

JULY 1983
ISSUE NUMBER 59

PRICE U.S. \$2.95
£1.25

THE ORIGINAL MAGAZINE FOR TRS-80™* OWNERS

H & E COMPUTRONICS INC.

**NOW INCLUDES
BUSINESS
COMPUTING.™**

Cover Photo by Harry Peterson
H&E COMPUTRONICS INC.
50 N. PASCACK ROAD
SPRING VALLEY, NEW YORK 10977
*TRS-80™ IS A TRADEMARK OF TANDY CORPORATION

U.S. POSTAGE
STANDARD
BULK RATE
Permit #58
New City, N.Y. 10955

FORWARDING & RETURN POSTAGE GUARANTEED

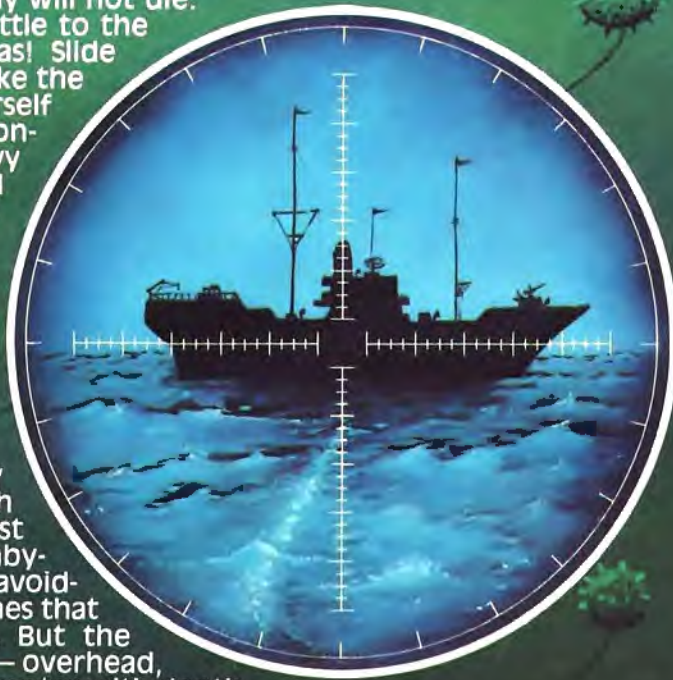
ABSOLUTELY RELENTLESS

The challenge of inner space — the fury of an enemy that seemingly will not die. This is SEA DRAGON — a battle to the death under the high seas! Slide into the Captain's chair, take the controls and prepare yourself for the most incredible non-stop action this side of Davy Jones' locker. SEA DRAGON puts you in control of a nuclear sub that's armed from stem to stern with enough firepower to take on King Neptune himself — and you'll need every missile, every torpedo, and every scrap of skill you can muster to survive.

The object of SEA DRAGON is to successfully navigate your sub through an underwater course past mountains and through labyrinthine passageways while avoiding clusters of explosive mines that rise from the seabottom. But the danger doesn't stop there — overhead, surface destroyers lace the water with depth charges; below, deadly attack bases and arcing lasers cut a killing swath that could reduce your sub to bubbling slag. But even these potentially lethal perils are dwarfed by the awesome menace that awaits you at the course's end.

SEA DRAGON — every possible "extra", is here to ensure your playing pleasure: exciting sounds, high score save, machine language graphics and an eye-popping scrolling seascape that extends the equivalent of over two dozen screens placed end-to-end, providing a diverse and unique challenge that will not diminish after repeated playings.

Nothing you've ever seen on your micro could possibly prepare you for this! You are ready now, ready for the ultimate in undersea action with a pace that is absolutely unyielding. SEA DRAGON — the arcade has finally come home.



FOR THE **APPLE** by John Anderson



FOR THE **ATARI** by Russ Wetmore



FOR THE **TRS-80** by Wayne Westmoreland & Terry Gilman



**ALL VERSIONS
ARE
JOYSTICK
COMPATIBLE!**



SEA DRAGON FEATURES

- Fantastic Scrolling Seascape
- Nearly Limitless Game Challenge
- High Score Save (disk version)
- Terrific Sound Effects
- Arcade Action Graphics™
- Apple version "talks" without special hardware!



**NOW FOR
THE
COLOR
COMPUTER!**

ORDERING INFORMATION	
APPLE 2 or APPLE 2 PLUS 48K Disk (DOS 3.3 required)	
042-0146	\$34.95
ATARI 32K Disk	
052-0146	\$34.95
ATARI 16K Tape	
051-0146	\$34.95
TRS-80 32K Disk	
012-0146	\$24.95
TRS-80 16K Tape	
010-0146	\$24.95
TRS-80 COCO 32K Tape	
080-0146	\$34.95

Adventure
INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.
PRICES SUBJECT TO CHANGE

To order, see your local dealer. If he does not have the program, then call 1-800-327-7172 (orders only please) or write for our free catalog.

Published by ADVENTURE INTERNATIONAL
a subsidiary of Scott Adams, Inc.
BOX 3435 • LONGWOOD, FL 32750 • (305) 830-8194

PUBLISHER

Howard Y. Gosman

BUSINESS MANAGER

Steven M. Kahan

EDITOR-IN-CHIEF

Hubert S. Howe, Jr., Ph.D.

BUSINESS EDITOR

Peter Shenkin, Ph.D.

MANAGING EDITOR

Martin Leffler

CONTRIBUTING EDITORS

Leo M. Conrad

Richard Kaplan

Spencer Koenig

Joseph Rosenman

Gordon Speer

A. A. Wicks

Steven M. Zimmerman, M.D.

ADVERTISING DIRECTOR

Kevin Rushalko

SALES MANAGERS

Valerie Furci

Rona Lowenfeld

Sheryl Prevot

DEALER SALES MANAGERS

Janet Lasher

ART DIRECTOR

Edmund Khaleel

OFFICE MANAGER

Beatrice Kahn

SOFTWARE MANAGER

Darlene Bell

CUSTOMER SERVICE

Robert Williams

INVENTORY CONTROL

James Withers

SHIPPING

Joan Gentry

PRODUCT DEVELOPMENT

Steven Kaplan

David Staub

PRODUCTION

Eileen Berman

Al Pizzo

Ruben B. Remigio

Louis Wetstein

MARKETING MANAGER

Andrew Hofer

PROGRAMMING MANAGERS

Roy Flynt

Nancy Rhodes

INVOICING MANAGER

Adele Damiano

SUBSCRIPTION FULFILLMENT

Janet Dillon

Karen Levine

JULY 1983

ISSUE NUMBER 59

CONTENTS**FEATURES**

- 9 Program Previews A. A. Wicks
Scripsit Made Clear/Zorlof Update
- 14 PIPS: Parts Inventory Planning Simulation Dennis P. Avola
Field Service Inventory Program
- 18 Practical Business Programs S. M. Zimmerman and L. M. Conrad
Month #7: Balance the Check Book
- 25 Twin Cities Computerist Goes Native Sallie Stephenson
Fred Yonke, Native American Computerist
- 29 A BASIC Calendar Anthony Scarpelli
A program to print calendars for any year
- 43 KOPYKAT Joe W. Rocke
Model III features for the Model I
- 48 Ask Richard Richard Kaplan
Questions about DBMSs, translators, number storage, more
- 52 Using NEWDOS/80 Disk Files, Part II John L. Gross
Some new features for "FF" files in Version 2
- 53 Maze James Gallagher
Build and print a rectangular maze
- 56 Hardware Review K. I. Brown
MDX-2 Interface Expansion Board from Micro-Design
- 58 Young Person's Math Program Robert V. Pritula
A program to help young children learn to solve math problems
- 59 Video's Visible Future: the Computer Connection Mike Shadick
"We ain't seen nothin' yet!"

REGULAR DEPARTMENTS

- 2 Bits and Pieces Howard Y. Gosman
Publisher's Remarks
- 4 The Crystal Ball
News and rumors of interest to TRS-80 owners
- 5 Beginner's Corner Spencer Koenig
Interpreters—the Grease in Interactive Computing
- 8 Letters to the Editor
Readers tell us what's on their minds
- 21 Pocket Computer Corner S. M. Zimmerman and L. M. Conrad
The simplex method of linear programming
- 55 Color Computer Corner
New products and speculations about the TRS-80 color computer
- 62 Computronics Classified
- 68 Advertising Directory

Entire contents copyright © 1983 by H & E Computronics, Inc. All rights reserved. Printed in the United States of America.

All correspondence should be addressed to: The Editor, H & E Computronics, Inc., 50 North Pascack Road, Spring Valley, NY 10977. Unaccepted manuscripts will be returned if accompanied by sufficient first class postage. H & E Computronics will not be responsible for the return of unsolicited manuscripts, cassettes, floppy diskettes, program listings, etc. not submitted with a self-addressed, stamped envelope. Opinions expressed by the authors are not necessarily those of H & E Computronics, Inc.

Material appearing in the *H & E COMPUTRONICS MAGAZINE* may be reprinted without permission by school and college publications, personal computing club newsletters, and nonprofit publications. Only original material may be reprinted; that is, you may not reprint a reprint. Each reprint must carry the following notice on the first page in 7-point or larger type:

Copyright © 1983 by *H & E Computronics, Inc.*, 50 North Pascack Road, Spring Valley, NY 10977.

Please send us two copies of any publication that carries reprinted material.

ADVERTISING RATES

Contact Advertising Director for rate card. Special discounts available for multiple insertions.

Kevin Rushalko
(603) 547-2970

For information about receiving copies of *COMPUTRONICS* in quantity contact:

U.S. and Canadian Distributor

H & E Computronics, Inc.
50 North Pascack Road
Spring Valley, New York 10977
Attention: Steven M. Kahan
Tel.: (914) 425-1535

International Distributor

Worldwide Media Service, Inc.
386 Park Avenue South
New York, New York 10016
Attention: *Sandra A. Joseph*
Cable: *WORLDMEDIA*
Telex: 620430 (WUI)
Tel.: (212) 686-1520

BITS AND PIECES

Howard Y. Gosman

THE MODEL 4

The big news this month is the new TRS-80 Model 4, which is an upgraded Model III with some really fantastic new features.

The standard 64K Model 4 with two disk drives sells for a surprisingly low \$1999—a very competitive price. You can upgrade the 64K computer to 128K with the addition of a single 64K RAM kit, which costs \$149. (A 16K Model 4 without disk drives is also available for \$999).

The Model 4 is compatible with all software that runs on the Model III—it will run the same TRSDOS 1.3 that the Model III uses (as well as LDOS and probably NEWDOS/80, DOSPLUS and other alternative operating systems). Also, Radio Shack will soon have a version of CP/M 3.0 for the Model 4 (for about \$200), which requires no modifications to the computer and will allow owners to use most of the thousands of already-existing CP/M programs on the Model 4.

Radio Shack has taken a hint

from the rest of the industry and included some very good software with the Model 4 (it all comes with TRSDOS 6.0). To start with, you get a disk-emulator program called MEMDISK, which allows you to use your extra memory as a superfast pseudo-disk drive, which will drastically cut down on disk accesses for many programs—you'll be able to load a long file into main memory in the blink of an eye.

The Model 4 will also come with its own printer-spooler software, which allows you to print out a long document while simultaneously running another program on screen. No more waiting for the printer to finish printing. Presumably, the size of your printer buffer will depend on the amount of the computer's memory you wish to dedicate to the spooler (which will be a good reason to have your Model 4 equipped with 128K). This is a very important feature which will probably be showing up soon as a

continued on page 6

The *H & E COMPUTRONICS MONTHLY NEWS MAGAZINE* is published by H & E Computronics, Inc., 50 North Pascack Road, Spring Valley, New York 10977. The *H & E COMPUTRONICS MONTHLY NEWS MAGAZINE* is not sponsored, nor in any way officially sanctioned by Radio Shack, a division of Tandy Corporation.

The purpose of the *H & E COMPUTRONICS MONTHLY NEWS MAGAZINE* is to provide and exchange information related to the care, use, and application of the TRS-80™ computer systems. H & E COMPUTRONICS, Inc. does not take any financial responsibility for errors in published materials. Users are advised to check and edit vital programs carefully.

The *H & E COMPUTRONICS MONTHLY NEWS MAGAZINE* encourages comments, questions, and suggestions. H & E COMPUTRONICS will pay contributors for articles and programs published in the magazine.

The *H & E COMPUTRONICS MONTHLY NEWS MAGAZINE* is typeset by Photonics, Ltd., 188 Highwood Ave., Tenafly, NJ 07670, and is printed by Kay Offset Printing Service, Inc., 154 Grand Street, New York, NY 10013.

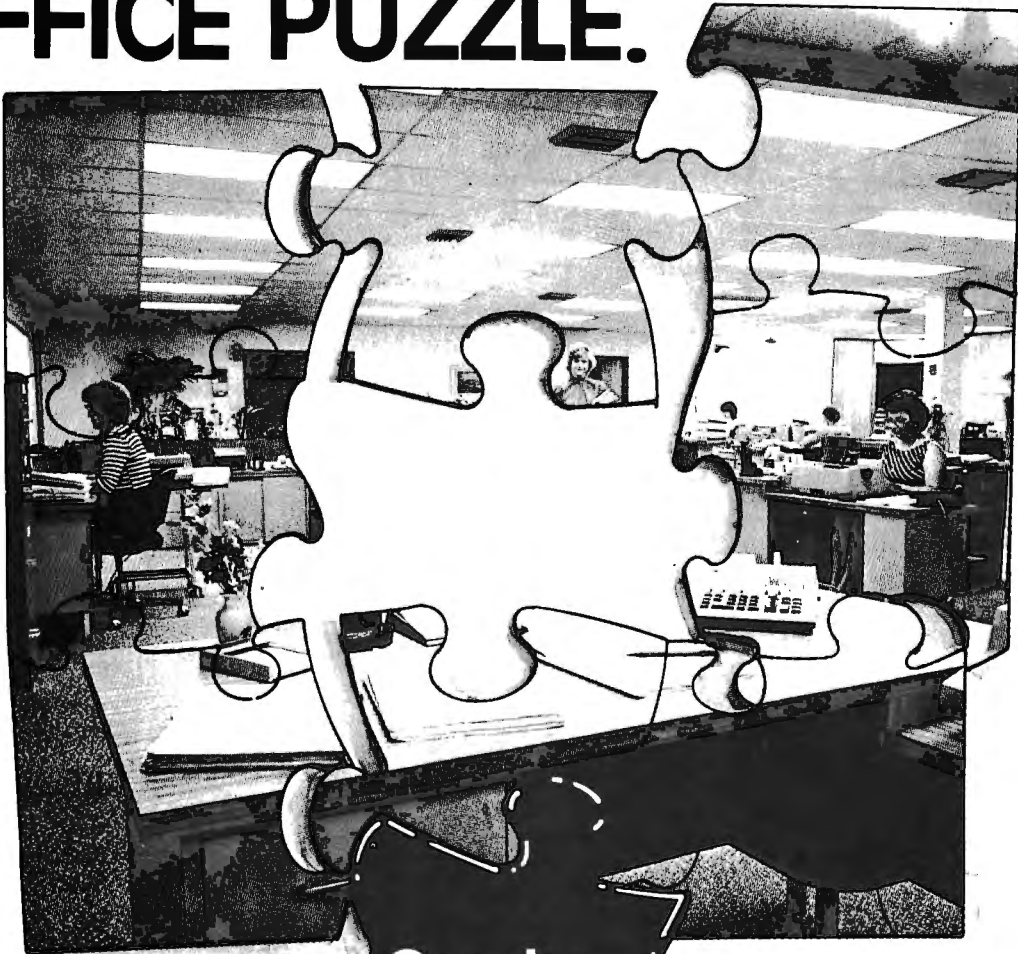
SUBSCRIPTION RATES

\$24 per year SURFACE MAIL	U.S. Only
\$36 per year FIRST CLASS MAIL	U.S.
\$36 per year AIR MAIL	Canada and Mexico
\$48 per year AIR MAIL	Outside U.S., Canada and Mexico
\$3 per copy Single Copies	U.S., Canada and Mexico
\$4 per copy Single Copies	Outside U.S., Canada and Mexico

Foreign subscriptions and sales should be remitted in U. S. funds drawn on a U.S. bank.

YOUR SUBSCRIPTION HAS EXPIRED. IF . . . THE NUMBER ABOVE YOUR NAME AFTER THE DASH ON YOUR MAILING LABEL IS 59 (OR LESS). THE NUMBER FOLLOWING THE DASH TELLS YOU THE LAST ISSUE THAT YOU WILL RECEIVE. For example, if your subscription number is 16429-59, your subscription expires with this issue (issue #59).

MULTIPLEXING, THE MISSING PIECE TO A BUSY OFFICE PUZZLE.



A Quadra-MAC (Multiplexor) can expand the capabilities, capacity, and speed of your office services with the minimum expenditure. The Quadra-MAC allows a company to hook up to four TRS-80 Model I and/or III computers to one hard drive. This immediately gives all of those computers the benefits of having their own hard drive but saves the company thousands of dollars because one hard drive can be shared by up to four computers instead of having to buy a hard drive for each unit.

The Quadra-MAC also allows each computer to use as much processor time as needed without affecting the other users because only the hard disk drive is being shared not the microprocessor.

In addition, each computer can run totally independently of the other computers without being affected by either hardware or software failure of another computer.

**Quadra-
MAC**

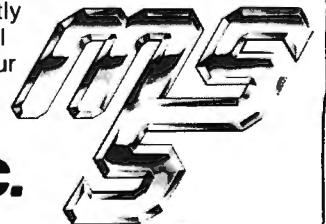
The Quadra-MAC also offers the following features:

- Up to 100 foot cables (allows up to 200 foot radius between two terminals)
- Includes a DOSPLUS Operating System
- Compatible with most hard drive systems
- Full File Locking with Multiplexing Features (prevents the file from being altered by more than one operator at the same time)
- No connector or buffer box cost

The Quadra-MAC is available for only \$995.00 which includes the complete DOSPLUS Operating Systems. The cable costs only \$2.50 per foot.

If MSS products are not currently carried by your local dealer, call Micro Systems Software for your closest dealer.

Dealer Inquiries Welcome.



MICRO-SYSTEMS SOFTWARE, INC.

4301-18 Oak Circle, Boca Raton, Florida 33431, Telephone: (305) 983-3390 Toll Free 1-800-327-8724

More "HELP" is on the way with these two NEW utilities for the NEWDOS-80 Model III users.

CAT/CMD is a totally DOS independent utility that offers the ability to read most major DOS directories from whatever your system disk might be. Displays the name, date, and type of DOS with the directory. You know it's on there, and now you can read it!\$16.50

RESTORE/CMD You can really "BRING'EM BACK ALIVE" with the latest alternative to an accidental or intentional file loss by the use of the "kill" command. RESTORE is a DOS command and could save you hours of typing in just seconds.\$16.50

Get CAT and RESTORE on one disk for \$29.95 *Apparat, Inc.

Mayday SOFTWARE

P.O. Box 88 • Rock Creek Road
Phillips, Wisconsin 54555
(715) 338-3986

VISA/M-C WELCOME

Personal checks require additional 14 days
All prices include shipping

THE ULTIMATE IN SOFTWARE UNPROTECTION

FPS-3 IS A FRONT PANEL SIMULATOR FOR THE TANDY CORP. TRS80 MODEL III **JUST FLIP A SWITCH AND!!!** THE PROGRAM IN MEMORY IS COPIED TO YOUR CHOICE OF DISK OR TAPE. TO RUN THE COPY SIMPLY BOOT THE DISK FROM RESET OR LOAD THE TAPE WITH THE SYSTEM COMMAND. YOU DO NOT NEED ANY TECHNICAL KNOWLEDGE TO USE THE FPS-3. ALL YOU NEED TO INSTALL THE FPS-3 ARE A HALF HOUR OF YOUR TIME AND A SCREWDRIVER. THE COST FOR A COMPLETE FPS-3 IS ONLY \$50.

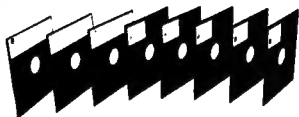
**WHAT THE SOFTWARE GODS
HAVE HIDDEN**

**THIS LITTLE
SHALL REVEAL DEVIL**

J.E.S. GRAPHICS, P.O. BOX 2752,
TULSA, OK. 74101 CALL 918 742 7104.

dy Jan **Dysan**
CORPORATION

better from inside out



at the lowest price!

Call our Modem Hotline (anytime) - 619-268-4488
for exclusive monthly specials. Our free catalog
contains more than 600 fantastic values.

ABC Data Products

(formerly ABM)

8868 CLAIREMONT MESA BLVD.
SAN DIEGO, CALIFORNIA 92123

ORDERS ONLY 800-854-1555 ITT TELEX 4992217 INFORMATION 619-268-3537

THE CRYSTAL BALL

News and Rumors of Interest to TRS-80

MISSED A PREDICTION

Well, we were wrong last month in our speculation that the Model 4 might turn out to be an OSBORNE-sized portable computer. Instead, they chose to stick with the basic design of the Model III. But we still believe that Radio Shack is working on a model to address the same market as the OSBORNE. Maybe they're just going to wait and see how well the Model 100 is received before going ahead with another portable. (And we're still waiting to see their new Korean-made Color Computer.)

MILITARY TRAINING WITH VIDEOGAMES

An important merger is developing between simulators used for military training purposes and videogames. A recently published study (by International Resource Development, Inc.) indicates that simulators are beginning to look more like videogames, and videogames are beginning to look more like rudimentary simulators.

At first glance, it might appear that simulators and videogames are radically different businesses. Simulators are high-ticket, low-volume products designed with the most serious of purposes. They are used to train personnel such as airplane pilots, astronauts, ship navigators and nuclear engineers. Videogames, on the other hand, are low-cost, high-volume entertainment products.

However, there are important similarities between simulators and videogames. Both attempt to replicate the "real world," and both, whether as a primary or secondary objective, teach certain skills. The simulator teaches how to use the primary device (such as a plane or ship), while the videogame teaches the skills necessary to master the videogame itself.

"Inevitably the two technologies will interconnect," says David Leducky, an IRD researcher. "Videogames will incorporate simulator principles in order to achieve greater real world likenesses, and simulator designers will come to understand the contribution fun makes to training, and add game aspects to simulator training."

Indeed, this is already occurring. Perceptronics manufactures a tank gunnery trainer, which the IRD report describes as follows: "It is a table top device with a sight, gunner controls, and scoreboard. Scenes of moving targets in a wide variety of environments are played back under microcomputer control from a laser-optical videodisc and presented in the sight. The gunner positions a reticle on the target and fires, with the number of hits and other aspects of his performance recorded."

Leducky maintains that such a design may prove to be beneficial. "It's not a bad strategy as far as training is concerned. Many of the youngsters entering the services have played videogames, and simulators designed in such a fashion may facilitate better and quicker learning."

According to IRD, there have been discussions within the simulator industry concerning the possible placement of simulators in recreation areas on military bases, thereby transforming (in the minds of the users, at least) that "training" into "entertainment." And, as Leducky stated half-seriously, "perhaps army recruiters should visit local videogame parlors to find the fighter pilots of tomorrow."

The IRD report notes that computer-based simulators are now used to train personnel in a wide variety of industries and skills, including general and commercial aviation, utilities, ship navigation and military and space applications. Simulators are also being used as research and development tools in such areas as new vehicle development, system development and human factors research.

MORE LISA-TYPE SYSTEMS

For some time now, we've been waiting to see the new "LISA-style" operating system being developed by VisiCorp, called VISI-ON. This will be implemented soon on the IBM Personal Computer, and is designed to give the same kind of cursor-mouse control and graphic screen windows found on the LISA and the earlier Xerox STAR system.

continued on page 6

BEGINNER'S CORNER

Spencer Koenig

Interpreters — the Grease in Interactive Computing

Howdy and how are you? Good grief! What an awful title for this month's topic. For any of those out there with sensitive sensibilities, my apologies. It was the best I could come up with on short notice, but the point is an accurate one.

No doubt, you've been reading and hearing about interpreters for quite a while. This is especially true if you've had a Radio Shack machine for some time.

Some of the first terms you come across in the beginner's manual were ROM and INTERPRETER. If you've done any reading on the subject, then you know that your basic interpreter is contained in ROM (read only memory), and for that reason is unalterable.

So, you ask yourself "what's the big deal? So there's a ROM chip that contains BASIC. So what!". As a matter of fact it is a big deal. The Models I and III were two of the early birds to have this kind of setup. Before that situation got started, you had a minimal boot system or monitor that was used to load in the interpreter, as you would any other program.

If you didn't have a ROM in the first 16k of your machine (first 12K actually), you'd have a 64K machine just like the big boys (S-100's and CP/M machines).

This is the reason that you'd have to make a hardware modification to be able to use CP/M. It's also a big deal, because now the big Radio Shack has a Model 4 that is supposed to be a Model 3+C/PM. How did they do it?

