you've got it all under control, the action speeds up.

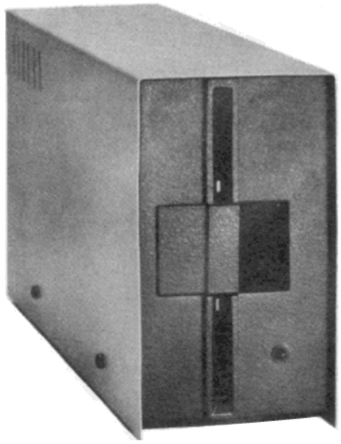
Choose the game speed, enemy bomb frequency and accuracy, shots on screen and the number of your bases. Move your base and simultaneously fire at the invaders -- you cannot do this in most similar games. Full sound effects add even more excitement to the incredible speed and action of INVADERS FROM SPACE. Fun for all ages and skill levels.
Protected tape....$14.95
Protected disk....$20.95

THE PROGRAM STORE · Dept K1 Box 9609 · 4200 Wisconsin Ave, NW · Washington, D.C. 20016

Item ______ Price ______
Postage $1.00
Total ______
name ______
addr ______
☐ CHECK ☐ VISA city ______ state ___ zip ______
☐ MASTERCARD MC Bank # ______ Card # ______ Exp ______

See List of Advertisers on page 306

Mini-Disk Storage Systems for TRS-80* Computers



Access... Offers More. Charges less.

from $295

Company/Drive Model	Price 1/	Flippy	Formatted Diskette Storage Capacity: Dbl-Density	Formatted Diskette Storage Capacity: Sgl-Density	100% Tested	48 hr. Burn-In	Warranty	Trial Period
40-TRACK DRIVES								
Access Unlimited								
AFD-100[1]	$295.00	no	180 Kbytes	102 Kbytes	yes	yes	yes	yes
AFD-100F[1]	329.00	yes	360 Kbytes	204 Kbytes	yes	yes	yes	yes
MTI TF-5	359.00	no	?	?	?	?	?	no
Midwest Comp. & Per. MPI/B-51	321.00	no	?	102 Kbytes	?	?	yes	no
Aerocomp Mdl 40-1	349.95	yes	?	?	yes	?	yes	yes
CPU Shop CCI-100	314.00	no	?	102 Kbytes	?	?	yes	no
AMI 40-track	325.00	no	?	?	?	?	?	no
80-TRACK DRIVES								
Access Unlimited								
AFD-200[1]	429.95	no	368 Kbytes	205 Kbytes	yes	yes	yes	yes
AFD-200F[1]	449.95	yes	736 Kbytes	410 Kbytes	yes	yes	yes	yes
MTI TF-8	639.00	no	?	200 Kbytes	?	?	?	no
Aerocomp 80-tk mdl	459.95	yes	?	?	yes	?	yes	yes
CPU Shop CCI-280	429.00	no	?	204 Kbytes	?	?	yes	no
AMI 80-track	560.00	no	?	?	?	?	?	no

1 As advertised in 80 Microcomputing, Jan. 1981.

Check this line-by-line comparison of Access 5-inch mini-disk systems for Model I and Model III computers. *Nobody* offers more benefits, better service or lower prices.

TOLL-FREE ORDER NO.
1-800-527-4196†
(orders and literature only)

Free trial offer

Use your brand new AFD drive system for up to 15 days. If you're not *completely* satisfied, pack it in the original shipping container and send it back to Access. We'll refund the price of the system (less shipping charges), no questions asked. (No refund for misuse or improper handling.)

Operating burn-in test — too

Every drive that leaves Access is not only 100% electrically tested and double-checked for mechanical alignment, but it is also given a full 48-hour operating burn-in test. You'll find a test list — checked and signed by one of our competent technicians — in the drive carton. If a drive has latent defects, the burn-in life test will weed them out. The drives we ship just keep on running. And running. And running.

About our warranty

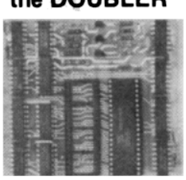
Venus de Milo has about the right number of fingers to count our warranty returns. Nevertheless, your new drive system is covered by our comprehensive 90-day limited warranty. The details are spelled out in the illustrated users manual included with each AFD drive system.

Software galore!

Games. Business programs. System software. We sell many of the best TRS-80* Model I and Model III programs, and at competitive prices. Make Access Unlimited your one-stop shopping center for all of your TRS-80* software, hardware and accessories. Save big! Call our toll-free order number, 1-800-527-4196† for free descriptive literature.

USE YOUR CREDIT CARD AND SAVE! VISA AND MASTER CARD CHARGES ARE NOT DEPOSITED UNTIL THE DAY YOUR ORDER IS SHIPPED. CALL NOW TOLL-FREE, 1-800-527-4196†.

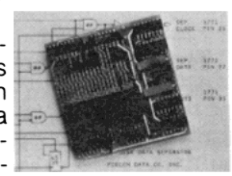
the DOUBLER™



Percom's new plug-in adapter for your Expansion Interface stores almost twice the data on a diskette track as a single-density system. You can store up to four times more data — depending on the type of drive — on one side of a diskette than you can store using a standard Model I mini-disk drive. Other features: Reads, writes and formats *either* single or double density minidiskettes. • Runs TRSDOS*, NEWDOS+‡ Percom OS-80™ or other single-density software *without* changing either software or hardware. Switch to double-density when convenient. • Includes DBLDOS™, a TRSDOS* compatible double-density operating system. • Includes on card, high-performance data separator circuit. • Installs *without* rewiring or trace cutting. • Introductory price, including DBLDOS and format conversion utility, only $219.95.

Permits Model III software to be read on Model I computers.

DATA SEPARATOR™



This PC board plug-in adapter for the TRS-80* virtually eliminates data read errors (CRC error — Track locked out!) which occur on high-density inner disk tracks, a problem that has plagued TRS-80* systems. The Percom Data Separator™ is installed in the Expansion Interface without modifying the host system. Caution: Opening the TRS-80* Expansion Interface may void the limited 90-day warranty: $29.95.

Disk System Interconnecting Cables

Improvement over RS cable design places drive 0, which includes the cable termination, at the end of the cable to eliminate the reflected noise of an unterminated cable. Better data integrity. Prices:

Two-Drive Cable $24.95
Four-Drive Cable 34.95

Power Line Filter

115/250 V, 50-400 Hz. Instructions included for easy installation in standard mini-box chassis: $19.95

Minidiskettes (Double-Density rated)

10 Disks in a convenient plastic organizer box $34.90
Single Disk 3.49

Disk Drive ID Tabs

1" x 1¼" self-adhering plastic drive identification tabs. Compatible silver with engraved black drive number. Two tabs (Nos. 0, 1): $2.50; three tabs (Nos. 0, 1, 2): $3.25; four tabs (0, 1, 2, 3): $4.50.

How to Order Order by calling Access Unlimited toll-free on 1-800-527-4196†. Mail orders also accepted. Orders may be charged to a VISA or MasterCard account or paid by a cashier's check, certified check or money order. We accept COD orders with 25% deposit. Sorry, we cannot accept personal checks. We pay shipping and insurance charges on orders over $1,000.00. Add approximate insurance and shipping charges for under $1,000.00. If in doubt about these charges, ask when you call in your order. Texas residents include 5% sales tax. Minimum order $20.00. Allow 2 to 4 weeks for delivery. **†Texas residents call (214) 494-0206.**

ACCESS UNLIMITED ✓229

315 N. Shiloh · Ste. D1 · Garland, TX 75042
(214) 494-0206

The interesting field of random access.

How to Handle Those Random Files

Ken Knecht
1340 West 3rd Street #130
Yuma, AZ 85364

TRS-80 random disk files are very useful for storing information, especially if the data must be changed later. This is true for data bases such as inventory, mailing lists, etc.

Random files are easy to use; easier, in fact, than sequential files with their involved punctuation requirements. One disadvantage of random files is wasted space if your records are less than 255 bytes. We'll discuss some ways to reduce this wasted space, but as long as the records are fixed at 255 bytes (256 in TRSDOS 2.3) this will be a problem.

Setting Up

When setting up random files, your first task is to determine the record format. We'll take a typical inventory record as an example:

Item Description—String—25 characters
Quantity in Stock—Integer—2 bytes
Price Each—Single Precision—4 bytes
Location—String—10 characters
Date of Last Sale—String—8 characters

Note that we have allocated only two bytes for the integer and four bytes for the single precision number.

To convert the single precision number stored in variable P to a four-byte string, use P$ = MKS$(P). Use Q$ = MKI$(Q) to convert the integer in variable Q to a two-byte string. To get the number back, use P = CVS(P$) or Q = CVI(Q$). This is faster than using Q$ = STR$(Q) or P$ = STR$(P) to store the number and Q = VAL(Q$) or P = VAL(P$) to get it back, and usually takes up less disk space. Remember that you can store only strings in a random file.

Look at Fig. 1. To put a string on the disk we must do several things. One of them is to point to the correct record, for example Record 5, with GET 1, 5. The file must have been opened with OPEN R, 1, FILE$, thus referencing it as file number one.

Before we can GET the record we must FIELD it. In this case we would use: FIELD 1, 25 AS A$, 2 AS B$, 4 AS C$, 10 AS D$, 8 AS E$. Note how the length of the variables matches the length of each item in our record.

You can use any variable names and sizes you desire. If you run your letters together without spaces, to save memory, be sure to put a space between the AS and C$, or the interpreter will read it as ASC$, and you'll get a syntax error. ASC is a reserved word.

To put a field in a record, first be sure it has been properly fielded. Assume the item description has been stored in I$, the price in P, the quantity in Q, the location in L$ and the date in DA$. Before you PUT the record you must use:

```
LSET A$ = I$: LSET B$ = MKI$(Q): LSET C$
  = MKS$(P):
LSET D$ = L$: LSET E$ = DA$
```

Then use PUT 1,5 to put the record in file 1, record 5. You can use RSET in the same way if you want the data right justified in the field. See Fig. 2 for LSET and RSET used with the sample file.

I have always found it safest to GET a record before you PUT it, even if it doesn't exist yet. Funny things sometimes hap-

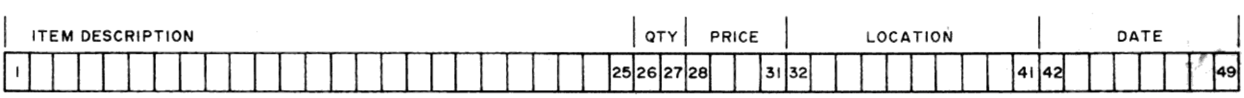


Fig. 1

pen if you don't. Note that this can't be done with a Model II.

Program Listing 1 is a sample program to enter some inventory items. Assume the file has been opened already. The LOF(1) + 1 in line 100 sets X to the next available record. LOF(1) is the last record (Last Of File).

Program Listing 2 enables us to read records. Again assume the file is already open.

Program Listing 3 enables us to change one item in the inventory.

Techniques

In a few cases I used LINEINPUT, when the data might contain a comma or other delimiter which would get an Extra Ignored if I used INPUT. When I asked for ANOTHER, I only looked at the first letter in the reply. That way Y or Yes or Yeah, etc. would work.

When printing the price, I used PRINTUSING to give a properly formatted display. In lines 580 to 620 I set the variable to the old value before using INPUT. This meant the old data would be in the variable if the user replied only Enter. This doesn't work in the lines that use LINEINPUT.

GOTO 30 is at the end of every module. Program Listing 4 explains that.

Put the four program segments together and you have a complete program. This is an example of modular programming.

Problems

The program wastes a lot of

```
100 X=LOF(1)+1
110 LINEINPUT"ITEM DESCRIPTION? ";I$
120 INPUT"QUANTITY";Q
130 INPUT"PRICE";P
140 LINEINPUT"LOCATION? ";L$
150 INPUT"DATE (01/02/80)";DA$
160 FIELD 1, 25 AS A$, 2 AS B$, 4 AS C$, 10 AS D$, 8 AS E$
170 GET 1,X
180 LSET A$=I$:LSET B$=MKI$(Q):LSET C$=MKS$(P)
190 LSET D$=L$:LSET E$=DA$
200 PUT 1,X:X=X+1
210 INPUT"ANOTHER";AN$:IF LEFT$(AN$,1)="Y" THEN 110
220 GOTO 30
```

Program Listing 1

```
300 LINEINPUT"ITEM DESCRIPTION? ";I$
310 FOR X=1 TO LOF(1)
320 FIELD 1, 25 AS A$, 2 AS B$, 4 AS C$, 10 AS D$, 8 AS E$
330 GET 1,X
340 IF INSTR(A$,I$)>0 THEN 360
350 NEXT X:PRINT"ITEM NOT FOUND":GOTO 410
360 PRINT"ITEM DESCRIPTION: ";A$
370 PRINT"QUANTITY: ";CVI(B$)
380 PRINTUSING"PRICE: $$####.##";CVS(C$)
390 PRINT"LOCATION: ";D$
400 PRINT"DATE: ";E$: PRINT
410 INPUT"ANOTHER";AN$: IF LEFT$(AN$,1)="Y" THEN 300
420 GOTO 30
```

Program Listing 2

```
500 LINEINPUT"ITEM DESCRIPTION? ";I$
510 FOR X=1 TO LOF(1)
520 FIELD 1, 25 AS A$, 2 AS B$, 4 AS C$, 10 AS D$, 8 AS E$
530 GET 1,X
540 IF INSTR(A$,I$)>0 THEN 560
550 NEXT X:PRINT"ITEM NOT FOUND":GOTO 660
560 PRINT"REPLY WITH <ENTER> IF YOU DO NOT WISH TO"
570 PRINT"CHANGE AN ITEM, OTHERWISE ENTER THE NEW DATA.":PRINT
580 I$=A$:PRINT"ITEM DESCRIPTION: ";I$;" ?";:LINEINPUT I$:
    IF I$="" THEN I$=A$
590 Q=CVI(B$):PRINT"QUANTITY: ";Q;:INPUT Q
600 P=CVS(C$):PRINTUSING"PRICE $$####.##";P;:PRINT" ";:INPUT P
610 L$=D$:PRINT"LOCATION: ";L$;" ?";:LINEINPUT L$:IF L$="" THEN
    L$=D$
620 DA$=E$:PRINT"DATE: ";DA$;" ";:INPUT DA$
630 LSET A$=I$:LSET B$=MKI$(Q):LSET C$=MKS$(P)
640 LSET D$=L$:LSET E$=DA$
650 PUT 1,X
660 INPUT"ANOTHER";AN$:IFLEFT$(AN$,1)="Y" THEN 500
670 GOTO 30
```

Program Listing 3

disk space; only 49 bytes are used in each 255 byte record. Also, it would be nice to be able to delete a whole record if an item was discontinued, and be able to print a sorted list of the whole inventory file.

Program Listing 5 rewrites the first module to use disk space more efficiently. With this small modification we store 245 bytes in each record (49 x 5) and waste only 10 bytes.

In line 10 we set RS to 49. This is the subrecord length. We then divide this into 255 to see how many subrecords we can fit into a record. We use INT( ) because we want an integer result. In line 160 we add a second FOR loop to keep track of the number of subrecords. Note that we start the loop with zero and count to one less than the number of subrecords.

In line 170 we calculate the number of subrecords already looked at—Y (our subrecord counter)*RS (the record size). We add the 1* and put every-

```
10 CLEAR 1000: LINEINPUT"FILE NAME? ";FILES$
20 OPEN"R",1,FILES$
30 CLS:PRINT"                    MENU"
40 PRINT"1: ADD ITEM"
50 PRINT"2: DISPLAY ITEM"
60 PRINT"3: CHANGE ITEM"
70 PRINT"4: FINISHED"
80 INPUT"CHOICE BY NUMBER";C:IF C<1 OR C>4 THEN 80
90 ON C GOTO 100, 300, 500, 95
95 CLOSE:END
```

Program Listing 4

```
100 RS=49: SU=INT(255/RS)
110 LINEINPUT"ITEM DESCRIPTION? ";I$
120 INPUT"QUANTITY";Q
130 INPUT"PRICE";P
140 LINEINPUT"LOCATION? ";L$
150 INPUT"DATE (01/02/80)";DA$
160 FOR X=1 TO LOF(1): FOR Y=0 TO SU-1
170 FIELD 1, (1 * RS * Y) AS Z$, 25 AS A$, 2 AS B$, 4 AS C$,
    10 AS D$, 8 AS E$
180 GET 1,X
190 IF LEFT$(A$,1) = CHR$(0) THEN 230
200 NEXT Y,X: X=LOF(1)+1
210 FIELD 1,25 AS A$, 2 AS B$, 4 AS C$, 10 AS D$, 8 AS E$
220 GET 1,X
230 LSET A$=I$: LSET B$=MKI$(Q): LSET C$=MKS$(P)
240 LSET D$=L$: LSET E$=DA$
250 PUT 1,X
260 INPUT"ANOTHER";AN$: IF LEFT$(AN$,1)="Y" THEN 110
270 GOTO 30
```

Program Listing 5

```
10 CLEAR 20000: LINEINPUT"FILE NAME? ";FILES$:OPEN"R",1,FILES$
20 RS=49: SU= INT(255/RS): C=1: DIM DA$(LOF(1) * SU,4)
30 FOR X=1 TO LOF(1): FOR Y=0 TO SU-1
40 FIELD 1,(1 * RS * Y) AS Z$, 25 AS A$, 2 AS B$, 4 AS C$,
   10 AS D$, 8 AS E$
50 GET 1,X: IF LEFT$(A$,1) = CHR$(0) THEN 80
60 DA$(C,0)=A$: DA$(C,1)=B$: DA$(C,2)=C$: DA$(C,3)=D$: DA$(C,4)=E$: C=C+1
80 NEXT Y,X: C=C-1
90 M=C: PRINT"NOW SORTING"
100 M=INT(M/2): IF M=0 THEN 260
110 J=1: K=C-M
120 I=J
130 L=I+M
140 IF DA$(I,0) < DA$(L,0) THEN 240
150 FOR X=0 TO SU-1
160 FOR Y=0 TO 2
170 A1=PEEK(VARPTR(DA$(I,X))+Y)
180 A2=PEEK(VARPTR(DA$(L,X))+Y)
190 POKE(VARPTR(DA$(I,X))+Y),A2
200 POKE(VARPTR(DA$(L,X))+Y),A1
210 NEXT Y,X
220 I=I-M: IF I<1 THEN 240
230 GOTO 130
240 J=J+1: IF J>K THEN 100
250 GOTO 120
260 PRINT"FINISHED SORT, NOW SAVING FILE IN NEW ORDER."
270 C1=1: FOR X=1 TO LOF(1): FOR Y=0 TO SU-1
280 FIELD 1,(1 * RS * Y) AS Z$, 25 AS A$, 2 AS B$, 4 AS C$,
    10 AS D$, 8 AS E$
290 GET 1,X
300 IF C1>C THEN LSET A$=CHR$(0): GOTO 320
310 LSET A$=DA$(C1,0)
320 LSET B$=DA$(C1,1): LSET C$=DA$(C1,2): LSET D$=DA$(C1,3):
    LSET E$=DA$(C1,4)
330 PUT 1,X: C1=C1+1
340 NEXT Y,X: CLOSE: CLEAR 1: RUN"INVENTRY"
```

Program Listing 6

✓See List of Advertisers on page 306

thing in parentheses to make Disk BASIC happy. If Y=0, we put nothing in Z$ (1*49*0=0), then as Y advances we step through the subrecords one by one. At each step we put the subrecords we already looked at in Z$, then disregard Z$.

In line 200 (note the added NEXT Y) we didn't find an empty record in the file, so we increment X to the first unused record and FIELD as for the first subrecord.

In line 190 we check A$ to see if the first character is a null. If it contains a valid subrecord, it would not have a CHR$(0) for the first character, because we are LSETing the data in A$. An unused subrecord would have a CHR$(0) in the beginning of A$, because the disk contains CHR$(0) until you put something there.

The next task is to find a subrecord to be deleted. In this case you'd use code similar to the module beginning at line 500 (change data) to search for the item, then put a CHR$(0) in the first byte of A$ when LSETing A$. It will thus be considered an empty subrecord when searching for a place to put a new subrecord. You could use MID$(I$,1,1)=CHR$(0). Be sure to add the additional subrecord code to all the modules; I only added it to the add data module.