It's really very simple. They make it possible to switch out the ROM and substitute another 16K of RAM that's usable by CP/M. You see, CP/M needs to have that lower (read that as lower numbered) memory for itself, and because the Radio Shack machines didn't allow that, a special version of CP/M, often called fake CP/M by those in the (snobbish) know, for the TRS-stuff.

As it turns out, you can go to

Radio Shack and pay them \$800.00, and they will cheerfully upgrade your (what was once a \$2500.00) machine to a brand spanking new \$2000.00 Model 4. Sounds like a great deal to me (just kidding—I wish they were). I'm sure Holmes electronics thinks it's terrific too. They advertise something similar for about \$300.00. Let's hear it for competition.

Well, this is getting a little off the track. The fact is, most of you have the ROM firmly imbedded, and so have a permanent BASIC interpreter.

What is an interpreter? Before we get to that, I'd like to go over a concept from the last topic, COMPILERS. A compiler is a program that takes input in the form of source code, usually high level (BASIC for example), and outputs another program in the form of machine code. This machine code, after some finagling with linkers and loaders, can be used by the processor directly. The advantages in most cases is that the speed of compiled code is much faster than interpreted code.

This brings us back to interpreters. An interpreter is a program also. This program takes an input in the form of high level source (i.e. BASIC) and outputs an action. That's really all it does, and if you think about it that's quite a lot.

One of the best analogies I've come across for how an interpreter works is this—consider the program to be like a child with a very bad memory, a piece of paper and a foreign language dictionary. This child has a job. It's required to follow instructions listed in some language. Spanish or French will do just fine.

As this person comes across a word, he has to look up the word. Once the word is looked up, it is translated. The translation is written down and the action is taken. Once the action is completed, the paper is thrown away. Then the next instruction is read, the word looked up, translated, written

down, the action taken and the paper thrown away.

This happens over and over again. Sounds pretty tedious, doesn't it, but this is very close to how your interpreter works and is the reason it's sooo slooow.

This is especially frustrating if you consider basic lines of code such as:

```
10 PRINT:PRINT:PRINT
```

For every occurrence of the word PRINT the process is repeated.

On the TRS-80, the interpreter looks at the source code and checks to see if it knows the word. If it doesn't know the word, it checks to see if the input was a number. If it wasn't a number, then you get an error message. This is the format for many interpreters.

If the interpreter does know the command, then it checks the address for the machine code associated with the command and jumps to that routine and executes it. This jump is like jumping to a subroutine in BASIC. Once the routine is done, the computer again looks to see the next word and goes through the process again.

As a matter of fact, it turns out that all programs ultimately boil down to be interpreted. That's why I was using the term interpreter/compiler in my last article.

It can be said that there are several types of interpreters. The first type is called a HARDWARE interpreter. This refers to the ultimate interpreter on all micro-computer chips. It's often located in the chip itself. You might have heard of it. It's called micro-code and is the bottom line interpreter.

This takes a "source" code that was translated by a compiler into machine code. The machine code is then converted into some action by the micro-code contained in the hardware interpreter.

The second type of interpreter is called a SOFTWARE interpreter (UCSD PASCAL, for example). This kind usually takes a source

continued on page 13



WHY BUY A JOY STICK TWICE?!

IF YOU HAVE AN ATARI COMPATIBLE JOY STICK, YOU CAN USE IT ON YOUR TRS-80 WITH OUR KIT

JOY STICK KIT*

Without Joy Stick Model I/III \$15.95
With Joy Stick Model I/III \$26.95

— **Be Your Own SYSOP !!!!** —

Bullet-80 Bulletin Board Model I/III
Version 8.0 \$150.00

Full line of H&E Computronics Software. 13% off any individual Versa Business Package
13% off All game software (from all manufacturers) if you mention this ad

Please call for information about ANY products. We have in stock a FULL LINE of Software from ALL Major Houses.

After-Market Computer Gallery**
P.O. Box 993 (Mail Order)
1 Franklin St. (Retail Outlet)
Danbury, CT 06810



Voice Line — 203 743-1299
Bullet-80 Computer Line — 203 744-4644
(300/1200 Baud)

PRICES DO NOT INCLUDE SHIPPING & HANDLING.

*Internal Installation Required. No trace cutting or electronics involved.

**A Division of Computer Services of Danbury

ARRANGER

100% Machine Language Disk Index Program for the TRS-80 Model I & III.
Automatically recognizes ALL major DOS's!

The Arranger is a master index system that automatically records the names of your programs, what disks those programs are on and type of DOS. Features include —

- Automatic single and double density recognition.
- Accepts LDOS, DOS+, TRSDOS, DBLDOS, NEWDOS/80, MULTIDOS . . .
- Works interchangeably with Model III, I double density.
- Capacity of 250 disks, 44 filenames/disk
- Quickly locates any amount of free granules
- Finds a program in less than 30 seconds!
- Alphabetizes 1500 filenames in 40 secs.!
- Option to sort by any extension (/BAS, /CMD, /???)
- Easily updates diskettes previously added with only 2 keystrokes.
- Backup function built in.
- Uses 1 to 4 drives, 35, 40 or 80 tracks.
- Radio Shack doubler compatible

Requires 32k / 1 disk minimum
JUST \$29.95

FREE SHIPPING

SATISFACTION GUARANTEED

Specify: TRS-80 Model number
(If you've added double density to your Model I, please indicate)

TRIPLE-D SOFTWARE

P.O. Box 642C PERSONAL CHECK
Layton, Utah 84041 VISA OR
(801) 546-2833 MASTERCARD

THE CRYSTAL BALL

continued from page 4

This will be just the beginning.

Microsoft and Micropro International are each working on similar operating systems, and we may even see such a system running on a TRS-80 before long. In recent computer magazines, Microsoft has also been advertising the Microsoft Mouse—which, although primarily designed for the IBM Personal Computer, will probably be available for other systems as well.

Don't forget about Apple. Their MACINTOSH system, a low-cost computer with similar operation to the LISA, will be appearing this summer, perhaps even by the time you read this column.

Since this type of system has only been available for a short time (and the cost is still prohibitive for most users), there's not a lot of really widespread reaction yet. But we'll be keeping a sharp eye on developments, and pass along any new information we come across.

HIGHER HARD DISK CAPACITIES

Seagate Technology will shortly announce two new 5-1/4 inch hard disk drives, following a trend toward higher capacities for mini hard disks. The new drives will reportedly have capacities of 25 and 33 Megabytes. Within another year or so, these capacities will shoot up to near or above 100 Megabytes. Slowly but surely, as prices fall, people will move more and more toward hard disk systems. Users who only own floppy disks will eventually find themselves in a situation similar to that confronting cassette users today—their storage media will be too limited in capacity, too unreliable, and too slow in retrieval to be practical with newer, more sophisticated software. ■

BITS AND PIECES

continued from page 2

built-in capability on virtually all new computers.

A "complete" telecommunication program is also included as a standard feature, making it possible to interface the Model 4 with other computers and on-line services. A facility is also provided to allow easy communication between the

Model 4 and the new TRS-80 Model 100 portable computer.

Another very good software package is the Model 4's "Job Control Language" which allows you to leave your computer completely unattended for long periods of time while it proceeds to run many different programs according to your instructions.

Another new item is a Model 4 version of Digital Research's CBASIC to be sold by Radio Shack. We don't yet know about the price or when it will be available.

Like the TRS-80 Model 12, the Model 4 has a "key-click" feature which gives audible feedback as you type any of the alphanumeric characters (naturally, if you don't want the key-click, you can turn it off). In addition, you will have complete control of an internal sound generator—you can set the pitch and duration with new commands in Model 4 BASIC.

There are other additions to the new BASIC. CHAIN and COMMON commands allow you to link programs together more efficiently, and a number of other sophisticated BASIC commands are now available, such as WHILE-WEND, WAIT, a command to reset the printer's line width from BASIC and more. The Model 4 BASIC will also allow 40-character long variable names! Instead of having AD\$, you can now use ADDRESS\$, and other fully descriptive variable names.

The Model 4 displays 80 characters by 24 lines, just like the Models II, 12 and 16, but we were wrong when we predicted that high-resolution graphics would be built-in. In fact, you have to install a graphics expansion board—the same one used for graphics on the Model III. The Model 4 keyboard has 3 user-programmable function keys above its 12-key numeric keypad, but otherwise the keyboard is just like the Model III's.

All in all, the Model 4 looks like a good upgrade for the Model III, and you can have your Model III upgraded to a Model 4. (That costs \$799, which seems a little steep.)

BUSINESS GRAPHICS

BIZGRAPH is a self-prompting business graphics program designed to work exclusively with the TRS-80 Model III GRAFYX SOLUTION board

continued on page 8

New Release

Now supports Mailing Lists, Form Letters, "ZAP-PROCESSING", and 18 more printer drivers.



**STILL ONLY
\$69.95**

IF YOU STILL THINK YOU HAVE TO SPEND \$200 FOR A GREAT WORD PROCESSING SYSTEM, THEN YOU NEED TO READ THIS AD!!

The Magnificent WORD PROCESSING SYSTEM

For the TRS-80 Model I and III

- Supports over 50 different popular printers including OKIDATA Microline 80, 82A, 83A, 84A, Qume, Centronics 737, 739, Radio Shack Line Printer IV, VI, Daisy Wheel II, EPSON MX-80, MX-100, Grafrax, Grafrax Plus, Gemini-10, Gemini-15, NEC PC-8023A-C, Spinwriter 5510, 5515, 5520, 5525, C. Itoh Prowriter 8510, Starwriter FP-1500, F-10, Tec 8500R, Smith-Corona TP-1, Brother HR-1, COMREX Com-Riter CR-1, IDS Microprism 480, and Diablo 630.
- Supports proportional space right-margin justifying on Centronics 737, 739, Radio Shack Line Printer IV, Daisy Wheel II, Grafrax Plus, NEC PC-8023A-C, Spinwriter 5510, 5515, 5520, 5525, C. Itoh Prowriter 8510, Starwriter FP-1500, F-10, and Diablo 630.
- Powerful Mailing List and Mail-Merge capabilities for personalizing standard legal documents and Form Letters, handling infinite number of data records per run, infinite number of data fields per data record, and data fields as large as up to 1000 characters each.
- Brand new feature called "ZAP-PROCESSING", allows you to display and edit any type of data or program file in "ZAP" (byte-hexidecimal) format.
- Any character or symbol your printer can print, even dot graphics, can be used in mid-line printing with the Special Character feature.
- Written in fast Z80 machine language with type-ahead key-stroke buffering for speed typing.
- Single key-stroke control of all editing functions for ease of use.
- Continuous on-screen display of word count, line count, and free memory count.
- Superscripts, subscripts, underlined, bolded, expanded and condensed type styles - combine and intermix within a line.
- Automatically justifies and word-wraps on the screen as you type.
- Search, Replace, and Global Search and Replace.
- Odd and even page user-definable headers, footers, and page number lines, with automatic page numbering.
- User-definable linespacing, sheet size, top, bottom, left, and right margins.
- Move blocks of text and copy blocks of text from disk, to disk, and within the text.
- Examine disk directory on any disk and kill files while editing.
- Powerful full-screen editing features for EDTASM and BASIC files, including automatic renumbering of lines.
- Built in function to dump contents of screen to printer.
- Print-previewing formats text, inserts headers, automatically numbers pages, etc. on the screen without printing it on paper.
- Page by page pausing capability for sheet fed printers.
- Supports both parallel and serial printers.
- Printer control code access.
- Works with NEWDOS, NEWDOS80, TRSDOS, MULTIDOS, LDOS, and DOSPLUS - Single or Double Density.
- Compatible with most all available spelling checker programs.

GUARANTEE

Many word processing systems claim theirs are the best, but few would dare guarantee them. Not us! We are confident that ZORLOF is the most useful word processing system on the market for under \$200. If you don't agree, return it within 30 days for a full refund.

Add \$2.00 shipping & handling. Florida residents add 5% sales tax. Checks require 3 weeks to clear banks.

ANITEK

SEE YOUR LOCAL
DEALER OR CALL

(305) 259-9397

MasterCard

VISA

ANITEK SOFTWARE PRODUCTS □ P.O. BOX 1136 □ MELBOURNE, FL. 32935 □ (305) 259-9397

**NODVILL
DIET
PROGRAM**

\$69.95

"TAKE A BYTE"

MAINTAIN, LOSE OR GAIN WEIGHT
With your TRS-80 Double Density Disk Model I or III
and
The NODVILL DIET PROGRAM "Take a Byte"

you can quickly and accurately

- Calculate caloric and nutritive food intake
- Evaluate nutritive value of your diet
- Compare daily diet to individual RDA Chart
- Create personalized daily meals and menus
- Plan varied daily menus based on sound nutrition
- Save records of daily meals and menus for future planning
- Print nutrition charts, food, meal, menu, and grocery lists

"Take a Byte" is a Modular BASIC Program

MAIN Program Menu

- (1) Recommended Daily Dietary Allowance (RDA) CHART
- (2) 733 Expandable Random Access FOOD LIST Data File
- (3) 28 Nutritional MEAL LIST Data File Examples
- (4) 7 Balanced DAILY MENU LIST Data File Examples
- (5) GROCERY LIST Program Module
- (6) 25 Page USER'S MANUAL including Charts and Tables

ALL Data Files can be EXPANDED and MODIFIED systematically and flexibly to reflect your personal diet, your choice of diet book, or your doctor's suggested diet.

For more information Call (703) 431-6449
To Order: Mail a check or money order for \$69.95 to:
NODVILL Software
24 Ford Road
Huntsville, Conn. 06032

MS-DOS is a trademark of the Intel Corporation.



**MAKE YOUR OWN
SIGNS IN MINUTES**

Reduction of an actual sign

The Banner Machine ©

- For the TRS-80 I & III with 32K tape or 48K disk
- For use on the Epson MX-80 with Graftrax
- Uses dot graphics instead of TRS-80 block graphics
- Menu-driven program
- Operation similar to a word processor
- Makes signs up to 10" tall by any length
- 10 sizes of letters from 3/4" - 8" high
- Mono or proportional spacing
- Automatic centering; Right and left justifying
- Makes borders of variable width up to 3/4"


Order The Banner Machine © — \$49.95 from

Virginia Micro Systems	Virginia Micro Systems 13646 Jeff Davis Highway Woodbridge, Virginia 22191	VISA MasterCard
	Phone (703) 491-6502	

**TRS-80 MODEL I T.M.*
GOLDPLUG - 80**

Eliminate disk re-boots and data loss due to poor contact problems at card edge connectors. The GOLD PLUG - 80 solders to the board card edge. Use your existing cables.

CPU/keyboard to expansion interface \$18.95
Expansion interface to disk, printer, RS232, screen printer (specify) \$9.95 ea
Full set, six connectors. . . \$54.95




EAP COMPANY
P.O. Box 14, Keller, TX 76248
(817) 498-4242
*TRS-80 is a trademark of Tandy Corp.

BITS AND PIECES
continued from page 6

(Micro-Labs' GRAFYX SOLUTION is a plug-in board which gives you 98,304 points in a 512 by 192 matrix).

BIZGRAPH will generate clear, accurate graphs for use in business applications, and will be useful for managers, small businessmen, and analysts. The BIZGRAPH package can create line graphs, bar charts, pie charts, area plots, histograms, and scatter plots. Data can be entered directly from the keyboard or read from a data file on disk—including VISICALC files.

You select a graph type, enter the data or a data file name, select additional options, and a graph is quickly displayed in fine detail. Multiple data sets can be combined in one graph. Another good feature is the ability to display the high-resolution screen along with the normal text and low-resolution screen.

The program provides automatic labeling of X and Y axis points using 85 characters per line. Forecasting future trends is possible using line fitting, quadratic, and third order linear regression analysis. Data smoothing using moving averages is also possible. Finished graphs can be saved on disk or printed on any of about 20 popular graphics printers.

The BIZGRAPH program, sample graphs, and manual sells for \$98 and can be ordered from Micro-Labs, Inc., 902 Pinecrest, Richardson, TX 75080; (214) 235-0915.

ON THE MODEL 16

Radio Shack has made the announcement that XENIX (from Microsoft) will be the standard operating system of the Model 16 from now on. This system allows expansion to three simultaneous users with the addition of two of Radio Shack's DT-1 data terminals to the system. The XENIX system requires 256K of memory and a hard disk drive, and can be run on Model II/12's that have the Model 16 upgrade. According to Radio Shack, all applications software sold by Radio Shack will be movable to the XENIX system. XENIX will be provided to all present Model 16 owners and packaged with all new systems.

Data Management Systems has released CDDS, a relational database management system that uses the Model 16's M68000 processor. It is a menu-driven program that uses an "English-subset" query language, and a variety of file structures are possible. For more information, contact Data Management Systems, 211 N. El Camino Real, Suite 101C, Encinitas, CA 92024; (619) 942-0744.

NEWBASIC 2.0

Modular Software Associates is offering NEWBASIC 2.0 for the Models I and III, which adds more than 40 new commands to BASIC, including commands for spooling, sound, RS-232 initialization, single-key entry of keywords, user-programmed keys, line labeling, and moving the READ/DATA pointer. You can create custom versions of NEWBASIC, combine assembly language graphics with BASIC programs, and more. For further information, contact Modular Software Associates, 209 18th Street, Huntington Beach, CA 92648; (714) 960-6668. ■

LETTERS TO THE EDITOR

Traffic Accidents

I am currently a sergeant in the traffic division at the Kalamazoo County Sheriff's Department and I wonder if any of your many readers know where I may obtain documentation of a program designed for use in scientific traffic accident investigation.

Specifically, I am searching for a program that I could use with my TRS-80 III with 2 disk drives (which I really like!), that would assist me in reconstructing collisions and thereby save me hours of calculating formulas, etc.

If you or your readers can help me, I would be grateful!!

Sgt. Mike Capman, Traffic Bureau
Kalamazoo County Sheriff's Dept.
1500 Lamont
Kalamazoo, Michigan 49001
(616) 383-8963

We do not know of such a program, but perhaps one of our readers will be able to help you out.

continued on page 13

PROGRAM PREVIEW

A. A. Wicks

This Month: Scripsit Made Clear/ZORLOF Update

Many users of Scripsit have voiced their annoyance with the method of instruction that is provided with this otherwise eminent word processing program. Owners know, but potential owners may not, that the only instruction that comes with the program is a set of cassette recordings, and a manual that is more of a reference notebook to the recordings than textual instruction.

The verbal instructions given on the recordings are excellent, and are directed to the person completely new to word processing—and probably only just becoming familiar with a computer. Perhaps many will find this approach quite satisfactory. Marketing feedback must have proved to Radio Shack that the public likes this approach, for they have continued the cassette method with Superscripsit. (The latter has some additional capabilities beyond Scripsit—at a much higher cost.)

Nevertheless, the cassette-and-notes method of learning Scripsit has indeed disgruntled many users. The more sophisticated initiate rapidly becomes impatient when listening to these tapes, and absolute newcomers find concentrating on concepts in a new field from a voice on a tape difficult and frustrating. The irksome, "Do it—now!" of the voice, as pleasant as the announcer may be, irritates after a session or two.

What is the result of this? Well, many worry through the recording sessions hour by hour, and then, also using the notebook, gradually acquire proficiency. Others (and I was one of them), listened to the first two tape sessions, and then began getting the answers in bits and pieces from the notebook, but mostly from a small reference card (in itself difficult to use and understand), that Radio Shack issues. Both of these methods are slow ways to learn something, and I know that in my case, I have always felt that there might be something in Scripsit that I have never uncovered (but I am not going to listen to those tapes to find out!). To the reader, I want to say that

this is not another put-down of Radio Shack—I happen to be one of their biggest supporters—but only a criticism of the instructional method.

This rather lengthy preamble is a lead-in to the review of a book I was delighted to receive, that could be considered the "missing" Scripsit manual. This book, written by J. S. Wilson, can more properly be described as a textbook on Scripsit word processing operations, delineating facts and instruction in a formal no-nonsense manner. No humorous asides or "cutesy" comments here—this book is for the serious learner.

BOOK SECTIONS

The book commences by providing a basic explanation of word processing to the newcomer. This is at the neophyte level, and proceeds to list a page of word processing terms as used throughout the book, in addition to being used in the "trade," so to speak. This initial information is followed by a well-detailed explanation of the TRS-80 keyboard, and the use of the keys, reset, etc., turning the computer on and off to avoid voltage "spikes," and so on. Much of this information will be superfluous to some readers, and may easily be skipped over by those who are well acquainted with the TRS-80. This portion is followed by instructions for loading Scripsit from tape and from disk, for both the Model I and Model III computers. This too, could be disregarded by trained users; however, for those who need the information, it is far more complete than either the Radio Shack Disk manual or the Scripsit guide. With this preliminary "education" out of the way, the book now moves into the actual operations involving Scripsit, which begins with Section 3.

This Section covers the basic operations in Scripsit, describing the message line and the control key that is used, and how to type in and store characters. The author then tells how to delete and change characters, storing of text and how scrolling takes place as you type.

★ **FREE SHIPPING** ★
Within Continental 48 States



MORE MAXI'S

MANAGER w/Utility (B.O.) ..\$119.95
MAXI UTILITY\$44.95
MAXI CRAS Mod I/III\$84.95
MAXI MAIL Mod III\$84.95
MAXI STAT Mod I/III.....\$164.95

LAZYWRITER Mdl I/III.\$149.95
NEWSCRIP 7.0 w/labels...\$119.95
LDOS Ver. 5.1 Mod I or III....\$114.95
DOSPLUS 3.5S/3.5D/3.5III....\$119.95
MULTIDOS 1.6 SD/DD/III.....\$89.95
GEAP w/DotWriter 1.5.....\$84.95
SUPERUTILITY + Ver. 3.0..\$64.95
DATA-WRITER 2.0 Mdl I/III.\$129.95

Auto dial/Ans. **LYNX** \$239.95
Mod I/III

LNW-Doubler 5/8

Includes Dosplus 3.4D \$199.95

RIBBONS

ZIP BOX RELOADS	1/2 Dz.	Dz.
Epson MX 70/80-20 Yds.....	24.00	42.00
Epson MX 100-30 Yds.....	30.00	52.00
NEC/Prowriter.....	21.00	36.00
Centronics 730/737/739/779 or LP-III/IV-16Yds.....	18.00	32.00

All ZIP BOXES are individually sealed black nylon and require no rewinding. Epson Reloads also available in red, blue, brown, green & purple. Any mix allowed.

CARTRIDGES	Each	Dozen
Epson MX70/80.....	7.50	80.00
Epson MX100.....	12.50	134.00
Prowriter 8510 & NEC 8023A.....	7.50	80.00
RS LP III/IV.....	6.50	70.00
RS LP VI/III.....	6.50	65.00
RS DSY WH II or DWP 410.....	6.50	70.00
RS DSY WHII - Nylon.....	6.50	70.00
MICRLNE 80/82A/83A/92.....	N/A	24.00
MICRLNE 84 1/2 x 40 yds.....	5.50	60.00
Diablo Hytype II - Multi Strike.....	6.50	70.00
Qume - Multi Strike.....	6.50	70.00
NEC Spin - Hi Yld - Multi Strike.....	7.00	75.00
Centronic 703/04/53.....	11.00	120.00

Minimum order 3 cartridges - any mix. For smaller quantities add \$1.50 per order. All our reloads and cartridges are manufactured by one of the oldest and most reputable ribbon Mfg's. in the country.

***** QUALITY GUARANTEED *****

**SEE OUR EXPANDED ADS IN
80 MICROCOMPUTING
SEND FOR YOUR FREE CATALOG.**

ORDERING INFORMATION

No credit cards at these low prices. Add \$2.00 on all COD orders. Certified Ck/MO/COD shipped immediately. Please allow 2 weeks for personal checks. For extra fast service phone in your COD order. Free shipping within Continental 48 states via UPS ground. For Canada, Hawaii, Alaska, applicable shipping and insurance charges apply. Prices subject to change without notice. New York State residents please add appropriate sales tax.

The items listed above are a cross-section of our product line. We carry the full line of most companies listed in the ad, plus much more. **SEND FOR YOUR FREE CATALOG.**

**146-03 25th Road, Dept. C
Flushing, New York 11354**

Mon-Fri (212) 445-7124 Sat. 10 A.M.-9 P.M. 10 A.M.-5 P.M.



FLEXIBLE DISKS WHY PAY MORE?

Prices Per Box (of 10 Pieces)

	VERBATIM		MAXELL	MEMOREX	3M SCOTCH
	OPTIMA	DATALIFE			
5 1/4" SS, DD	#XL614-01 \$46.00	#MD-525-01 \$25.00	#MD-1M \$30.00	#3481 \$25.00	#744D-0. \$25.00
5 1/4" DS, DD	#XL624-01 \$59.00	#MD550-01 \$34.00	#MD2-DM \$41.00	#3491 \$34.00	#745-0 \$36.00
5 1/4" SS, QUAD	—	#MD577-01 \$33.00	#MD1-DDM \$40.00	#3504 \$37.00	#746-0 \$35.00
5 1/4" DS, QUAD	—	#MD557-01 \$42.00	#MD2-DDM \$49.00	#3501 \$47.00	#747-0 \$47.00
8" SS, DD	#XL34-6100 \$47.00	#FD34-8000 \$33.00	#FD1-1200 \$40.00	#3090 \$34.00	#741-0 \$33.00
8" DS, DD	#XL34-6201 \$58.00	#DD34-4001 \$40.00	#FD2-1200 \$45.00	#3102 \$40.00	#743-0 \$42.00

VERBATIM 8" or 5 1/4" HEAD CLEANING KIT \$10.00
 REPLACEMENT 8" or 5 1/4" CLEANING DISKS (10 pieces)..... \$16.00
 SCOTCH 8" or 5 1/4" HEAD CLEANING KIT \$32.00
 DISK MINDER 5 1/4" PLASTIC Holds 75 DISKS with 5 separators... \$20.00

Send (2) 20¢ stamps for list of all available diskettes!!

S. D. C. & S. Co., Inc.

85-07 1/2 Jamaica Avenue, Woodhaven, NY 11421
 (212) 849-8600

ORDERING INSTRUCTIONS: Money Orders, Bank Checks shipped same day.
 Personal checks must clear (15 days). Add \$3.00 shipping and handling per order.

He then mentions screen width for display, how it is changed, and how to determine the width in use at the moment, because even though the screen display is limited, *Scripsit* allows viewing up to 132 characters.

Section 4 describes how to acquire text from tape or disk. In this regard, *Scripsit Made Clear* comes with a cassette, too. But in this case, it is not a verbal instructional cassette, but rather several example texts on both sides of the tape for use while working with the book—thus saving you the necessity of typing in material for practice operations. This also allows direct references to be made to specific areas of the examples. Disk users may transfer the contents of this cassette to disk, and the method of doing this is clearly explained. This type of exposition occurs frequently throughout this book—whenever some action is required that is associated with *Scripsit*, but is not necessarily a direct function of *Scripsit*.

Following this, the method of "loading and chaining" files is aptly placed at this point in the text—this being the method whereby one

file may be appended to another in memory to become one new file. Searching for a file name is covered here, too. With *Scripsit*, you may not call for disk directory information directly from within *Scripsit*; unfortunate, but true. Mr. Wilson has, therefore, provided detailed information regarding leaving *Scripsit*, getting a Directory listing, and then returning to *Scripsit*. In the course of this explanation he also describes how to delete a document file from a disk. None of this is unique, and the standard methods of doing this are what is explained; however, some one who has had little experience in this area will find these explanations very helpful.

Saving text on tape or disk follows in Section 5, and once again, considerable detail is provided concerning tape operations, loading and verifying tapes, screen indications, and so forth. This detail could easily be a distraction if it were not for the fact that users familiar with such operations can recognize these areas and pass by. For those who need this type of guidance it can be a boon, and easily makes the text a "stand-alone" document. Saving to

disk, together with formatting instructions closes this Section.

The next Section, 6, takes the reader and student into the advanced procedures of *Scripsit*. In fact, many users of *Scripsit* who feel comfortable with it but do not believe they are utilizing this word processor to the fullest extent, could very well pick up this book and start here. However, this is not recommended in my opinion, without at least a cursory scanning of that which precedes Section.

Text insertion of characters, words, sentences and paragraphs opens the Section, followed by a good clarification of a more difficult to understand operation: the insertion of blank lines, from one to many. As easily as insertions are explained, deletions of the same grammatical groups are also described, plus the process of deletion to end-of-text.

Much more is presented in this Section—splitting paragraphs, indentations, interchange of text, tabulations. For the last-named, I would have liked to have seen some typical screen examples portrayed in the text; however, the screen sampling may be sufficient. In *Scripsit*, tabbed lines must be ended with an end-of-line symbol, the author points out. This hint may have been in the *Scripsit* tapes, but if it was, then I overlooked it—but it is a good hint, similar to many throughout the book.

Several short examples for the reader to type in are provided, in addition to the examples on the cassette, in this Section when hanging indentations are explained. It is interesting to note that the examples themselves are statements regarding functions of *Scripsit*, which serve to reinforce the learning process. Incidentally, hanging indentation with *Scripsit* is difficult and time-consuming. The explanation helps, but does not reduce the involvement, unfortunately. The Repeat Command, covered in this part of this Section, also provides a rather useful example—it may be used to make duplicate copies of a tape "save."

Printing of text, probably the most complex function of *Scripsit* (although Block Marking is also tricky), is given 12 pages of thorough explanation in Section 7. Page format commands, widow line suppression, copy markers for restricted printing, comment lines,

etc., are all adequately covered here. The Section concludes with a "letter-starter" file, as the author calls it. This file allows you to type the name and address of the sender and recipient of a letter only once, even though this is needed for both envelope and letter. It looks complex, but the telling is more than the doing, and, once performed, appears to be a useful shortcut that will be used over and over.

Section 8 clarifies one of the most useful but easier functions of *Scripsit*—Searching. A unique example of "search and count" uses an inventory that allows *Scripsit* to search out and count the number of times a certain item appears in the inventory.

The last text section provides 18 pages of extremely useful information on working with "Blocks" within *Scripsit*—that is, portions of text identified for treatment such as moving them from one place to another in a document, page numbering, headers and footers (information to be placed on every page top or bottom), and hyphenation of word divisions. The latter operation, with *Scripsit* (and other word processors), is, at best, tedium. This is especially so if the text line is justified, i.e., both sides of the text being a straight line vertically. You have a choice in printing justified text of either not having any end-of-line word division (with hyphens), or possibly many divisions. In the former situation, a large number of blank spaces may appear between some words. In this case, you, as the author, must decide what should be divided and whether or not the place the word processing program selects for division is indeed correct by accepted writing standards.

There is no doubt that the latter operation is difficult, and there is always a tendency to avoid the whole problem by letting the computer follow the first method mentioned above. *Scripsit Made Clear* is no exception—even while the author is explaining the method of word division and hyphenation, large blank spaces appear in many of the sentences, and few word divisions occur anywhere in the book. This would not have occurred had the composition been performed using proportional spacing, but in this text some of these blank areas are distracting. (I have been

advised by Mr. Wilson that the next printing of the book will be "reset in a better type style." If this means that the word spacing will be improved, this will be welcome.)

PRODUCTION

Scripsit Made Clear is printed on good quality white paper, with green covers of soft card stock front and back. Perfect binding (glued spine), is used and this does not allow the book to remain flat when open. I feel strongly that any publication that will be used for a reference at the computer, should be bound in such a manner that it is possible to keep it open at all times at any page chosen.

Printing is by the offset press process, and composition appears to be by either an impact printer or Selectric typewriter. The typeface is a good easy-to-read font. I was not intent on nit-picking, and noted only four typographical errors. There may be more, but this is a good record for a text of this size. Also, there are some other minor errors; such as the example given for Comment Lines, which omits a required asterisk, although the text correctly states that it is needed.

The Table of Contents is double-column, and is presented by Section heading, sub-heading and where required, double sub-heading. While complete, as far as the Contents pages are concerned, this publication would benefit the reader more if an Index was also included. A document that will eventually be used for frequent reference after being used initially for study, needs a good Index.

The book includes an Appendix containing four page-long tables. These are: Control Key Commands, Message Line Commands (Special Commands), Print Format Instructions, and Error Messages and Possible Causes. (The messages as shown on the cards are briefer than in the Radio Shack manual, however.) These tables have been included also as full-size separate Reference Cards, one on each side of a good quality plastic-laminated card. This is an excellent idea and a very handy reference when working with *Scripsit*. Not only are the commands or coded references included, but they are also briefly explained as to their use. I do not like to make comparisons, because usually a comparison is merely a

subjective opinion, but these reference cards are an improvement over the Radio Shack card issued for *Scripsit*. Incidentally, at the bottom of one of the cards, the author has included the keyboard layout as it appears in the *Scripsit* manual, complete with key function designators in red. This too, will save new *Scripsit* operators from needing to have more than one document on hand while using the program.

Although no biography of the author appears in the book, I suspect that he is an erudite educator, but, if not, then he should be writing more instructional-type books. The writing is easy-going in impeccable English, yet does not address the audience in a juvenile or jocular manner as many computer texts are prone to do.

Overall, I am greatly impressed with this publication. For sheer value, it is outstanding, as the price of \$14.95 includes the book, the cassette of example texts, and the two (four-subject) reference cards, all attractively packaged in a plastic sleeve envelope. In addition, a Supplement is included that covers a variety of subjects—How to Make Duplicate Copies of *Scripsit* Disk or Tape, High Speed Tape Operations (Model III only), changing *Scripsit* 3.2 to a high speed tape, and a special note about Percom Disk units.

Scripsit Made Clear by J. S. Wilson. Delegate Books, 3 Kellidon Drive, Felton, CA 95018. \$14.95 plus \$2.00 postage (90 cents tax in California).

ZORLOF—An Update

Another word processing program, ZORLOF, was reviewed in this column in the February 1983 issue. At the time, I mentioned several possible improvements that could be made to the program, and criticized some items. But overall, the review was upbeat, readers may recall, and in its form at the time, the program was generally a good one.

Anitek, the producers of the program, have since improved the program, partly based on some of my comments, and they have added some features that were in process at the time. I cannot see any of the former deficiencies

Can your VisiCalc® Sort?

Sort the rows
or columns of a
VisiCalc
spread sheet.

It can with VIS\Bridge/SORT™ from Solutions, Inc.

The sorted spread sheet still contains all the formulas and values from the unsorted original. Up to 5 rows or columns may be used as sort keys. Each key may be alpha or numeric and either ascending or descending.

VIS\Bridge/SORT is available for the Apple® II+ and III, the IBM PC™ and the TRS-80® I, II/16, and III.

\$89 plus \$4 shipping and handling from Solutions, Inc.

Order 802 229 0368. Box 989, Montpelier, VT 05602.

Mastercard and Visa. Dealer inquiries welcomed.

Also available: VIS\Bridge/REPORT™ for \$79 and

VIS\Bridge/DJ™ (Dow Jones) for \$445.

All VIS\Bridge products are trademarks of Solutions, Inc. VisiCalc™ is a trademark of VisiCorp. TRS-80® is a trademark of Tandy Corp. IBM PC™ is a trademark of IBM Corp. Apple® is a trademark of Apple Computers, Inc.

Date	Contribution
2/05/83	\$225.00
2/09/83	\$450.00
2/11/83	\$1,500.00
2/15/83	\$390.00
2/19/83	\$2,000.00
2/23/83	\$945.00

ORIGINAL
Jones,
Billings, J.
Mares, P.
Davis, N.
Franks, B.
Howard, R.

Date	Contribution
2/19/83	\$2,000.00
2/11/83	\$1,500.00
2/23/83	\$945.00
2/09/83	\$450.00
2/15/83	\$390.00
2/05/83	\$225.00

SORTED
BY \$ AMOUNT
Franks, B.
Mares, P.
Howard, R.
Billings, J.
Davis, N.
Jones, R.

existing now. For your interest, a few of the improvements are provided below. Anitek will upgrade your copy of the program if you are a present owner.

First (and I am so grateful for this!), the Keyboard Click feature can be cancelled without any problem now. And the "case of the missing cursor" that plagued me while using ZORLOF has been solved. By reducing the blink rate of the cursor, Anitek has brought the cursor back to life for my computer and anyone else who may have had the same problem.

Now for the improvements—they are considerable, and they are impressive. The first one will probably only be of interest to persons familiar with software manipulation—Anitek addresses the hackers. The feature is called "ZAP Processing," and permits you to patch directly to a machine language or data file. To implement the function you move the cursor to the file name in the directory, and press a control key and one other. The file then appears on the screen in "zap" format. Once on the screen you merely scroll to any place displayed,

and make your changes directly. Full instructions for this are provided in the manual. From time to time, Anitek proposes to issue zap changes, which anyone who feels capable may make directly to save time. Anyone who does not, and cannot do it otherwise may have this done by Anitek.

The second modification to ZORLOF allows form letters to be produced, with an infinite number of data records per file, an infinite number of data fields per data record, and a size allowance of up to 1000 characters per record—certainly an expansive capability! The ZORLOF disk includes two examples, a text file and a data file, that permits you to become fully familiar with the operation of this function within ZORLOF—and both examples are excellent ones, incidentally. In addition, the editing facilities for BASIC and EDTASM files have been enhanced to handle any length of program lines; you will recall that this was previously restrictive. Also, there is now an end-of-sentence character to prevent blank characters from appearing between sentences at the end

of a line.

Nine more printer drivers have now been added, bringing the total to over 50 different printer configurations (some printers are included twice in this total—once for mono-space and once for proportional spacing). It is difficult to think of a printer that may have been omitted, and if there are, there is a driver which may work—"undefined," which is the category that assumes no special or unique features.

Yes, the manual has been improved and expanded, too. One of my previous complaints, lack of an adequate Table of Contents, has been corrected with a vengeance. From an almost useless six-line list, the new Table of Contents now runs to three full pages, and every numbered paragraph with its title has now been listed. No Appendix, however; I would like to see this eventually included, or even a sorted alphabetized Table of Contents would help to locate information quickly for reference. The manual is still issued in the previous sturdy binder, but, as mentioned in the previous review, it will not lie flat, unfortunately—not an ideal situation for a reference manual. And if you try to make it lie flat while open, the binder will pop open.

One of the most useful improvements to the documentation has been provided by completely reprinting the Reference Card (front and back on a single card). This card, with Printer Commands on one side, and Editing Functions on the other, is now alphabetic in listing the commands and functions. This in itself now makes the card perfectly useful. But in addition, Anitek has added the page number reference for the function on the right hand side of each item. This is done with only one other word processor reference card that I know of, and you will agree is of great value to the user. As a bonus, and as in the previous release of this program, a plastic sleeve is included for the Reference Card, which adds to its inherent stiffness, and will help to keep it clean.

ZORLOF remains priced at \$69.95, with all warranties as previously reviewed still in effect. This probably puts this word processing program in the category of being more program for less

money than any other, considering all of the features that are offered for both the Model I and Model III. Anitek has paid attention to the problems of the first release and have responded quickly and positively—which rates them highly in my estimation.

A. A. Wicks
30646 Rigger Road
Agoura, CA 91301 ■

LETTERS TO THE EDITOR

continued from page 8

Response to One Letter

Although I no longer subscribe to your magazine, I thought you might like to know about the response to one letter you published from me in your April 1982 issue. The subject of my letter was how to utilize a teletype as a printer with a TRS-80.

In April, I received an inquiry from Michigan, in May, one from New York. In June, I got a phone call from another person in Michigan. Around that time I got a phone call from Texas.

Today (January 11, 1983), I got a letter from Porto Alegre, Brazil. The guy has a "Digitus-100 48K Level II." I never heard of it, but he says it works like a TRS-80 Model I Level II. He has an Olivetti BCN TTY which he is trying to use as a printer.

All those who furnished a return address have been sent a reply from me.

I enjoyed your magazine very much, and I thought you might want to know about the response to the only letter I ever sent you.

R. H. Long
2106 Valleywood Dr.
Carrollton, TX 75006 ■

Printer Problems

I own the TRS-80 Model I Level II with cassette, and a Super Brain EMAKO 20 printer. A problem has developed that you might be able to help me with.

After using the machine for three years with no trouble, the LPRINT command suddenly developed a bug. After typing in or cassette loading a program using this command and running it, I get a syntax

error on any line using LPRINT. To correct the problem I simply press ENTER and re-run the program. After doing this for all lines in which the LPRINT command appears, the program works.

After fighting it for some time, I tried cooling the unit with a fan. This seems to help eliminate the bug for that session. However, reloading any program with LPRINT commands, after the machine has been running for a while, the trouble starts again.

My question is: is the CPU at fault, or what?

One other question: I see many interesting programs in magazines, but they are for disk operation. Is there some simple conversion for getting around the OPENs and CLOSEs in these programs?

In asking this question, you can see that I am not an experienced programmer.

Any help will be appreciated.

Larry Hoffman
1-15-10 Toyotama-miniami
Nerima-ku, Tokyo 176

The LPRINT problem (which is certainly unusual) sounds more like a RAM error than anything else. We would suggest that you investigate the low RAM (in the keyboard case) first.

As for your other question, there is no simple way to change disk programs to work with cassettes. First, it depends on the type of disk access being used. Random files (if the OPEN statement says "R") would be impossible. Sequential files (OPEN "I" or "O") could possibly be converted, provided that you were not putting too much data on the cassettes.

Simply delete the OPEN statements and insert something that tells you to ready the cassette. Then, change any INPUT #1 or PRINT #1 statements to PRINT #-1 or INPUT #-1. Finally, delete the CLOSE statements, or change them to simply print a message to remove the cassette.

BEGINNER'S CORNER

continued from page 5

and converts it into an intermediate language.

This intermediate language is then interpreted by a software interpreter, which runs on your

machine (P-code interpreter). As is the case with PASCAL, the intermediate language is the "MACHINE CODE" for some hypothetical computer. In essence, the hypothetical machine is a simulation of someone's vision of the ideal machine.

The last classification is called (again) a SOFTWARE interpreter, but there is a difference. This version is like your BASIC interpreter. The source is directly interpreted by your machine to be run on your machine and not some imaginary machine. This interpreter is designed to take full advantage of the options your machine contains. It's said to be MACHINE DEPENDENT.

The second type of interpreter seems to be becoming very popular. It has several advantages over the kinds that run BASIC. As you are probably aware, new and better machines are coming off the assembly line every day. This kind of development is driving software writers batty. They can't seem to keep up. For every new machine that comes out, software must be provided. This means that the software has to be converted to the machine code of the new machine.

A case in point is the IBM/PC machine. The IBM uses an 8088 processor. This processor is upward compatible with the 8080. That means that machine code written on a 8080 can conceivably run without change on the 8088. Certain considerations, such as changes in input and output procedures, have to be re-worked but, on the whole, most programs will be able to run.

An example of the kinds of mistakes that can happen also occurred with the IBM/PC. One of those famous makers of FORTRAN for the TRS-80 cross-compiled (compiled the same source for a different machine) the source for the IBM. One of those small differences between the TRS-80 and the IBM/PC is that the IBM has a little more memory on board (48K versus up to 512K IBM).

Unfortunately, the source was not changed to take advantage of the new facilities. So, a user of this fantastic machine writes a

continued on page 20

PIPS: PARTS INVENTORY PLANNING SIMULATION

Dennis P. Avola

ABSTRACT

Discuss field service inventory, and you will get at least two points of view. This classical discussion often occurs between the field service technician and the parts depot manager.

Field service technicians historically over-stock their trunk supply of "spare" parts to avoid costly incomplete service calls because of lack of parts. These service technicians, whose primary objective is to satisfy the customer, can't comprehend others who attempt to reduce, reassign, and minimize service parts stocking levels. Imminently depot managers are often cited as the "culprits" by service technicians.

Those who manage and control national and regional service parts depots are accustomed to claims of inventory stock-outs and misallocments by service technicians. These depot managers have financial performance objectives to reduce the over all cost of inventory like; the cost of money invested in inventory, obsolescence, spoilage, insurance costs and the maintenacnce and ordering costs of securing the inventory.

The optimal solution is a balance between both points of view: The cost of inventory stock versus the penalty cost of an inventory stock-out. As depicted in exhibit A, the optimum stock level occurs when the total cost curve extends to its lowest point. Inventory levels to the left asnd right of the optimum stocking level are considered under and over stocked, respectively.

One factor which significantly affects the optimum stocking level is the lead time to replenish a service part. Exhibit B illustrates the impact of different lead times and parts usage on the optimum inventory level. If the lead time to replenish a particular part increases, and the optimum stocking level is not adjusted accordingly, then an under-stock situation is created.

Consequently, the field service technician obsorbs the costly expense of a incomplete service call due to an inventory stockout.

This paper introduces a field service inventory planning program that addresses both the service technicians concern about part stock-outs, and the depot managers objectives to minimize total inventory costs.

INTRODUCTION

"PIPS" (Parts Inventory Planning Simulation) is a field service inventory planning and control program for the service depot manager. "PIPS" calculates the optimum inventory stock level based on the following input factors:

- Parts usage (per specified time);
- Part cost (manufacturing/purchase);
- Lead time to replenish;
- Penalty cost of a part stock-out.

Parts may be grouped by product category or A,B,C classifications. "PIPS" is user friendly and menu-driven for easy operation.

MENU

(1) BUILD A NEW PARTS DATA FILE

This menu selection begins by prompting the user to enter some pertinent information about the parts data file. This information as depicted in screen 1 is germane to the entire data file. The user is then prompted to enter for each part number the necessary usage, cost, product group, and lead time information. The part number man consist of 10 alpha/numeric characters. PIPS can accomodate a maximum of 300 part items

PARTS INVENTORY PLANNING SIMULATION MODEL
OPTIMUM STOCKING LEVEL

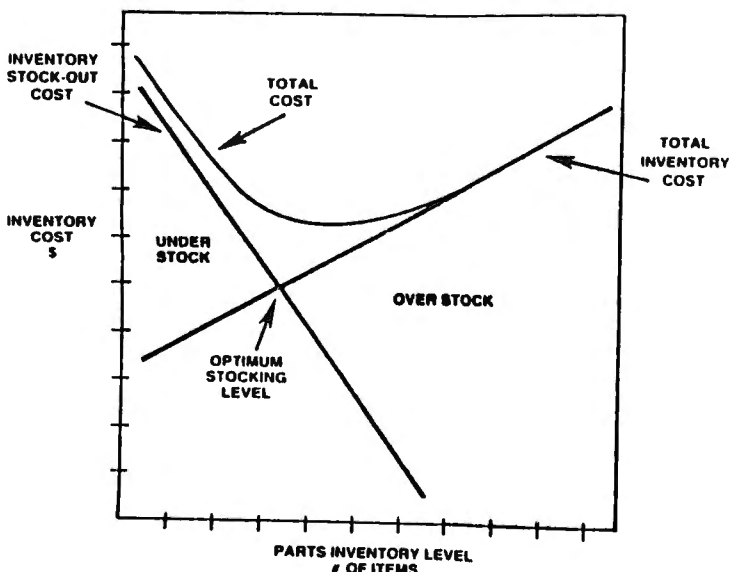


Exhibit A

IMPACT OF PARTS USAGE AND LEAD TIME TO REPLENISH ON
OPTIMUM STOCKING LEVEL

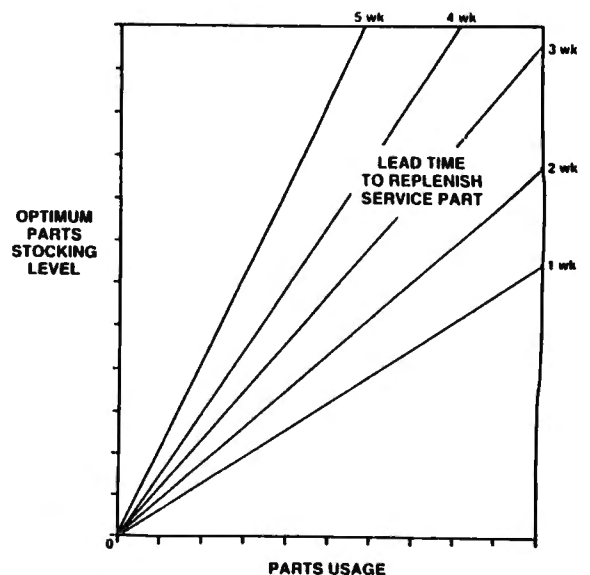
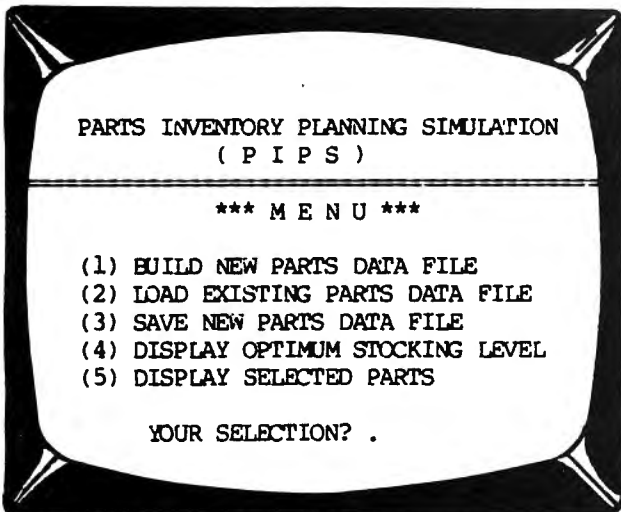
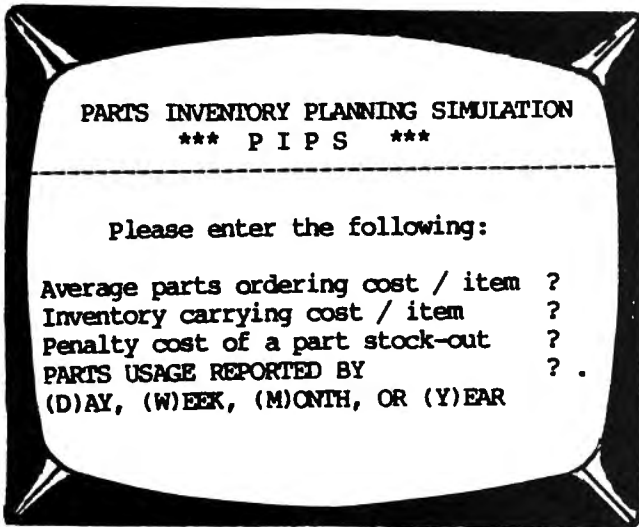


Exhibit B

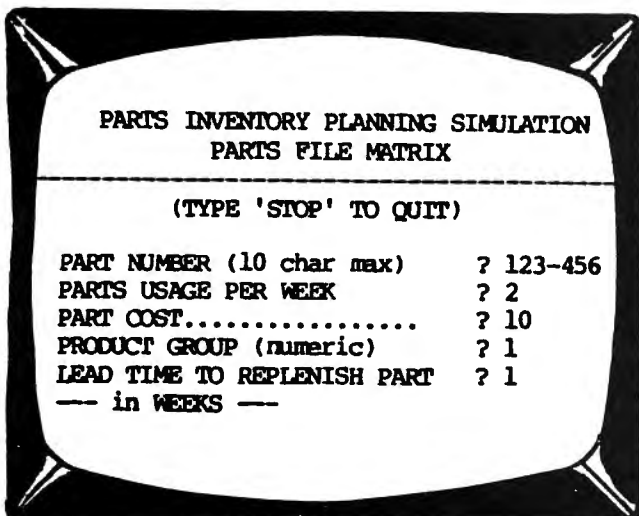


Screen #1

depending on your microcomputer's RAM (memory) capacity. A change routine allows for corrections to the parts data file. (Note: the program automatically adjusts inconsistent information relating to parts usage and lead time to replenish.)