To sort the data you'll have to put it into an array in memory. In this case, we'll use the array DA$. To calculate the array size you'd use DIM DA$(LOF(1)*SU, 4) after the file was opened and you calculated SU. Be sure that the program sees this DIM statement only once while the program is running or you'll get a double dimension error. Note also that this won't work if you add any records to the file after you DIM the array.

To get around these problems I'd use a separate subroutine called by the main program to call the sort program. The sort program in turn would recall the main program when it finished: 1000 CLOSE:RUN"SORT".

The menu would call line 1000 when a sort was requested. See Program Listing 6 for the sort routine.

CLEAR 20000 in line 10 should be changed to reflect the amount of string space available after the program is loaded.

Line 300 ascertains that any unused subrecords have a CHR$(0) in their first byte to show they are unused. The CLEAR 1 in line 340 makes sure there is enough room to reload the inventory program.

The sort is a shell sort. The VARPTR routine speeds up the program. BASIC now doesn't have to do any garbage collecting as string space keeps filling up with temporary variables.

Be sure that the string space in line 10 is ample to hold the inventory for the sort. The sort only looks at the item description. To use another column make it a

B	I	N		I	7	I	3		

LSET

		B	I	N		I	7	I	3

RSET

Fig. 2

Make waves in the German computer market!

Advertise in . . .

CHIP

The German-language computer market has an exceptional sales potential. CHIP, the leading German magazine for desktop computers will help you to make this market your own. Over 65% of the CHIP readership deals with computers in their business or profession. In addition, the CHIP audience has made computing a hobby for which it is willing to spend money. The German market for personal computing has grown by leaps and bounds since CHIP first appeared in 1978—proven by its paid circulation of 50,047*. Get yourself into the European market today by advertising in CHIP. *IVW Auditing (ABC equivalent), 3rd quarter 1980

GENERAL ADVERTISING RATES*

SIZE	1x	3x	6x	12x
1 pg.	1565	1410	1330	1252
¾ pg.	1174	1057	998	939
⅔ pg.	1044	940	888	835
½ pg.	783	705	666	626
⅓ pg.	544	490	462	436
¼ pg.	410	369	348	328

*U.S. Dollars effective 6/1/81. Rates subject to change in accordance with exchange rates.

DEADLINES

Issue	On Sale Date	Closing Date	Last Date Materials in Germany
Sept.	Aug. 31	July 20	July 27
Oct.	Sept. 28	Aug. 17	Aug. 24
Nov.	Oct. 26	Sept. 14	Sept. 21
Dec.	Nov. 30	Oct. 19	Oct. 26
Jan.	Dec. 28	Nov. 16	Nov. 23
Feb.	Jan. 25	Nov. 30	Dec. 7

SAVE
Take advantage of the best interest rates in years.

FOR MORE INFORMATION CONTACT:

Piergiorgio Saluti
European Representatives
(603) 924-7296

WAYNE GREEN INC. PETERBOROUGH, NH 03458

user selected variable or change it in line 140.

This is no comparison for a machine language sort, but it hobbles along at a fair speed for a BASIC sort. The big disadvantage is that the whole thing must fit into memory. By running the sort as a separate program, we can use maximum string space for the file. If this sort is too slow, or your RAM is not ample for the file, look into Racet's DOSORT program. This is a very fast machine language sort and does not require that the file be in RAM.

Printing

You have seen how we display the data in several of the modules. Now print the headings across the screen, and then the data, using tabs to keep the separation right. You know the lengths of the data, so this shouldn't be a problem. In the case of the price and quantity you'll have to guess at the maximum lengths.

You can use a program like this to store most any data base. My intention was only to give you a few examples of using random files to store data. I hope you give these techniques a try in your next program. They are worth studying if you are learning to use random access. ■

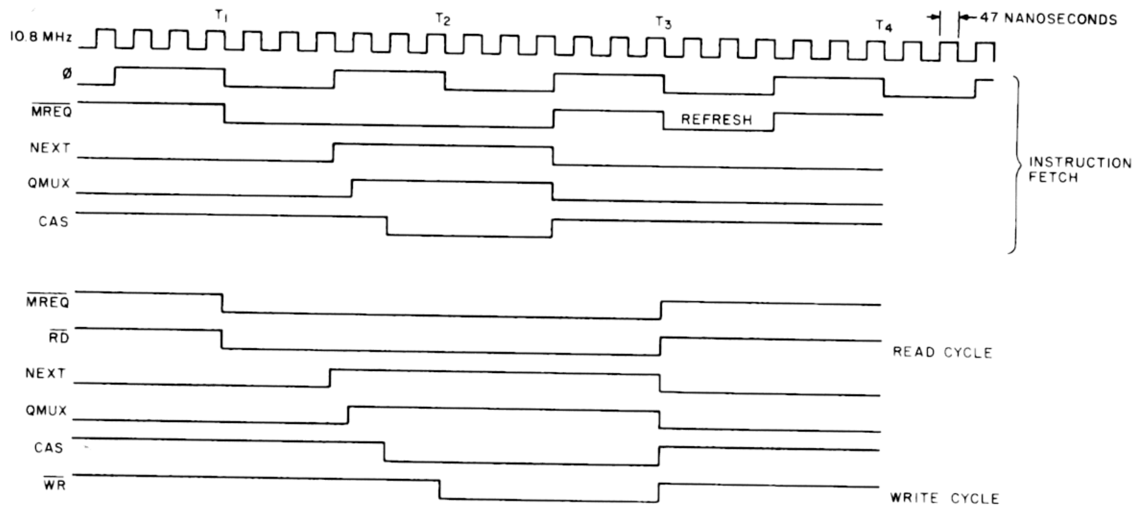


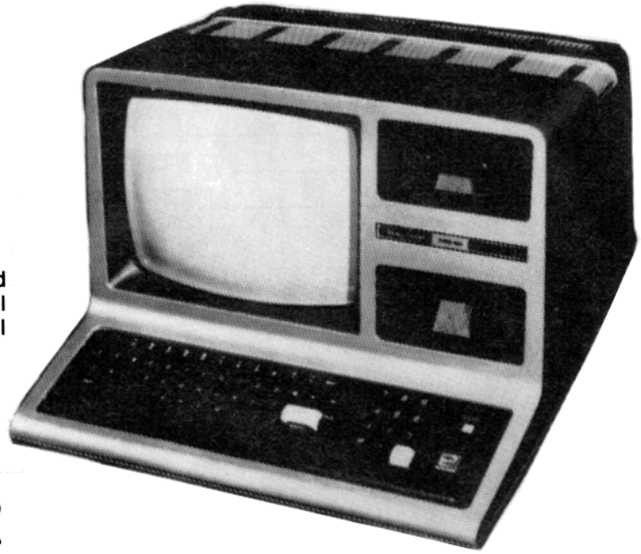
Fig. 3

FIRST WE REDUCED THE PRICE . . . NOW, WE HAVE INCREASED THE PERFORMANCE

TRS-80® MOD III

FIRST . . . We lowered the prices of the Model III from $2499* to

$1899



*We have taken the basic 16K Model III computer, expanded the memory to 48K and added our ultra-reliable MTI Dual Disk Drive system. System is fully compatible with MOD III DOS and peripherals.

MOD III PLUS - PROFESSIONAL SYSTEM

Now we have enhanced the basic system to increase the processing speed and increase the display capability from 64 x 16 characters to 80 x 24 characters. This allows complete word processing capability on the MOD III. Speed-up, the basic processing speed has been upgraded by adding a Z-80B processor. Programs run many times faster compatibly without the use of special software. **$2799**

MOD III/80

Upgraded MOD III, two disk system. Now you can get 80 track Disk Drives allowing 350K bytes per drive or 700K bytes per disk drive. An Outstanding Value. .... AS LOW AS **$2249**

MTI has a complete assortment of COMPUTER KITS - & Accessories

MTI DISK DRIVES for MOD III

Internal Kits - 40 TK.
- Disk Drive 1 .... $649
- Disk Drive 2 .... $279

External Add-on Kits - 40 TK.
- Disk Drive 3 .... $379
- Disk Drive 4 .... $359

- Model III DOS & Manual ... $21.95
- 16K RAM Kit $59
- RS-232 Serial Interface $95

Ask About Our Extended Warrantee Plan

MICROCOMPUTER TECHNOLOGY, INC.
3304 W. MACARTHUR, SANTA ANA, CA 92704
★ PHONE (714) 979-9923 ★ TELEX #678401TAB IRIN

✓28

*Uses MTI Memory, Disk Drives & Components

MasterCard VISA

®TRS-80™ IS A REGISTERED TRADEMARK OF TANDY CORPORATION

ALL PRICES CASH DISCOUNTED
FREIGHT FOB FACTORY
ASK FOR FREE CATALOG

MTI

MICROCOMPUTER TECHNOLOGY INCORPORATED

SUMMER SALE

CLEARANCE SPECIAL
SHUGART SA400 (TF-3)

35 track disk drive includes power supply and chassis.

LIMITED SUPPLY

$289

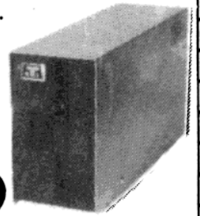
FREE FULL YEAR WARRANTY

with purchase of new 40 TK.

TANDON DISK DRIVE

comes complete with power supply, chassis, cable & DOS PLUS

ONLY $409



DOUBLE DENSITY SPECIAL!

MTI Doubler Pkg. includes doubler and microsystems DOS PLUS 3.3D, run single and double density, plus a box of Verbatim diskettes.

$275

DISK STORAGE SPECIAL

Buy 5 boxes of diskettes at $30.00 ea. and get a $30 Value Flip-Sort

FREE

LEXICON II ACOUSTIC MODEM SPECIAL CLOSEOUT

$119

Memory Kit (16K)	$43.00
AC Isolator (6 socket)	$49.95
Disk Head Cleaner	$19.95
Diskettes Verbatim (10) hard ring 5" s.s	$30.00

Disk Drive Sale!
with power supply and chassis

Shugart SA400L, 40 trk.	$319
Pertec FD200, 40 trk.	$349
TF-5 MPI B51, 40 trk.	$349
TF-11 Tandon, 40 trk.	$309
TDH-1 Dual sided drive, 35 trk.	$419
TF-3M Drive Sys. 3 Shugart 40 trk. (1 cabinet)	$599

SOFTWARE SALE

DOS+ 3.3	$99
DOS+ 3.3D	$99
Disk & System Diagnostic	$49

INTERFACES - MOD I

Interface 16K	$349
Interface 32K	$419

MONTHLY SPECIAL - NEW FOR TRS-80®

TF-15 80 TRACK DISK DRIVE

Double your capacity. 200K bytes of storage. Complete with power supply and chassis.

$479

TF-17 DUAL 80 TRACK DISK DRIVE

Quadruple your capacity. 400K bytes of storage. Complete with power supply and chassis.

$659

SYSTEMS — MOD I

Interface 32K, TF-11 disk drive cable and DOS 3.3

$799

MOD I Disk Expansion Systems

• 2 TANDON TF-11	$618
• 1 Two-Drive Cable	$26
• 1 Expansion Interface 32K	$419
• 1 DOS+ 3.3	$99
TOTAL LIST PRICE	$1162
SPECIAL PRICE ONLY	**$1029**

★ BARE DRIVES FOR ANY MICROCOMPUTER

Pertec FD200	$282	FD250	$359
Shugart SA400 35 trk.	$269	SA400L 40 track	$279
MPI B51	$279	B52	$349
MPI B91	$399	B92	$525
TANDON 40 trk.	$259	TANDON 80 trk.	$339
TANDON D/40 trk.	$339	TANDON D/80 trk.	$504

MTI

MICROCOMPUTER TECHNOLOGY, INC.

3304 W. MACARTHUR, SANTA ANA, CA 92704

★ PHONE (714) 979-9923 ★ TELEX #678401TAB IRIN

®TRS-80™ IS A REGISTERED TRADEMARK OF TANDY CORPORATION

MasterCard VISA

ALL PRICES CASH DISCOUNTED
FREIGHT FOB FACTORY
ASK FOR FREE CATALOG

Time is data where the Z-80 is concerned.

High Speed Data Tapes

Jim Glosser
152 Clover Ave.
Marion, OH 43302

If you've created large data files which need to be stored on tape, you know that data tapes are slow. BASIC has no choice but to use great amounts of time to load or save tapes. But there is an alternative. A machine code subroutine can reduce this time by up to 90 percent.

The Problem

BASIC has to use the PRINT #-1 and INPUT #-1 statements to create and read tapes. Each time the statements are used, your tape recorder turns on, writes or reads, and turns off. This happens even in loop statements. The statement FOR X=1 TO 50:PRINT #-1,A$(X): NEXT will cause 50 ons, 50 writes and 50 offs.

Each time the tape is turned on to write, a leader of 256 zeros is written to the tape before the data.

Table 1 shows the 500 baud tape time for a string of 200 array elements averaging 50 characters each. Out of a total tape time of 16.3 minutes, only 2.7 minutes (17 percent) is actually used for the data.

Time tests on the demonstration program (Program Listing 2) resulted in a subroutine tape time of 10 seconds versus a BASIC tape time of 1 minute 55 seconds. This reduces recorder time 91 percent. The savings will vary according to the amount of actual data being saved. Time savings could never be less than 50 percent with the worst condition of 255 bytes in each string.

200 Elements × 50 characters	= 10,000 bytes
200 Leaders × 256 zeros	= 51,200 bytes
10,000 bytes × 8 bits per byte / 500 baud × 60 sec/min	= 2.7 minutes data time
51,200 bytes × 8 bits per byte / 500 buad × 60 sec/min	= 13.6 minutes leader time
Total Data Tape Time	= 16.3 minutes

Table 1. String Array Tape Time.

```
6FD0  .................
6FE0  ..........BBBBBA
6FF0  AAAATEST BTEST A
```

Table 2. ASCII Dump.

The greatest time-waster in BASIC is caused by string handling. Look at the following listing, then look at the string space dump in Table 2.

```
10 CLEAR 48
20 T$ = "TEST"
30 A$ = T$ + "A":B$ + T$ + "B"
40 A$ = STRING$(5,"A"):B$ =
   STRINGS$ (5,"B")
```

A$ and B$ are defined in line 30, then re-defined in line 40. But a look at the 48 bytes of string data (Table 2) shows both the original and current values of A$ and B$.

CLEAR 48 reserves 48 bytes of memory for string values and tells BASIC that nothing currently written there is of any value. So BASIC sets a marker to the highest address in the string space (6FFFH) and starts writing there.

The T$ string is exactly defined in the BASIC program, so a marker (VARPTR) in another area of memory is written to point to those exact five bytes "TEST" in the program statement.

But strings A and B in lines 30 and 40 are not exactly defined. In each case, the string has to be put together in the buffer. And since the buffer doesn't exactly match a program statement, the exact string must be moved to the string space. Then the VARPTR's for A$ and B$ are adjusted to point to the correct number of bytes at their exact location in the string spaces. But there are two A$s and two B$s in the string space.

```
6FD0  ....................
6FE0  ............BBBBBA
6FF0  AAAATEBBBBBAAAAA
```

Table 3. ASCII Dump.

The clear statement in line 10 says that there are no valid values stored in the string space. So when BASIC puts together string "TEST A", it writes it at the first location in the string space. Then it writes the starting location (6FFA) and the

ARG 1	1 =	Load any array but must be used to load the first array from tape. Value 1 must be used unless you know that the string space is already compressed.
	2 =	Load subsequent (2nd, 3rd, etc.) arrays. This value saves the amount of time required for a string space compression. If string variables have been created or changed since the first array load, then you must use value 1 instead.
	3 =	Write tape of string array.
	4 =	Verify the tape saved array against the values in memory.
ARG 2		Use VARPTR to identify the array. VARPTR returns an integer value which can be used to locate the array. The array element number (0, 1, 2, etc.) must be the same number used in ARG 3. (The subroutine doesn't check this.) *Caution:* Creating additional variables after a VARPTR can cause the VARPTR value to become invalid.
ARG 3		Starting element number of array. This identifies the first element that the subroutine is supposed to use.
ARG 4		Last array element number to be used.

Table 4. Arguments.

length (6) in the VARPTR for A$. "TEST B" is put in the next available space and the starting location (6FF4H) and length (6) are written in the B$ VARPTR.

When A$ is changed to "AAAAA", BASIC doesn't bother to see if A$ is previously defined. (The previous definition isn't valid anymore.) BASIC just writes the new A$ in the next available string space and writes the new location (6FEFH) and length (5) in the A$ VARPTR.

Now the string space contains 26 bytes of memory for additional progressive writes, 10 bytes of valid data, and 12 bytes of garbage. BASIC will continue the progressive writing of strings until its need for more space will take it past location 6FD0H. At that time, it will call ROM routine 28E6H.

ROM call 28E6H reads all the string VARPTRs looking for pointers to valid data. When it finds valid data, the data is moved to the start of the string space in the same progressive manner previously seen. If we call 28E6H after line 40, the string space will be rewritten as illustrated in Table 3.

The current values of A$ and B$ are written in location 6FF6H through 6FFFH. No VARPTRs point to the rest of the data in the string space, and the next string will be written in the string space starting at the "E" at location 6FF5H (right over the old values).

Input from Tape

If all tape input were numeric, then BASIC could probably cope with data tapes written without all those starts, stops and leaders. At worst, it would need a slight delay between data elements.

But input has to accept string data, also. It must be constantly ready to turn off the tape recorder to allow time for call 28E6H to compress the string space, making room for additional progressive writes.

If BASIC knew its input were the first of a series with no string writes in between, then it could call 28E6H before turning on the tape. With the string space compressed, it could handle a continuous stream of string data.

But input doesn't tell BASIC enough. So, BASIC assumes it may need to stop the tape between any two data statements.

Print statements write all those zeros so input can stop the tape after any data statement and restart it after calling 28E6H. The zeros provide the time for the tape to start and get up to speed before it reads another statement.

Those 51,200 zeros in Table 1

0	= Operation OK
−1	= Checksum error during read
−2	= Number of elements error in ARG 3 or ARG 4.
−3	= Out of string space
−4	= Invalid number in ARG 1
−5	= Array identified in ARG 2 not string or not single dimension.
+1 or higher	= Number of errors detected during verify, if any.

Table 5. Return Values.

✓See List of Advertisers on page 306

Program Listing 1.

```
                   00100 ;
                   00110 ;QUICK STRING ARRAY TAPE SUB-ROUTINE
                   00120 ;
                   00130 ;NON-GSF INITIALIZATION
                   00140 ;
                   00150 ;WRITTEN BY JIM GLOSSER
                   00160 ;
408E               00170          ORG     408EH          ;USR JUMP ADDRESS
408E FBFD          00180 DEFUSR   DEFW    BEGIN
FFFF               00190 ENDMEM   EQU     65535          ;32767 OR 49151 OR 65535
FDF0               00200          ORG     ENDMEM-527     ;START OF THIS PROGRAM
FDF0 00            00210 NUMARG   DEFB    0
FDF1 0000          00220 ARG1     DEFW    0
FDF3 0000          00230 ARG2     DEFW    0
FDF5 0000          00240 ARG3     DEFW    0
FDF7 0000          00250 ARG4     DEFW    0
FDF9 0000          00260 NUMELM   DEFW    0
FDFB CD7F0A        00270 BEGIN    CALL    0A7FH          ;GET ARGUMENT IN 'HL'
FDFE E5            00280          PUSH    HL             ;SAVE ARGUMENT
FDFF AF            00290          XOR     A              ;ZERO 'A'
FE00 57            00300          LD      D,A            ;ZERO 'D'
FE01 3AF0FD        00310          LD      A,(NUMARG)     ;ARG INDEX ADDER
FE04 5F            00320          LD      E,A            ;PUT IN 'E'
FE05 21F1FD        00330          LD      HL,ARG1        ;START OF STORAGE
FE08 19            00340          ADD     HL,DE          ;INCREMENT FOR 2ND, 3RD
FE09 19            00350          ADD     HL,DE          ;& 4TH ARGUMENTS
FE0A E5            00360          PUSH    HL             ;ARG STORAGE ADDR
FE0B DDE1          00370          POP     IX             ;TO 'IX'
FE0D E1            00380          POP     HL             ;GET ARGUMENT BACK
FE0E DD7500        00390          LD      (IX),L         ;STORE ARGUMENT FROM 'HL'
FE11 DD7401        00400          LD      (IX+1),H
                   00410                                 ;'A' CONTAINS NUMARG
FE14 FE03          00420          CP      3              ;DONE YET?
FE16 280A          00430          JR      Z,ARGDUN       ;YES - JUMP
FE18 3C            00440          INC     A              ;NEXT ARG INDEX ADDER
FE19 32F0FD        00450          LD      (NUMARG),A     ;STORE IT
FE1C 210000        00460          LD      HL,0           ;RESET 'HL' FOR NEXT ARG
FE1F C39A0A        00470          JP      0A9AH          ;GET NEXT ARG
FE22 AF            00480 ARGDUN   XOR     A              ;ZERO 'A'
FE23 32F0FD        00490          LD      (NUMARG),A     ;RESET FOR NEXT TIME
                   01000 ;CHECK FOR VALID STRING ARRAY AND ARRAY LENGTH
                   01010 ;
40FB               01020 ARRAY    EQU     40FBH          ;START OF ARRAY POINTERS
40FD               01030 ENDARR   EQU     40FDH          ;FREE SPACE POINTER
FE26 DD2AFB40      01040 CHECK    LD      IX,(ARRAY)     ;POINT TO START OF ARRAYS
FE2A DDE5          01050          PUSH    IX             ;PUT IN 'HL'
FE2C E1            01060          POP     HL
FE2D ED4BF3FD      01070          LD      BC,(ARG2)      ;VARPTR TARGET ARRAY
FE31 B7            01080          OR      A              ;RESET CARRY
FE32 ED42          01090          SBC     HL,BC          ;IS VARPTR AN ARRAY?
FE34 F278FE        01100          JP      P,BADARR       ;NO - JUMP
FE37 DDE5          01110 FINDAR   PUSH    IX
FE39 E1            01120          POP     HL
FE3A 23            01130 NXTARR   INC     HL             ;POINT TO ARRAY LENGTH
FE3B 23            01140          INC     HL
FE3C 23            01150          INC     HL
FE3D 4E            01160          LD      C,(HL)         ;PUT LENGTH IN 'BC'
FE3E 23            01170          INC     HL
FE3F 46            01180          LD      B,(HL)
FE40 09            01190          ADD     HL,BC          ;END OF ARRAY
FE41 ED4BF3FD      01200          LD      BC,(ARG2)      ;VARPTR ADDR
FE45 B7            01210          OR      A              ;RESET CARRY
FE46 E5            01220          PUSH    HL             ;SAVE END OF ARRAY LOCATION
FE47 ED42          01230          SBC     HL,BC          ;IS VARPTR THIS ARRAY?
FE49 E1            01240          POP     HL             ;GET BACK BEFORE BRANCH
FE4A F253FE        01250          JP      P,CHKARR       ;YES-JUMP
FE4D 23            01260          INC     HL             ;POINT TO START NEXT ARRAY
FE4E E5            01270          PUSH    HL             ;PUT IN 'IX'
FE4F DDE1          01280          POP     IX
FE51 18E7          01290          JR      NXTARR         ;IS NEXT ARRAY TARGET?
FE53 DDE5          01300 CHKARR   PUSH    IX             ;START OF TARGET ARRAY
FE55 E1            01310          POP     HL
FE56 7E            01320          LD      A,(HL)         ;ARRAY TYPE
FE57 FE03          01330          CP      3              ;IS IT A STRING ARRAY?
FE59 201D          01340          JR      NZ,BADARR      ;NO -JUMP
FE5B 110500        01350          LD      DE,5           ;ADVANCE TO NUM OF DIM
FE5E 19            01360          ADD     HL,DE          ;POINT 'HL' TO NUM OF DIM
FE5F 7E            01370          LD      A,(HL)         ;NUM OF DIMENSIONS
FE60 FE01          01380          CP      1              ;IS IT SINGLE?
FE62 2014          01390          JR      NZ,BADARR      ;NO - JUMP
FE64 23            01400          INC     HL             ;POINT TO SIZE
FE65 5E            01410          LD      E,(HL)         ;PUT SIZE IN 'DE'
FE66 23            01420          INC     HL
FE67 56            01430          LD      D,(HL)
FE68 2AF7FD        01440          LD      HL,(ARG4)      ;LAST ELEMENT TO USE
FE6B 23            01450          INC     HL             ;ADJUST FOR ELEMENT 0
FE6C B7            01460          OR      A              ;RESET CARRY
FE6D EB            01470          EX      DE,HL          ;SWAP REGISTERS
FE6E ED52          01480          SBC     HL,DE          ;IS LAST ELEM TOO HIGH?
FE70 21FEFF        01490          LD      HL,-2          ;CODE IF ERROR
FE73 FA9A0A        01500          JP      M,0A9AH        ;TOO HIGH - JUMP TO BASIC
FE76 1806          01510          JR      START          ;ARRAY OK
FE78 21FBFF        01520 BADARR   LD      HL,-5          ;RETURN VALUE
FE7B C39A0A        01530          JP      0A9AH          ;RETURN TO BASIC
FE7E ED57          02000 START    LD      A,I            ;GET INTERRUPT STATUS
FE80 F5            02010          PUSH    AF             ;SAVE IT
FE81 F3            02020          DI
FE82 3AF1FD        02030          LD      A,(ARG1)
FE85 3D            02040          DEC     A              ;WAS IT 1?
FE86 2811          02050          JR      Z,INARLD       ;1ST ARRAY LOAD
FE88 3D            02060          DEC     A              ;WAS IT 2?
FE89 2811          02070          JR      Z,NXARLD       ;SUBSEQ. ARRAY LD
FE8B 3D            02080          DEC     A              ;WAS IT 3?
FE8C CA48FF        02090          JP      Z,TAPEWR       ;WRITE TAPE FOR VERIFY
FE8F 3D            02100          DEC     A              ;WAS IT 4?
FE90 CA93FF        02110          JP      Z,VERIFY       ;VERIFY
FE93 21FCFF        02120          LD      HL,-4          ;INVALID ARG1 RETURN
FE96 C3F5FF        02130          JP      RETURN
40A7               02140 BUFFER   EQU     40A7H          ;BASIC BUFFER
                   02150 ;
FE99 CDE628        02160 INARLD   CALL    28E6H          ;COMPRESS STRING SPACE
FE9C AF            02170 NXARLD   XOR     A              ;ZERO 'A'
FE9D 47            02180          LD      B,A            ;ZERO 'B'
FE9E 4F            02190          LD      C,A            ;ZERO 'C'
FE9F CD30FF        02200          CALL    CHKCNT         ;CHECK ARG4 VS. ARG3
FEA2 23            02210          INC     HL             ;ADJUST
FEA3 22F9FD        02220          LD      (NUMELM),HL    ;SAVE COUNT ELEM'S READ
FEA6 2AA740        02230          LD      HL,(BUFFER)    ;POINTER TO BASIC BUFFER
FEA9 DD2AF3FD      02240          LD      IX,(ARG2)      ;VARPTR TO ARRAY
FEAD CD1202        02250          CALL    0212H          ;TURN RECORDER ON
FEB0 CD9602        02260          CALL    0296H          ;FIND SYNC BYTE
                   02270                                 ;& TURN ON STARS
FEB3 AF            02280 NXBYRD   XOR     A              ;ZERO 'A'
FEB4 CD3502        02290          CALL    0235H          ;READ CHAR FROM TAPE
```

```
FEB7 FEØF      Ø23ØØ          CP    ØFH          ;END TAPE MARK?
FEB9 285C      Ø231Ø          JR    Z,ENDRD      ;YES - JUMP
FEBB FEØD      Ø232Ø          CP    ØDH          ;END STRING?
FEBD 28Ø7      Ø233Ø          JR    Z,STRSTR     ;YES - JUMP
FEBF 77        Ø234Ø          LD    (HL),A       ;PUT CHAR IN BUFFER
FECØ 23        Ø235Ø          INC   HL           ;POINT NEXT BUFFER SPACE
FEC1 81        Ø236Ø          ADD   A,C          ;ADD CHAR TO CHECKSUM
FEC2 4F        Ø237Ø          LD    C,A          ;SAVE CHECKSUM
FEC3 Ø4        Ø238Ø          INC   B            ;NUM OF CHAR'S IN BUFFER
FEC4 18ED      Ø239Ø          JR    NXBYRD       ;GET NEXT CHAR
FEC6 C5        Ø24ØØ STRSTR   PUSH  BC           ;NUM CHAR'S & CHECKSUM
FEC7 AF        Ø241Ø          XOR   A            ;ZERO 'A'
FEC8 B8        Ø242Ø          CP    B            ;IS STRING COUNT Ø - NULL?
FEC9 282A      Ø243Ø          JR    Z,NULL       ;YES - JUMP
FECB ED5BD64Ø  Ø244Ø          LD    DE,(4ØD6H)   ;PTR NEXT STRING SPACE
FECF 78        Ø245Ø          LD    A,B          ;STRING CHAR'S IN BUFFER
FEDØ 4F        Ø246Ø          LD    C,A          ;TO REG 'C'
FED1 AF        Ø247Ø          XOR   A            ;ZERO 'A'
FED2 47        Ø248Ø          LD    B,A          ;ZERO 'B'
FED3 EB        Ø249Ø          EX    DE,HL        ;SWAP REGISTERS
FED4 ED42      Ø25ØØ          SBC   HL,BC        ;NEW NEXT STR SPACE
FED6 EB        Ø251Ø          EX    DE,HL
FED7 C1        Ø252Ø          POP   BC           ;GET BACK BEFORE BRANCH
FED8 2AAØ4Ø    Ø253Ø          LD    HL,(4ØAØH)   ;STRING SPACE LOWER LIMIT
FEDB ED52      Ø254Ø          SBC   HL,DE        ;IS THERE ROOM FOR STRING?
FEDD F242FF    Ø255Ø          JP    P,OUTSTR     ;NO - JUMP
FEEØ C5        Ø256Ø          PUSH  BC           ;SAVE CHECKSUM AGAIN
               Ø257Ø                             ;& NUM OF CHARACTERS
FEE1 2AA74Ø    Ø258Ø          LD    HL,(BUFFER)  ;ADDR OF BUFFER
FEE4 78        Ø259Ø          LD    A,B          ;CHAR'S IN BUFFER
FEE5 3D        Ø26ØØ          DEC   A            ;ADJUST FOR LOCATION
FEE6 4F        Ø261Ø          LD    C,A          ;TO 'C'
FEE7 AF        Ø262Ø          XOR   A            ;ZERO 'A'
FEE8 47        Ø263Ø          LD    B,A          ;ZERO 'B'
FEE9 Ø9        Ø264Ø          ADD   HL,BC        ;END OF STRING IN BUFFER
FEEA Ø3        Ø265Ø          INC   BC           ;RESTORE NUM OF CHAR'S CNT
FEEB ED5BD64Ø  Ø266Ø          LD    DE,(4ØD6H)   ;NEXT STRING SPACE
FEEF EDB8      Ø267Ø          LDDR               ;MOVE STRING FROM BUFFER
               Ø268Ø                             ;TO STRING SPACE
FEF1 ED53D64Ø  Ø269Ø          LD    (4ØD6H),DE   ;POINT TO NEXT STRING SPACE
FEF5 C1        Ø27ØØ NULL     POP   BC           ;GET BACK CHECKSUM
               Ø271Ø                             ;& NUM OF CHARACTERS
FEF6 DD7ØØØ    Ø272Ø          LD    (IX),B       ;VARPTR, NUM OF CHAR'S
FEF9 DD23      Ø273Ø          INC   IX           ;VARPTR, STRING ADDR
FEFB 13        Ø274Ø          INC   DE           ;START OF STRING
FEFC DD73ØØ    Ø275Ø          LD    (IX),E       ;STORE ADDR, LOW ORDER
FEFF DD23      Ø276Ø          INC   IX
FFØ1 DD72ØØ    Ø277Ø          LD    (IX),D       ;STORE ADDR, HIGH ORDER
FFØ4 DD23      Ø278Ø          INC   IX           ;NEXT VARPTR
FFØ6 2AF9FD    Ø279Ø          LD    HL,(NUMELM)  ;ARRAY ELEMENTS TO READ
FFØ9 2B        Ø28ØØ          DEC   HL           ;SUBTRACT 1
FFØA 22F9FD    Ø281Ø          LD    (NUMELM),HL  ;ELEMENTS LEFT TO READ
FFØD CD2CØ2    Ø282Ø          CALL  Ø22CH        ;BLINK STAR
FF1Ø AF        Ø283Ø          XOR   A            ;ZERO 'A'
FF11 47        Ø284Ø          LD    B,A          ;SET CHAR COUNT = Ø
FF12 2AA74Ø    Ø285Ø          LD    HL,(BUFFER)  ;BASIC BUFFER ADDR
FF15 189C      Ø286Ø          JR    NXBYRD       ;START READING NEXT STRING
FF17 CD35Ø2    Ø287Ø ENDRD    CALL  Ø235H        ;READ CHECKSUM ON TAPE
FF1A B9        Ø288Ø          CP    C            ;CHECKSUMS SAME?
FF1B 2ØØD      Ø289Ø          JR    NZ,RDCKER    ;NO - JUMP
FF1D 2AF9FD    Ø29ØØ          LD    HL,(NUMELM)  ;NUM ELEMENTS LEFT
FF2Ø 7C        Ø291Ø          LD    A,H          ;IS IT ZERO?
```

Program continues

✓See List of Advertisers on page 306

```
FF21 B5                     OR      L
FF22 2018                   JR      NZ,NMELER     ;NO - JUMP
FF24 210000       OK        LD      HL,Ø          ;'OK' RETURN VALUE
FF27 C3F5FF                 JP      RETURN
FF2A 21FFFF       RDCKER    LD      HL,-1         ;CHECKSUM ERR. RETURN VALUE
FF2D C3F5FF                 JP      RETURN
FF30 2AF7FD       CHKCNT    LD      HL,(ARG4)     ;LAST ELEM
FF33 ED5BF5FD               LD      DE,(ARG3)     ;1ST ELEM
FF37 B7                     OR      A             ;RESET CARRY
FF38 ED52                   SBC     HL,DE         ;LAST GREATER THAN 1ST?
FF3A DØ                     RET     NC            ;YES - RETURN
FF3B E1                     POP     HL            ;CLEAR RETURN ADDRESS
                                                  ;OUT OF STACK
FF3C 21FEFF       NMELER    LD      HL,-2         ;NUM ELEM'S ERROR RET VALUE
FF3F C3F5FF                 JP      RETURN
FF42 21FDFF       OUTSTR    LD      HL,-3         ;OUT STRING SPACE RET VALUE
FF45 C3F5FF                 JP      RETURN
                  ;
FF48 CD3ØFF       TAPEWR    CALL    CHKCNT        ;CHK ARG4 VS. ARG3
FF4B 23           TAPEW1    INC     HL            ;ADJUST ELEM WRITE COUNT
FF4C E5                     PUSH    HL            ;PUT IN 'DE'
FF4D D1                     POP     DE
FF4E DD2AF3FD               LD      IX,(ARG2)     ;VARPTR 1ST STRING
FF52 CD1202                 CALL    Ø212H         ;TURN RECORDER ON
FF55 CD8702                 CALL    Ø287H         ;WRITE LEADER & SYNC BYTE
FF58 AF                     XOR     A             ;ZERO 'A'
FF59 4F                     LD      C,A           ;SET CHECKSUM = Ø
FF5A 181Ø                   JR      FSELWR
FF5C 3EØD         NXELWR    LD      A,ØDH         ;CR TO END STRING
FF5E CD6402                 CALL    Ø264H         ;WRITE BYTE
FF61 1B                     DEC     DE            ;SUB 1 ELEMENTS TO WRITE
FF62 7A                     LD      A,D           ;ANY MORE TO WRITE?
FF63 B3                     OR      E
FF64 281F                   JR      Z,ENDWR       ;NO - JUMP
FF66 DD23                   INC     IX            ;POINT TO NEXT VARPTR
FF68 DD23                   INC     IX
FF6A DD23                   INC     IX
FF6C DD7EØØ       FSELWR    LD      A,(IX)        ;STRING CHAR COUNT
FF6F FEØØ                   CP      Ø             ;IS IT NULL STRING?
FF71 28E9                   JR      Z,NXELWR      ;YES - JUMP
FF73 47                     LD      B,A           ;STRING CHAR TO WRITE
FF74 DD6602                 LD      H,(IX+2)      ;ADDR OF STRING TO 'HL'
FF77 DD6EØ1                 LD      L,(IX+1)
FF7A 7E           NXBYWR    LD      A,(HL)        ;GET CHAR FROM STRING
FF7B CD6402                 CALL    Ø264H         ;WRITE CHARACTER
FF7E 81                     ADD     A,C           ;ADD CHAR VALUE TO CHECKSUM
FF7F 4F                     LD      C,A           ;SAVE NEW CHECKSUM
FF8Ø 23                     INC     HL            ;POINT TO NEXT CHAR
FF81 1ØF7                   DJNZ    NXBYWR        ;ANY MORE CHAR'S TO WRITE?
                                                  ;YES - JUMP
FF83 18D7                   JR      NXELWR        ;GO ON TO NEXT ELEMENT
FF85 3EØF         ENDWR     LD      A,ØFH         ;END TAPE MARK
FF87 CD6402                 CALL    Ø264H         ;WRITE END TAPE MARK
FF8A 79                     LD      A,C           ;CHECKSUM
FF8B CD6402                 CALL    Ø264H         ;WRITE CHECKSUM
FF8E 21ØØØØ                 LD      HL,Ø          ;'OK' RETURN VALUE
FF91 1891                   JR      OK
                  ;
FF93 CD3ØFF       VERIFY    CALL    CHKCNT        ;CHECK ARG4 VS. ARG3
FF96 23           TAPEV1    INC     HL            ;ADJUST ELEM VERIFY COUNT
FF97 AF                     XOR     A             ;ZERO 'A'
FF98 57                     LD      D,A           ;ZERO 'D'
FF99 5F                     LD      E,A           ;ZERO 'E'
FF9A D5                     PUSH    DE            ;ERROR COUNT
FF9B E5                     PUSH    HL
FF9C D1                     POP     DE            ;NUM OF ELEM TO VERIFY
FF9D DD2AF3FD               LD      IX,(ARG2)     ;VARPTR TO FIRST ELEMENT
FFA1 CD1202                 CALL    Ø212H         ;TURN TAPE RECORDER ON
FFA4 CD9602                 CALL    Ø296H         ;FIND SYNC BYTE
FFA7 AF                     XOR     A             ;ZERO 'A'
FFA8 4F                     LD      C,A           ;SET CHECKSUM = Ø
FFA9 1818                   JR      FSELVF
FFAB CD35Ø2       NXELVF    CALL    Ø235H         ;READ CHAR FROM TAPE
FFAE FEØD                   CP      ØDH           ;CARRIAGE RET]RN?
FFBØ 28Ø3                   JR      Z,CROK        ;YES - JUMP
FFB2 E3                     EX      (SP),HL       ;GET ERROR COUNT
FFB3 23                     INC     HL            ;ADD 1 TO IT
FFB4 E3                     EX      (SP),HL       ;STORE ERROR COUNT
FFB5 1B           CROK      DEC     DE            ;SUB 1 ELEMENTS TO VERIFY
FFB6 7A                     LD      A,D           ;ANY LEFT?
FFB7 B3                     OR      E
FFB8 2827                   JR      Z,ENDVER      ;NO - JUMP
FFBA DD23                   INC     IX            ;POINT TO NEXT VARPTR
FFBC DD23                   INC     IX
FFBE DD23                   INC     IX
FFCØ CD2CØ2                 CALL    Ø22CH         ;BLINK STAR
FFC3 DD7EØØ       FSELVF    LD      A,(IX)        ;HOW LONG IS STRING
FFC6 FEØØ                   CP      Ø             ;IS IT NULL?
FFC8 28E1                   JR      Z,NXELVF      ;YES - JUMP
FFCA 47                     LD      B,A           ;PUT LENGTH IN 'B'
FFCB DD6602                 LD      H,(IX+2)      ;ADDR OF STRING \O 'HL'
FFCE DD6EØ1                 LD      L,(IX+1)
FFD1 CD35Ø2       NXBYVF    CALL    Ø235H         ;READ NEXT CHAR ON TAPE
FFD4 BE                     CP      (HL)          ;SAME AS CHAR IN MEMORY?
FFD5 28Ø3                   JR      Z,BYOK        ;YES - JUMP
FFD7 E3                     EX      (SP),HL       ;GET ERROR COUNT
FFD8 23                     INC     HL            ;ADD 1
FFD9 E3                     EX      (SP),HL       ;SAVE ERROR COUNT
FFDA 81           BYOK      ADD     A,C           ;ADD CHAR VALUE TO CHECKSUM
FFDB 4F                     LD      C,A           ;SAVE CHECKSUM
FFDC 23                     INC     HL            ;POINT NEXT CHAR IN MEMORY
FFDD 1ØF2                   DJNZ    NXBYVF        ;MORE CHAR'S IN STRING?
                                                  ;YES - JUMP
FFDF 18CA                   JR      NXELVF        ;VERIFY NEXT ELEMENT
FFE1 CD35Ø2       ENDVER    CALL    Ø235H         ;READ NEXT BYTE FROM TAPE
FFE4 FEØF                   CP      ØFH           ;IS IT END OF TAPE MARK?
FFE6 28Ø3                   JR      Z,ENTPOK      ;YES - JUMP
FFE8 E3                     EX      (SP),HL       ;ADD 1 TO ERROR COUNT
FFE9 23                     INC     HL
FFEA E3                     EX      (SP),HL
FFEB CD35Ø2       ENTPOK    CALL    Ø235H         ;READ CHECKSUM FROM TAPE
FFEE B9                     CP      C             ;ARE CHECKSUM THE SAME?
FFEF 28Ø3                   JR      Z,CKSMOK      ;YES - JUMP
FFF1 E3                     EX      (SP),HL       ;ADD 1 TO ERROR COUNT
FFF2 23                     INC     HL
FFF3 E3                     EX      (SP),HL
FFF4 E1           CKSMOK    POP     HL            ;GET TOTAL ERROR COUNT,Ø=OK
FFF5 CDF8Ø1       RETURN    CALL    Ø1F8H         ;TURN TAPE RECORDER OFF
FFF8 F1                     POP     AF            ;GET SAVED INTERRUPT STATUS
FFF9 E2FDFF                 JP      PO,GOBACK     ;JUMP IF INTERRUPTS WERE
                                                  ;DISABLED UPON ENTRY INTO
                                                  ;SUB-ROUTINE
FFFC FB                     EI                    ;TURN ON INTERRUPTS
FFFD C39AØA       GOBACK    JP      ØA9AH         ;RETURN TO BASIC
                  ;
ØØ72                        END     72H           ;JUMP TO BASIC
```

NEW COLLEGE BOARDS 81/82 for TRS-80 PET, APPLE™

The best way to sharpen your skills for the College Boards is to work on actual examinations. Each of these program sets confronts the user with a virtually limitless series of questions and answers. Each program is based on past exams and presents material of the same level of difficulty and in the same form used in the College Board examination. Scoring is provided in accordance with the formula used by College Boards.