Screen #2



Screen #3

(2) LOAD EXISTING PARTS DATA FILE

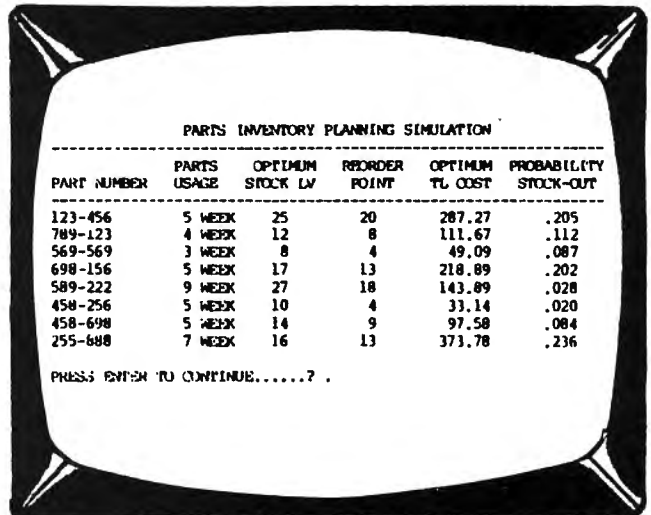
This menu selection loads an existing parts data file stored on disk as specified by the user.

(3) SAVE NEW PARTS DATA FILE

This menu selection saves a new parts data file on disk as specified by the user.

(4) DISPLAY OPTIMUM STOCKING LEVEL

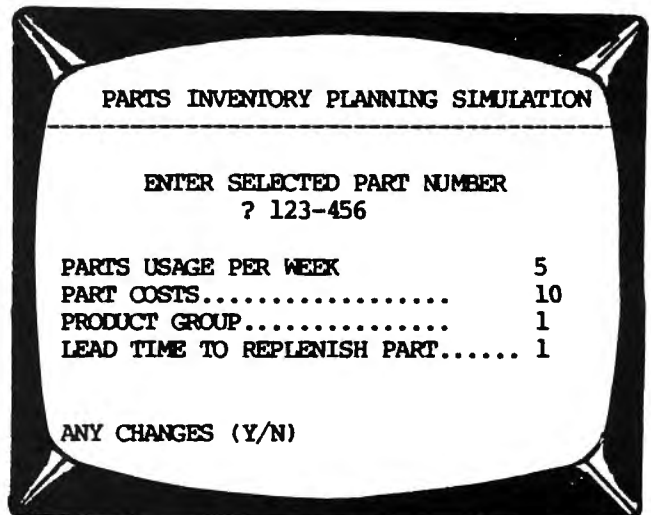
This menu selection displays on the CRT or line printer the optimum stocking level, reorder point, dollar value of the optimum stock level and the probability of a part stock-out (screen 4) for each part number on the file. PIP S calculates the above output for each part number before displaying it on the screen. (Note: Display time may vary per part number.)



Screen #4

(5) DISPLAY SELECTED PARTS

This menu selection allows the user to review, edit, or change any one part number on file (screen 5). Additionally, the user or the calculations of the optimum stock level (screen 6) for the part number selected (screen 5).



Screen #5

PART NUMBER 123456				
STOCK LEVEL	INVENTORY COST	STOCK-OUT COST	TOTAL INV COST	PROBABILITY STOCK-OUT
0	0.00	131.62	131.62	0.723
1	10.00	131.62	141.62	0.723
2	20.00	66.91	86.91	0.368
3	30.00	25.35	55.35	0.139
4	40.00	7.56	47.56	0.042
5	50.00	1.85	51.85	0.010
6	60.00	0.38	60.38	0.002
7	70.00	0.07	70.07	0.000

Screen #6

CONCLUSION

The net gain from the Parts Inventory Planning Simulation (PIPS) program is increased field service profitability and improved customer satisfaction. By establishing an optimum stock level and reorder point for all your field service parts, you will generate significant material and labor cost savings. Computer-aided inventory planning models like PIPS should assist service depot managers in achieving their financial objectives and minimizing the occurrence of inventory stock-outs.

PROGRAM NOTES

PIPS was written on a TRS-80 Model III (TRSDOS) microcomputer system. The program requires a minimum of 32K RAM and one disk drive. For additional information write to the author.

PROGRAM LISTING

```

10 REM PARTS INVENTORY PLANNING SIMULATIONS "PIPS"
20 REM BY DENNIS P. AVOLA
25 CLEAR 3000: DIM A(5,300), PA(200), TC(200), S(200), C(200),
  US(300), P(200), PS(300)
30 E$="% % ## % % ## ## ## ##"
  #.###"
32 E1$=" ## ## ##.#"
35 E2$="% % % ##.# ##.# ## ##"
  ##.# #.###"
40 X$=" ## ##.## ##.## ##.##"
  #.###"
45 X1$=" ## ##.## ##.## ##.##"
  #.###"
50 E3$=" ## ##"
  ##.#.##"
90 GOTO 5000
100 CLS: PRINT TAB(15)"PARTS INVENTORY PLANNING SIMULATION"
110 PRINT TAB(22)"*** P I P S ***"
120 PRINT STRING$(63,"-")
125 PRINT: PRINT TAB(17)"Please enter the following:": PRINT
130 PRINT TAB(8)"(1) Average parts ordering cost / item";:
  INPUT " "; C1
135 PRINT TAB(8)"(2) Inventory carrying cost / item";: INPUT
  " "; C2

```

```

140 PRINT TAB(8)"(3) Penalty cost of a part stock-out";: INPUT
  " "; C3
141 C4=C1+C2
145 PRINT TAB(8)"(4) PARTS USAGE REPORTED BY": PRINT TAB(12)
  "(D)AY, (W)EEK, (M)ONTH, or (Y)EAR": PRINT@624,: INPUT Z$
147 GOSUB 9000
150 REM INPUT MATRIX
152 FOR I=1 TO 300
155 CLS: PRINT TAB(15)"PARTS INVENTORY PLANNING SIMULATION"
160 PRINT TAB(23)"PARTS FILE MATRIX": PRINT STRING$(63,"-")
165 PRINT TAB(20)"(TYPE 'STOP' TO QUIT)": PRINT
170 PRINT TAB(12)"PART NUMBER (8 char max)": PRINT@362,:
  INPUT P$(I)
175 IF P$(I)="STOP" THEN E=I-1: GOTO 5000
180 PRINT TAB(12)"PARTS USAGE PER "; Z$,: PRINT@426,:
  INPUT A(1,I)
185 GOSUB 1500
190 PRINT TAB(12)"PART COST.....": PRINT@490,:
  INPUT A(2,I)
195 PRINT TAB(12)"PRODUCT GROUP (numeric)": PRINT@554,:
  INPUT A(3,I)
200 PRINT TAB(12)"LEAD TIME TO REPLENISH PART": PRINT
  TAB(12)"--- in "; US(I); "S ---": PRINT@618,: INPUT A(4,I)
205 GOSUB 1700
210 CLS: PRINT TAB(15)"PARTS INVENTORY PLANNING SIMULATION"
215 PRINT TAB(20)"CHANGE ROUTINE": PRINT
  STRING$(63,"-")
220 PRINT: PRINT TAB(15)"(1) PART NUMBER"; TAB(34)P$(I)
230 PRINT TAB(15)"(2) PARTS USAGE"; TAB(33)A(1,I); " PER
  "; US(I)
240 PRINT TAB(15)"(3) PART COSTS"; TAB(33)A(2,I)
250 PRINT TAB(15)"(4) PRODUCT GROUP"; TAB(33)A(3,I)
255 PRINT TAB(15)"(5) LEAD TIME "; TAB(33)A(4,I); " "; US(I); "S"
260 PRINT: PRINT@896,"ANY CHANGES (Y/N)": INPUT A$:
  IF A$<>"Y" AND A$<>"N" THEN 260
265 IF A$="N" THEN 295
270 PRINT@896,"CHANGE ROW NUMBER ";: INPUT G: IF G<1 OR G>5
  THEN 270
275 IF G=1 THEN PRINT@925,"CHANGE TO";: INPUT R$: GOTO 290
280 PRINT@925,"CHANGE TO";: INPUT R
285 A((G-1),I)=R: GOTO 210
290 P$(I)=R$: GOTO 210
295 NEXT I
300 REM CALCULATIONS
305 GOSUB 1000
310 FOR I = 1 TO E
315 S(I)=0: C(I)=0
320 IF U=5 AND A=1 THEN GOSUB 8100
321 IF U=5 AND A=2 THEN GOSUB 1010
322 M=A(4,I)/(1/A(1,I))
325 M5=-M*(LOG(2.71828)/LOG(10))
330 FOR X=0 TO 200
340 PA(X)=0: IF X=0 OR X=1 THEN K1=0: J=0: GOTO 390
350 FOR J=0 TO X-1
360 IF J=0 OR J=1 THEN K1=0: GOTO 375
365 IF J=2 THEN K1=.3010: GOTO 375
370 GOSUB 600
371 K1=G4
375 P0=(M5+(J*(LOG(M)/LOG(10))))-K1
376 P(X)=10>P8
380 PA(X)=P(X)+PA(X): NEXT J: GOTO 405
390 P(X)=10>M5: PA(X)=P(X): GOTO 405
395 P(X)=10>(M5+(LOG(M)/LOG(10))): PA(X)=PA(X)+P(X)
405 PA(X)=1-PA(X)

```

continued on page 49

BeaLin Corporation

OFFERS YOU...

TRAX-SW

80/100 (Revised Version)

Now you can retain your *TRS-80 Block graphics capability while adding all of the features of **Grafrax-plus. Such as...

- Underline Mode
- Subject/Superscript
- Line Drawing Graphics

TRAX-SW	\$99.95
GRAFTRAX-PLUS (MX 80/100 Specify)...	\$69.95
GRAFTRAX 80	\$65.00
GRAFTRAX 80 (Modified for MX 100)	\$89.95

(Not available for serial MX-80 Printers.)

Plug in board supplied with or without **Grafrax-Plus.

LNW MODEL II 125 K/4MHZ OPERATION \$1995.00

Supports Model I systems and CPM. 5-8 Double density controller supports Hi-Res B&W and Color TRS-80 Model I compatible; supplied with a NEC JB 1201 Hi-Res green phosphor monitor, DOSPLUS 3.4 and LNW Basic (Special prices for package purchases - call for quotation)

LNW EXPANSION SYSTEM \$369.95

TEAC THIN LINE DISK DRIVES 40TK/SS/DD \$260.00

(one year warranty/price includes case and power supply)

PRINTER SWITCHES \$129.95

Manual/Software Selectable-Mod I, III...\$119.95 Mod II, XII, & XVI.

Switch between two printers, use your computer to produce draft copies on one printer. Then make final copies on letter quality printer without switching cables.

(MODEL II, XII, & XVI requires NO cable modification.)

VERBATIM DATA LIFE DISKETTS (Box of 10) \$26.95

PAGE-IT \$29.95

A disk utility that prints basic programs in easily readable format.

- COMPATIBLE WITH MOST OPERATING SYSTEMS
- HIGHLIGHTS REM STATEMENTS
- INDENTS CODE — LINE NUMBERS STAND OUT
- UNPACKS BASIC CODE
- DOUBLE SPACE OPTION
- SUPPORTS SPECIAL OPTIONS OF 15 PRINTERS

HAYES SMARTMODEM	\$229.00
1200 BAUD VERSION	\$599.00

(Call or write for information and prices.)

USE OUR DATA LINE TO BROWSE THROUGH THE MANY ITEMS WE HAVE AVAILABLE. YOU MAY PLACE ORDERS OR HAVE SPECIFIC QUESTIONS ANSWERED. DATA LINE FORMAT IS 300 BAUD, 7 BIT, NO PARITY. SATISFY ALL OF YOUR COMPUTING NEEDS, COMPLETE SYSTEMS, DISK DRIVES, PRINTERS, EPSON RIBBONS & REPACKS, PAPER PRODUCTS, PLUS MUCH MORE. CALL OR WRITE FOR FREE CATALOG.

Software shipping and handling — \$3.50

PHONE VOICE (301) 490-2744 MODEM (301) 730-2229

*Trademark of Tandy Corp.
**Trademark of Epson

BeaLin Corp. 9335 Old Scaggsville Rd. Laurel, Md. 20707

VISA OR MASTERCARD ACCEPTED — DEALERS INQUIRIES WELCOME

PRACTICAL BUSINESS PROGRAMS

S. M. Zimmerman and L. M. Conrad

Month #7: BALANCE the check book

Copyright© 1983 Zimmerman & Conrad

There are two utility programs included in our general ledger package the average business person will find very useful. The task of balancing a check book is no fun. BALANCE, this month's program, makes the job a lot easier.

Next month we will publish MILES, a program designed to aid the business person when calculating automobile costs. Next month is the last of this series detailing our general ledger package. For those wanting to look up when we published the previous parts our publication schedule is listed below:

No	Month	Program
1.	GLMENU (Controls use of routines).
2.	NAME & START (Initializes chart of accounts and firms name).
3.	TRANSACTION (Inputs monthly transactions).
4.	CPA (Produces Balance sheet & Income and Expense statements, Profit & Loss statement).
5.	UPDATE & YEAR (Changes chart of accounts and performs year end closing tasks).
6.	TRIAL (Produces trial balance).
7.	BALANCE (Balances check book).
8.	MILES & MOVE (Calculates gas mileage and moves files between disks).

We will now detail the operation of our check balancing routine.

RUNNING BALANCE

This program has proven to be a most useful program. It is designed to aid in balancing a check book. The first few questions you may expect to encounter are shown below with some dummy numbers we created as answers to illustrate the program.

THIS PROGRAM IS AN AID IN RECONCILIATION OF YOUR CASH ACCOUNT WITH YOUR BANK STATEMENT

ENDING BALANCE OF CHECK BOOK? 300.00

BANK'S BALANCE FOR PERIOD? 1000.00

The idea behind the act of balancing a check book is to get the adjusted bank's balance to equal your balance or to get the adjusted check book balance to equal the bank's balance.

The next question encountered is relative to uncleared checks. Again we show the questions with some dummy answers.

INPUT ITEM NO. & NOT CLEARED CHECK 0,0 WHEN COMPLETE

ITEM 1? 100,89.77

ITEM 2? 101,99.63

ITEM 3? 0,0

After you have typed in 0,0 the computer will summarize the data you have just typed.

NOT CLEARED CHECKS

NO	ID NO	ITEM	SUM
1	100	89.77	89.77
2	101	99.63	189.40

NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3 TO ADD, -4 TO CONTINUE?-

The action menu you see above follows each of the sets of input data in this program. You can correct any item at this time if you wish. It is also possible to make corrections later if necessary.

We will assume you are happy with the above and have typed -4 to complete the statement and move on to the next item, which is as follows:

INPUT ITEM NO & NOT CLEARED DEPOSITS 0,0 WHEN COMPLETE?
ITEM 1 ? 0,0

We answered 0,0 to the first item above, which means there were no uncleared deposits to adjust for.

In order to not get tied up in the minute details of this program, we will assume there are no additional corrections, and we shall rapidly move through the program by showing the questions and a set of possible answers. All possible outcomes will not be covered in order to keep the review as simple as possible.

NOT CLEARED DEPOSITS

NO	ID NO	ITEM	SUM
1	0	0	0

NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3 TO ADD, -4 TO CONTINUE? -4

INPUT ITEM NO. & NOT CLEARED DEPOSITS 0,0 WHEN COMPLETE
ITEM 1? 0,0

NOT CLEARED DEPOSITS

NO	ID NO	ITEM	SUM
1	0	0	0

NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3 TO ADD, -4 TO CONTINUE? -4

INPUT ITEM NO. & NOT RECORDED CHECKS 0,0 WHEN COMPLETE
ITEM 1? 0,0

NOT CLEARED DEPOSITS

NO	ID NO	ITEM	SUM
1	0	0	0

NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3 TO ADD, -4 TO CONTINUE ? -4

INPUT ITEM NO. & NOT RECORDED DEPOSITS 0,0 WHEN COMPLETE
ITEM 1? 0,0

NOT CLEARED DEPOSITS

NO	ID NO	ITEM	SUM
1	0	0	0

NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3 TO ADD, -4 TO CONTINUE? -4

INPUT ITEM NO. & ERRORS AS + INFAVOR OF BANK -
 OTHERWISE 0,0 WHEN COMPLETE
 ITEM 1? 0,0
 ERRORS AS + INFAVOR OF BANK - OTHERWISE 0,0 WHEN COMPLETE
 NO ID NO ITEM SUM
 1: 0 0 0
 NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3 TO ADD,
 -4 TO CONTINUE? -4

BOOK START 300.00
 BANK 1000.00
 ADJUST BOOK 489.40
 DIFFERENCE (BANK- BOOK) 510.60

ACTION MENU
 P PRINTER
 RS RESTART
 R RETURN TO GLMENU?-

As you can see, the bank's result does not agree with the adjusted check book balance. This is a usual condition for many persons. Even when using the computer, errors are made.

The next step is to search for errors. Let's say we found a few checks which have not cleared. We do not have to start all over. If we elect to answer the last question with RS, then the next thing on the screen will be the following:

RECYCLE MENU
 1 ORIGINAL DATA
 2 NOT CLEARED CHECKS
 3 NOT CLEARED DEPOSITS
 4 NOT RECORDED CHECKS
 5 NOT RECORDED DEPOSITS
 6 ERRORS
 SELECT A NUMBER?

The selection of any of the above puts us back to that part of the program. You must now walk through the data already input with an ENTER until you get a listing of the data set you requested. At this point, you may make whatever corrections you wish. When you type -4 and hit the ENTER key, you will again be returned to the summary routine.

EXAMINING THE PROGRAM

Line 10 starts by clearing some space for string variables. Line 20 prints the title of the program and line 30 sets the dimension which limits the number of adjustments which can be made. If you have more than 250 transactions to input you will have to change the value of D in line 30. We set the value for a computer with 16k of memory and have never had the need to change the number.

Lines 40 and 50 define the titles to be used for the various parts of the program, and line 60 starts the input of the data for the program. The input of data continues through line 310.

In line 320 the balance is calculated. We used double precision variables for this task. Lines 320-380 print out the results, an action menu, and then routes the flow of the program in a manner as directed by the menu.

Line 385 is for those who have an operating system with a printer line counter. It may be left off for those without such a system.

The remaining steps are designed to output the results to the printer.

```

10 CLS: CLEAR 1000: REM "BALANCE"
20 KK=0: PRINT "THIS PROGRAM IS AN AID IN RECONCILIATION OF
YOUR CASH ACCOUNT
WITH YOUR BANK STATEMENT"
30 D=250: DIM X$(D,5), X%(D,5): REM 250 DIMENSION LIMIT
40 A$(1)="NOT CLEARED CHECKS": A$(2)="NOT CLEARED DEPOSITS":
A$(3)="NOT RECORDED CHECKS"
50 A$(4)="NOT RECORDED DEPOSITS": A$(5)="ERRORS AS + IN FAVOR
OF BANK - OTHERWISE"
60 INPUT "ENDING BALANCE OF CHECK BOOK": EB#
70 INPUT "BANK'S BALANCE FOR PERIOD": BB#: IF KK=1 THEN 320
80 FOR J=1 TO 5
90 PRINT "INPUT ITEM NO. & ": A$(J): PRINT " 0,0 WHEN
COMPLETE"
100 I=0
110 I=I+1: PRINT "ITEM "; I: INPUT X$(I,J), X%(I,J): IF X$(I,J)>0
THEN 110
120 PRINT A$(J): N(J)=I-1: K=0
130 PRINT "NO ID NO ITEM SUM": A$="### ###"
#####.## #####.##"
140 S#(J)=0: FOR I=1 TO N(J): S#(J)=S#(J)+X$(I,J)
150 K=K+1: PRINT USING A$; I, X$(I,J), X%(I,J), S#(J): IF K=13
THEN K=0: INPUT "ENTER TO CONTINUE": DU$
160 NEXT I
170 K=0: INPUT "NO TO CHANGE, -1 TO INSERT, -2 TO DELETE, -3
TO ADD, -4 TO CONTINUE": KZ
180 IF KZ=-1 THEN 240
190 IF KZ=-2 THEN 280
200 IF KZ=-4 THEN IF KK=1 THEN 320 ELSE 310
210 IF KZ=-3 THEN I=N(J): GOTO 110
220 IF KZ<1 THEN 170
230 INPUT "ITEM NO, AMOUNT": X%(KZ,J), X#(KZ,J): GOTO 130
240 INPUT "NO INSERT, ITEM NO, AMOUNT": V, CN%, Z#: N(J)=N(J)+1
250 FOR I=N(J) TO V STEP -1
260 X$(I,J)=X$(I-1,J): X%(I,J)=X%(I-1,J)
270 NEXT I: X#(V,J)=Z#: X%(V,J)=CN%: GOTO 130
280 INPUT "NUMBER TO DELETE": V: N(J)=N(J)-1
290 FOR I=V TO D: X$(I,J)=X$(I+1,J): X%(I,J)=X%(I+1,J): IF
X$(I,J)=0 THEN 130
300 NEXT I: GOTO 130
310 NEXT J
320 BA#=EB#+S#(1)-S#(2)-S#(3)+S#(4)-S#(5):
Z$="###.###.###.###.##"
330 PRINT "BOOK START ";: PRINT USING Z$; EB#: PRINT "BANK =";
340 PRINT USING Z$; BB#: PRINT "ADJUSTED BOOK =";: PRINT USING
Z$; BA#: PRINT "DIFFERENCE (BANK-BOOK)":: PRINT USING Z$; BB#-BA#
350 PRINT "ACTION MENU": PRINT " P PRINTER": PRINT "
RS RESTART": INPUT " GL RETURN TO GLMENU": K$
360 IF K$="GL" THEN LOAD "GLMENU", R
370 IF K$="RS" THEN 450
380 IF K$<<"P" THEN 350
385 INPUT "AUTO LINE COUNTER (Y/N)": LC$: IF LC$="Y" THEN
CMD"FORMS(T)"
390 INPUT "TITLE, DATE": T$, D$: LPRINT T$, D$
400 LPRINT "BOOK START ";: LPRINT USING Z$; EB#: LPRINT "BANK
=";: LPRINT USING Z$; BB#
410 LPRINT "ADJUSTED BOOK =";: LPRINT USING Z$; BA#: LPRINT
"DIFFERENCE (BANK-BOOK)":: LPRINT USING Z$; BB#-BA#
420 FOR J=1 TO 5: LPRINT A$(J)

```

```

430 S#(J)=0: FOR I=1 TO N(J):S#(J)=S#(J)+X#(I,J): LPRINT
USING A$;I,X$(I,J),X#(I,J),S#(J):NEXT I
440 NEXT J:GOTO 350
450 CLS:PRINT "RECYCLE MENU":PRINT " 1 ORIGINAL
DATA": PRINT " 2 NOT CLEARED CHECKS"
460 PRINT " 3 NOT CLEARED DEPOSITS": PRINT " 4
NOT RECORDED CHECKS": PRINT " 5 NOT RECORDED
DEPOSITS"
470 PRINT " 6 ERRORS"
480 INPUT "SELECT A NUMBER";N:KK=1
490 J=N-1:IF N>1 AND N<7 THEN 90
500 GOTO 60

```

SUMMARY

The check balancing routine reviewed this month is one of the most used programs in our general ledger set. This program aids greatly the task of balancing your checkbook with the bank's results. The manner in which we have written this program is such that you need not use our general ledger program to find this program of value. It is a powerful addition to the general ledger package.