SAT, PSAT, N.M.S.Q.T., set includes 20 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, and Mathematics. Price $149.95

EDUCATOR EDITION - SAT, PSAT includes all of the above programs plus detailed solutions and explanations for each problem plus drill exercises. SAT set includes 26 programs $229.95

GRADUATE RECORD EXAMINATION set includes 23 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, Mathematics, Logical Diagrams, Analytical Reasoning. $199.95

EDUCATOR EDITION - Graduate Record Exam Set includes 33 programs. $289.95

Owners of our initial College Board series can upgrade their package to the College Board 81-82 specs. including the all new reading comprehension, sentence completion plus expanded vocabulary and mathematics sections for $69.95.

ALL TIME SUPER STAR BASEBALL & SUPER STAR BASEBALL

ALL TIME SUPER STAR BASEBALL Sample Lineup

B. Ruth	T. Williams
L. Gehrig	J. Foxx
J. DiMaggio	H. Greenberg
J. Jackson	R. Hornsby
G. Sisler	H. Wilson
S. Musial	B. Terry
T. Cobb	M. Mantle
W. Mays	H. Aaron
C. Young-P	W. Johnson-p

SUPER STAR BASEBALL Sample Lineup

D. Parker	J. Rice
W. Stargell	H. Aaron
W. Mays	L. Brock
P. Rose	R. Carew
O. Cepeda	H. Killebrew
C. Yazstremski	R. Allen
W. McCovey	R. Leflore
R. Jackson	R. Zisk
G. Brett	B. Madlock
R. Guidry-P	T. Seaver-p

Performance is based on the interaction of actual batting and pitching data. Game can be played by one or two players with the computer acting as a second player when desired. Players select rosters and lineups and exercise strategic choices including hit and run, base stealing, pinch hitting, intentional walk, etc. Highly realistic, there are two versions, ALL TIME SUPER STAR BASEBALL, and SUPER STAR BASEBALL featuring players of the present decade. Each includes about 50 players allowing nearly an infinite number of roster and lineup possibilities.

***Both Games $24.95**

SWORD OF ZEDEK

Fight to overthrow Ra, The Master of Evil. In this incredible adventure game, you must confront a host of creatures, natural and supernatural. To liberate the Kingdom, alliances must be forged and treasures sought. Treachery, deceit and witchcraft must be faced in your struggles as you encounter wolves, dwarves, elves, dragons, bears, owls, orcs, giant bats, trolls, etc. Each of the twelve treasures will enhance your power, by making you invisible, invulnerable, more eloquent, more skillful in combat, etc., etc., as you explore the realms of geography, both on the surface and underground. Dungeons, temples, castles, mountains, etc., are all a part of the fantastic world of Ra. Each game is unique in this spectacular and complex world of fantasy. $24.95

ISAAC NEWTON

Perhaps the most fascinating and valuable educational game ever devised - ISAAC NEWTON challenges the players (1-4) to assemble evidence and discern the underlying "Laws of Nature" that have produced this evidence. ISAAC NEWTON is an inductive game that allows players to intervene actively by proposing experiments to determine if new data conform to the "Laws of Nature" in question. Players may set the level of difficulty from simple to fiendishly complex.

In a classroom setting the instructor may elect to choose "Laws of Nature" in accordance with the complete instruction manual provided.

For insight into some of the basic principles underlying **ISAAC NEWTON** see *Godel, Escher, Bach* by Douglas R. Hofstadter, Chapter XIX and Martin Gardner's "Mathematical Games" column in **Scientific American**, October, 1977 and June, 1959. $24.95

TIME TRAVELER

The best of the adventure games. Confronts player with complex decision situations and the demand for real time action. Using the Time Machine, players must face a challenging series of environments that include; The Athens of Pericles, Imperial Rome, Nebuchadnezzar's Babylon, Ikhnaton's Egypt, Jerusalem at the time of the crucifixion, The Crusades, Machiavelli's Italy, the French Revolution, the American Revolution, and the English Civil War. Deal with Hitler's Third Reich, Vikings, etc. At the start of each game players may choose a level of difficulty . . . the more difficult, the greater the time pressure. To succeed you must build alliances and struggle with the ruling powers. Each game is unique. $24.95

*ALL PROGRAMS AVAILABLE FOR TRS-80, APPLE II & PET

*Programs for APPLE or TRS-80 are on

☐ **disk or** ☐ **cassette, *please specify.***

All programs require 16K ● TRS-80 programs require LEVEL II BASIC ● APPLE programs require Applesoft BASIC

Krell Software

Send check or money order to

21 Milbrook Drive, Stony Brook, NY 11790

(516) 751-5139

✓375

✓See List of Advertisers on page 306

are an insurance policy to make sure you can read all the data you saved, even if the recorder has been turned off for a string compression.

Machine Language Speed

We've all heard about the 4½ hour Radio Shack mailing list sort that in machine code will run in nine seconds. Machine language has to be faster! Right?

No! The BASIC Interpreter is itself a machine language program. And all identical instructions run at the same speed in the Z-80 CPU, regardless of whether they are BASIC Interpreter instructions or a subroutine.

Then why the difference between the 4½ hours and the nine seconds? Simple! The subroutine has four hours, 29 minutes and 51 seconds less work to do.

BASIC has all kinds of conditions to consider and respond to. A machine language subroutine is written to respond to a very limited set of conditions and in a very precise manner. We won't save four hours with this one, but almost 14 minutes out of 16 ain't bad.

We will provide the subroutine exactly four instructions and then turn it loose.

- •What to do (Write, Verify or Read)
- •What array to use
- •With which element to start
- •With which element to stop

Your BASIC program gives the subroutine accurate data concerning the array and the number of elements. The array must be singly dimensioned, DIM A$(300), and the last element must be a valid element such as 290.

Arguments (Instructions)

Four specific argument values (see Table 4) are required by the subroutine. Arguments are passed to the subroutine by using a variable defined as an integer and using the OR operator to assign consecutively the four arguments to the same variable. Line 580 of the demonstration program (Listing 2) shows the preferred manner of passing argument values.

The POKE *****, 0 statement in line 550 makes sure that NUMARG in Listing 1 is initialized to (0) before branching to the subroutine. The subroutine will normally re-initialize itself. However, if an error occurs in line 580, the subroutine can be looking for ARG 2,3 or 4 upon reentry. The POKE statement ensures that the subroutine will accept the arguments in their proper order.

Note the caution concerning ARG 2 (Table 4). Creating additional simple variables will cause array VARPTR's to be relocated. To prevent that, A2 = VARPTR(T$(A3)): is the last variable defined before going to the subroutine transfer line (Listing 2, line 570). Variable A is also defined before A2 for the same reason. (See Listing 2, line 130.)

I use a single, long array string for each complete data record and I extract individual fields with the MID$ function. Numeric values are included with the string data by using the STR$(X) function.X = VAL (MID$ (Y(N),P,L) extracts the numeric data.

My approach causes some slowing of program performance, but simplifies data saves and reduces array overheads.

```
                    DEMONSTRATION PROGRAM

100 CLEAR 1000:DEFINT A-Z:DIM T$(26)
110 REM DISK USERS MUST EXECUTE DEFUSR IN LINE 120 AS FOLLOWS
                    16K = DEFUSR &H7DFB
                    32K = DEFUSR &HBDFB
                    48K = DEFUSR &HFDFB
120 REM DEFUSR=&H****
130 A=0:A3=1:A4=26
140     REM   BUILD TEST STRINGS
150 FOR X=1TO15:T$(X)=STRING$(5,X+64)+STR$(X)
160 FOR X1=1TO5:T$(X)=T$(X)+CHR$(X1+64):NEXT:NEXT
170 T$(16)="":      REM NULL STRING
180 FOR X=17TO26:T$(X)=STRING$(10,X+64):NEXT
190 CLS:FOR X=1TO26:PRINT X;TAB(6)T$(X):FOR Y=1 TO 100:NEXT:NEXT
200     REM   TEST FUNCTIONS
210 INPUT"ENTER TO CONTINUE";X
220 CLS:PRINT"1 = QUICK TAPE WRITE"
230 PRINT"2 = QUICK TAPE VERIFY"
240 PRINT"3 = QUICK TAPE READ"
250 PRINT"4 = REGULAR TAPE WRITE"
260 PRINT"5 = REGULAR TAPE READ"
270 PRINT"6 = DISPLAY TEST ARRAY"
280 INPUT"ENTER FUNCTION";X:IF X<1 OR X>6 GOTO 280
290 ON X GOTO 310 , 340 , 370 , 410 , 440 , 190
300     REM   QUICK TAPE WRITE
310 GOSUB 490 :A=0:A1=3:GOSUB 550
320 IF A<>0 THEN GOTO 510 ELSE PRINT"QUICK WRITE DONE":GOTO 210
330     REM   QUICK TAPE VERIFY
340 GOSUB 490 :A=0:A1=4:GOSUB 550
350 IF A<>0 THEN GOTO 510 ELSE PRINT"VERIFY DONE":GOTO 210
360     REM   QUICK TAPE READ
370 GOSUB 470 :A=0:A1=1:GOSUB 550
380 IF A<>0 THEN GOTO 510 ELSE PRINT"QUICK READ DONE"
390 INPUT"ENTER TO SEE TEST ARRAY";X:GOTO 190
400     REM   REGULAR TAPE WRITE
410 GOSUB 490 :FOR X=1TO26:PRINT#-1,T$(X):NEXT
420 PRINT"REGULAR WRITE DONE":GOTO 210
430     REM   REGULAR TAPE READ
440 GOSUB 470 :FOR X=1TO26:INPUT#-1,T$(X):NEXT
450 PRINT"REGULAR READ DONE":GOTO 390
460     REM   RESET ARRAY TO NULLS BEFORE READ
470 FOR X=1TO26:T$(X)="":NEXT
480     REM   CHECK RECORDER
490 INPUT"ENTER WHEN RECORDER READY";X:RETURN
500     REM   ERROR HANDLING
510 IF A>0 THEN PRINT"VERIFY DETECTED";A;"ERRORS":GOTO 210
520 PRINT"SUB-ROUTINE ERROR CODE ";A;"RETURNED":GOTO 210
530     REM   INITIALIZATION FOR START OF USR SUB-ROUTINE
540 REM IN LINE 550, POKE VALUE '0' AS FOLLOWS:
                    16K = POKE 32240,0
                    32K = POKE -16912,0
                    48K = POKE -528,0
550 POKE *****,0
560     REM   USR SUB-ROUTINE ENTRY
570 A2=VARPTR(T$(A3))
580 A=USR(A1) OR USR(A2) OR USR(A3) OR USR(A4)
590 RETURN
```

Program Listing 2.

✓See List of Advertisers on page 306

170- 180	DEFUSR points BASIC to machine language subroutine.
270- 490	BEGIN accepts the arguments (instructions) from BASIC and stores them for program use. It keeps returning to BASIC until has all four arguments.
1000-1530	CHECK looks at the array to make sure that it's a string array and that it is sized as large or larger than the last element number to be used.
2000-2130	START determines the operation needed and proceeds to read, verify or write routines.
2160-	INARLD compresses the string space.
2170-2270	NXARLD sets pointers to accept a new array and turns on the tape recorder.
2280-2390	NXBYRD reads characters from the tape until a carriage return (OD) or end of tape mark (OF) is read. Characters are stored in a buffer and character value is added to a checksum.
2400-2860	STRSTR checks for enough string space to store the string, then moves it from the buffer to the string space and writes the location and length in VARPTR. It initializes for the next string and returns to NXBYRD.
2870-3080	ENDRD reads the checksum from tape and compares it with the checksum calculated in NXBYRD.
3100-3190	TAPEWR initializes for tape writing and turns on the tape recorder.
3200-3280	NXELWR writes the CR (OD) after a string and checks to see if more strings need to be written to tape.
3290-3420	FSELWR writes the string characters to the tape and adds the character value to the checksum.
3430-3480	ENDWR writes the end of tape mark (OF) and the checksum to tape.
3500-3630	VERIFY initializes for verification and turns on tape recorder.
3640-3690	NXELVF checks for CR (OD) after each string.
3700-3830	CROK checks for more array elements to verify and sets pointer to next.
3840-3890	NXBYVF compares character read from tape and character in memory and increments error count if not same.
3900-3950	BYOK adds character value to checksum and checks for end of string.
3960-4010	ENDVER verifies end of tape mark.
4020-4080	ENTPOK reads checksum from tape and compares with calculated checksum.
4090-4150	RETURN turns on interrupts, turns off tape and goes back to BASIC with return value in HL.

Program Summary

When the subroutine passes control back to BASIC, it returns a value in HL. This value is assigned by BASIC to the variable used at the beginning of the USR branch line in the demonstration listing. Reading the variable after control returns to BASIC, will tell you the results of the subroutine's operation.

Return values are listed in Table 5. Those values are demonstrated in Listing 2. Table 6 should give you some hints about array storage. This should help you with modifications.

Using the Subroutine

To assemble the subroutine set memory size to 32240, 48624 or 65008 for 16, 32 or 48K respectively. Load the program from tape with the System command. After the program loads, answer the second prompt with / Enter. Your BASIC program is now ready to load.

Loading in Disk BASIC destroys the USR link address at 408EH, so it must be re-established with the DEFUSR instruction as demonstrated in line 120 of Listing 2.■

```
BASIC PROGRAM TO CREATE ARRAYS.
10 DIM AA%(3)    'CREATE INTEGER ARRAY
20 DIM BB!(3)    'CREATE SINGLE PRECISION ARRAY
30 DIM CC#(3)    'CREATE DOUBLE PRECISION ARRAY
40 DIM DD$(3)    'CREATE STRING ARRAY
50 DIM E$(2,2)   'CREATE 2 DIMENSION ARRAY
60 END

THE POINTER TO THE START OF ARRAYS IS AT LOCATION 40FBH.
      57A0H
40FB  A0 57 29 58 BF 56 04 04 04 04 04 04 04 04 04 04

THE POINTER TO THE START OF FREE SPACE (FOLLOWING ARRAYS)
IS AT LOCATION 40FDH.
      5829H
40FD  29 58 BF 56 04 04 04 04 04 04 04 04 04 04 04 04

DUMP OF ARRAY STORAGE SPACE (57A0H to 5829H).
       #     %     &  *    $1
       2  A  A    11  1     4  (0)   (1)   (2)   (3)
57A0  02 41 41 0B 00 01 04 00 00 00 00 00 00 00 00 00
       #     %     &  *    $1
       4  B  B    19  1     4     (0)         (1)
57B0  04 42 42 13 00 01 04 00 00 00 00 00 00 00 00 00
                               #     %     &  *    $1
          (2)         (3)      8  C  C    35  1     4
57C0  00 00 00 00 00 00 00 00 08 43 43 23 00 01 04 00

                (0)                     (1)
57D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

                (2)                     (2)
57E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
       #     %     &  *    $1
       3  D  D    15  1     4    (0)      (1)    (2)
57F0  03 44 44 0F 00 01 04 00 00 00 00 00 00 00 00 00
                   #     %     &  *    $1    $2
           (3)      3     E    32  2     3     3 (0,0)
5800  00 00 00 00 03 00 45 20 00 02 03 00 03 00 00 00

           (1,0)   (2,0)    (0,1)    (1,1)    (2,1)
5810  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

       (0,2)    (1,2)    (2,2)   START OF FREE SPACE
5820  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

KEYS

#=Array Type
  2=Integer
  3=String
  4=Single Precision
  8=Double Precision
%=Array Name
&=Number of bytes from this location to last byte of array.
*=Number of dimensions

$1=Size-1st dimension
$2=Size-2nd dimension, etc.
(n)=Array element
NOTE: Number of bytes per array element is equal to array type.

String array elements consist of VARPTR which gives size and starting location of string.

Table 6. This is a detail of the BASIC array storage arrangement.

This month we mean it!

The Real Rules of 78s

R. L. Conhaim
15506 Kiamichi Rd., Apt. 1
Apple Valley, CA 92307

No one wants to pay any more than necessary, with the high cost of borrowing money these days. But, few people realize that early payoff can save something from the interest portion of a loan. The question is, how much can you save?

There's a formula that many banks and loan companies use called The Rule of 78's. It gets its name from the fact that the numerical value of the months in a year, when added together, total 78. That is, 1 + 2 + 3 + ... + 12 = 78. The formula used to determine the rebate is:

$$\text{Rebate} = \frac{(n - k + 1)(n - k)}{n^2 + n} \times FC$$

where n = total number of payments in the contract, k = the number of payment periods already passed at early payoff, and FC = the total finance charge (interest) in the original loan.

Take the case of a person with an original contract of 36 months which has a total interest of $467.24. How much could he save if he paid off at the 15th month? Plug the numbers into the formula and out comes $162.06 or almost 35 percent of the original finance charge.

A pocket calculator with square root capability could give you the answer to the problem. But, suppose you'd like to see how much could be saved at the 16th, 17th and all subsequent months? That's where your computer outshines the laborious calculator method. With a simple program like that shown in Listing 1, you can print out all the answers in just a couple of seconds.

The program is short and straightforward. You just enter the three variables which are printed out for recordkeeping purposes. If you don't need a printout, change the LPRINT s in lines 60, 70, 80, 130 and 140 to PRINT commands.

The variable X is used to stop the program for every 12 output lines so the answers won't scroll off the screen. Typing CONT restarts the printout where it left off. Line 120 stops the execution of the program when the payoff month equals the number of months in the contract.

The left bracket symbol ([) in line 100 is the exponentiation sign, and in many computers and printers is shown as an up arrow (↑).■