Steven M. Zimmerman, Ph.D.
College of Business and Management Studies
University of South Alabama
Mobile, Alabama 36688

Leo M. Conrad
Imagineering Concepts
P.O. Box 9843
Mobile, Alabama 36691-0843 ■

BEGINNER'S CORNER

continued from page 13

program that should fit on the PC and finds, to his chagrin, that his compilation will crash due to lack of memory. The compiler put certain elements too close together, and at some point the STACK and DATA collided (don't worry about what a stack is, just know that the only good collision is between you and your money). That's not a good thing to do.

These kinds of problems don't happen only with machine language programs. High level languages may be easier to write and debug. They should, therefore, be easier to transport (microcommuting—get it?). Right? The answer is "not necessarily." It depends on how much software and how much time you have. The job can be done, but it's a problem. Try converting from Model 3 BASIC to Color BASIC and you'll get the general idea.

One way to get out of this problem is to have a situation where all machines are said to be one machine (the ultimate machine) able to run one standard language. That is what the developers of PASCAL tried to do.

They created a two-level system where you, the programmer, can write a program one time and it should be able to run on any other machine equipped with a P-code interpreter. As it turns out, someone forgot to design and DICTATE what would be the "standard" PASCAL.

Because every machine has different capabilities,

each implementor of PASCAL added certain features to his language. Options such as graphics were not included as part of the standard set of commands. As a matter of fact, Pascal wasn't supposed to become a real language at all but was developed to serve as a learning tool (see MODULA by the inventor of PASCAL, Nicklas Wirth). What this resulted in was a standard that ends up being a subset to the PASCAL for each machine.

Other language developers are trying to accomplish something similar. The Forth community (they like being called a "community." It gives them a sense of identity. They even have reunions and meetings and things to discuss changes in the language everyone will accept in the future.) is also trying to develop a transportable language (Pascal doesn't have a community, just a large following and a good financial base).

Forth is called an interpretive compiler and is very different from any other language I've come across. There are times when words are interpreted (i.e. result in action being taken) and times when the language can be extended to include new user-defined words (compiled).

It's very easy (and fun) to program in but can be difficult to read, especially if the programmer wants to be difficult to read. Complaints like this can be made for any language but Forth and APL seem to be favorites for this kind of harangue (I always wanted to use that word).

The users of this format refer to language as the FOURTH generation language. The reason it's spelled FORTH is because Charles H. Moore, the inventor of FORTH, didn't have enough room on his IBM machine for the 6th character. The language is said to be 10 to 20 times faster than BASIC. This puts it at the same speed as compiled PASCAL.

Ultimately, everyone is trying to accomplish the same ends. They want a language that is easy to use, easy to write, easy to maintain and modify. The reason interpreters were invented at all was for programs that would be run a few times and then changed to suit new needs.

The big advantage of interpreters boils down to the catch word of microcomputing, and that is "INTERACTIVE." Interactive means that the machine works with you by asking questions as you develop your program and/or flags errors as they come up. This is supposed to make programming easier and faster. Just as quickly as the errors come up, you can change the code and make corrections.

Well, we live in an imperfect world, and as you can see, we have quite a few imperfect solutions (FORTH sure looks good if you like reverse polish notation ... 4 3 + = 7). It seems that everyone is trying to get to the ultimate user friendly machine (look at LISA—the Japanese have a copy called the MONA) and the ultimate user friendly language. I hope they keep trying. I love all the new toys that result.

That about covers this topic. I hope you feel encouraged by all this, because it means that they just might succeed. Keep hacking, and most of all, have fun.

Spencer Koenig
153-27 73 Avenue
Flushing NY 11367 ■

POCKET COMPUTER CORNER

Steven M. Zimmerman, Ph.D. and Leo M. Conrad

This Month: The Simplex Method Of Linear Programming

Copyright© 1983 Zimmerman and Conrad

The Simplex method of linear programming is a technique for finding the values of decision variables that will maximize a profit function or minimize a cost function. This pocket computer program is designed to solve for a maximization problem with up to ten variables and four constraints, or five variables and nine constraints. The exact limitation for a maximization problem may be calculated using the following relationship:

$$(\text{Variables} + \text{constraints} + 3) * (\text{constraints} + 1) \text{ less than } 107$$

Because of the limited memory (1.9K) of your pocket computer, the program was designed assuming you have some knowledge of the Simplex method. You must be able to set up the Simplex tableau from the objective function and constraints. Both minimization and maximization problems can be handled, but you must know how each is set up. In addition, you must learn how to read the results in the Simplex tableau. Included in the decision will be sections on how to set up the two types of problems and how to identify the answers. The trade-off was between capacity and ease of operation. In this program, the option selected was to maximize the amount of capacity available.

SETTING UP A MAXIMIZATION PROBLEM

A typical maximization problem is as follows:

$$\text{Maximize } F(X1, X2, X3, X4) = 12 * X1 + 16 * X2 + 17 * X3 + X4$$

Subject to the constraints:

$$\begin{aligned} X1 + 2 * X2 + 3 * X3 + 4 * X4 &\leq 120 \\ 8 * X1 + 16 * X3 &\leq 222 \end{aligned}$$

and built into the program solution procedure, all values of X must be greater than or equal to zero.

When working with a maximization problem, all constraints must be of the "less than or equal to" a limit. If you have a constraint such as:

$$X1 + 5 * X2 \geq 555$$

you must multiply both sides of the inequality by negative one to change the direction of the constraint as follows:

$$-X1 - 5 * X2 \leq -555$$

Once the constraint has been set up in this manner, you can proceed with the problem.

The set up of the sample problem as a Simplex tableau starts as follows:

		Variables				Constraints			
		X1	X2	X3	X4	S1	S2		
Constraints	S1	:	:	:	:	1	0	:	:
	S2	:	:	:	:	0	1	:	:
	coefficients					0	0		0
		objective function							

The rows are labeled S1 and S2. S1 is for slack variable one and is associated with the first constraint. S2 is for slack variable two and is associated with the second constraint. The one in the S1 row S1 column is placed there automatically, as is the one in the S2 row and S2 column. The positions where you see zeros always are given these values at the beginning of a set up.

The zero in the far right position is the current value of the objective function. The value starts at zero and changes as changes in the value of the decision variables X1, X2, X3, and X4 change.

The next step is to add the value of the coefficients of the objective function to the tableau. The objective function is:

$$F(X1, X2, X3, X4) = 12 * X1 + 16 * X2 + 17 * X3 + X4$$

The coefficients are 12, 16, 17, and 1. The values are changed to negative quantities and add to the tableau as shown:

		Variables				Constraints			
		X1	X2	X3	X4	S1	S2		
Constraints	S1	:	:	:	:	1	0	:	:
	S2	:	:	:	:	0	1	:	:
	coefficients					0	0		0
		objective function							

If any of the coefficient had been negative they would have become positive numbers as the results of this step. These values and the two zeros to the right on the third line are called indicators. The computer will continue to try to make improvements in the solution as long as any of these indicators are negative. The last position in line three is not an indicator, it is the value of the objective function.

The constraints are:

$$\begin{aligned} X1 + 2 * X2 + 3 * X3 + 4 * X4 &\leq 120 \\ 8 * X1 + 16 * X3 &\leq 222 \end{aligned}$$

Adding the first constraint to the tableau, the following is the result:

		Variables				Constraints			
		X1	X2	X3	X4	S1	S2		
Constraints	S1	1	2	3	4	1	0	120	:
	S2	:	:	:	:	0	1	:	:
	coefficients					0	0		0
		objective function							

The values added to line one are a 1, 2, 3, 4, and 120, corresponding to the first constraint. Adding the second constraint results in:

	Variables				Constraints			
	X1	X2	X3	X4	S1	S2		
Constraints	S1	: 1	: 2	: 3	: 4	: 1	: 0	: 120
	S2	: 8	: 0	: 16	: 0	: 0	: 1	: 222
		-12	-16	-17	-1	0	0	0
		coefficients						
		objective function						

The coefficients of X2 and X4 are zero in the last constraint because they are not there. In the tableau above there are two zeros in the location referring to these two variables in the S2 row.

This completes the construction of the tableau. Removing the labels we have:

Row 1:	1	2	3	4	1	0	120
Row 2:	8	0	16	0	0	1	222
Row 3:	-12	-16	-17	-1	0	0	0

This is the information which must be entered into the computer. The computer assumes the material being entered is a maximization problem, and will label its input and output according to the procedures just reviewed.

SETTING UP A MINIMIZATION PROBLEM

Assume you want to minimize the following problem:

$$\text{Minimize } F(X1, X2) = 120 * X1 + 222 * X2$$

Subject to:

$$\begin{aligned} X1 + 8 * X2 &\geq 12 \\ 2 * X1 &\geq 16 \\ 3 * X1 + 16 * X2 &\geq 17 \\ 4 * X1 &\geq 1 \end{aligned}$$

The task of setting up the Simplex tableau begins with the following:

	Constraints				Variable		
	S1	S2	S3	S4	X1	X2	
Constraints	X1	: :	: :	: :	: 1	: 0	: :
	X	: :	: :	: :	: 0	: 1	: :
					0	0	0

The size of the tableau is the same as in the maximization example. The principal difference is the labels. In the maximization case, where the variables were located, constraints now appear in the minimization case. Similarly, the constraints now become variables. In this case X1 row is matched to X1 column with a one and X2 row is matched to X2 column in a similar manner.

The objective function is:

$$F(X1, X2) = 120 * X1 + 222 * X2$$

Adding the coefficients of the objective function to the tableau results in the following:

	Constraints				Variable		
	S1	S2	S3	S4	X1	X2	
Constraints	X1	: :	: :	: :	: 1	: 0	: 120
	X	: :	: :	: :	: 0	: 1	: 222
					0	0	0

There is no change in sign of the values. They are added to the right side of the tableau in the case of the minimization problem.

The constraints are now added. Let's start with the first constraint:

$$1 * X1 + 8 * X2 \geq 12$$

	Constraints				Variable		
	S1	S2	S3	S4	X1	X2	
Constraints	X1	: 1	: :	: :	: 1	: 0	: 120
	X	: 8	: :	: :	: 0	: 1	: 222
		-12			0	0	0

The value of the coefficients appear on each row near the row labeled with the variable of concern. The value the variables are to be less than appear on the bottom with a negative sign added. If the value was negative, then it would be changed to a positive number.

Adding the remaining constraints results in:

	Constraints				Variable			
	S1	S2	S3	S4	X1	X2		
Constraints	X1	: 1	: 2	: 3	: 4	: 1	: 0	: 120
	X	: 8	: 0	: 16	: 0	: 0	: 1	: 222
		-12	-16	-17	-1	0	0	0

Removing all labels, the material is made ready to put into the computer:

Row 1:	1	2	3	4	1	0	120
Row 2:	8	0	16	0	0	1	222
Row 3:	-12	-16	-17	-1	0	0	0

Examine the results with care. They are exactly the same as in the case of the maximization problem. The computer's labels will assume a maximization problem, but this minimization problem will be solved at the same time the maximization problem is solved. The concept we are using is that the maximization problem is the primal problem and the minimization problem is the dual problem. (Which is primal depends on where you start).

RUNNING THE PROGRAM

After turning your pocket computer on, place it in the DEFINABLE or RUN modes. Type R. <CR> <CR>

means carriage return—press one of the white keys marked ENTER) or RUN <CR> and you will see:

SIMPLEX
CON

To obtain the maximum possible capacity, abbreviations have been used. # CON is asking the question, "number of constraints." The computer assumes that you are typing a maximization problem. In the maximization problem, there were two constraints: type 2 <CR>.

VAR

This question is asking for the number of variables. Type 4 <CR> for four variables.

ROW 1
?

The numbers in row 1 are: 1 2 3 4 1 0 120. Type 1 <CR> and then 2 <CR> until the entire row has been entered.

ROW 2
?

In an manner similar to the above, the values of row 2—8 0 16 0 0 1 222—are entered.

ROW 3
?

The final row—12 -16 -17 -1 0 0 0—is entered one value at a time.

If you have your printer attached and ready to run, it will reproduce the input data as shown:

```

ROW 1.
      1.
      2.
      3.
      4.
      1.
      0.
      120.
ROW 2.
      8.
      0.
      16.
      0.
      0.
      1.
      222.
ROW 3.
     -12.
     -16.
     -17.
      -1.
       0.
       0.
       0.

```

There will be a delay for the computer to perform its calculations. In this case, the delay is approximately one minute long. The larger the tableau, the longer delay you can expect.

EXAMINING THE RESULTS FOR THE MAXIMIZATION CASE

The computer will print the following results:

```

ROW 1.
      0.
      1.
      0.5
      2.
      0.5
     -0.0625
     46.125
ROW 2.
      1.
      0.
      2.
      0.
      0.
      0.125
     27.75
ROW 3.
      0.
      0.
     15.
     31.
      8.
      0.5
     1071.
X
      2.
      1.

```

The answer to the problem is contained within the above. Let's rewrite the output as a maximization tableau:

	Variables				Constraints		
	X1	X2	X3	X4	S1	S2	
						
X2 :	0	1	0.5	2	0.5	-0.0625	46.125
Constraints						
X1 :	1	0	2	0	0	0.125	27.75
						
	0	0	15	31	8	0.5	1071.
	coefficients						
	objective function						

The reason we know the label of row one is X2 and the label of row two is X1 is that the output following the X was a 2 and a 1. At the end of row one is 46.125. This means X2=4.125. At the end of row two is 27.75. This means X1=27.75. Since X3 and X4 do not appear on a row, their values are: X3=0, X4=0.

The objective function is:

$$\text{Maximize } F(X1, X2, X3, X4) = 12 * X1 + 16 * X2 + 17 * X3 + X4$$

Putting the values of X1=27.75, X2=46.125, X3=0, and X4=0 into the equation we get: 1071. The number

in the lower right hand corner of the tableau is 1071!
Typing the values of the X's into the constraints we get:

For constraint number one:

$$\begin{aligned} X1 + 2 * X2 + 3 * X3 + 4 * X4 &\leq 120 \\ 27.75 + 2 * 46.125 + 3 * 0 + 4 * 0 &\leq 120 \\ 120 &\leq 120 \end{aligned}$$

For constraint number two:

$$\begin{aligned} 8 * X1 + 16 * X2 &\leq 222 \\ 8 * 27.75 + 16 * 0 &\leq 222 \\ 222 &\leq 222 \end{aligned}$$

The answers are within the defined constraints.

EXAMINING THE RESULTS FOR THE MINIMIZATION CASE

Redefining the labels on the tableau for minimization and typing the results we obtain:

	Constraints				Variables		
	S1	S2	S3	S4	X1	X2	
Variables	S2	: 0	: 1	: 0.5	: 2	: 0.5	: -0.0625: 46.125
	S1	: 1	: 0	: 2	: 0	: 0	: 0.125 : 27.75
		0	0	15	31	8	0.5 1071.

The last row in the X1 column has the value 8. This means X1=8. The last row in the X2 column is 0.5. This means X2=0.5. Examining the objective function:

$$F(X1,X2)=120 * X1 + 222 * X2$$

and typing the results:

$$\begin{aligned} F(8,0.5) &= 120 * 8 + 222 * 0.5 \\ &= 1071. \end{aligned}$$

The value of the objective function is the same for both the minimization and maximization case.

Checking the constraints:

For constraint number one:

$$\begin{aligned} 1 * X1 + 8 * X2 &\geq 12 \\ 1 * 8 + 8 * 0 &\geq 12 \\ 72 &\geq 12 \end{aligned}$$

For constraint number two:

$$\begin{aligned} 2 * X1 &\geq 16 \\ 2 * 8 &\geq 16 \\ 16 &\geq 16 \end{aligned}$$

For constraint number three:

$$\begin{aligned} 3 * X1 + 16 * X2 &\geq 17 \\ 3 * 8 + 16 * 0.5 &\geq 17 \\ 16 + 8 &\geq 17 \\ 24 &\geq 17 \end{aligned}$$

For constraint number four:

$$\begin{aligned} 4 * X1 &\geq 1 \\ 4 * 8 &\geq 1 \\ 32 &\geq 1 \end{aligned}$$

All constraints are satisfied, and the solution is optimal.

DEGENERATE SOLUTIONS

The theory used for the development of the program sometimes results in a problem which will not yield a result. When this happens, you can usually find an acceptable solution by changing the order of the constraints.

EXAMINING THE PROGRAM

The program was designed to use the first nine memory locations A, B, C, D, E, F, G, H, I for programming, and all other locations for data memory. This is one of the reasons the computer is able to handle as large a problem as it can. Location A contains the number of constraints, location B contains the number of variables. Locations C, D, E, G, and H are used as working variables and loop counters. Location F is the pivotal column, while location I is the pivotal row.

Lines 1-6 are used to input the data. Lines 10 and 11 print the tableau on display or printer. Lines 20-23 check to see if there are any negative indicators. If there are no negative indicators, line 22 sends the program to line 70 to print the final results. If there is a negative indicator, the program proceeds to line 30 with F defined as the column with the negative indicator, the pivotal column.

Lines 30-33 select the row with the best chance for transferring. This row is called the pivotal row, and variable I contains this information.

Lines 40-42 divide the Ith row of the old tableau by the pivot and defines a row for the new tableau.

Lines 50-51 keep track of the row labels. This is the information printed after the X in the final results.

Lines 60-66 redefine all the rows of the new tableau, with the exception of row I already defined. The program then recycles to line 20 to see if any additional negative indicators exist.

Lines 70-72 call line 10 as a printing subroutine and then print the row labels.

The program uses line numbers to structure the tasks being performed. To maximize the program capacity the line numbers were kept as low as possible.

PROGRAM LISTING

```

1: PAUSE "SIMPLEX": CLEAR
2: INPUT "# CON "; A: INPUT "# VAR "; B
4: D=10: FOR C=1 TO A+1: PAUSE "ROW "; C
5: FOR E=1 TO B+A+1: INPUT A(D): D=D+1: NEXT E: NEXT C: GOSUB 10
6: GOTO 20

10: D=10: FOR C=1 TO A+1: PRINT "ROW "; C
11: FOR E=1 TO B+A+1: PRINT A(D): D=D+1: NEXT E: NEXT C: RETURN

20: D=A*(B+A+1)+10: F=1
21: IF A(D)<0 THEN 30
22: F=F+1: D=D+1: IF D-10<=(A+1)*(A+B+1) THEN 70
23: GOTO 21

30: D=9+F: G=(A+1)*(A+B+1)+11: H=A+B+10: E=999999
31: FOR C=1 TO A: A(G)=99999: IF A(D)>0 LET A(G)=A(H)/A(D)
32: IF E>A(G) LET E=A(G): I=C
33: D=D+A+B+1: G=G+1: H=H+A+B+1: NEXT C
  
```

continued on next page

TWIN CITIES COMPUTERIST GOES NATIVE

Sallie Stephenson

"It's not hard to write a good program if you know what you want the computer to do"—Fred says—pointing to an outburst of ATARI cassettes he's already filled with games of his own design. Fred Yonke is a Native American, of Minneapolis, Minnesota, a freelance graphic artist who has recently found that home computer games are relaxing and challenging at the same time.

"Centipede was the first game I played," he says. "I was just having fun playing all the games, and then I thought I'd like to try designing some of my own.

For my game designing, the ATARI offers more colors than any other home computer—that's what I like—and I'm up to my ears in speaker sounds. Most of the games I design have rockets and explosions in them. Actually," Fred explains, "I can create a whole variety of sounds."

"Missile Blasters," "A is for Alpha," "Pluto," and "Mind Suggestion" are just some of the games he birthed. The jackets he has designed for these video games are exciting, with tribal overtones of bright bold hues, which comes from his Cherokee heritage.

Yet, the geometric shapes and lines he chooses are very futuristic. One of his games called "Profession" is a fast-action playing field with a line of characters that appear and disappear suddenly. The object is to hit any of the graphic-block characters before they disappear without running into boundary lines or having one of the other team's characters zap you.

Already I'm thinking of a lot more games," Fred says. "I've even got some of the members of my old tribe at Red Lake Indian Reservation playing my games. They have one computer there that has a modulator/demodulator (modem), using regular telephones, so we can play back and forth."

Though Fred has no plans to market his games yet, he says he's still having a lot of fun with it. "It's really child's play, but—what the heck—a child can accomplish almost anything, right?"

Sallie Stephenson
Cedar Square West, Suite F-810
1601 South Fourth Street
Minneapolis, MN 55454 ■

POCKET COMPUTER CORNER continued from previous page

```
40:D=9+F+(I-1)*(A+B+1):D=A(D)
41:FOR C=10+(I-1)*(A+B+1) TO 10+(I-1)*(A+B+1)+A+B
42:A(C)=A(C)/D:NEXT C
```

```
50:C=(A+1)*(A+B+2)+9+I
51:A(C)=F
```

```
60:FOR C=1 TO A+1:IF C=1 THEN 66
61:D=9+F+(C-1)*(A+B+1):D=A(D)
62:G=10+(I-1)*(A+B+1)
63:FOR E=10+(C-1)*(A+B+1) TO 10+(C-1)*(A+B+1)+A+B
```

```
65:A(E)=A(E)-D*A(G):G=G+1:NEXT E
66:NEXT C:GOTO 20
70:GOSUB 10:PRINT "X"
71:C=(A+1)*(A+B+2)+10
72:FOR D=C TO C+A-1:PRINT A(D):NEXT D
```

Steven M. Zimmerman, Ph.D.
College of Business and Management Studies
University of South Alabama
Mobile, Alabama 36605

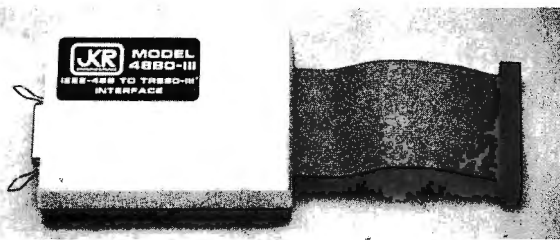
Leo M. Conrad
Imagineering Concepts
P.O. Box 9843
Mobile, Alabama 36691-0843 ■

IEEE-488 to TRS-80 Interface

Our IEEE-488 to TRS-80 interface adds powerful GPIB control capability to your TRS-80 Model I or Model III Computer. Simply plug the interface into the I/O Bus on your computer, and load the RAM resident interface software (supplied). With additional software also provided the interface can then be used immediately.

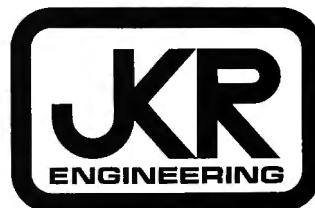
Compare these features for only \$299.

- Will operate with Basic Compilers.
- Can be used in Assembly programs.
- Relocatable thru-out memory.
- Uses less than 1K of memory.
- Supports TRSDOS, NEWDOS & LDOS.



Model 4880-I or Model 4880-III
with Disk or Tape Price \$299.

Add \$2 for shipping & handling per order.
Add \$3 for COD orders. CA res add 6% Sales Tax.



1687 Pinewood Way
Milpitas CA. 95035

To order call
(408) 263-7139

INTRODUCTION TO FRACTIONS AND SPELLBOUND

Airdrie Ferguson

Copyright © 1981 Educomp

INTRODUCTION TO FRACTIONS

Introduction to fractions is an "intrinsic programmed learning tutorial" that teaches students about fractions. It is completely self-explanatory. Instructions are given on the screen when you run it. Good luck!

```
1 *** AIRDRIE FERGUSON BOX 40206 CASUARINA NT 5792 AUSTRALIA **
2 CLS: CLEAR2000: RANDOM: PRINTCHR$(23): PRINT@92, "AN"; : PRINT@208,
"INTRODUCTION TO"; : FORJ=1T0500: NEXT: GOSUB53: CLS: PRINT@896, "C)
1982"; : PRINT@953, "EDUCOMP"; : FORJ=1T0500: NEXT: CLS
3 A$=CHR$(50)+CHR$(128)+CHR$(26)+STRING$(3,24)+STRING$(3,45)+
CHR$(128)+CHR$(26)+STRING$(3,24)+CHR$(51)+CHR$(128): PRINT
TAB(20)"THERE ARE TWO SECTIONS,": PRINT"THE FIRST OF WHICH, IS
AN INTRINSIC PROGRAMMED LEARNING TUTORIAL"
4 PRINTTAB(12)"THE SECOND SECTION CONSISTS OF EXERCISES": PRINT
TAB(7)"WHICH USE THE KNOWLEDGE PRESENTED IN THE TUTORIAL.":
PRINT
TAB(7)"THE OCCURENCE OF A STUDENT ERROR IN ANY EXERCISE,": PRINT
TAB(13)"GENERATES A RETURN TO THE TUTORIAL AND"
5 PRINTTAB(8)"A REVIEW OF THE AREA RELATING TO THAT STUDENT
ERROR": PRINTTAB(1)"FOLLOWED BY A FURTHER PRESENTATION OF THE
RELEVANT EXERCISE.": PRINTTAB(21)"COMPLETE PROFICIENCY,": PRINT
TAB(26)"10 OUT OF 10"
6 PRINTTAB(10)"IS A PREREQUISITE OF EACH FOLLOWING EXERCISE":
QS="": PRINT@910, "PRESS <ENTER> WHEN READY TO CONTINUE";
7 QS=INKEY$: IFQ$="" THEN ELSE IFQ$=CHR$(13) THEN 8 ELSE 7
8 GOT015
9 REM
10 CLS: R=RND(8)+1: GOSUB22: R$=STR$(R): PRINTCHR$(23): PRINT@266,
"1"+CHR$(26)+STRING$(2,24)+STRING$(3,45)+CHR$(26)+STRING$(3,24)
+R$: : FORJ=1T01000: NEXT: PRINT@338, "MEANS"; : FORJ=1T0500: NEXT: FOR
K=1T05: PRINT@266, " "; : PRINT@350, " "; : FORJ=1T0100: NEXT
11 PRINT@266, "1"; : PRINT@350, "1"; : FORJ=1T0100: NEXT: NEXT: PRINT
@522, STRING$(20,179); : FORJ=1T0500: NEXT: PRINT@354, "DIVIDED BY";
R$: : FORK=1T05: PRINT@394, " "; : PRINT@376, " "; : FORJ=1T0100: NEXT:
PRINT@392, R$: : PRINT@374, R$: : FORJ=1T0100: NEXT: NEXT
12 FORJ=1T0500: NEXT: PRINT@522, STRING$(20/R,191); : FORJ=1T01000:
NEXT: PRINT@522, STRING$(20/R,191)+STRING$(20,128); : FORJ=1T0500:
NEXT: PRINT@668, "IS"; : FORJ=1T0500: NEXT: PRINT@778, STRING$(
20,191); : FORJ=1T0700: NEXT: PRINT@916, "DIVIDED BY"; R$:
13 FORJ=1T02000: NEXT: PRINT@668, " "; : PRINT@778, STRING$(20,128);
: PRINT@916, STRING$(12,128); : FORJ=1T0500: NEXT: PRINT@654, "IS ONE
"; RR$: " OF"; : FORJ=1T0500: NEXT: PRINT@778, STRING$(20,191); : FOR
J=1T02000: NEXT: T=T+1: IFT=5 THEN T=0: GOT014 ELSE 9
14 IFQ$="" THEN 14 ELSE 14
15 CLS: PRINTCHR$(23): PRINT@284, "2"; : PRINT@346, STRING$(3,45); :
PRINT@412, "3"; : FORJ=1T0500: NEXT: IFB THEN 20 ELSE FORK=1T03: PRINT
@220, " "; : FORJ=1T050: NEXT: PRINT@220, CHR$(92); : FORJ=1T050: NEXT:
NEXT: PRINT@148, "NUMERATOR"; : FORJ=1T0500: NEXT: FORK=1T03
16 PRINT@476, " "; : FORJ=1T050: NEXT: PRINT@476, CHR$(91); : FORJ=1
T050: NEXT: NEXT: PRINT@530, "DENOMINATOR"; : FORJ=1T0500: NEXT: FORK=1
T03: PRINT@342, " "; : FORJ=1T050: NEXT: PRINT@342, CHR$(94); : FORJ=1
T050: NEXT: NEXT: PRINT@324, "VINCULUM"; : FORJ=1T01500: NEXT
17 PRINT@16, "THE NUMERATOR"; : PRINT@132, "IS READ AS A CARDINAL
NUMBER"; : FORJ=1T02000: NEXT: FORS=1T05: PRINT@220, " "; : PRINT@296,
" "; : FORJ=1T0100: NEXT: PRINT@220, CHR$(92); : PRINT@296, "TWO"; :
FORJ=1T0100: NEXT: NEXT: FORJ=1T02000: NEXT: PRINT@552, "S";
18 PRINT@642, " ARE READ AS ORDINAL NUMBERS"; : FORJ=1T02000: NEXT:
FORS=1T05: PRINT@422, " "; : PRINT@476, " "; : FORJ=1T0100: NEXT:
PRINT@476, CHR$(91); : PRINT@422, "THIRDS"; : FORJ=1T0100: NEXT: NEXT:
FORJ=1T01500: NEXT: PRINT@354, "=";
19 PRINT@358, STRING$(5,45); : FORJ=1T01500: NEXT: IFQ$="" THEN 19 ELSE 19
```

```
GOT0101 ELSE 1: GOT015
20 FORAA=284T0266STEP-2: PRINT@AA, A$; : NEXT: PRINT@338, "MEANS 2
DIVIDED BY 3"; : FORJ=1T02000: NEXT: CLS: FORK=2T09: PRINTCHR$(23):
PRINT@350, CHR$(176); : PRINT@412, STRING$(3,140)CHR$(197); : PRINT
@478, CHR$(131); : FORJ=1T01000: NEXT: PRINT@348, K;
21 PRINT@412, STRING$(3,45)CHR$(128)=""CHR$(128)"1"; : PRINT@476,
K; : FORJ=1T01000: NEXT: NEXT: GOT09
22 ONR-1GOT023,24,25,26,27,28,29,30
23 RR$="HALF": RETURN
24 RR$="THIRD": RETURN
25 RR$="FOURTH": RETURN
26 RR$="FIFTH": RETURN
27 RR$="SIXTH": RETURN
28 RR$="SEVENTH": RETURN
29 RR$="EIGHTH": RETURN
30 RR$="NINTH": RETURN
31 CLS: FORK=1T05: CLS: PRINTCHR$(23)
32 A=RND(7)+1: B=RND(7)+2: IFA>=B THEN 32 ELSE GOSUB43: GOSUB34: FOR
AA=1T0B: PRINTSTRING$(20/B,191)+CHR$(128); : NEXT: FORJ=1T0500: NEXT
: PRINT@330, A; " OF THESE "A$; : FORJ=1T0500: NEXT: PRINT: PRINT:
PRINT: FORAA=1T0A: PRINTSTRING$(20/B,191)+CHR$(128); : FORJ=1T0100
33 NEXT: NEXT: FORJ=1T0250: NEXT: PRINT@706, "ARE "; F$(1); " OF ONE
WHOLE SET"; CHR$(26); : FORJ=1T0500: NEXT: PRINT: PRINT: FORAA=1T0B:
PRINTSTRING$(20/B,191)+CHR$(128); : FORJ=1T0100: NEXT: NEXT: FORJ=1
T02000: NEXT: NEXT: IFQ$="" THEN 34 ELSE 34
34 ONB-1GOT035,36,37,38,39,40,41,42
35 A$="HALVES": RETURN
36 A$="THIRDS": RETURN
37 A$="FOURTHS": RETURN
38 A$="FIFTHS": RETURN
39 A$="SIXTHS": RETURN
40 A$="SEVENTHS": RETURN
41 A$="EIGHTHS": RETURN
42 A$="NINTHS": RETURN
43 F$(1)="": F$(1)=CHR$(27)+STR$(A)+CHR$(26)+STRING$(2,24)+
STRING$(3,45)+CHR$(26)+STRING$(3,24)+STR$(B)+CHR$(193)+CHR$(27)
: RETURN
44 CLS: PRINTCHR$(23): FORF=1T05: PRINT@412, CHR$(188); : PRINT@346,
STRING$(3,140); : PRINT@284, CHR$(143); : FORJ=1T0250: NEXT: PRINT
@346, STRING$(3,45); : PRINT@282, F; : PRINT@410, F; : FORJ=1T0250: NEXT:
PRINT@354, "="; : FORJ=1T0370: NEXT: CLS: PRINTCHR$(23): NEXT
45 CLEAR1000: FORK=1T05: CLS
46 X=RND(8): Y=RND(9): IFX>Y THEN 46 ELSE X$=STR$(X): Y$=STR$(Y): Z$=
STR$(RND(8)+1): W$=STR$(VAL(X$)*VAL(Z$)): U$=STR$(VAL(Y$)*VAL
(Z$)): XX$=X$: YY$=Y$: GOSUB52: PRINTCHR$(23): PRINT@270, F$(1); :
PRINT@280, "X 1 = "; F$(1); : FORJ=1T01500: NEXT: XX$=Z$: YY$=Z$
47 GOSUB52: PRINT@282, F$(1); : FORJ=1T01500: NEXT: PRINT@216, "X"; :
PRINT@280, " "; : PRINT@344, "X"; : FORJ=1T01500: NEXT: PRINT@278,
STRING$(3,45); : FORJ=1T01500: NEXT: E=208: IFVAL(W$)>9 THEN E=E-2:
W$=STR$(INT(VAL(W$)/10))+STR$(VAL(W$)-(INT(VAL(W$)/10)*10))
48 PRINT@E, " "; W$; " "; : FORJ=1T01500: NEXT: F=336: IFVAL(U$)>9
THEN F=F-2: U$=STR$(INT(VAL(U$)/10))+STR$(VAL(U$)-(INT(VAL(U$)
/10)*10))
49 PRINT@F, " "; U$; " "; : PRINT@272, " --- "; : FORJ=1T01500:
NEXT: PRINT@288, "AND"; : FORJ=1T0500: NEXT: PRINT@520, "ARE
EQUIVALENT FRACTIONS"; : FORJ=1T0500: NEXT: FORS=1T05: PRINT@852,
STRING$(22,128); : FORJ=1T0100: NEXT: PRINT@852, "EQUAL VALUE";
50 FORJ=1T0100: NEXT: NEXT: FORJ=1T01000: NEXT: NEXT: IFQ$="" THEN 138
51 GOT059
52 F$(1)=CHR$(27)+CHR$(128)+XX$+CHR$(26)+STRING$(2,24)+STRING$(
3,45)+CHR$(26)+STRING$(3,24)+YY$: RETURN
53 C$=CHR$(191)+STRING$(3,131)+CHR$(128)+CHR$(191)+STRING$(
2,131)+CHR$(189)+CHR$(128)+CHR$(190)+STRING$(2,131)+CHR$(189)
```

```

+CHR$(128)+CHR$(190)+STRING$(2,131)+CHR$(189)+CHR$(128)
+CHR$(131)+CHR$(171)+CHR$(151)+CHR$(131)+CHR$(128)+CHR$(191)
54 C$=C$+CHR$(128)+CHR$(190)+STRING$(2,131)+CHR$(189)+CHR$(128)
+CHR$(191)+CHR$(180)+CHR$(128)+CHR$(191)+CHR$(128)+CHR$(190)
+STRING$(2,131)+CHR$(141)+CHR$(26)+STRING$(41,24)
55 C$=C$+CHR$(191)+STRING$(2,140)+CHR$(194)+CHR$(191)+CHR$(140)
+CHR$(188)+CHR$(135)+CHR$(128)+CHR$(191)+STRING$(2,140)
+CHR$(191)+CHR$(128)+CHR$(191)+CHR$(197)+CHR$(170)+CHR$(149)
+CHR$(194)+CHR$(191)+CHR$(128)+CHR$(191)+CHR$(194)+CHR$(191)
56 C$=C$+CHR$(128)+CHR$(191)+CHR$(170)+CHR$(149)+CHR$(191)
+CHR$(128)+CHR$(139)+STRING$(2,140)+CHR$(180)+CHR$(26)
+STRING$(41,24)
57 C$=C$+CHR$(191)+CHR$(196)+CHR$(191)+CHR$(128)+CHR$(130)
+CHR$(189)+CHR$(128)+CHR$(191)+CHR$(194)+CHR$(191)+CHR$(128)
+CHR$(175)+STRING$(2,176)+CHR$(159)+CHR$(194)+CHR$(170)
+CHR$(149)+CHR$(194)+CHR$(191)+CHR$(128)+CHR$(175)
+STRING$(2,176)
58 C$=C$+CHR$(159)+CHR$(128)+CHR$(191)+CHR$(128)+CHR$(139)
+CHR$(191)+CHR$(128)+CHR$(172)+STRING$(2,176)+CHR$(159):CLS:
PRINT@333,C$::FORJ=1TO500:NEXT:FORK=1TO150:X=RND(191)+15680:
POKEX,128:POKEX+1,128:POKEX-1,128:NEXT:CLS:RETURN
59 FORK=1TO5:CLS:PRINTCHR$(23)
60 A=RND(8):B=RND(9):IFA=>BTHEN60ELSEC=RND(8)+1:D=A*C:E=B*C:
IFD>9THEN61ELSEPRINT@336,D::PRINT@348,D::GOTO62
61 H=INT(D/10):HH=INT((D/10-INT(D/10)+.05)*10):A$=STR$(H)+
STR$(HH):PRINT@334,A$:PRINT@346,A$:
62 IFE>9THEN63ELSEPRINT@464,E::PRINT@476,E::GOTO64
63 M=INT(E/10):MM=INT((E/10-INT(E/10)+.05)*10):C$=STR$(M)+
STR$(MM):PRINT@462,C$:PRINT@474,C$:
64 PRINT@420,"-":PRINT@430,"1":PRINT@408,"=":PRINT@400,
STRING$(3,45)::PRINT@412,STRING$(3,45)::FORJ=1TO1000:NEXT:
PRINT@364,C::PRINT@428,STRING$(3,45)::PRINT@492,C::FORJ=1TO2000
:NEXT:PRINT@420," "":PRINT@484,"-":PRINT@356,"-":
65 PRINT@418,STRING$(5,45)::FORJ=1TO2000:NEXT:PRINT@346,STRING$
(11,128)::PRINT@356,A::FORJ=1TO2000:NEXT:PRINT@474,STRING$
(11,128)::PRINT@484,B::PRINT@412," --- "":PRINT@408,
" =":FORJ=1TO1000:NEXT:PRINT@408," AND":FORJ=1TO500:NEXT
66 PRINT@646,"ARE EQUIVALENT FRACTIONS":FORJ=1TO500:NEXT:FOR
S=1TO5:PRINT@852,STRING$(22,128)::FORJ=1TO100:NEXT:PRINT@852,
"EQUAL VALUE":FORJ=1TO100:NEXT:NEXT:FORJ=1TO1000:NEXT:NEXT:IF
QQTHEN132
67 FORK=1TO10:CLS:A=RND(8)+1:ONAGOSUB69,70,71,72,73,74,75,76,77
:PRINTCHR$(23):PRINT@80,"TO FIND A "A$:,"":PRINT@194,"BY HOW
MUCH WOULD YOU DIVIDE ?":
68 PRINT@414," "":FORJ=1TO100:NEXT:PRINT@414,"?":FORJ=1TO100:
NEXT:Q$="":Q$=INKEY$:IFQ$=""THEN68ELSEIFVAL(Q$)=ATHEN78ELSESC=0
:QQ=1:GOTO9
69 REM
70 A$="HALF":RETURN
71 A$="THIRD":RETURN
72 A$="FOURTH":RETURN
73 A$="FIFTH":RETURN
74 A$="SIXTH":RETURN
75 A$="SEVENTH":RETURN
76 A$="EIGHTH":RETURN
77 A$="NINTH":RETURN
78 SC=SC+1:PRINT@660,SC;"CORRECT":PRINT@412,A::FORJ=1TO500:
NEXT:IFSC=10THENSCL=0:GOTO79ELSE67
79 REM
80 CLS
81 FORK=1TO10:CLS
82 A=RND(8):B=RND(8)+1:ONBOSUB86,87,88,89,90,91,92,93,94:
IFA=>BTHEN82ELSEC=B-A:PRINTCHR$(23):PRINT@92,A::PRINT@156,
STRING$(3,45)::PRINT@220,B::PRINT@336,"HOW MANY "B$:PRINT
@402,"ARE ASKED FOR ?":
83 PRINT@464,STRING$(17,45)CHR$(128)STRING$(3,45)::Q$=""
84 PRINT@438,"?":FORJ=1TO100:NEXT:PRINT@438," "":FORJ=1TO100:
NEXT:Q$=INKEY$:IFQ$=""THEN84ELSEIFVAL(Q$)=ATHEN95ELSESC=0:QQ=1:
GOTO31

```

```

85 GOTO100
86 REM
87 B$="HALVES":RETURN
88 B$="THIRDS":RETURN
89 B$="FOURTHS":RETURN
90 B$="FIFTHS":RETURN
91 B$="SIXTHS":RETURN
92 B$="SEVENTHS":RETURN
93 B$="EIGHTHS":RETURN
94 B$="NINTHS":RETURN
95 PRINT@436,A::PRINT@564,B::FORJ=1TO500:NEXT:PRINT@656,"HOW
MANY "B$:PRINT@724,"WILL BE LEFT ?":PRINT@846,A::PRINT@870,
B::PRINT@910,STRING$(3,45)CHR$(128)+"CHR$(128)STRING$(3,45)
CHR$(128)"="CHR$(128)STRING$(3,45)CHR$(128)"="CHR$(128)"1":
96 FORL=974TO998STEP12:PRINT@L,B::NEXT:Q$=""
97 PRINT@860," "":FORJ=1TO100:NEXT:PRINT@860,"?":FORJ=1TO100:
NEXT:Q$=INKEY$:IFQ$=""THEN97ELSEIFVAL(Q$)=CTHEN99
98 NEXT:GOTO100
99 SC=SC+1:PRINT@858,C::FORJ=1TO500:NEXT:PRINT@20,SC;"CORRECT":
:FORJ=1TO500:NEXT:GOTO98
100 IFSC=10THENSCL=0:GOTO101ELSESC=0:GOTO81
101 REM
102 CLS
103 FORK=1TO10:CLS
104 A=RND(7)+1:ONAGOSUB106,107,108,109,110,111,112,113:B=RND(7)
+2:ONBOSUB114,115,116,117,118,119,120,121,122:IFA=>BTHEN104
ELSEA$="1"+CHR$(26)+STRING$(2,24)+STRING$(3,45)+CHR$(26)+
STRING$(3,24)+STR$(B)+CHR$(195)+STRING$(2,27):X=(34-4*A)+256
105 PRINTCHR$(23):FORL=1TOA:PRINT@X,AA$:X=X+8:NEXT:FORJ=1TO250
:NEXT:GOTO123
106 REM
107 A$="TWO":RETURN
108 A$="THREE":RETURN
109 A$="FOUR":RETURN
110 A$="FIVE":RETURN
111 A$="SIX":RETURN
112 A$="SEVEN":RETURN
113 A$="EIGHT":RETURN
114 REM
115 REM
116 B$="THIRDS":RETURN
117 B$="FOURTHS":RETURN
118 B$="FIFTHS":RETURN
119 B$="SIXTHS":RETURN
120 B$="SEVENTHS":RETURN
121 B$="EIGHTHS":RETURN
122 B$="NINTHS":RETURN
123 PRINT@532,A$+CHR$(128)+B$:FORJ=1TO500:NEXT:PRINT@732,
STRING$(3,45)::PRINT@798,"?":Q$=""
124 PRINT@670,"?":FORJ=1TO100:NEXT:PRINT@670," "":FORJ=1TO100:
NEXT:Q$=INKEY$:IFQ$=""THEN124ELSEIFVAL(Q$)=ATHEN126ELSESC=0:
B=0:QQ=1:GOTO3
125 GOTO128
126 Q$=""
127 PRINT@668,A::PRINT@798," "":FORJ=1TO100:NEXT:PRINT@798,"?":
FORJ=1TO100:NEXT:Q$=INKEY$:IFQ$=""THEN127ELSEIFVAL(Q$)=B
THEN129ELSESC=0:B=0:QQ=1:GOTO15
128 SC=0:GOTO103
129 SC=SC+1:PRINT@796,B::FORJ=1TO250:NEXT:PRINT@20,SC;"CORRECT":
:FORJ=1TO500:NEXT:IFSC=10THENSCL=0:GOTO130ELSENEXT:GOTO132
130 FORJ=1TO1000:NEXT:CLS:FORJ=1TO500:NEXT:PRINT@154,
"CONGRATULATIONS !":PRINT@274,"YOU HAVE SUCCESSFULLY
COMPLETED":PRINT@416,"AN":PRINT@538,"INTRODUCTION TO":
FORJ=1TO500:NEXT:GOSUB53:FORJ=1TO1000:NEXT:END
131 GOTO131
132 CLS:QQ=0:FORL=1TO10:AA=278:BB=406
133 D=RND(9):C=RND(8)+1:IFD=>CTHEN133ELSEE=RND(8)+1:A=D*E:
A$=STR$(A):IFA>9THENA$=STR$(INT(A/10+.005))+STR$(A/10-
INT(A/10))*10):AA=AA-2

```

```

134 B=C*E:BS=STR$(B):IFB>9THENBS=STR$(INT(B/10+.005))+
STR$(B/10-INT(B/10))*10):BB=BB-2
135 PRINTCHR$(23):PRINT@AA,A$;:PRINT@BB,B$;:PRINT@418,C;:
PRINT@292,"?";:PRINT@342,STRING$(3,45)CHR$(128)="CHR$(128)
STRING$(3,45);:Q$=""
136 FORJ=1TO100:NEXT:PRINT@292," ";:FORJ=1TO100:NEXT:
PRINT@292,"?";:Q$=INKEY$:IFQ$=""THEN136ELSEIFVAL(Q$)=DTHEN137
ELSEQ=1:SC=0:GOTO51
137 SC=SC+1:PRINT@290,D;:PRINT@84,SC;"CORRECT";:FORJ=1TO500:
NEXT:CLS:NEXTT:IFSC=10THENS=0:GOTO138
138 CLS:Q=0:FORT=1TO10:AA=288:BB=416
139 D=RND(9):C=RND(8)+1:IFD>=CTHEN139ELSEE=RND(8)+1:A=D*E:
A$=STR$(A):IFA>9THENA$=STR$(INT(A/10+.005))+STR$(A/10-
INT(A/10))*10):AA=AA-2
140 B=C*E:BS=STR$(B):IFB>9THENBS=STR$(INT(B/10+.005))+
STR$(B/10-INT(B/10))*10):BB=BB-2
141 PRINTCHR$(23):PRINT@276,D;:PRINT@404,C;:PRINT@290,"?";:
PRINT@BB,B$;:PRINT@340,STRING$(3,45)CHR$(128)="CHR$(128)
STRING$(3,45);:Q$=""
142 FORJ=1TO100:NEXT:PRINT@290," ";:FORJ=1TO100:NEXT:
PRINT@290,"?";:Q$=INKEY$:IFQ$=""THEN142ELSEIFHTHENH=0:Q$=Z$+Q$:
GOTO144ELSEIFA>9THENH=1:GOTO143ELSE144
143 Z$=Q$:GOTO142
144 IFVAL(Q$)=ATHEN145ELSEQ=1:SC=0:GOTO45
145 SC=SC+1:PRINT@AA,A$;:PRINT@84,SC;"CORRECT";:FORJ=1TO500:
NEXT:CLS:NEXTT:IFSC=10THEN130ELSEEND

```

SPELLBOUND

First, it is necessary to insure that a blank cassette without a leader is available for audio purposes, and of course a cassette recorder compatibly connected to the computer.

Also, as the program is for the TRS-80 Model III, alteration to the port addresses in lines 19 and 34 will be necessary for the Model I, for which the program was originally written.

SPELLBOUND uses the drill learning process commonly in use in earlier days but proved effective where more sophisticated educational methods fall down. While the program is self-explanatory and to a large extent auto-instructional, the following notes will speed up the familiarization process:

The supervisor (teacher or parent) should have a selected list of words with which the student has exhibited difficulty, the blank audio cassette (leaderless), and the cassette recorder connected and ready.

Load SPELLBOUND and RUN.

The number of words in the list is requested, and instructions for insertion of the cassette are issued depending on whether the audio tape has or has not already been prepared.