```
10 REM "RULE OF 78 REBATE CALCULATION"
20 CLS:INPUT "ENTER TOTAL FINANCE CHARGE";FC
30 INPUT "ENTER TOTAL NUMBER OF MONTHS IN CONTRACT";N
40 INPUT "ENTER NUMBER OF PAY-OFF MONTH";K
50 CLS:LPRINT "TOTAL FINANCE CHARGE = ";FC
60 LPRINT "TOTAL MONTHS IN CONTRACT ";N
70 LPRINT "FIRST PAY-OFF MONTH ";K
80 LPRINT TAB(10); "PAY-OFF MONTH             REBATE"
90 X=0
100 RB= (((N-K+1)*(N-K))/(N[2+N))*FC
120 IF K=N GOTO 200
130 LPRINT TAB(15);K;
140 LPRINT USING "              ##,###.##";RB
150 X=X+1
160 IF X=12 GOTO 180 ELSE 170
170 K=K +1:GOTO 100
180 X=0 :STOP
190 GOTO 170
200 END
```

Program Listing 1

```
TOTAL FINANCE CHARGE =  467.24
TOTAL MONTHS IN CONTRACT  36
FIRST PAY-OFF MONTH  15
          PAY-OFF MONTH             REBATE
               15                   162.06
               16                   147.33
               17                   133.30
               18                   119.97
               19                   107.34
               20                    95.41
               21                    84.19
               22                    73.66
               23                    63.84
               24                    54.72
               25                    46.30
               26                    38.59
               27                    31.57
               28                    25.26
               29                    19.64
               30                    14.73
               31                    10.52
               32                     7.02
               33                     4.21
               34                     2.10
               35                     0.70
```

Sample Problem

SINCE YOU COULDN'T WAIT......

WE DIDN'T.....!

IMB

INTRODUCES THE FIRST IN A SERIES OF ACTION PACKED, HIGH RESOLUTION COMPUTER SIMULATIONS.

METEOR STORM © 1981

CONTAINS HIGH-RESOLUTION, SOUND ENHANCED LASER BLASTS; COCKPIT PERSPECTIVE; VARIABLE SKILL LEVEL; AUTOMATIC MULTI-GAME SCORE TALLY AND JOYSTICK OPERATION. (JOYSTICK NOT INCLUDED.)

EXCLUSIVELY DESIGNED FOR

TRS-80* COLOR
EXTENDED COMPUTER

Send check or money order $12.95
IMB: ILLUSTRATED MEMORY BANKS
P.O. BOX 289
WILLIAMSTOWN, MA 01267 ✓467

* A TRADEMARK OF TANDY CORPORATION

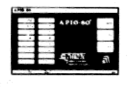
New Products

For the TRS-80*

MODELS 1&3

Digital I/O Port

+teaching newsletter



8 TTL outputs, 6 inputs; assembled & tested PC board; control the outside world from BASIC or TBUG; lowest price available $47 . . . sample newsletter $2.50

TTY-interface

HARD COPY

.....low cost printout

convert any ASCII or Baudot teletype to a TRS-80 printer; no expansion interface needed $62; optional loop supply $30; write for more info

LIGHT * PEN

with sophisticated software

add excitement to your software, LTPEN-80 adds four new functions to BASIC; in an instant, pick any (X,Y) point or character position on the screen simply by pointing the pen!; enhance games . . . simplify user entry $27

all prices include postage . . . call or write for more information

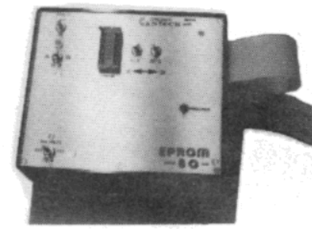
also . . . need a printer but lack a Teletype? see William Strube ad in this issue.

SYNTEX for more info. call 717/733-4769
electronic innovations
Box 4034 LANCASTER PA 17604

* TRS-80 IS A TRADEMARK OF TANDY CORP. (C) 80 BY SYNTEX ELECTRONIC INNOVATIONS

EPROM-80 FOR TRS-80

Programmer for 2700 Type EPROMs (2704/08, 2758, 2716, 2732, 2732A, Mostec 2764, Intel 2764)



The logical solution to EPROM programming! One program for all type EPROMs. Timing under software control. 28 pin zero insertion force socket. No external power supplies required. 36" ribbon cable connects directly to expansion interface. Average programming time 100 seconds per 1K bytes. Includes program on disc and cassette. Program operated in two modes. Mode 1: EPROM type select. Program prompts user to set 4 toggle switches depending upon EPROM type. Mode 2: Type of action: Reads one EDTASM file from disc or cassette (assembled anywhere) into buffer. Reads EPROM data into buffer. Writes buffer data to EPROM. Automatic verify after write. And More. Minimum requirements: TRS-80 LEVEL 2, 16K RAM . . . . . . PRICE $290.00.

CANTECH DATA ✓483
7826 W. ARGYLE, NORRIDGE, ILL 60656
PHONE (312) 887 0161

LISP

INTERPRETER FOR THE TRS-80*

SEE THE AUGUST, '79 ISSUE OF BYTE TO FIND HOW EASY IT IS TO USE LISP.

- INCLUDES MORE THAN 50 FUNCTIONS & PREDICATES.
- WRITTEN IN ASSEMBLY LANGUAGE FOR HIGH-SPEED OPERATION.
- FULL ARITHMETIC CAPABILITY.
- LISP SOURCE FROM KEYBOARD, CASSETTE, OR DISK.
- OUTPUT TO SCREEN OR PRINTER
- COMPLETE SYNTAX CHECKING WITH DETAILED ERROR CODES.
- USER'S MANUAL
- SUPPLIED ON CASSETTE FOR 16K-48K LEVEL II.

PRICE: 49.95
USER'S MAUAL ONLY $5.00 (APPLICABLE TO LATER PURCHASE OF SOFTWARE).

CYBER INNOVATIONS
WORLD TRADE CENTER
P. O. BOX 58657
DALLAS, TEXAS 75258 ✓259

*TRADEMARK OF TANDY CORPORATION

TRS-80 & OTHER NEEDS FILLED FOR LESS

+ + + COMPATIBLE DISK DRIVES WITH POWER SUPPLY AND CASE—90 DAY WARRANTY + + +

* 40 TRACK (102,400 BYTE/DISK) DISK DRIVE WITH P.S. AND CASE **$299**
* 8 IN. DRIVE & P.S./CASE **$689** WITH P.S./CASE FOR 3 DRIVES **$879**
* 80 TRACK (204.8K BYTE) **399**
* 4-DRIVE CABLE **$28** ** 10 DISKS-5 IN. @ **$24**-8 IN. @ **$36** HARD CASE **$3 & 5**

BASE 2 PRINTER **$599** EPSON MX—80 PRINTER—CALL MICROLINE 80 PRINTER **$399**

* CENTRONICS 737 **$699** MICROLINE 82 **$549** CABLE @ **$25**
* MICROLINE 83 **$829**
* LEEDEX 12 IN B&W MONITOR **$114** LEEDEX 12 IN B&G MONITOR **$138**
* APF 10 IN B&W MONITOR **$109**
* 300 BAUD MODEM (ORIG/ANS) $115. AND UP + + + + + + + + 16K MEMORY SET (200 NANO) **$23**
* 16K MODEL III RADIO SHACK SYSTEM **$849**
* APPLE, ATARI, RADIO SHACK HARDWARE/SOFTWARE DISCOUNTED, A/R, A/P, G/L, P/R FOR **$200 or $59 ea.** (MODEL 1) & **$329 or $80 ea.** (MODEL 2). APPLICATIONS INTERACT & ARE COMPLETE & PROFESSIONAL. WILL RUN ON OTHER COMPUTERS. THIS IS A SPECIAL INTRODUCTORY PRICE.
* ASK FOR FREE FLYER WITH OUR LOW PRICES—DEALER INQUIRIES INVITED MASS. RESIDENTS ADD 5% TAX—F.O.B. TEWKSBURY—FREIGHT EXTRA.

M/C, VISA OR CHECK ACCEPTED. TRS-80 IS A REG. TRADEMARK OF TANDY CORP. ✓105

OMNITEK SYSTEMS 1899 MAIN ST., TEWKSBURY, MASS. 01876 CALL 617/851-4580

Model 488-80B For Model 1 Operation

Model 488-80C For Model 3 Operation

IEEE-488 TO TRS-80* INTERFACE

Everything needed to add *powerful* BASIC GPIB-488 controller capability to TRS-80 Model 1 or 3, Level 2 or DOS with a minimum of 16K.

488-80B or 488-80C $245.
+ shipping, insurance & tax

WHEN ORDERING SPECIFY DISK OR TAPE

SCIENTIFIC ENGINEERING LABORATORIES
11 Neil Drive • Old Bethpage, NY 11804
Telephone: (516) 694-3205 ✓291

**Trademark of Tandy Corp.*
There is no affiliation between Scientific Engineering Laboratories and Tandy Corp. or Radio Shack.

No. 4 UNBELIEVABLE OPPORTUNITY!

You've Written a Fantastic Game? Then We'd Like to Publish It!

We're looking for hot GAME programs:

ARCADE (HI-SPEED GRAPHICS)
ADVENTURE FORMAT
FANTASY WARGAMING
BOARD GAMES
LOGIC & PUZZLE GAMES

There's Gold in them there Games! Write for our free Programmer's Kit today.

INSTANT SOFTWARE, INC.
Submissions Dept.
Peterborough, NH 03458 ✓2

Build a better error trap and bugs will beat a path from your programs.

To Err is . . . Forbidden—Part II

John D. Adams
13126 Tripoli Ave.
Sylmar, CA 91342

```
10 ON ERROR GOTO 30000
30000 IF ERR/2+1=11 THEN RESUME NEXT
30010 PRINT"ERROR NUMBER ";ERR/2+1;"IN LINE ";
      ERL
30020 INPUT"PRESS ENTER TO CONTINUE";Z$:RESUME
      10
```

When the TRS-80 discovers an error, it shifts to either command or edit mode. This seems perfectly sensible: If the machine cannot do what you have asked it to, or, if continued execution will produce nothing but garbage, the best thing is to stop. Stopping execution poses no problem *unless* you have non-program data stored in memory. In this case, there is no way to restart without losing the stored data.

Level II BASIC incorporates a short set of statements that set up an error trapping routine. These routines deal with errors in two ways. One handles loss of data. Another allows the computer to ignore an error of little consequence that occurs in a program so that execution can continue. For example, if I were taking logarithms from numbers generated internally and there was a chance a negative number could pop up, I might want to simply skip that number and go on.

Record Student Grades

I use a homemade program to record my students' grades. During the course of the program, individual test grades are used as divisors. When a student is absent for a test, a zero is entered until he takes a makeup, at which time the program changes the grade. Since zero cannot be used as a divisor, a potential exists for a division by zero error. As the program is tight on RAM space, I did not want to add another module to deal with this eventuality.

The error trapping routine below was developed to handle this problem. Its structure illustrates how such routines work.

The ON ERROR GOTO instruction in line 10 initiates the routine, substantially altering the way in which errors are processed. When an error is discovered, execution doesn't shift to command or edit mode, but proceeds to the indicated line number for further instructions. This process works much like a subroutine, and the instruction, like GOSUB, works as a paired instruction. When the routine has finished, a RESUME is necessary. Obviously, this instruction must be read *before* an error occurs and should be placed at the beginning of the program.

The routine could be terminated at this point by adding the line: 30000 RESUME NEXT. This would cause the computer to ignore the error and branch back to the next program statement to continue execution. In my grade program, this is what I wanted if the error was an attempt at division by zero. But, if it was some other error that materially altered program output, that error would also be ignored.

The 80 reports errors using abbreviations; these codes appear on pages B/2 and B/3 of the manual. Computers deal with everything as numbers, however, and their conversion into letters is for human convenience. The error condition is carried internally as a number, which can be used in error trapping to designate specific errors. The instruction ERR/2 + 1 in lines 30000 and 30010 is for this purpose. Entering PRINT ERR/2 + 1 after an error occurs will return a number which identifies the error. The number codes for errors appear on page B/1 of the manual.

In my grade program, the error I want ignored is division by zero, error code 11. Line 30000 means, "If the error you have found is division by zero, forget it and pick up execution on the line or statement following the error. If it is not, then drop to line 30010."

PRINT ERL returns the line number in which the error happened. Line 30010 tells us what and where the error is, and line 30020 stops execution so we can make some decisions. Possibly the error involves inoperable data, such as trying to take the square root of a negative number. In this case, we just want to go back to make another data entry. On the other hand, if the error is one that needs fixing, we are going to have to leave the execute mode for repairs.

Dumping Data

After adding, deleting or editing lines, we cannot continue execution and running the program will cause data loss. If there is considerable data involved, a module can be built into the error trapping routine itself to dump data to tape before you hit the break key. This makes a matching loading routine necessary.

There are defaults for the ERL instruction: If no error has been made, it returns a zero; if the error was made in direct mode, such as trying to use INPUT in command mode, the number 65535 will be returned.

The RESUME statement is similar to RETURN; its action is similar, but options are available which enhance its effect. Used alone, as RESUME 0, it will return to the statement in which the error was committed. Run these lines:

```
10 FOR X = -10 TO 10
20 PRINT SQR(X)
30 NEXT:END
```

Since we cannot take square roots of negative numbers, an FC message is returned. Now add these lines:

```
5 ON ERROR GOTO 100
100 X=X+1:RESUME (or RESUME 0)
```

The trapping routine lets us increment X until the data becomes operable.

A line number may also be specified for the return point. The following lines illustrate this:

```
5 ON ERROR GOTO 100
10 FOR Y=1 TO 25
20 X=RND(20)-10
30 PRINT X,SQR(X)
40 NEXT
100 RESUME 500
500 X=-X:PRINT SQR(X):"I"
510 NEXT
```

Although square roots of negative numbers cannot be expressed as real values, they may be represented as complex numbers after the real and imaginary parts are computed. When a negative number is processed, an FC error will result, which sends execution to line 100. The RESUME 500 in that line further branches to the indicated line number for complex number computation and printout, after which we are returned to the loop in line 10. In this case the routine is acting much like a flag which detects and processes negative numbers.

RESUME NEXT returns to the statement following that in which the error was made. This is useful if we want to ignore all negative numbers in a square root program. In my grading program, this is what I wanted.

Using the ON ERROR GOTO instruction is like flipping a light switch—it will stay on until we turn it off. It may be that we want this routine to operate in one part of a program, but not in another. In that case, the error trapping process is disabled by the instruction ON ERROR GOTO 0. The function can now be turned on and off at will.

The error trapping routine discussed here is a simple one. Error trapping routines can be built to do some remarkable tasks. These routines are more complex to construct and should be tested to see that they are working properly. An additional instruction, ERROR n, is provided for that use. Look at the following lines:

```
5 ON ERROR GOTO 1000
10 ERROR 8
20 .......
```

Line 10 causes the computer to behave exactly as it would if an undefined line error had been found. Any of the recognized errors can be simulated by using the number code for that error. In this way, routines can be checked for proper operation.

Error trapping routines are very useful in debugging programs. Using the ON ERROR GOTO nnn instruction as line 1, and a suitable routine at line 30000, the program can be cleaned up and these lines deleted after corrections are made. We've come a long way from Level I's WHAT, HOW and SORRY! ■

Subscription Problem?

80 Microcomputing does not keep subscription records on the premises, therefore calling us only adds time and doesn't solve the problem.

Please send a description of the problem and your most recent address label to:

80 Microcomputing
Subscription Dept.
PO Box 981
Farmingdale, NY 11737

Thank you and enjoy your subscription.

Of ARPS and Moogs and 80s.

A Quick Riff on Synthesizers

Dave Keen and Dave Dischert
RD 1, Box 432
State Highway 83
Cape May Court House, NJ 08210

Do you remember what occupied your spare time before you bought your computer? Was it ham radio? Stamp collecting? Model railroading?

We played music. We used the big ARPs and Moog synthesizers for live performing and recording. That old flame was ignited again with the recent appearance of several music synthesizers for the TRS-80.

Tone generation is not too difficult on the '80. Many of the games on the market utilize the notes and sound effects created by toggling flip flops at the cassette port.

But music is more than a sequence of notes. It requires polyphonic voices, that is, many notes played simultaneously; oscillator waveform selection (square, sine, sawtooth, triangle, pulse); dynamic control of the composite waveform's harmonics; most important, a low pass filter with variable cut-off frequency (and preferably variable Q or resonance); and a volume control.

Such bare bones features are found on even the cheapest piano keyboard style electronic synthesizers.

ARPS and Moogs

Let's briefly examine how sounds are created on performance synthesizers.

In 1964, Robert Moog produced a system of dc voltages that controlled various parameters such as frequency, filtering and volume.

A piano style keyboard, set up to output dc voltages in relation to each key, connects to one or more oscillators. These VCOs (voltage controlled oscillators), are capable of producing several waveforms. Each shape is made up of different overtones (see Fig. 1). It is important to be able to select these shapes as they determine the resulting sound.

A sine wave is a pure tone with no overtones. Square waves have a hollow sound. Squeeze the sides of a square wave together and you get a pulse waveform. This reedy, nasal tone is useful in duplicating saxophone sounds. Violins and brass effects can be simulated by first starting with a sawtooth.

Most manufacturers use oscillators which respond to one volt per octave. If, for example, an oscillator played a middle C, then raising its input to two volts would produce a C one octave higher. You can see the precision needed in designing oscillator controllers. If the voltage is off by as little as one-twelfth volt, you're playing in another key!

Random noise generators, creating white, pink or low frequency sounds, are also used to effect explosions, surf, wind, earthquakes, as well as percussion instruments.

As shown in Fig. 3, output from various sources are mixed into one composite signal as they move down to their next modifiers, the VCF, (voltage controlled filter), and then to the VCA (voltage controlled amplifier).

Envelope generators monitor these latter two devices. They create a slowly varying dc voltage that controls the cut-off frequency of the filter and the attenuation (volume) of the amplifier.

When the musician hits a key, three things happen. A dc voltage is produced to feed the oscillators; a gate pulse is created for as long as the key is depressed (typically zero volts when no key is held down, 10 volts when one is); and a trigger pulse, or short spike of voltage, is produced when the key is initially depressed. The gate and trigger pulses are used to tell the envelope generators to start their pre-programmed voltage patterns.

Take a look at the graphic representation

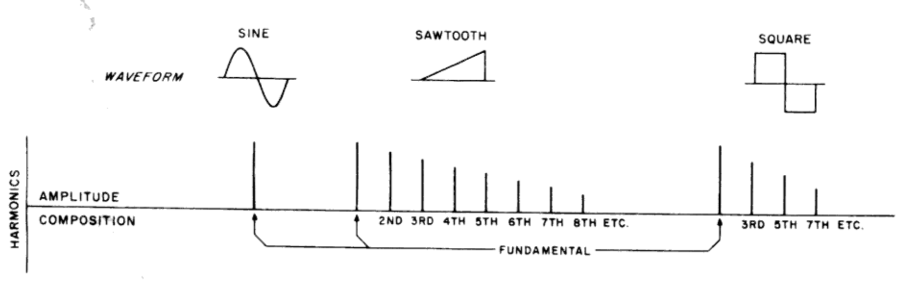


Fig. 1. Waveforms and Their Composition

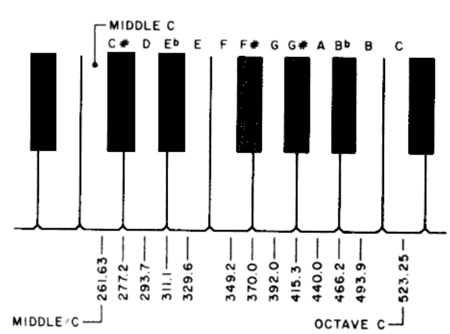


Fig. 2. The Frequencies of the 12-Note Octave Standard

"An envelope with a fast attack time and slow release time would be used to control the amplifier."

of the sound of a piano note being struck (Fig. 4). First, the sound is loud and then gradually gets softer. Here, an envelope with a fast attack time and slow release time would be used to control the amplifier. Also, as the sound diminishes, the higher harmonics are lost. Therefore, another generator is employed to control the filter.

From here the signal goes to the amplifier, speakers and then to you. Of course, you could alter the signal in other monstrous ways: ring modulation, reverb, echo, phase shifting, waveform inverting, phase locked loops. . . HELP!

Now that you know a little about the creation and modification of sound you are in a better position to choose the computer synthesizer that will best meet your expectations.■

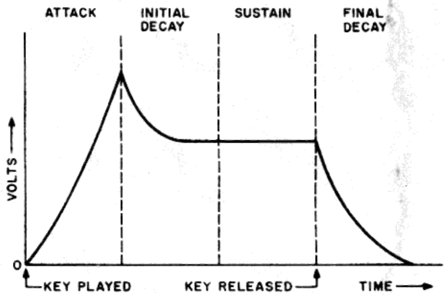


Fig. 4. Envelope Generator Parameters: On a typical keyboard synthesizer, voltage is developed by the envelope generator when a key is played. Voltage rises to a level preset by a potentiometer, then falls to a level determined by another potentiometer and stays there until the key is released. Upon release, the voltage falls to zero at a rate determined by a fourth control.

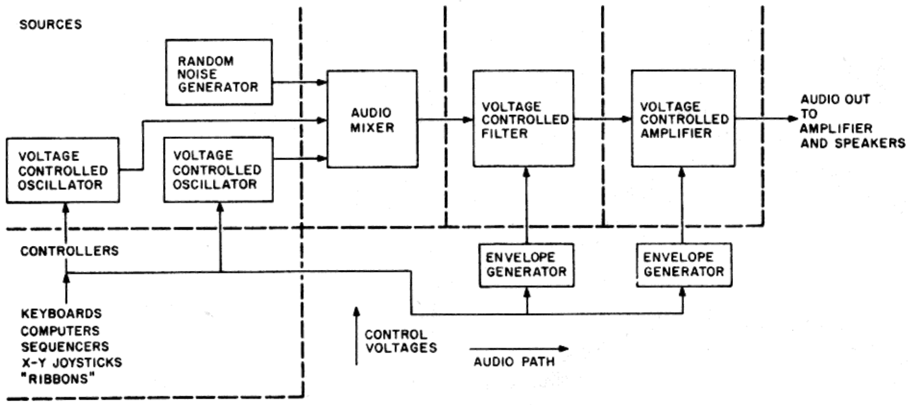


Fig. 3. Signal Flow and Control Layout of Performance Synthesizer

✓See List of Advertisers on page 306

Some sneaky ways to get more from your 16-bit registers.

Undocumented Instructions

Brian Cameron
284 Albert St.
Waterloo, Ontario
Canada N2L 3T8

Have you ever needed an extra general purpose register? And wouldn't it be great to be able to split those, sometimes unused, 16-bit index registers in half and get four more eight-bit work registers? Well, it can be done—if you know how to access them.

The key to these new registers and their instructions, is to precede an H or L type instruction with a special hex code. A hex code of DD tells the CPU that you are going to use the IX registers and a code of FD tells it you're using the IY index register.

Let's look at an example of how this is done. Normally the LD A,H (hex 7C) will move the contents of the H register into the A register. But if a hex DD is put in front of the load instruction, then the contents of the high-order byte of the IX index register are moved into the A register:

```
DEFB 0DDH ;gives key to IX register
LD    A,H   ;moves high byte of IX into A
```

If a hex FD is used in place of the DD, then the high-order byte of the IY register would be moved into the A register. In order to reference the low-order byte of the index registers, use instructions that normally refer to the L register. In an example the load instruction would read, LD A,L.

You can load into (and from) both index registers as well as the general registers A, B, C, D, and E. The HL register pair cannot be used since they now refer to the index registers. If the following instructions were executed—DEFB 0FDH LD H,L—they would result in the lower half of the IY register moving into the upper part of the same register.

The load immediate instructions also works on these registers. For example, DEFB 0DDH LD H,2 will load the number two into the high order byte of the IX register. Add, and add with carry, and subtract, and subtract with borrow instructions can be performed between the high and low bytes of the index registers and the data in the accumulator. It is also possible to increment, decrement, compare, AND, OR, and EXOR between the accumulator and index registers.

Another instruction on the general purpose registers, including the H and L registers, will take the contents of the register, multiply by two and add one ($r = r \cdot r + 1$). I called this DUPINC in the table. This requires two bytes. The first contains the hexadecimal value CB. The second contains the register the operation is carried out on.

In Table 1 HX and LX are used as names for the high-order and low-order bytes of the IX register. I use HY and LY for the IY register. The lowercase n, used in the immediate type instructions, is the symbol for a number.

By using these undocumented instructions, not only do you free up extra registers for general work, but a certain amount of protection is added to the security of your programs. Most disassemblers and monitors do not support these extra instructions and will display what appears as data areas right in the middle of your programs.■

*Editor's Note: These opcodes are not tested during production of the Z80 chip by Zilog. It is possible, in a small number of cases, that any of these opcodes may not work.

Table 1.

MNEMONIC		OP CODE
LD	HX,A	DD67
LD	HX,B	DD60
LD	HX,C	DD61
LD	HX,D	DD62
LD	HX,E	DD63
LD	LX,A	DD6F
LD	LX,B	DD68
LD	LX,C	DD69
LD	LX,D	DD6A
LD	LX,E	DD6B
LD	HY,A	FD67
LD	HY,B	FD60
LD	HY,C	FD61
LD	HY,D	FD62
LD	HY,E	FD63
LD	LY,A	FD6F
LD	LY,B	FD68
LD	LY,C	FD69
LD	LY,D	FD6A
LD	LY,E	FD6B
LD	HX,n	DD26
LD	LX,n	DD2E
LD	HY,n	FD26
LD	LY,n	FD2E
ADD	A,HX	DD84

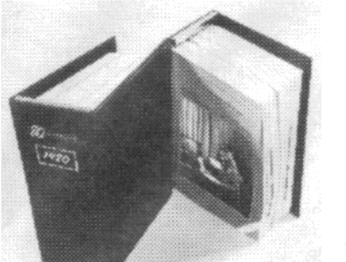
Table continues

ADD	A,LX	DD85
ADD	A,HY	FD84
ADD	A,LY	FD85
SUB	A,HX	DD94
SUB	A,LX	DD95
SUB	A,HY	FD94
SUB	A,LY	FD95
INC	HX	DD24
INC	LX	DD2C
INC	HY	FD24
INC	LY	FD2C
AND	HX	DDA4
AND	LX	DDA5
AND	HY	FDA4
AND	LY	FDA5
XOR	HX	DDAC
XOR	LX	DDAD
XOR	HY	FDAC
XOR	LY	FDAD
DUPINC	A	CB37
DUPINC	B	CB3Ø
DUPINC	C	CB31
DUPINC	D	CB32
DUPINC	E	CB33
DUPINC	H	CB34
DUPINC	L	CB35
DUPINC	(HL)	CB36
LD	A,HX	DD7C
LD	B,HX	DD44
LD	C,HX	DD4C
LD	D,HX	DD54
LD	E,HX	DD5C
LD	A,LX	DD7D
LD	B,LX	DD45
LD	C,LX	DD4D
LD	D,LX	DD55
LD	E,LX	DD5D
LD	A,HY	FD7C
LD	B,HY	FD44
LD	C,HY	FD4C
LD	D,HY	FD54
LD	E,HY	FD5C
LD	A,LY	FD7D
LD	B,LY	FD45
LD	C,LY	FD4D
LD	D,LY	FD55
LD	E,LY	FD5D
LD	HX,LX	DD65
LD	LX,HX	DD6C
LD	HY,LY	FD65
LD	LY,HY	FD6C
ADC	A,HX	DD8C
ADC	A,LX	DD8D
ADC	A,HY	FD8C
ADC	A,LY	FD8D
SBC	A,HX	DD9C
SBC	A,LX	DD9D
SBC	A,HY	FD9C
SBC	A,LY	FD9D
DEC	HX	DD25
DEC	LX	DD2D
DEC	HY	FD25
DEC	LY	FD2D
OR	HX	DDB4
OR	LX	DDB5
OR	HY	FDB4
OR	LY	FDB5
CP	HX	DDBC
CP	LX	DDBD
CP	HY	FDBC
CP	LY	FDBD

✓See List of Advertisers on page 306

80 microcomputing™

BINDERS



order yours today

Keep your library of 80 Microcomputing safe from loss or damage in these handsomely appointed binders with rich dark green covers and gold lettering. Each binder holds 12 issues making an EXCELLENT REFERENCE HANDBOOK. Several binders form a quality library you can be proud of.

\$7.50 each . . . 3 for \$21.75 . . . 6 for \$42.00 . . .

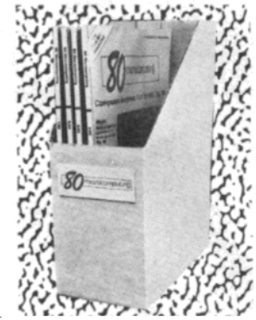
Postage paid in USA. Foreign orders please include \$2.50 for postage.

Specify 1980 or 1981

Send check or money order only to:
80 MICROCOMPUTING BINDERS
P.O. Box 5120, Phila., PA 19141

Allow 6-8 weeks for delivery. Please no C.O.D. orders. no phone orders.

is HARD COPY STORAGE a problem?



Here's the ideal way to keep your growing collection of *80 Microcomputing* in order! The *80 Microcomputing* Library Shelf Boxes. . . . sturdy, corrugated, white, dirt-resistant boxes will keep your issues of *80 Microcomputing* orderly and available for constant reference. Self-sticking labels are available for the boxes, too, not only for *80* but also for *Kilobaud/Microcomputing, 73 Magazine, CQ, QST, Ham Radio, Personal Computing, Interface Age, Byte* and *Radio Electronics*. Ask for whichever labels you want with your box order. Each box holds a full year of the above magazines. Your magazine library is your prime reference: Keep it handy and keep it neat with these strong library shelf boxes. One box (BX1000) is \$2.00, 2–7 boxes (BX1001) are \$1.50 each, and 8 or more boxes (BX1002) are \$1.25 each. Be sure to specify which labels we should send. Shipping and handling charges are \$1.00 per order. Call in your credit card orders on our toll free line 800-258-5473, or use the order card in the back of the magazine and mail to:

80 microcomputing™ Peterborough, NH 03458

Please allow 4–6 weeks for delivery. No COD orders accepted.

While they last . . .

the complete year of 1980 80 Microcomputing (in its own shelf box . . . a \$2.00 value—Free) for only \$25.00.*

Did you miss the 2,256 pages of useful information published for your TRS-80** in *80 Microcomputing* last year? Now is the perfect time to catch up for only \$25.00.*

In the first year of *80 Microcomputing* here were 335 articles on your TRS-80—2½ times more than any other computing magazine. Also in 1980 there were new product reviews, news and columns and best of all hundreds of dollars worth of useable programs.

So to unlock the key to your TRS-80 and double its value. . . send in today for the complete year of 1980. *80 Microcomputing* (in its own shelf box) for only \$25.00.*

We have a limited number of these complete sets so send in the card today or call our toll free #800-258-5473 and charge it to your VISA, MC or AE card.

**plus shipping and handling (\$3.00) **TRS-80 is a Trademark of Tandy*

Yes, I want a year of *80 Microcomputing* for \$25.00 plus \$3.00 for shipping and handling. Plus I'll receive a shelf box to store them in.

☐ Check enclosed for \$________________

Name ________________________________

Address ______________________________

City ____________________ State ________ Zip ________

Please allow 4 to 6 weeks for delivery

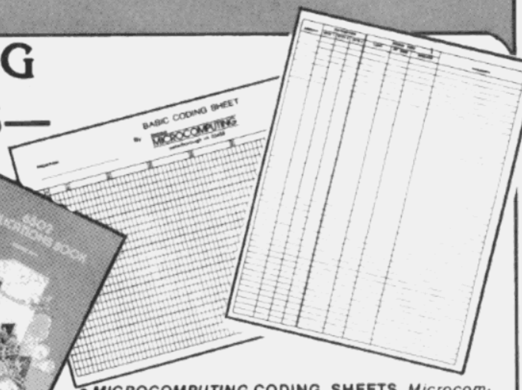
80 Microcomputing • 80 Pine Street • Peterborough, N.H. 03458

Toll free ordering number 800-258-5473

80BI

80 microcomputing™ bookshelf

—PROGRAMMING COOKBOOKS—



—Z-80—

● **INSIDE LEVEL II**—BK1183—For machine language programmers! This is a comprehensive reference guide to the Level II ROMs, allowing easy utilization of the sophisticated routines they contain. It concisely explains set-ups, calling sequences, variable passage and I/O routines. Part II presents an entirely new composite program structure which unloads under the SYSTEM command and executes in both BASIC and machine code with the speed and efficiency of a compiler. Special consideration is given to disk systems. $15.95.*

● **PROGRAMMING THE Z-80**—BK1122—by Rodnay Zaks. Here is assembly language programming for the Z-80 presented as a progressive, step-by-step course. This book is both an educational text *and* a self-contained reference book, useful to both the beginning and the experienced programmer who wish to learn about the Z-80. Exercises to test the reader are included. $14.95.*

● **Z-80 SOFTWARE GOURMET GUIDE AND COOKBOOK**—BK1045—by Nat Wadsworth. Scelbi's newest cookbook! This book contains a complete description of the powerful Z-80 instruction set and a wide variety of programming information. Use the author's ingredients including routines, subroutines and short programs, choose a time-tested recipe and start cooking! $16.99.*

● **Z-80 ASSEMBLY LANGUAGE PROGRAMMING**—BK1177—by Lance A. Leventhal. This book thoroughly covers the Z-80 instruction set, abounding in simple programming examples which illustrate software development concepts and actual assembly language usage. Features include Z-80 I/O devices and interfacing methods, assembler conventions, and comparisons with 8080A/8085 instruction sets and interrupt structure. $16.99.*

TEST EQUIPMENT —LIBRARY—

● **VOL. I COMPONENT TESTERS**—LB7359— . . how to build transistor testers (8), diode testers (3), IC testers (3), voltmeters and VTVMs (9), ohmmeters (8 different kinds), inductance (3), capacity (9), Q measurement, crystal checking (6), temperature (2), aural meters for the blind (3) and all sorts of miscellaneous data on meters . . . using them, making them more versatile, making standards. Invaluable book. $4.95.*

● **VOL. II AUDIO FREQUENCY TESTERS**—LB7360— . . . jam packed with all kinds of audio frequency test equipment. If you're into SSB, RTTY, SSTV, etc., this book is a must for you . . . a good book for hi-fi addicts and experimenters, too! $4.95.*

● **VOL. III RADIO FREQUENCY TESTERS**—LB7361—Radio frequency waves, the common denominator of Amateur Radio. Such items as SWR, antenna impedance, line impedance, rf output and field strength; detailed instructions on testing these items includes sections on signal generators, crystal calibrators, grid dip oscillators, noise generators, dummy loads and much more. $4.95.*

● **VOL. IV IC TEST EQUIPMENT**—LB7362—Become a troubleshooting wizard! In this fourth volume of the 73 TEST EQUIPMENT LIBRARY are 42 home construction projects for building test equipment to work with your ham station and in servicing digital equipment. Plus a cumulative index for all four volumes for the 73 TEST EQUIPMENT LIBRARY. $4.95.*

A NEW PROGRAMMING —SERIES—

The Microprocessor Software Engineering Series by John Zarrella provides common sense descriptions of advanced computer system topics for engineers, programmers and development managers. Each volume is a self-contained review of a software engineering topic, explaining fundamental concepts in easy-to-understand language and describing sophisticated software tools and techniques. Detailed glossary of technical jargon is included in each volume. This series will help you find the solutions to your software problems.

● **OPERATING SYSTEMS: CONCEPTS AND PRINCIPLES**—BK1193—Presents an overview of the basic operating system types, their components and capabilities. $7.95.*

● **WORD PROCESSING AND TEXT EDITING**—BK1194—Provides a firm basis for understanding word processing terminology and for comparing systems. $7.95.*

● **SYSTEM ARCHITECTURE**—BK1195 —Presents a detailed overview of advanced computer system design including object architecture and capability-based addressing. $9.95.*

—6502—

● **PROGRAMMING THE 6502 (Third Edition)**—BK1005—Rodnay Zaks has designed a self-contained text to learn programming, using the 6502. It can be used by a person who has never programmed before, and should be of value to anyone using the 6502. The many exercises will allow you to test yourself and practice the concepts presented. $12.95.*

● **6502 APPLICATIONS BOOK**—BK1006—Rodnay Zaks presents practical-application techniques for the 6502 microprocessor, assuming an elementary knowledge of microprocessor programming. You will build and design your own domestic-use systems and peripherals. Self-test exercises included. $12.95.*

● **6502 ASSEMBLY LANGUAGE PROGRAMMING**—BK1176—by Lance A. Leventhal. This book provides comprehensive coverage of the 6502 microprocessor assembly language. Leventhal covers over 80 programming examples from simple memory load loops to complete design projects. Features include 6502 assembler conventions, input/output devices and interfacing methods, and programming the 6502 interrupt system. $16.99.*

● **6502 SOFTWARE GOURMET GUIDE AND COOKBOOK**—BK1055—by Robert Findley. This book introduces the BASIC language programmer into the realm of machine-language programming. The description of the 6502 structure and instruction set, various routines, subroutines and programs are the ingredients in this cookbook. "Recipes" are included to help you put together exactly the programs to suit your taste. $12.95.*

● ***MICROCOMPUTING* CODING SHEETS** *Microcomputing*'s dozen or so programmers wouldn't try to work without these handy scratch pads, which help prevent the little errors that can cost hours and hours of programming time. Available for programming is Assembly/Machine Language (PD1001), which has columns for address, instruction (3 bytes), source code (label, op code, operand) and comments; and for BASIC (PD1002) which is 72 columns wide. 50 sheets to a pad. $2.39.*

—8080/8080A—6800—

● **8080A/8085 ASSEMBLY LANGUAGE PROGRAMMING**—by Lance Leventhal—BK1004—Assembly language programming for the 8080A/8085 is explained with a description of the functions of assemblers and assembly instructions, and a discussion of basic software development concepts. Many fully debugged, practical programs are included as is a special section on structured programming. $15.99.*

● **8080 PROGRAMMING FOR LOGIC DESIGN**—BK1078—Ideal reference for an in-depth understanding of the 8080 processor. Application-oriented and the 8080 is discussed in light of replacing conventional, hard-wired logic. Practical design considerations are provided for the implementation of an 8080-based control system. $9.50.*

● **8080 SOFTWARE GOURMET GUIDE AND COOKBOOK**—BK1102—If you have been spending too much time developing simple routines for your 8080, try this new book by Scelbi Computing and Robert Findley. Describes sorting, searching, and many other routines for the 8080 user. $12.95.*

● **6800 PROGRAMMING FOR LOGIC DESIGN**—BK1077—Oriented toward the industrial user, this book describes the process by which conventional logic can be replaced by a 6800 microprocessor. Provides practical information that allows an experimenter to design a complete micro control system from the "ground up." $9.50.*

● **6800 SOFTWARE GOURMET GUIDE AND COOKBOOK**—BK1075—Like its culinary cousin, *The 8080 Gourmet Guide*, this book by Scelbi Computing and Robert Findley describes sorting, searching and other routines—this time for the 6800 user. $12.95.*

—COOKBOOKS—

● **CMOS COOKBOOK**—BK1011—by Don Lancaster. Details the application of CMOS, the low power logic family suitable for most applications presently dominated by TTL. Required reading for every serious digital experimenter! $10.50.*

● **TVT COOKBOOK**—BK1064—by Don Lancaster. Describes the use of a standard television receiver as a microprocessor CRT terminal. Explains and describes character generation, cursor control and interface information in typical, easy-to-understand Lancaster style. $9.95.*

● **TTL COOKBOOK**—BK1063—by Don Lancaster. Explains what TTL is, how it works, and how to use it. Discusses practical applications, such as a digital counter and display system, events counter, electronic stopwatch, digital voltmeter and a digital tachometer. $9.50.*

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to 80 Microcomputing Bookshelf ● Peterborough NH 03458. Be sure to include check or detailed credit card information.

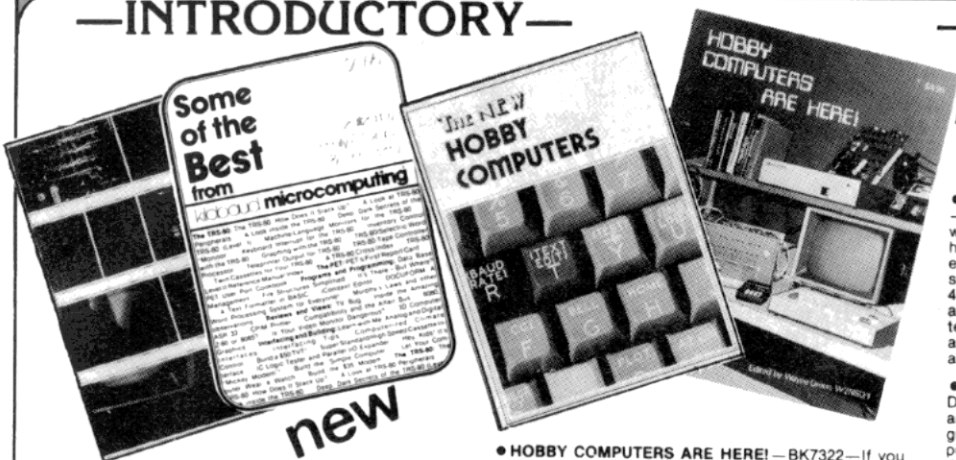
No C.O.D. orders accepted. All above add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write Customer Service at the above address.

FOR TOLL FREE ORDERING CALL 1-800-258-5473

PRICES SUBJECT TO CHANGE WITHOUT NOTICE

80 microcomputing bookshelf

—INTRODUCTORY—



new

● **YOUR FIRST COMPUTER**—BK1191—by Rodnay Zaks. Whether you are using a computer, thinking about using one or considering purchasing one, this book is indispensable. It explains what a computer system is, what it can do, how it works and how to select various components and peripheral units. It is written in everyday language and contains invaluable information for the novice and the experienced programmer. (The first edition of this book was published under the title "An Introduction to Personal and Business Computing".) $7.95.*

● **SOME OF THE BEST FROM KILOBAUD MICROCOMPUTING**—BK7311—A collection of the best articles that have appeared in Kilobaud MICROCOMPUTING. Included is material on the TRS-80 and PET systems, CP/M, the 8080/8085/Z-80 chips, the ASR-33 terminal. Data base management, word processing, text editors and file structures are covered too. Programming techniques and hard-core hardware construction projects for modems, high speed cassette interfaces and TVTs are also included in this large format, 200 plus page edition. $10.95.*

● **HOBBY COMPUTERS ARE HERE!**—BK7322—If you want to come up to speed on how computers work . . . hardware and software . . . this is an excellent book. It starts with fundamentals and explains the circuits, and the basics of programming, along with a couple of TVT construction projects, ASCII-Baudot, etc. This book has the highest recommendations as a teaching aid. $4.95.*

● **UNDERSTANDING AND PROGRAMMING MICROCOMPUTERS**—BK7382—A valuable addition to your computing library. This two-part text includes the best articles that have appeared in 73 and Kilobaud Microcomputing magazines on the hardware and software aspects of microcomputing. Well-known authors and well-structured text helps the reader get involved. $10.95.*

● **THE NEW HOBBY COMPUTERS**—BK7340—This book takes it from where "HOBBY COMPUTERS ARE HERE!" leaves off, with chapters on Large Scale Integration, how to choose a microprocessor chip, an introduction to programming, low cost I/O for a computer, computer arithmetic, checking memory boards . . . and much, much more! Don't miss this tremendous value! Only $4.95.*

— INTRODUCTION — TO MICROCOMPUTERS (VOL. 0–III)

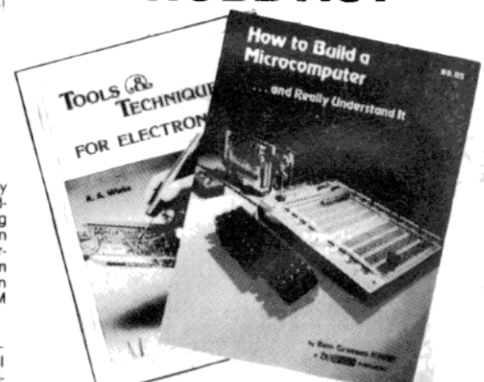
● **AN INTRODUCTION TO MICROCOMPUTERS, VOL. 0**—BK1130—The Beginner's Book—Written for readers who know nothing about computers—for those who have an interest in how to use computers—and for everyone else who must live with computers and should know a little about them. The first in a series of 4 volumes, this book will explain how computers work and what they can do. Computers have become an integral part of life and society. During any given day you are affected by computers, so start learning more about them with Volume 0. $7.95.*

● **VOL. I**—BK1030—**2nd Edition completely revised.** Dedicated to the basic concepts of microcomputers and hardware theory. The purpose of Volume I is to give you a thorough understanding of what microcomputers are. From basic concepts (which are covered in detail), Volume I builds the necessary components of a microcomputer system. This book highlights the difference between minicomputers and microcomputers. $12.99.*

● **VOL. II**—BK1040 (with binder)—Contains descriptions of individual microprocessors and support devices used only with the parent microprocessor. Volume II describes all available chips. $31.99*

● **VOL. III**—BK1133 (with binder)—Contains descriptions of all support devices that can be used with any microprocessor. $21.99*

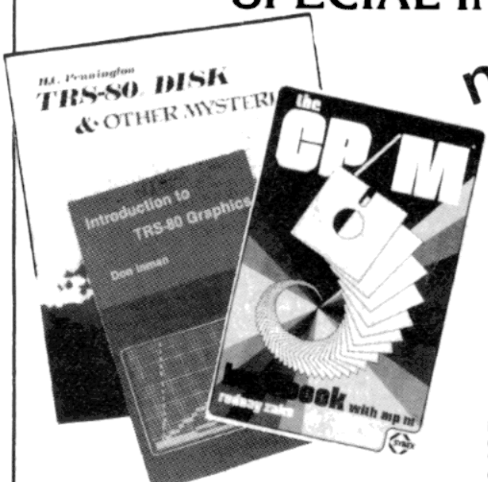
FOR THE ELECTRONICS — HOBBYIST —



● **TOOLS & TECHNIQUES FOR ELECTRONICS**—BK7348—by A. A. Wicks is an easy-to-understand book written for the beginning kit builder as well as the experienced hobbyist. It has numerous pictures and descriptions of the safe and correct ways to use basic and specialized tools for electronic projects as well as specialized metal working tools and the chemical aids which are used in repair shops. $4.95.*

● **HOW TO BUILD A MICROCOMPUTER—AND REALLY UNDERSTAND IT**—BK7325—by Sam Creason. The electronics hobbyist who wants to build his own microcomputer system now has a practical "How-To" guidebook. This book is a combination technical manual and programming guide that takes the hobbyist step-by-step through the design, construction, testing and debugging of a complete microcomputer system. Must reading for anyone desiring a true understanding of small computer systems. $9.95.*

—SPECIAL INTERESTS—



new

● **MICROSOFT BASIC DECODED AND OTHER MYSTERIES**—BK1186—by James Farvour. From the company that brought you *TRS-80 DISK AND OTHER MYSTERIES*! Contains more than 6500 lines of comments for the disassembled Level II ROMs, six additional chapters describing every BASIC subroutine, with assembly language routines showing how to use them. Flowcharts for all major routines give the reader a real insight into how the interpreter works. $29.50.

● **THE CP/M HANDBOOK (with MP/M)**—BK1187—by Rodnay Zaks. A complete guide and reference handbook for CP/M—the industry standard in operating systems. Step-by-step instruction for everything from turning on the system and inserting the diskette to correct user discipline and remedial action for problem situations. This also includes a complete discussion of all versions of CP/M up to and including 2.2, MP/M and CDOS. $14.95*.

● **INTRODUCTION TO TRS-80 GRAPHICS**—BK1180—by Don Inman. Dissatisfied with your Level I or Level II manual's coverage of graphics capabilities? This well-structured book (suitable for classroom use) is ideal for those who want to use all the graphics capabilities built into the TRS-80. A tutorial method is used with many demonstrations. It is based on the Level I, but all material is suitable for Level II use. $8.95.*

● **TRS-80 DISK AND OTHER MYSTERIES**—BK1181—by Harvard C. Pennington. This is the definitive work on the TRS-80 disk system. It is full of detailed "How to" information with examples, samples and in-depth explanations suitable for beginners and professionals alike. The recovery of one lost file is worth the price alone. $22.50.*

● **MICROPROCESSOR INTERFACING TECHNIQUES**—BK1037—by Austin Lesea & Rodnay Zaks—will teach you how to interconnect a complete system and interface it to all the usual peripherals. It covers hardware and software skills and techniques, including the use and design of model buses such as the IEEE 488 or S-100. $15.95.*

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to 80 Microcomputing Bookshelf • Peterborough NH 03458. Be sure to include check or detailed credit card information.

No C.O.D. orders accepted. All above add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write Customer Service at the above address.

FOR TOLL FREE ORDERING CALL 1-800-258-5473

PRICES SUBJECT TO CHANGE WITHOUT NOTICE

Why use their flexible discs:

Ampex, Athana, BASF, Caelus, Control Data, Dysan, IBM, Inmac, K-Line, Maxell, Nashua, Scotch, Shugart, Syncom, 3M, Verbatim or Wabash

when you could be using

MEMOREX

for as low as $1.99 each?

Find the flexible disc you're now using on our cross reference list... then write down the equivalent Memorex part number you should be ordering.

Product Family	Product Description	Memorex Part Number (3201-)	CE quant. 100 price per disc ($)	Ampex	Athana	BASF	Caelus	Dysan	IBM	K-Line	Maxell	Nashua	Scotch 3M	Shugart	Syncom	Verbatim	Wabash	Inmac	Control Data
	IBM Compatible (128 B/S, 26 sectors)	3060	2.19	5FD-111110	473071	53428	CM-F11	800506	2305830	40013	FD1-128	FD-1	740-0	S/A 100	15002	FD34-1000	F111111X	7870-K	421602
Flexible Disc 1s	IBM Compatible (128 B/S, 26 sectors) w/ W.P.N.	3062	2.24	–	–	–	–	–	–	–	–	–	740-0	–	–	–	–	–	–
	IBM Compatible (128 B/S, 26 sectors) w/ W.P.N. & Hub ring	3064	2.55	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Single-Headed Drives	IBM Compatible (128 B/S, 26 sectors) REVERSIBLE	1729	3.35	5FD-113110	473072	54431	–	–	–	40015	–	FD-2	740/2-0	–	15150	FF34-2000	F171111X	7860-K	–
Single-Density Media	IBM System 6 Compatible	3066	2.19	–	473077	54561	–	800509	1669959	40014	–	–	740-0 056	–	15003	FD60-1000	F116111X	–	–
	IBM Compatible (256 B/S, 15 sectors)	3109	2.19	5FD-111210	473073	–	–	800584	2305845	40040	–	–	740-3600	–	15005	FD36-1000	F112111X	7861-K	–
	IBM Compatible (512 B/S, 8 sectors)	3110	2.19	–	473074	–	–	800585	1669954	40044	–	–	–	–	15004	FD60-1000	F113111X	7889-K	–
	Shugart Compatible, 32 hard sector	3015	2.19	5FD-211010	470901	53802	CM-F21	101/1	–	40016	FH1-32	FD-132	740-32	S/A-101	15025	FD32-1000	–	7890-K	421322
	Shugart Compatible, 32 hard sector REVERSIBLE	3025	3.35	5FD-213010	–	–	–	–	–	40017	–	–	740/2-32	–	15151	FF32-2000	–	7880-K	–
	Wang Compatible, 32 hard sector w/Hub ring	3087	2.50	–	–	54491	–	–	–	–	–	–	740-32RM	–	–	–	F37A411X	–	–
	CPT 8000 Compatible	3045	2.79	–	–	–	–	–	–	–	–	–	–	–	15226	–	–	–	–
Flexible Disc 1d	IBM Compatible (128 B/S, 26 sectors)	3090	2.95	5FD-121010	474071	54568	–	3740/1D	–	40047	FD1-128/M2100	FD-1D	741-0	–	–	FD34-8000	F131111X	7857-K	423002
	Soft Sector (128 B/S, 26 sectors) REVERSIBLE	3093	3.99	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Single-Headed Drives	Shugart Compatible, 32 hard sector	3091	2.95	5FD-221010	470801	54596	–	101/1D	–	40024	FH1-32D	–	741-32	S/A-103	15075	FD32-8000	F33A410X	7887-K	423322
Double-Density Media	Shugart Compatible, 32 hard sector REVERSIBLE	3094	3.99	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	Wang Compatible, 32 hard sector w/Hub ring	3088	3.20	–	–	–	–	–	–	–	–	–	–	–	–	–	F22A411X	–	–
Flexible Disc 2s	Soft Sector (Unformatted)	3101	3.84	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	Soft Sector (128 B/S, 26 sectors)	3113	3.84	–	–	54428	–	800814	1766870	–	–	–	–	S/A-150	15153	FD10-4026	F121111X	–	–
Double-Headed Drives	Soft Sector (256 B/S, 15 sectors)	3106	3.84	–	473477	54226	–	800815	2736700	40043	FD2-256D	–	742-0	–	15154	FD10-4015	F122111X	7856-K	424612
Single-Density Media	32 Hard Sector	3108	3.84	–	–	–	–	–	–	–	FH2-32	–	–	–	–	–	–	–	–
Flexible Disc 2d	Soft Sector (Unformatted)	3102	3.49	–	473485	–	–	DY150	–	40028	FD2-XDM	FD-2D	743-0	–	15103	DD34-4001	–	–	425002
	Soft Sector (128 B/S, 26 sectors)	3115	3.49	–	–	–	–	–	–	–	–	–	–	S/A-150	–	–	–	–	–
Double-Headed Drives	Soft Sector (256 B/S, 26 sectors)	3103	3.49	–	473471	54325	–	800817	1766872	40019	FD2-256D	–	743-0/256	–	15101	DD34-4026	F144111X	7858-K	425602
Double-Density Media	Soft Sector (512 B/S, 15 sectors)	3114	3.49	–	473472	54479	–	800818	1669044	40039	–	–	743-0/512	–	15100	DD34-4015	F145111X	–	425612
	Soft Sector (1024 B/S, 8 sectors)	3104	3.49	–	473473	54485	–	800819	1669045	40020	–	–	743-0/1024	–	15102	DD34-4008	F147111X	7859-K	425622
	32 Hard Sector	3105	3.49	5FD-321010	470851	–	–	101/2D	–	40021	FH2-32D	–	743-32	S/A-151	15125	DD32-4000	F34A411X	7881-K	425322
	Burroughs B-80 Compatible 32 Hard Sector	3092	3.49	–	–	–	–	–	–	–	–	–	–	–	–	–	F34A611X	–	–
	Soft Sector (1024 B/S, 8 sectors) w/ Hub Ring	3116	3.75	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Flexible Disc FD Memorex 651 or Equiv. Drive Compatible	FD VI (Vinyl Jacket)	30712003	2.95	–	470651	–	CM-F31	FDIV	–	40002	–	FD-165	511-0	–	15026	FD65-1000	F61A111X	7910	–
Mini Flexible Disc 1s	Soft Sector (Unformatted)	3401	1.99	–	475001	54256	–	104/1	–	40500	MD1	MD 1	744-0	S/A-104	15300	MD525-01	M11A211X	7897	441002
5¼" Single-Headed Drives	10 Hard Sector	3403	1.99	–	475010	54257	–	107/1	–	40501	–	MD 110	744-10	S/A-107	15325	MD525-10	M41A211X	7898	441102
Single-Density Media	16 Hard Sector	3405	1.99	–	475016	54258	–	105/1	–	40502	MH1	MD 116	744-16	S/A-105	15326	MD525-16	M51A211X	7899	441162
	Soft Sector (Unformatted) W/Hub Ring	3431	2.19	–	–	–	–	–	–	–	–	–	–	–	–	MD525-01	–	–	–
	10 Hard Sector, w/Hub Ring	3433	2.19	–	–	–	–	–	–	–	–	–	–	–	–	MD525-10	–	–	–
	16 Hard Sector, w/Hub Ring	3435	2.19	–	–	–	–	–	–	–	–	–	–	–	–	MD525-16	–	–	–
Mini Flexible Disc 1d	Soft Sector	3417	2.24	–	–	54646	–	104/1D	–	–	–	–	–	–	–	MD540-01	–	–	–
5¼" Single-Headed Drives	10 Hard Sector	3418	2.24	–	–	54649	–	107/1D	–	–	–	–	–	–	–	MD540-10	–	–	–
Double-Density Media	16 Hard Sector	3419	2.24	–	–	54652	–	105/1D	–	–	–	–	–	–	–	MD540-16	–	–	–
Mini Flexible Disc 2d	Soft Sector	3421	2.74	–	–	54624	–	104/2D	–	–	MD2-D	–	745-0	S/A-154	–	MD550-01	–	–	–
5¼" Double-Headed Drives	10 Hard Sector	3423	2.74	–	–	54627	–	107/2D	–	–	–	–	745-10	S/A-157	–	MD550-10	–	–	–
Double Density Media	16 Hard Sector	3425	2.74	–	–	54630	–	105/2D	–	–	MH2-D	–	745-16	S/A-155	–	MD550-16	–	–	–

Memorex Flexible Discs...The Ultimate in Memory Excellence

Quality
Memorex means quality products that you can depend on. Quality control at Memorex means starting with the best materials available. Continual surveillance throughout the entire manufacturing process. The benefit of Memorex's years of experience in magnetic media production, resulting, for instance, in proprietary coating formulations. The most sophisticated testing procedures you'll find anywhere in the business.

100 Percent Error Free
Each and every Memorex Flexible Disc is certified to be 100 percent error free. *Each* track of *each* flexible disc is tested, individually, to Memorex's stringent standards of excellence. They test signal amplitude, resolution, low-pass modulation, overwrite, missing pulse error and extra pulse error. They are torque-tested, and competitively tested on drives available from almost every major drive manufacturer in the industry including drives that Memorex manufactures. Rigid quality audits are built into every step of the manufacturing process and stringent testing result in a standard of excellence that assures you, our customer, of a quality product designed for increased data reliability and consistent top performance.