TAPE NOT PREPARED The program prompts the insertion of each word typed followed by an audio presentation for the cassette to record. Following the typing of each word, the command SPEAK NOW appears, followed by about 10 seconds, during which time the recorder activates and the dictated message follow, viz. "FOLLOW: The police began to FOLLOW the man . . . spell FOLLOW" or some such. This continues until all words have been prepared.

TAPE ALREADY PREPARED It is only necessary for the student to follow prompts, and the tape will play the dictated test and prompt the typing of the student attempts to spell them.

Correct results are followed by progression through the list at the student's own rate, but an error requires that the whole exercise be restarted.

Options exist to select a repeat of the list, prepare a fresh one, or finish.

It has been noted that, with some equipment, difficulty

has arisen where the relays activating the recorders do not respond. This has proved to be a mechanical fault and not inherent in the program.

```

1 FORZQ=28672TO28690:READNX:POKEZQ,NX:NEXT:POKE16526,0:
POKE16527,112:DATA58,37,1,254,73,40,6,175,62,4,211,255,201,175
,62,2,211,236,201
2 **** AIRDRIE FERGUSON BOX 40206 CASUARINA NT 5792 AUSTRALIA
**
3 CLS:PRINTCHR$(23):PRINT@340,"SPELLBOUND":CLEAR8000:DEFINTA-Z:
DIMLL$(25),L$(26),TL$(100),TW$(100):FORDT=1TO26:READL$(DT):NEXT
:DATA"A","B","C","D","E","F","G","H","I","J","K","L","M","N",
"O","P","Q","R","S","T","U","V","W","X","Y","Z":CLS
4 PRINT@896,"(C) 1981":PRINT@953,"EDUCOMP";
5 FORJ=1TO500:NEXT:CLS:PRINT@397,"PLEASE TYPE HOW MANY WORDS IN
THIS LIST";:FORJ=1TO750:NEXT:PRINT@663,"THEN PRESS <ENTER>":
PRINT@541,"";:INPUTN:CLS:GOSUB26:PRINT"INSERT REWOUND AUDIO
CASSETTE":PRINT"REMOVE AUX AND EAR PLUGS"
6 PRINT"PRESS 'RECORD' AND 'PLAY' TOGETHER":PRINT"PRESS
<ENTER>":INPUTYZ:CLS
7 REM *** CREATE WORDS ***
8 CLS:C=0:FORL=1TON:TW$(L)="" :PRINT@33,"TYPE WORD 'L' AND PRESS
<ENTER>":KK=1000:GOSUB21:IFZYTHEN18ELSEGOSUB25:CLS:NEXTL:GOSUB
28:CLS:PRINT"INSERT REWOUND AUDIO CASSETTE":PRINT"REMOVE AUX
AND EAR PLUGS":PRINT"PRESS 'PLAY'"
9 PRINT"PRESS <ENTER>":INPUTYZ:GOTO11
10 CLS:PRINT"REWIND AUDIO CASSETTE":PRINT"REMOVE AUX AND EAR
PLUGS":PRINT"PRESS PLAY":PRINT"PRESS <ENTER>":INPUTYZ
11 WW=0:FORW=1TON:GOTO14
12 FORV=1TO3:PRINT@P+64," ";:FORJ=1TO100:NEXT:PRINT@P+64,CHR$(
91);:FORJ=1TO100:NEXT:NEXT:FORJ=1TO1000:NEXT:CLS:GOTO10
13 CLS:GOSUB14:NEXTW:GOTO19
14 CLS:IFW=1THENYS="FIRST"ELSEIFW=NTHENYS="LAST"ELSEY$="NEXT"
15 X=USR(0):PRINTCHR$(23):PRINT@406,Y$ WORD":FORD=1TO8700:NEXT
:CLS:PRINTCHR$(23):PRINT@2,"SPELLBOUNDSPELLBOUNDSPELLBOUND";
:PRINT@962,"SPELLBOUNDSPELLBOUNDSPELLBOUND";:GOSUB16:GOTO18
16 BX=0:P=351-LEN(TW$(W)):IFP/2<<INT(P/2)THENP=P+1
17 Q$="" :Q$=INKEY$:IFQ$=""THEN17ELSEIFASC(Q$)<65ORASC(Q$)>90
THEN17ELSEW=WW+1:DD=ASC(Q$)-64:IFL$(DD)<>TL$(WW)THEN12ELSE
BX=BX+1:PRINT@P,CHR$(ASC(Q$));:IFBX=LEN(TW$(W))THENFORJ=1TO500:
NEXT:RETURNELSEP=P+2:GOTO17
18 CLS:NEXT:IFW=N+1THEN19ELSEGOSUB28:IFZYTHEN10
19 PRINTCHR$(23):PRINT@258,"PRESS <R> TO REPEAT THIS LIST":
PRINT@388,"PRESS <N> FOR A NEW LIST":PRINT@514,"PRESS <F> IF
YOU ARE FINISHED"
20 Q$="" :Q$=INKEY$:IFQ$=""THEN20ELSEIFQ$="N"THEN5ELSEIFQ$="R"
THENCLS:GOTO10ELSEIFQ$="F"THENCLS:FORJ=1TO2000:NEXT:GOTO22
ELSE20
21 Z$="" :Z$=INKEY$:IFZ$=""THEN21ELSEIFZ$=CHR$(13)RETURNELSE
IFASC(Z$)<65ORASC(Z$)>90THEN21ELSEC=C+1:D=ASC(Z$)-64:TL$(C)=
L$(D):TW$(L)=TW$(L)+CHR$(D+64):PRINT@KK,CHR$(D+64);:KK=KK+1:
GOTO21
22 END
23 '
24 '
25 CLS:PRINTCHR$(23):X=USR(0):FORJ=1TO500:NEXT:FORJ=1TO700:
PRINT@410,"SPEAK";:PRINT@410," ";:NEXT:PRINT@410,"STOP";:
FORJ=1TO1000:NEXT:CLS:RETURN
26 PRINT"HAS THE AUDIO TAPE BEEN PREPARED ? (Y OR N)"
27 Q$="" :Q$=INKEY$:IFQ$=""THEN27ELSEIFQ$="Y"THENZY=1:CLS:GOTO8
ELSEIFQ$="N"THENCLS:RETURNELSE27
28 CLS:FORLL=1TON:PRINTTAB(31)TW$(LL):NEXT:PRINT@896,"IF NO
ERRORS PRESS '0'":PRINT"IF AN ERROR PRESS '1'";
29 Q$="" :Q$=INKEY$:IFQ$=""THEN29ELSEIFQ$="0"THENCLS:RETURNELSE
IFQ$="1"THENZY=1:GOTO8ELSE29

```

Airdrie Ferguson
Box 40206
Casuarina, NT 5792
Australia ■

A BASIC CALENDAR

Anthony T. Scarpelli

This article is about a calendar program that I originally wrote in the language FORTH. Dr. Howe asked if it could be written in BASIC. I looked at the program and said, "why not?" What it does is to generate patterns on the screen. If you like a particular pattern, you can print it. Then a calendar for any year you choose is printed right after the pattern. An example of the printout is shown in figure 1.

I like this program because it can produce nice patterns, and the calendar it makes allows you to see the whole year at a glance. It's also nice to give away, and is a way to show off your computer. I don't like the BASIC program, though, because it's too slow. We'll get into the speeds of the various routines later, but for now let me explain how a FORTH program is turned into a BASIC one.

FORTH is a language of words. Instead of line numbers, GOTO's and GOSUB's, in FORTH we create words to do things. When we start to look at the subroutines in the program, we will see that there are a lot of them. Practically every one of the subroutines was a FORTH word. So instead of saying "GOSUB 1000," in FORTH we just use the word for that routine. In addition, as we can use subroutines within subroutines in BASIC, we use the words that we have created to create new words. It's very easy to do this in FORTH, but it can turn into a mess if we are not careful creating subroutines in BASIC.

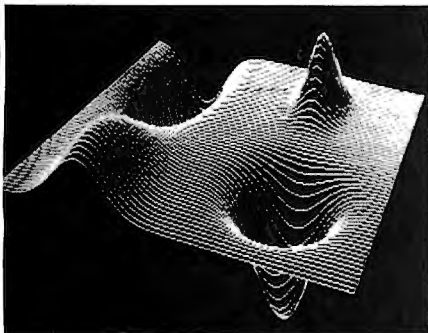
FORTH is inherently somewhat structured, so I tried to make this BASIC program as structured as possible, too. To me, structured means understandable. To make this possible, I write things in little chunks and then put them all together. Creating a program in FORTH means defining a word, making it work, then using it to make other words. The best way to write a FORTH program is to make the definitions of words short, and have many definitions.

In BASIC we can create a subroutine, test it and debug it, then use it in other subroutines. The difficulty about doing this in BASIC is that, because of the line numbers, the program flows from the top to the bottom. You call subroutines from the early line numbers, and the subroutines come later. This means you have to use higher line numbers for the subroutines, and then connect them together in the smaller line numbers. This can be tricky when trying to convert from FORTH to BASIC, as I found out.

If you look at lines 420 to 500, you will see the total program. It merely jumps from subroutine to subroutine. Each subroutine was written, tested, and debugged, and finally connected together here. This makes changing and debugging very easy. Simply by putting an apostrophe (') (REM) before any of the "GOSUBs" the whole routine can be skipped during the testing stage. Now we'll get to the routines.

The initialization routine from lines 120 to 390 are

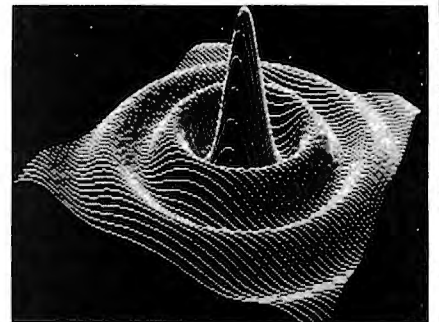
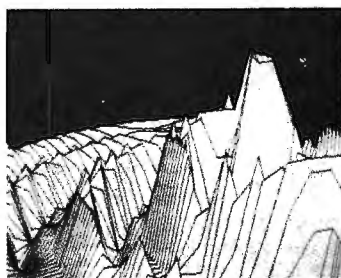
Surface Plot The Grafyx Solution[™] with Depth



Outstanding Grafyx. SURFACE PLOT lets you broaden your imagination by creating amazing three-dimensional views with Grafyx Solution. Micro-Labs' Grafyx Solution is a plug-in, clip-on board which gives you 98,304 points in a 512 x 192 matrix. That's sixteen times as many points as a standard Model III! Another unique feature is the ability to display the hi-res screen along with the normal text and low-res screen.

Flexible Grafyx. SURFACE PLOT allows you to enter an equation of the form $Z=F(X,Y)$ where Z is the height above the surface for a given X, Y coordinate. For example, entering the equation $Z=10-X^2$ draws a hill. The final picture can be viewed from any position in space so you can see an image from underneath, above, or even inside a hill or valley on the plot surface. You can also specify the size of the resulting image.

Complete Grafyx. The program automatically removes hidden lines for



best results. The documentation contains complete instructions and sample equations so that you will have your computer hard at work without delay. The finished plot can be saved on disk or printed on any of 20 popular printers.

The Grafyx Solution package is shipped from stock and includes the board, 44 programs including an 80-Column display driver and Extended Graphics Basic, and a 54 page manual all for \$299.95. The SURFACE PLOT program, twelve hi-res pictures, and manual is \$39.95. Shipping is free on pre-paid or COD orders. (Tx. residents add 5% sales tax.)

MICRO-LABS, INC. 214-235-0915
902 Pinecrest, Richardson, Texas 75080

typical. We have set up an array for the the month strings, one for the number of days in the month, a string for the days of the week, and an array called "TM" which holds three months of numbers. When I first started to write the program in FORTH, I wondered how I was going to print three different months across. After all, a printer prints only one line at a time, and we're trying to print across a page the three weeks of three different months.

It was simple, after I thought about it. You fill an array with all the numbers in the three months, and where there are no numbers to print, you fill in the spaces with zeros. Thus, each month actually consists of 42 characters. At the beginning there are empty characters, then the numbers of the month, then finally empty characters until the end. There are three sets of these. All we need to do, then, is to print seven characters, increment a pointer by 42, print seven more characters, and so on. Then decrement the pointer by 119 to start again at the next week of the first month.

We can see how we fill this array "TM" in the subroutine at line 4000. Remember that these little routines were originally FORTH words, and were converted practically verbatim, so to speak, so some of them may contain codes that were needed in FORTH, but not in BASIC. I have tried to keep this at a minimum, but some have slipped in. It doesn't matter, though, for the routines still work. So let's start with the routine at 4110.

When we come to this routine we have already computed the day of the week the month starts on. This is "ST" for start day. So we fill the start of the array with zero's up to the start day. We first check to see if the start day begins on the seventh. If it does, we skip this routine. The reason is to avoid a large blank space during the first week. This makes the calendar more pleasing to look at.

Next, we fill the middle part of the routine. It just loads the array with the days of the month up to the number of days in the particular month which is "LD" for last day.

Finally we fill up the rest of the month, up to the 42nd character, with zeros.

The subroutine at 4270 calculates the last day of the month. The subroutines at 4320 and 4350 calculate how many zeros are used in the last week (LW). The subroutine at 4110 calls all these subroutines and fills up one month, and finally the subroutine at 4000 does this three times for the three months across.

In FORTH these routines were written in a different order than presented here. The innermost subroutine was written first, and then we went downward to the last routine, which in the BASIC version became the first routine, at 4000.

In order to know where the start day (ST) begins, we have to do some calculations. These are listed in lines 3000 to 3090. The first calculation determines how many days there are from year zero to the year that the calendar is for. This is a large number, so I used double precision here, in case you want to have a calendar for the year 20,000, (we should live so long!). Then we have to find out how many leap years there were in the next line, and then we need what I call a fudge factor. This calculation is due to the inexactitude in earth's orbit around the sun.

Line 3040 gets us the total days which is used in the next line to determine the day of the week the year starts on. Then we do some more fudging, taken from my FORTH program, and finally we make a flag so that we know if the year is a leap year. If it is, of course, we want to add a one to the 28 in February.

Since BASIC is an offspring of FORTRAN, it is highly mathematical. The math routines are a little easier to implement in BASIC than in FORTH. However, these routines run a lot faster in FORTH.

Now that we have our array all filled up with three months of information, we have to get it printed out in a nice and regular fashion. This is done from lines 5000 to 5610.

The main routine starts at line 5000. This line sets the printer to expanded print when printing the year of the calendar. Then we print the year, then go back to normal print mode. If your printer uses different codes, these lines are the ones to change, or merely get rid of the codes if your printer can't use them.

Lines 5050 to 5070 prints the four quarters of the year. Line 5080 just cancels the the expanded mode which was set before we printed the pattern.

The subroutine at 5160 is self explanatory.

If we jump to the subroutine at 5460, we see that it just prints the strings from the array that contains the names of the months. Next, at line 5560, we print the string that is the names of the days three times.

Then back to the subroutine at line 5220. We must first fill the array, and then zero our count (C) variable. Since there are six lines to be printed, we set up our loop for that. We first tab over eight spaces, then jump to the subroutine that prints the three weeks, then decrement our count by 119. We mentioned this before.

The subroutine at 5310 will print three weeks. The increment at line 5330 causes a jump to the next months location in the array. The subroutine at 5380 prints one week of data. It first gets the number in the array, and checks to see if it is a zero, and if so prints three spaces. Otherwise it prints it using a format that eliminates the spaces that normally precede and follow a number.

The words that you see on the first line of each of these subroutines is the actual word that I used in my FORTH program. It's much easier calling up words when you need them then subroutine line numbers. For instance the FORTH version of the subroutine from lines 5310 to 5360 is as follows:

```
:PRNT3WK 30 DO PRNTWK 42 C + ! LOOP CR;
```

A short explanation will show you the differences and similarities between the two languages. The colon starts a FORTH definition, and the word to be defined follows it. Next we set up a loop that will execute three times starting at zero. The body of the loop is between the words "DO" and "LOOP." As you can see from the BASIC program, at this point we call a subroutine. In FORTH we just use the word that was previously defined, in this case "PRNTWK." Next we take the value 42 and add it to our count "C," with the word "+!," which stands for plus-store. Finally, after the loop we do a carriage return, "CR," and the semi-colon ends

continued on page 39

BUSINESS COMPUTING™

THE PUBLICATION FOR
SERIOUS SMALL BUSINESS
COMPUTER USERS

PUBLISHED BY

COMPUTRONICS!
MATHEMATICAL APPLICATIONS SERVICE™

50 N. PASCACK ROAD
SPRING VALLEY, NEW YORK 10977

(914) 425-1535

PUBLISHER: Howard Y. Gosman
EDITOR: Andrew Hofer

ADVISORY EDITORS: Dr. Peter Shenkin
Richard Kaplan
Beatrice Kahn
Steven Kahan

The purpose of BUSINESS COMPUTING is to provide and exchange information of interest to owners of under-\$10,000 computers. BUSINESS COMPUTING depends on reader participation for much of the information in each issue. Please send questions, comments, stories, press releases and any other information of interest to: The Editor, BUSINESS COMPUTING, 50 North Pascack Road, Spring Valley, New York 10977.

*** OSBORNE INTRODUCES A NEW COMPUTER ***

Osborne Computer Corp. has introduced a new, more powerful computer which will be available in two models -- the EXECUTIVE 1 and the EXECUTIVE 2. The EXECUTIVE 1 is available now, and the planned EXECUTIVE 2 will have a 16-bit processor (8088) that will make the unit software-compatible with the IBM Personal Computer.

The EXECUTIVE computers are portable units that look very much like the OSBORNE-1 (and may actually be considered an upgrade of the OSBORNE-1), but they have twice the memory, more disk storage, an amber 80-column display (7" diagonal) and may soon offer the industry's first built-in hard disk drive for a portable computer. The EXECUTIVE 1 features 128K of memory, while the EXECUTIVE 2 will have 256K.

Osborne will follow its tradition of supplying lots of good software with the new computers. Both computers are equipped with CP/M 3.0, the p-system, WORDSTAR, SUPERCALC, MBASIC, CBASIC, and the PERSONAL PEARL database management system. The EXECUTIVE 2 will include CP/M-86, MS-DOS 2.0, and GW BASIC.

Other features include a soft keyboard which allows the user to program the cursor control and numeric keys for special functions and two built-in character sets plus a utility for designing your own typestyles. Both computers will be capable of emulating several terminals, including the IBM 3270.

The EXECUTIVE 1 will list for \$2495, while the EXECUTIVE 2's price has not yet been announced. The company will also have a special on the OSBORNE 1 -- they will add the PERSONAL PEARL system to the other software packages sold with the OSBORNE 1, without raising the price. This offer will last through June.

Osborne has been very successful -- they virtually created the market for portable computers, which is now the fastest growing market in small computers. The proof: Osborne's sales went from \$6 to \$70 million last year!

***** TEXAS INSTRUMENTS OFFERS FREE 192K MEMORY BOARD *****

In a promotional program for their TI Professional Computer, Texas Instruments is now offering a 192K memory expansion board free to all owners who purchase the computer between April 1 and June 30, 1983. Since the memory card normally retails for \$700, this is a very worthwhile offer. This much extra memory will give new owners all the capacity needed to run many of the new integrated software packages that require expanded memory.

The price of the Professional Computer will remain at \$2595, which includes the computer with 64K, keyboard, video display, CRT controller, and disk. Undoubtedly, this promotion will be a big help in increasing TI's sales.

***** HOW TO GET RICH WITH YOUR MICROCOMPUTER *****

This is the title of a new book on strategies for making money with personal computers. HOW TO GET RICH WITH YOUR MICROCOMPUTER provides a lot of good information on how to set up, start and operate a profitable microcomputer-based business. Many of these businesses can be started with little investment, without extensive computer training, and can be operated from your own home.

Written in non-technical language, this book is the result of research and interviews with successful entrepreneurs, and the author's own experience running a computerized business from his home. Here is a sample of some of the many topics discussed:

Word Processing Service	Software Publishing
Income Tax Service	How to Advertise
Mailing Lists	Selecting a Computer
Business Primer	Selling Computer Supplies
Low Cost Startups	Computer Games

HOW TO GET RICH WITH YOUR MICROCOMPUTER is a 96-page, 8 by 11 sized book, and sells for \$12. The publisher even guarantees your satisfaction -- if you don't like the book, you can return it for a refund within 30 days. To order, write to Home Business News, 1221 Beaver Pike, Jackson, OH 45640.

***** COMMODORE SHIPPING THREE NEW SYSTEMS, WORKING ON FIVE MORE ****

COMMODORE has started to ship three new computers to its dealers-- the B 500, the C 128 and the PET 64. But they may be rushing things just a bit -- they've been shipping the computers without instruction manuals or even power cords.

The C 128 is a color computer based on COMMODORE's 6509 processor. The computer will support a 40-column screen and is expandable to 896K.

The C 128 sells for \$795.

The B 500 is their new business computer, and it's virtually identical to the C 128 except that it has an 80-column screen and does not support color. The B 500 will cost \$895.

The PET 64, the most recent upgrade of their educational series, is priced at \$695.

COMMODORE is also working on five other new systems, including a portable computer called the SX 100, and a "Lisa-type" computer based on the Z8000 processor.

The SX 100 will reportedly be available within about 90 days, and will feature a 5-inch screen. A green-screen version will be available for less than \$1000, and a full color version for about \$500

The "Lisa-type" computer will have a built-in monitor and presumably a computer "mouse" or similar cursor-pointing device like Apple's LISA computer. This system will be expandable to 896K and sell for under \$3000.

Two upgrades of the COMMODORE business line will also be released soon -- the B 700 and the BX 700. The BX 700 will use three different chips, COMMODORE's 6509, the Zilog Z80, and the Intel 8088, with the ability to switch back and forth between them for different software applications.

Still another new computer will be released -- the EXECUTIVE 64, with 128K of memory and a group of software packages included.

*** NEW MAINFRAME-TO-MICRO CONNECTION ***

VisiCorp (owners of VISICALC and other VISI- programs) and Informatics General Corporation have announced a cooperating pair of software products that provide an "intelligent" connection between personal computers and large databases on IBM mainframe computers. Designed primarily for use in medium to large corporations, the new software gives personal computer users access to corporate information maintained in centralized databases, while providing data processing efficiency, security and access control needed by corporate management information systems directors and data processing managers. The following is an overview of the advantages of this type of system, provided by VisiCorp and Informatics.

The new software will help eliminate confusion caused by duplication and inconsistency of data being manipulated by personal computer users. It will ensure that all users will be accessing the same data which is kept up-to-date, accurate and consistent in the central database.

Establishing even a "dumb" link between two computers can be a complicated and frustrating task. The mechanics of actually communicating between a mainframe and a microcomputer are straightforward, but require an extensive knowledge of communications

protocol. Many communications packages (called "terminal emulators") can handle these protocols, but handling the mechanics alone is not enough. Most mainframe systems are not well suited for first-time or casual users because of their complex sign-on procedures and command languages. Terminal emulation simply passes these characteristics through to the user. The result is communication that requires constant user intervention, and received data that is useful only for viewing and/or printing. The personal computer user so actually wants to DO something with the incoming data must manipulate it extensively, or sometimes even re-enter it by hand into a spreadsheet program or database manager. This is clearly ridiculous.

These two problems prevent users from gaining the main benefits of personal computer use. The users want to save time and improve productivity, which they cannot do if they are answering questions posed by the mainframe or learning complex mainframe systems. The users want to manipulate information, not just look at it. That is why the most popular application for personal computers (after word processing) is the electronic spreadsheet.

What is needed is an "intelligent" link; one which makes communications an easy process to learn. This requires taking advantage of the power of the microcomputer to shield users from the complicated process of signing onto the mainframe, composing a query for the database, waiting for it to run, and reformatting the data so it can be used by a personal computer application where it is understandable and available to the user.

This calls for a pair of programs: one that runs on the mainframe, which accepts queries from the personal computer and extracts the answers; and another running on the personal computer, which aids the user in framing the questions, communicates the query to the mainframe, and accepts the results in a form which can be viewed, printed, or -- most importantly -- manipulated in other programs.

This description brings up a few questions. Why use mainframe data at all? Why not just keep the data on personal computers? If mainframe data must be used, why not just use terminals to access it? Well, personal computers are powerful tools which allow manipulation of data in ways that are impossible using a terminal; but an intrinsic attribute of large corporations is that data needs to be shared among many users. This raises a number of issues:

QUALITY OF DATA: Many people trust computer output more than manual data. In the case of personal computers, this trust is sometimes misplaced. Recipients assume data was entered correctly; not always a valid assumption. This is a classic problem: garbage in, garbage out.

DUPLICATION OF DATA: Not only does this waste time during data entry, and space during data storage, but it poses a problem of updating. If data is changed in one place, is it changed everywhere?