Customer-Oriented Packaging
Memorex's commitment to excellence does not stop with a quality product. They are proud of their flexible discs and they package them with pride. Both their packaging and their labeling have been designed with your ease of identification and use in mind. The desk-top box containing ten discs is convenient for filing and storage. Both box labels and jacket labels provide full information on compatibility, density, sectoring, and record length. Envelopes with multi-language care and handling instructions and color-coded removable labels are included. A write-protect feature is available to provide data security.

Full One Year Warranty — Your Assurance of Quality
Memorex Flexible Discs will be replaced free of charge by Memorex if they are found to be defective in materials or workmanship within one year of the date of purchase. Other than replacement, Memorex will not be responsible for any damages or losses (including consequential damages) caused by the use of Memorex Flexible Discs.

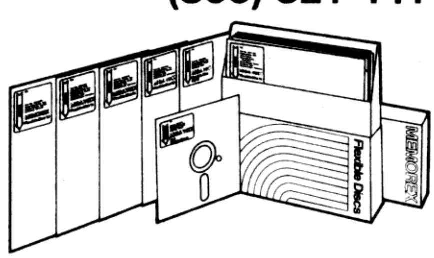
Quantity Discounts Available
Memorex Flexible Discs are packed 10 discs to a carton and 10 cartons to a case. Please order only in increments of 100 units for quantity 100 pricing. We are also willing to accommodate your smaller orders. Quantities less than 100 units are available in increments of 10 units at a 10% surcharge. **Quantity discounts** are also available. Order 500 or more discs at the same time and deduct 1%; 1,000 or more saves you 2%; 2,000 or more saves you 3%; 5,000 or more saves you 5%; 10,000 or more saves you 7%; 25,000 or more saves you 8%; 50,000 or more saves you 9% and 100,000 or more discs earns you a 10% discount off our super low quantity 100 price. Almost all Memorex Flexible Discs are immediately available from **CE**. Our warehouse facilities are equipped to help us get you the quality product you need, when you need it. If you need further assistance to find the flexible disc that's right for you, call the Memorex compatibility hotline. Dial 800-538-8080 and ask for the *flexible disc hotline* extension 0997. In California dial 800-672-3525 extension 0997.

Buy with Confidence
To get the fastest delivery from **CE** of your Memorex Flexible Discs, send or phone your order directly to our Computer Products Division. Be sure to calculate your price using the **CE** prices in this ad. Michigan residents please add 4% sales tax. Written purchase orders are accepted from approved government agencies and most well rated firms at a 10% surcharge for net 10 billing. All sales are subject to availability. All sales are final. Prices, terms and specifications are subject to change without notice. Out of stock items will be placed on backorder automatically unless **CE** is instructed differently. International orders are invited with a $20.00 surcharge for special handling in addition to shipping charges. All shipments are F.O.B. Ann Arbor, Michigan. No COD's please. Non-certified and foreign checks require bank clearance.

Mail orders to: **Communications Electronics**, Box 1002, Ann Arbor, Michigan 48106 U.S.A. Add $8.00 per case or partial-case of 100 8-inch discs or $6.00 per case of 100 5¼-inch mini-discs for U.P.S. *ground* shipping and handling in the continental U.S.A. If you have a Master Charge or Visa card, you may call anytime and place a credit card order. **Order toll-free** in the United States. Call anytime 800-521-4414. If you are outside the U.S. or in Michigan, dial 313-994-4444. **Dealer inquiries invited.** All order lines at **Communications Electronics** are staffed 24 hours.

Copyright ©1981 Communications Electronics™

Order Toll-Free!
(800) 521-4414



For Data Reliability—Memorex Flexible Discs

COMMUNICATIONS ELECTRONICS™

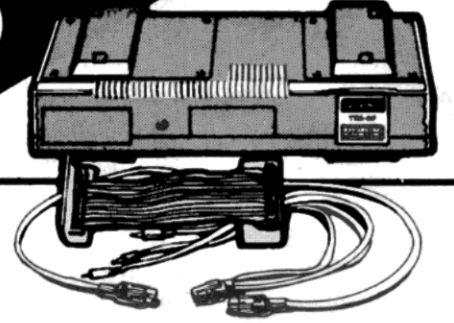
263

Computer Products Division

854 Phoenix □ Box 1002 □ Ann Arbor, Michigan 48106 U.S.A.
Call TOLL-FREE (800) 521-4414 or outside U.S.A. (313) 994-4444

✓See List of Advertisers on page 306

...PRODUCT ...SPECIALS..... for the TRS-80®



TRS-80® Expansion Interface

Factory sealed Model I expansion interfaces. Includes cables, TRS-80 interface card edge, expansion board card edge, line printer card edge, bus extension and floppy disk card edge.
Available with 16k or 32k RAM memory

Cat No. 3001 16k Expansion Interface — $375
Cat No. 3002 32k Expansion Interface — $400

LDOS from LOBO Drives

The most powerful disk operating system yet released for the TRS-80. Supports up to 8 disk drives in any combination. 5¼", 8", single or double sided. Even hard disk. Comes complete with extensive user documentation and one year free maintenance from LOBO.

Cat No. 3053 TRS-80, MOD I, 32k, Disk — $139.00

ATLANTEAN ODYSSEY
From Interpro

This is the new graphics version of the classic undersea adventure. Visit underwater cities and see strange painted shapes as you search for the lost city of Atlantis.

Cat No. 3088 TRS-80, MOD I, 48k, Disk — $29.95

STAR WARRIOR
From Automated Simulations

Be an interstellar avenger and remove the despots from power in this action packed adventure. Graphics and sound. Tough opponents provide a great challenge.

Cat. No. 3057 TRS-80, MOD I 32k, Cass. — $39.95
Cat No. 3058 TRS-80, MOD I, 32K, Disk + TRSDOS — $39.95

How To Order
Write or phone. Pay by check, M/C, Visa or COD's. (COD's add $1.40 addl.) Orders must specify this ad to qualify for prepaid shipping. Offer expires Aug. 1, 1981.

HW ELECTRONICS
(Formerly Hobbyworld Electronics)
19511 Business Center Dr., Dept. V7
Northridge, Ca. 91324
(800) 423-5387 (Toll free outside CALIF.)
(213) 886-9200 (in CALIF.)

****MENTION THIS AD AND WE PREPAY UPS GROUND SHIPPING AND HANDLING****

Send For Our FREE HW Catalogue

Fill in the coupon below to receive your free copy of our 44 page catalogue. Contains a wide selection of products for you TRS-80 computer.

Name ____________________
Company ____________________
Address ____________________
City ____________________
State ____________ **Zip** ________

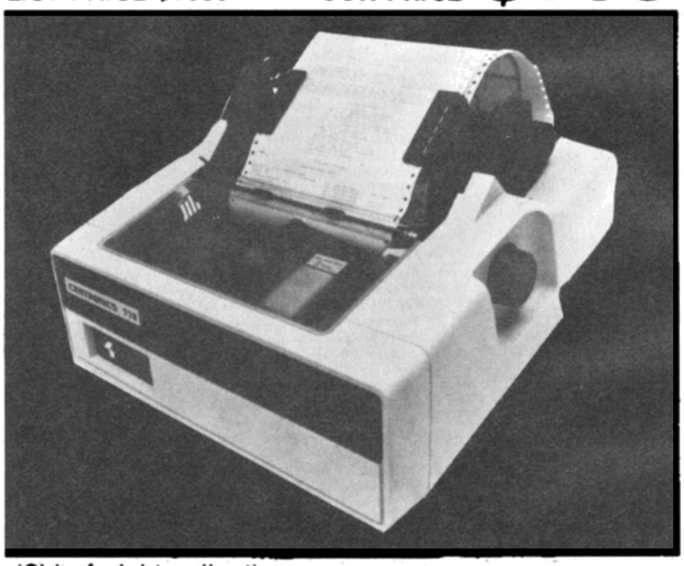
V7

Now save even more with factory rebates on Centronics® printers for the TRS-80®

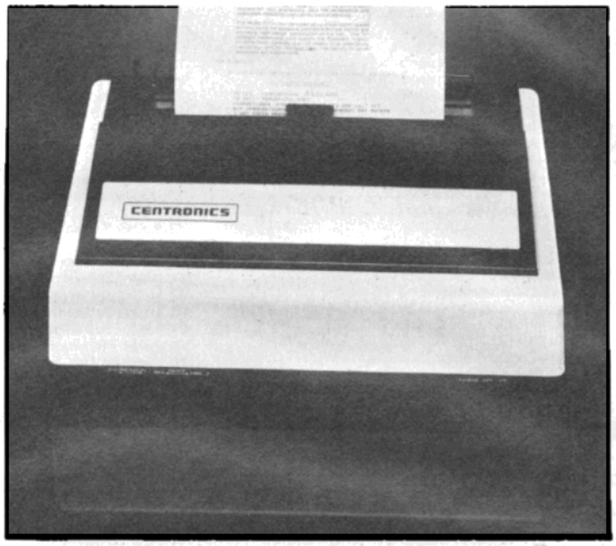
CENTRONICS 779

Same as Radio Shack Line Printer I

LIST PRICE $1350 ***OUR PRICE*** **$799**



(Ship freight collect)



CENTRONICS 737

Featuring Correspondence, Quality and Proportional Spacing *LIST $995*

OUR PRICE **$799.**

less rebate of **150.**

net cost: ***$649***

(add $7.50 for shipping)

Receive factory Rebates from Centronics with coupon from **MMM**

CENTRONICS 730

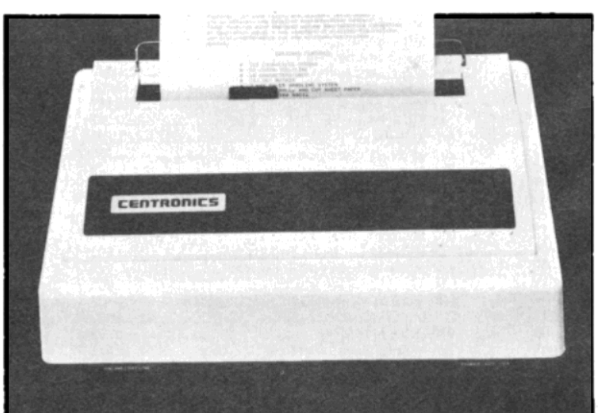
Same as Radio Shack Line Printer II

LIST $795
OUR PRICE **$599.**

less rebate of **100.**

net cost: ***$499***

(add $7.50 for shipping)



Disc Drive Values for TRS-80

Your Choice

5¼" Disk Drives

Shugart, MPI-51, or Tandon
w/case & Power Supply

$499 VALUE

MMM PRICE ONLY

$329
2 for $638

24

TM TRS-80 is a trademark of Radio Shack

Limited offer expires July 31st

Mini Micro Mart, Inc. 1618 James Street, Syracuse, NY 13203
(315) 422-4467 TWX 710-541-0431

LIST OF ADVERTISERS

For further information from our advertisers, please use the Reader Service card.

READER SERVICE

This card valid until September 30, 1981

Reader Service: To receive more information from any of the advertisers in this issue of *80 Microcomputing*, circle the number on the postage-paid Reader Service Card that corresponds with the Reader Service number on the ad in which you are interested. You will find numbers, preceded by a ✓, near the logo of each advertiser. Complete the entire card, drop into a mailbox and in 4-6 weeks you will hear from the advertiser directly.

Please help us to bring you a better magazine—by answering these questions.

I. What is your age?
- ☐ A. under 18
- ☐ B. 18-22
- ☐ C. 23-40
- ☐ D. 41-60
- ☐ E. over 60

II. What is your occupation?
- ☐ 1. Professional
- ☐ 2. Engineer
- ☐ 3. Data processing
- ☐ 4. Business
- ☐ 5. Education
- ☐ 6. Technician
- ☐ 7. Student
- ☐ 8. Other

III. What are your primary applications of your TRS-80 (check only two)?
- ☐ A. Business
- ☐ B. Games
- ☐ C. Home
- ☐ D. Education
- ☐ E. Scientific
- ☐ F. Control
- ☐ G. Music

IV. Your TRS-80, is it a
- ☐ 1. Level I
- ☐ 2. Level II
- ☐ 3. Model II
- ☐ 4. Don't own one yet

V. What peripherals do you have (check all that apply)?
- ☐ A. Expansion interface
- ☐ B. Disk
- ☐ C. Printer

VI. How much have you spent on hardware?
- ☐ 1. less than $500
- ☐ 2. $500-1,000
- ☐ 3. $1,000-2,000
- ☐ 4. $2,000-4,000
- ☐ 5. $4,000-6,000
- ☐ 6. more than $6,000

VII. How much have you spent on software?
- ☐ A. less than $100
- ☐ B. $100-250
- ☐ C. $250-500
- ☐ D. $500-1,000
- ☐ E. more than $1,000

VIII. What is your level of education?
- ☐ 1. Post-graduate
- ☐ 2. College
- ☐ 3. High school

IX. How many people read your copy of 80 Microcomputing?
- ☐ A. 1
- ☐ B. 2
- ☐ C. 3
- ☐ D. 4 or more

X. If you are not a subscriber please circle number 500.

80 Microcomputing • POB 2743 • Clinton IA 52735

1	6	11	16	21	151	156	161	166	171	301	306	311	316	321	451	456	461	466	471
2	7	12	17	22	152	157	162	167	172	302	307	312	317	322	452	457	462	467	472
3	8	13	18	23	153	158	163	168	173	303	308	313	318	323	453	458	463	468	473
4	9	14	19	24	154	159	164	169	174	304	309	314	319	324	454	459	464	469	474
5	10	15	20	25	155	160	165	170	175	305	310	315	320	325	455	460	465	470	475
26	31	36	41	46	176	181	186	191	196	326	331	336	341	346	476	481	486	491	496
27	32	37	42	47	177	182	187	192	197	327	332	337	342	347	477	482	487	492	497
28	33	38	43	48	178	183	188	193	198	328	333	338	343	348	478	483	488	493	498
29	34	39	44	49	179	184	189	194	199	329	334	339	344	349	479	484	489	494	499
30	35	40	45	50	180	185	190	195	200	330	335	340	345	350	480	485	490	495	500
51	56	61	66	71	201	206	211	216	221	351	356	361	366	371	501	506	511	516	521
52	57	62	67	72	202	207	212	217	222	352	357	362	367	372	502	507	512	517	522
53	58	63	68	73	203	208	213	218	223	353	358	363	368	373	503	508	513	518	523
54	59	64	69	74	204	209	214	219	224	354	359	364	369	374	504	509	514	519	524
55	60	65	70	75	205	210	215	220	225	355	360	365	370	375	505	510	515	520	525
76	81	86	91	96	226	231	236	241	246	376	381	386	391	396	526	531	536	541	546
77	82	87	92	97	227	232	237	242	247	377	382	387	392	397	527	532	537	542	547
78	83	88	93	98	228	233	238	243	248	378	383	388	393	398	528	533	538	543	548
79	84	89	94	99	229	234	239	244	249	379	384	389	394	399	529	534	539	544	549
80	85	90	95	100	230	235	240	245	250	380	385	390	395	400	530	535	540	545	550
101	106	111	116	121	251	256	261	266	271	401	406	411	416	421	551	556	561	566	571
102	107	112	117	122	252	257	262	267	272	402	407	412	417	422	552	557	562	567	572
103	108	113	118	123	253	258	263	268	273	403	408	413	418	423	553	558	563	568	573
104	109	114	119	124	254	259	264	269	274	404	409	414	419	424	554	559	564	569	574
105	110	115	120	125	255	260	265	270	275	405	410	415	420	425	555	560	565	570	575
126	131	136	141	146	276	281	286	291	296	426	431	436	441	446	576	581	586	591	596
127	132	137	142	147	277	282	287	292	297	427	432	437	442	447	577	582	587	592	597
128	133	138	143	148	278	283	288	293	298	428	433	438	443	448	578	583	588	593	598
129	134	139	144	149	279	284	289	294	299	429	434	439	444	449	579	584	589	594	599
130	135	140	145	150	280	285	290	295	300	430	435	440	445	450	580	585	590	595	600

Name ______________________

Address ______________________

City ______________ State ______ Zip ______

BOOKS

80 Microcomputing
Peterborough NH 03458

Please send me the following 80 Microcomputing products:

Qty.	Catalog #	Title	Unit Price	Total

Add $1 shipping/handling ______

Total ______

Enclosed $ ______ ☐ Check ☐ M.O.

Bill: ☐ AE ☐ MC ☐ VISA

Card # ______________________

Exp date ______________ Interbank # ______________

Signature ______________________

Name ______________________

Address ______________________

City ______________ State ______ Zip ______

Please allow 4–6 weeks for delivery

No C.O.D. orders accepted.

7-81

SUBSCRIPTION

80 Microcomputing POB 981
Farmingdale NY 11737

80 Microcomputing subscribers
SAVE 40%
Subscribe now.

Please allow 4–6 weeks for delivery

☐ New Subscription ☐ Renewal

☐ 1 year—$18 ☐ 2 years—$30 ☐ 3 years—$45

☐ Enclosed $ ______ ☐ Check ☐ M.O.

Bill: ☐ MC ☐ VISA ☐ AE ☐ Me

Card # ______________ Exp. date ______

Signature ______________ Interbank # ______

Name ______________________

Address ______________________

City ______________ State ______ Zip ______

Canada: $20 1 year only, US funds
Other foreign: $28 one year only, US funds

317B9

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 217 CLINTON IA 52735

POSTAGE WILL BE PAID BY ADDRESSEE

80 microcomputing T.M.

POB 2743
Clinton IA 52735

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 780 FARMINGDALE NY 11737

POSTAGE WILL BE PAID BY ADDRESSEE

80 microcomputing TM

Subscription Dept.
POB 981
Farmingdale NY 11737

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 80 PETERBOROUGH NH 03458

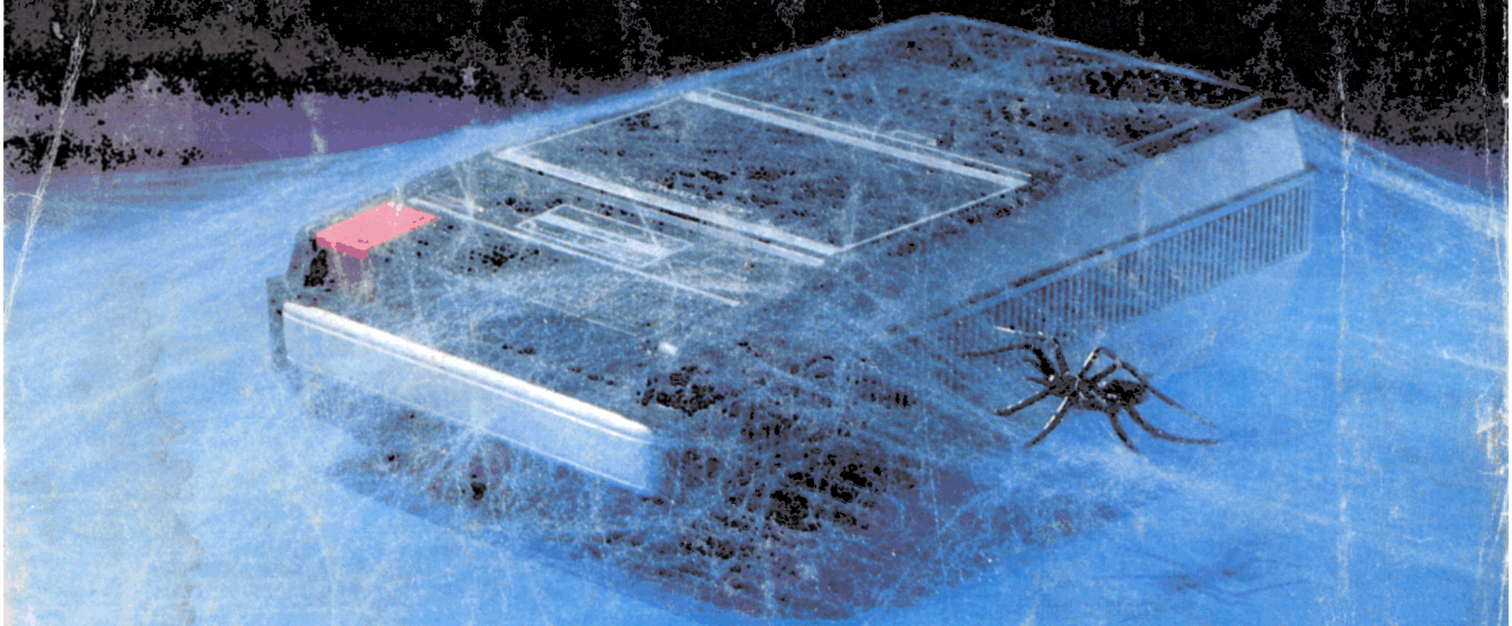
POSTAGE WILL BE PAID BY ADDRESSEE

80 microcomputing TM

Peterborough NH 03458

Att. Mail Order

TIRED OF WAITING?



Frustrating isn't it! No matter how much you speed up your program it still seems to take forever to save data onto a cassette. Wouldn't it be great if someone could design a mass storage system with the speed of a disk, but at half the cost? *Exatron* did, the *Exatron Stringy Floppy (ESF)*.

Totally self-contained, the ESF is an extremely fast, reliable, and economical alternative to cassette or disk storage of programs or data. All of the ESF's operations are under the computer's control, with no buttons, switches, knobs or levers to adjust or forget.

The ESF uses a miniature tape cartridge, about the size of a business card, called a wafer. The transport mechanism uses a direct drive motor with only one moving part. Designed to read and write digital data only, the ESF suffers from none of the drawbacks of cassettes - without the expense of disks.

Several versions of the ESF are available, for the *TRS-80, Apple, PET, OSI* and an *RS 232* unit. Even the slowest of the units is 15 times faster than a cassette, and all are as reliable as disk drives - in fact a lot of users say they are *more* reliable!

excellence in electronics

exatron

To get further information about the ESF give Exatron a call on their Hot Line *800-538 8559* (inside California 408-737 7111).

If you can't wait any longer then take advantage of their 30 day money-back guarantee, you've nothing to lose but time!

181 Commercial Street
Sunnyvale, CA 94086

TRS-80, Apple and PET are trademarks of Tandy, Apple and Commodore respectively.