CURRENCY OF DATA: Much of the information in large corporations changes daily, with some changing weekly, monthly, or yearly. Use of outdated information results in bad conclusions as surely as

typographic errors.

COMPARABILITY OF DATA: Firms with multiple personal computers are finding there can be apples-to-oranges comparisons attempted when one manager is storing results quarterly by units, while another stores them monthly by dollars, for example.

SECURITY OF DATA: Other than locking a diskette in a desk drawer there are few reliable techniques for ensuring limited access to personal computer data. Security surrounding most such machines is relatively light.

All of these problems have been solved, or are in the process of solution, on mainframe systems which have been coping with them for years. Many are being solved through the use of mainframe database management systems. The DBMS provides the tools, in combination with user procedures, to ensure that data is correct, up-to-date, stored only once, kept in a well-defined format and secure from unauthorized access.

If keeping data on a mainframe makes sense for all of these reasons, why not manipulate it there, too, via a computer terminal? The primary reason is simply cost-effectiveness. Many terminals are at remote sites and incur communications bills for the time they spend hooked to the mainframe, and even when located in the same building as the mainframe, there are costs associated with mainframe processing time. Mainframes are great as filing cabinets and calculators. It is an expensive waste of their time to "hold hands" with terminals. Many applications done on personal computers would, if done on a mainframe, tie it up so as to render it useless for other tasks. In short, much of what is done on personal computers simply is not practical for most mainframes.

The ideal system would extract the specifically needed data from a mainframe database, summarize it, and present it in manipulatable form to the personal computer. This involves a machine-to-machine dialogue in which the personal computer and the mainframe "shake hands" and pass information back and forth without unnecessary operator intervention. In the case of the VisiCalc/Informatics offering, this process is automatic and guaranteed to "download" correct information to the personal computer's applications program.

The new software, called "VISIANSWER" (initially designed for the IBM Personal Computer) and "ANSWER/DB" (for IBM mainframes and other mainframes), minimizes the amount of time the user spends communicating with the mainframe. Most of the time, the user is simply moving the cursor through a group of menus in the VISIANSWER program. Communications with the mainframe is required only briefly at the start of the session, and again briefly after the user's question is composed and ready for loading up into the mainframe. Final contact is established when the mainframe is ready to download the results into the personal computer.

The VISIANSWER program simplifies the process of requesting information from the mainframe's database, allowing the user to define

the request in an easy-to-understand form. Then the program poses the question in a form the mainframe can understand, and brings back the resulting information in a form that the personal computer's applications programs can understand -- specifically, VISICALC and other VISI-series programs.

VISIANSWER is for use in internal corporate systems (a previously announced product, the VISILINK program, ties a personal computer to a public database of business/economic information maintained by Data Resources, Inc.). The two packages together allow planners and managers to integrate, in a single spreadsheet, data from the outside world with internal corporate data. A manufacturing firm, for instance, could extract raw material prices from DRI and actual manufacturing costs from its corporate database, and use them as the basis for its manufacturing cost spreadsheet.

Mainframes and micros both have their places in the corporate information hierarchy -- both together can easily accomplish tasks that are unsuitable for either alone.

Mainframes will handle large tasks requiring uniformity. They will be used as data repositories, keeping information consistent, current, comparable and secure.

Personal computers will be used for small, ad hoc, personal tasks with individualized approaches. They will manipulate summaries prepared by the mainframes, and help generate accurate, up-to-date reports, charts and graphs.

For more information about the VISIANSWER and ANSWER/DB programs contact VisiCorp, 2895 Zanker Road, San Jose, CA 95134, (408) 92-6081; or Informatics General Corp., 21031 Ventura Blvd., Woodlnd Hills, CA 91364, (213) 887-9040.

*** NEW PORTABLE COMPUTER FROM NEC ***

NEC Home Electronics (USA) Inc., has introduced a portable computer for salesmen, managers who travel continuously, and busy executives who take work home. Adaptable to both office and home environments, NEC's new PC-8200 can communicate with many larger computers in offices, and also serve as a home computer for family finances and programming at home.

This machine is very similar to the new Radio Shack Model 100, and is going to be the Model 100's major competition (in fact, it may well be that it is manufactured by the same Korean firm that makes the Model 100).

The battery-operated PC-8200 comes with built-in software -- a text editing package, telecommunications software, and BASIC. Like the Model 100, it features an 8-line by 40-character LCD display with full cursor addressing, upper and lowercase characters, graphics characters, and other special symbols. For greater screen capacity, a future NEC option will allow users to plug into a full-screen display

(at home or in the office) with a CRT adapter.

The 8-bit PC-8200 offers standard 16K bytes of RAM, expandable to 64K, and an additional 32K with a RAM cartridge. A full 32K of ROM is standard, but can be expanded to 64K, with an additional 32K of ROM with a plug-in cartridge (the ROM will provide more built-in software for the computer).

The NEC's external RAM and ROM cartridges give it the programming capabilities of small business systems. Usually, data entered in business computers is stored on disk. NEC has incorporated cartridge technology, typical of home computers, into a business configuration and successfully bridged two markets. For additional storage capacity, the portable provides a floppy disk expansion port.

The PC-8200 is capable of communicating with virtually any other computers using its RS232 port and telecommunications software. It can download and upload data to any computer that uses ASCII data files (most micros and minicomputers do).

In addition to the RS232 port, NEC has included ports for cassette, printer, bar code reader, and floppy disk drives, as well as a general purpose expansion bus for future expansion.

NEC is developing software for finance, sophisticated text editing, and business applications which should be available sometime this summer.

For more information contact NEC Home Electronics (USA) Inc., 1401 Estes Avenue, Elk Grove Village, IL 60007; (312) 228-5900.

*** NEW MULTI-WINDOW OPERATING SYSTEM ***

Perhaps the most interesting new product demonstrated at the COMDEX/SPRING show was an operating system (or "operating environment") that can give the IBM Personal Computer multi-tasking, multiple-window features, like Apple's LISA computer. The new system, called DESQ, is likely to cause Apple and VisiCorp (who is developing a similar system called Visi-On) some alarm -- DESQ will sell for only \$395.

DESQ will allow IBM owners to run several different programs simultaneously, with the screen split into several "window" displays. According to Quarterdeck (the manufacturer), DESQ can run any applications software available for the PC -- for instance, running and displaying dBase II, VisiCalc, and WordStar all at the same time, AND pass data back and forth between the different programs.

*** DIGITAL RESEARCH BASIC ***

Here's another highlight from the COMDEX/SPRING show. Digital Research, the company which created the CP/M operating system, has introduced their own "Personal BASIC" which is designed for their

16-bit CP/M-86, MP/M-86 and Concurrent CP/M.

For a long time, their CP/M has dominated the market for small computer operating systems, while other companies (Microsoft in particular) have held the market for higher-level programming languages.

Now Microsoft is pushing XENIX, their multi-tasking, multi-user operating system for 16-bit computers, while Digital Research is trying to grab the market for a 16-bit BASIC, and at the same time increase their sales of CP/M-86.

Digital's Personal BASIC can run ANY program written in Microsoft BASIC, and, according to Digital, it outperformed Microsoft BASIC in more than 60 benchmark tests run on an IBM PC. Personal BASIC includes debugging aids that allow you to trace the execution of your program line-by-line or single-step, and continuously monitor the value of a given variable, plus an improved line editor. A couple of other good new features: when saving or loading a file, you will no longer have to put the filename in quotes, and, for the first time, it will be possible to issue simple CP/M commands directly from BASIC.

Personal BASIC will sell for \$150 and shipments will probably start by the end of May.

*** INEXPENSIVE INK-JET PRINTER ***

Docutel/Olivetti has announced a new ink jet dot matrix printer, which will reportedly sell for under \$575. Ink jet technology, which until now has been reserved for large and somewhat expensive computer printers, offers substantially quieter operation, as no striking blow is used to create character images or graphic dots.

The new printer, called the PR2300, uses a single-jet printing head to direct carbon particles onto regular paper in a 7 by 7 dot matrix. The paper is sensitized with electrical impulses in a pattern corresponding to the desired character or graphic pattern and attracts the carbon particles where they are permanently affixed. A similar process is used in electrostatic copying machines.

The PR2300 prints a full 96 ASCII characters with seven other foreign language and symbol sets or software selectable characters. It prints at up to 50 bidirectional lines per minute for a full 80 columns (that's about 120 characters per second). Three printing pitches, Pica (10 CPI), Elite (12 CPI) and Mikron (15 CPI), are available with variable line spacing. The PR 230 offers normal, compressed, double height, double width and bold printing with either single or double-line underlining.

The PR2300 is a compact 15" by 10.5" by 4.5", weighs 11.5 pounds, and uses a standard Centronics parallel or RS232C serial interface.

the definition. Short and sweet, and very powerful.

That's all there is to printing a calendar. If your printer is capable of printing the TRS-80 graphics, then you can have a nice pattern printed above the calendar. The pattern generator routine starts at line 1000. When I implemented this pattern routine from the FORTH routine, the first thing I noticed about it was its slowness. It took about four minutes to print a full pattern to the screen. It takes four seconds in FORTH! However, rather than write a routine that takes less time, I leave that to you, and I'll stick to the FORTH calendar.

Since I put in a speed-up modification in my computer, line 1010, after it initializes the X and Y variables, will turn the mod on. If you have a speed-up mod, take out the apostrophe before the colon, or put in your own turn-on statement.

The next line sets the up the variable that actually determines how full the pattern becomes. When you see the pattern on the screen, it looks very nice and full, however, when it prints out, it looks a lot more scanty. By increasing the 100 in this line to 200, your pattern will fill up more, and look better when printed out, but it will take longer to produce too.

The loop in lines 1030 to 1070 takes "C" and plots the pixels with the GOSUBs in line 1050. The subroutines, from line 1310 to 1380, are just some of the reasons why it takes so long to print a pattern, but these were how I set it up in FORTH. Each subroutine calls another subroutine which is located in the group at lines 1400 to 1430. These routines get a random number, and then they are added together to give us a value from zero to six. This then is used as the variable in line 1120 that GOTO's a routine that will increment "X" by one or increments or decrements "Y" by one or two. Once we have a new value for "X" and "Y," we have to check to see if it's going to write a pixel off the screen. This is checked in lines 1130 to 1150. Finally in lines 1160 to 1190 we set the pixel on in all four quarters of the screen.

I know that all this jumping around slows things down. By removing the spaces, GOSUB's, and by combining statements onto one line, this routine can be considerably improved. However, it is easier to understand when written this way. Also, you can put your own pattern routine here if you like. When you first see the pattern that this routine produces, I'm sure you'll like it. You can sit and watch it all day long. If you don't speed it up, though, that's how long it will take.

Now that we have a nice pattern on the screen, we have to get it to the printer. What we need here is a screen dump routine, and we will find that starting at line 2000. The first two lines of this routine reset the printer and put it into an emphasized mode. When you are debugging this program, I suggest you turn line 2010 off. Printing the calendar without this line will cut the printing time in half.

In line 2020 we have a variable "O" for offset. If your MX-80 printer is not in the TRS-80 mode, then you have to add 32 to all the graphic block character numbers in order to print them. This offset is for that purpose.

Line 2040 prints the top line of the border that surrounds this pattern, and line 2060 prints the two

surrounds the pattern, and line 2060 prints the two sides of the border. Now to print the screen.

Video memory starts at 15360. The memory consists of 1024 bytes of RAM, and each letter, space or other symbol you see on your screen is somewhere in there. We can PEEK it like any other memory. When the pattern is on the screen, the only characters in video RAM are spaces (ASCII decimal 32) or graphic characters from decimal 129 to 191. If we PEEK a space character, we don't need any offset. However, if we get hold of a graphic block, we want to add the offset for the printer.

This is all done in the loop starting at line 2080. Since we have 16 lines on the screen, this is the loop limit, and since there are 64 characters in a line, this is the inner loop at line 2100. Simple.

The rest of the lines from 2170 to 2210 print the rest of the border.

Except for the instructions routine, and the few routines at the end of the program, that's it. The program is very modular so it is easy to understand and modify. If you are interested in the FORTH version of this program, send me a SSAE and I'll send you the listing. The BASIC version prints out the entire calendar in about four and a half minutes when in the emphasized mode. The FORTH version does it in three and a half minutes, which is as fast as the printer can print it.

I hope you like your calendars. Now, at any time you want, you can print up a good looking new year for yourself.

Names of some variables:

CN = MONTH COUNT
DA = ARRAY THAT HOLDS THE NUMBER OF DAYS IN A MONTH
FU = FUDGE FACTOR IN MATH ROUTINE
IC = INCREMENT FOR ARRAY FILL-UP
LD = LAST DAY IN THE ARRAY FILL-UP
LF = LEAP YEAR FLAG
LP = TOTAL LEAP YEARS IN MATH ROUTINE
LW = LAST WEEK
MO = STRING ARRAY THAT HOLDS NAMES OF MONTHS
NX = HOLDS THE NUMBER OF DAYS IN THE NEXT MONTH
O = PRINTER OFFSET
SB = SUBTOTAL DAYS IN MATH ROUTINE
ST = START DAY OF THE YEAR AND MONTH
T = TAB COUNT FOR PRINTOUT
TD = TOTAL DAYS UP TO PRESENT YEAR
TM = ARRAY THAT HOLDS THREE MONTHS OF DATA
WE = STRING OF THE NAMES OF THE DAYS OF THE WEEK
YE = THE YEAR OF THE CALENDAR

Most of the other variables are used for indexes, counting and temporary storage.

PROGRAM LISTING

```
10 REM *** CALENDAR PROGRAM ***
20 '
30 'INITIALIZATION
35 '
40 CLEAR 0: CLEAR 200
50 DIM MO$(12), TM(127), DA(12)
60 WE$=" S M Tu W Th F S "
```

```

70 MOS(1)=" Jan "
80 MOS(2)=" Feb "
90 MOS(3)=" Mar "
100 MOS(4)=" Apr "
110 MOS(5)=" May "
120 MOS(6)=" Jun "
130 MOS(7)=" Jul "
140 MOS(8)=" Aug "
150 MOS(9)=" Sep "
160 MOS(10)=" Oct "
170 MOS(11)=" Nov "
180 MOS(12)=" Dec "
185 '
190 DA(1)=31
200 DA(2)=28
210 DA(3)=31
220 DA(4)=30
230 DA(5)=31
240 DA(6)=30
250 DA(7)=31
260 DA(8)=31
270 DA(9)=30
280 DA(10)=31
290 DA(11)=30
300 DA(12)=31
310 '
320 'MAIN PROGRAM
330 '
340 GOSUB 6000 ' INSTRUCTIONS
350 GOSUB 1000 ' PRODUCE PATTERNS
360 GOSUB 2000 ' PRINT SCREEN
370 GOSUB 3000 ' DO MATH
380 GOSUB 5000 'PRINT CALENDAR
390 GOSUB 9000 'ANOTHER CALENDAR?
500 END
997 '
998 ' PATTERN ROUTINE
999 '
1000 CLS:RANDOM
1010 X=1:Y=23 :OUT254,1 'SPEED UP
1020 C=100+RND(200)
1030 FOR B=1 TO C
1040   FOR I=1 TO 8
1050     ON I GOSUB 1280,1290,1300,1310,1320,1330,1340,1350
1060   NEXT I
1070 NEXT B
1075 OUT254,0 'SLOW DOWN
1080 GOSUB 7000:IF Z$="P" THEN RETURN
1090 CLS:GOTO 1010
1100 ON A GOTO 1180,1200,1220,1240,1260,1170,1170
1110 IF Y<1 THEN Y=1
1120 IF Y>47 THEN Y=47
1125 IF X>127 THEN X=1
1130 SET(X,Y)
1140 SET(127-X,Y)
1150 SET(X,47-Y)
1160 SET(127-X,47-Y)
1170 RETURN
1180 X=X+1
1190 GOTO 1110
1200 Y=Y+1
1210 GOTO 1110
1220 Y=Y-1
1230 GOTO 1110
1240 Y=Y+2
1250 GOTO 1110
1260 Y=Y-2
1270 GOTO 1110
1280 GOSUB 1390:GOSUB 1400:RETURN

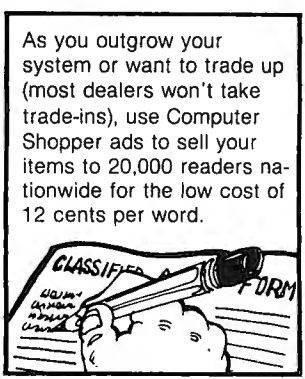
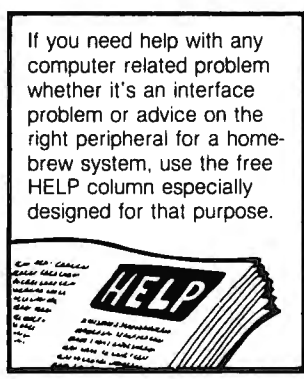
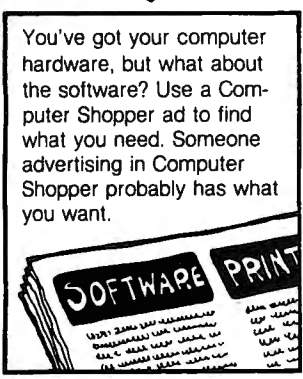
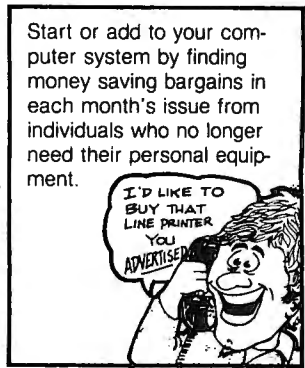
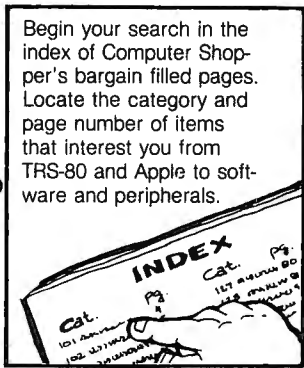
```

```

1290 GOSUB 1370:GOSUB 1400:RETURN
1300 GOSUB 1390:GOSUB 1400:RETURN
1310 GOSUB 1370:GOSUB 1400:RETURN
1320 GOSUB 1390:GOSUB 1400:RETURN
1330 GOSUB 1370:GOSUB 1400:RETURN
1340 GOSUB 1380:GOSUB 1400:RETURN
1350 GOSUB 1370:GOSUB 1400:RETURN
1360 GOTO 1280
1370 D=RND(3):RETURN
1380 E=RND(3):D=RND(3):RETURN
1390 F=RND(3):E=RND(3):D=RND(3):RETURN
1400 A=D+E+F:A=A-3:GOTO 1100:RETURN
1410 RETURN
1996 '
1997 'SCREEN PRINT ROUTINE
1998 ' FOR USE WITH MX-80 PRINTERS
1999 '
2000 LPRINT CHR$(27);CHR$(64); 'RESET PRINTER
2002 LPRINT CHR$(27);CHR$(69); 'EMPHASIZED MODE
2005 O=32:T=6 'OFFSET & TAB
2008 LPRINT:LPRINT TAB(T);
2010 LPRINT CHR$(151+O);STRING$(70,131+O);CHR$(171+O)
'TOPLINE
2015 LPRINT TAB(T);
2020 LPRINT CHR$(149+O);STRING$(70," ");CHR$(170+O) 'SIDES
2030 X=15360 'START OF VIDEO MEMORY
2040 FOR I=1 TO 16
2050   LPRINT TAB(T);CHR$(149+O);" ";
2060   FOR J=0 TO 63
2065     A=PEEK(X+J):IF A=32 THEN A=A-32
2070     LPRINT CHR$(A+O);
2080   NEXT J
2090   LPRINT " ";CHR$(170+O)
2100   X=X+64
2110 NEXT I
2115 LPRINT TAB(T);
2120 LPRINT CHR$(149+O);STRING$(70," ");CHR$(170+O) 'SIDES
2125 LPRINT TAB(T);
2130 LPRINT CHR$(181+O);STRING$(70,176+O);CHR$(186+O)
'BOTLINE
2140 LPRINT:LPRINT
2150 RETURN
2997 '
2998 ' MATH ROUTINES
2999 '
3000 SB#=YE*365+1 'SUBTOTAL DAYS
3010 LP=INT((YE-1)/4) 'LEAPYEAR
3020 FU=INT((((YEAR-1)/100)+1)*3/4) 'FUDGE FACTOR
3030 TD#=(SB#+LP)-FU 'TOTAL DAYS
3040 ST=((INT(TD#/7))*-7)+TD# 'START DAY
3050 IF ST=0 THEN ST=7
3060 ST=ST-1
3070 LF=YE/4 'LEAP YEAR FLAG
3080 IF LF=INT(YE/4) THEN LF=1 ELSE LF=0
3090 RETURN
3997 '
3998 'FILL ARRAY ROUTINE
3999 '
4000 FOR J=1 TO 3 'FILL THREE MONTHS
4010   GOSUB 4060
4020 NEXT J
4030 IC=0
4040 RETURN
4050 '
4060 GOSUB 4280 'FILL ONE MONTH
4070 GOSUB 4120
4075 IC=IC+42
4100 RETURN
4110 '

```

How to Buy or Sell Computer Equipment and Software



Computer Shopper is THE nationwide publication for buying, selling and trading Micro and Mini-computer equipment and software. Each issue has over 100 pages full of bargains of new and used equipment.

You can save hundreds of dollars by getting the equipment you need from the hundreds of classified ads individuals place in Computer Shopper every month.

Now is the time for you to join over 20,000 other computer users who save time and money with a subscription to Computer Shopper.

Subscribe today and get your first issue and a classified ad absolutely FREE. Type or print your ad on a plain piece of paper and send it along with your subscription.

Just fill in the coupon or MasterCard or VISA holders can phone for faster service and start making your computer dollar go further today.

Cut out and mail to: **COMPUTER SHOPPER**
P.O. Box F535 • Titusville, FL 32780

Yes, I'll try Computer Shopper, I understand that if I'm not satisfied with my first issue I can receive a full refund and keep the first issue free.

- 1 year \$10.00 (\$30.00 in Canada)
- I have enclosed my free classified ad.
- I want to use my free ad later, send me a coupon.

NAME: _____

ADDRESS: _____

CITY: _____

STATE: _____ ZIP: _____

 **COMPUTER SHOPPER**
P.O. Box F535 • Titusville, FL 32780
Telephone: 305-269-3211

```

4120 IF ST=7 THEN ST=0:GOTO 4190 'FILL START
4140 FOR I=1 TO ST
4150 TM(IC+I)=0
4160 NEXT I
4170 '
4190 FOR I=1 TO LD 'FILL MIDDLE
4200 TM(IC+I+ST)=I
4210 NEXT I
4220 '
4225 GOSUB 4330 'FILL END
4230 FOR I=1 TO LW
4240 TM(IC+I+ST+LD)=0
4250 NEXT I
4260 GOSUB 4338
4265 RETURN
4270 '
4280 CN=CN+1 'MONTH COUNT
4290 NX=DA(CN) 'NEXT MONTH
4300 IF NX=28 AND LF=1 THEN LD=NX+1 ELSE LD=NX 'LAST DAY
4310 RETURN
4320 '
4330 LW=(43-LD)-ST 'LAST WEEK
4335 RETURN
4336 '
4338 GOSUB 4330 'START MONTH
4340 IF LW>7 THEN LW=LW-7
4350 LW=LW-1
4360 ST=7-LW
4370 RETURN
4997 '
4998 'PRINT CALENDAR ROUTINE
4999 '
5000 LPRINT CHR$(27);CHR$(83); 'EXPANDED PRINT
5005 LPRINT TAB(20);YE 'PRINT NUMBER OF YEAR
5010 LPRINT CHR$(27);CHR$(84) 'CANCEL EXPANDED PRINT
5011 LPRINT
5013 K=0
5015 FOR H=1 TO 4 'PRINT YEAR
5022 GOSUB 5040
5023 NEXT H
5024 LPRINT CHR$(27);CHR$(70); 'CANCEL EMPH. MODE
5025 FOR I=1 TO 3
5026 LPRINT
5027 NEXT I
5028 CN=0
5029 RETURN
5030 '
5040 GOSUB 5310 'PRINT THREE MONTHS
5050 GOSUB 5410
5060 GOSUB 5090
5070 LPRINT
5075 RETURN
5080 '
5090 GOSUB 4000 'PRNT3MON
5095 C=0
5100 FOR I=1 TO 6
5105 LPRINT STRING$(8," ");
5110 GOSUB 5160
5120 C=C-119
5130 NEXT I
5145 RETURN
5150 '
5160 FOR L=1 TO 3 'PRNT3W
5170 GOSUB 5220
5180 C=C+42
5190 NEXT L
5200 LPRINT
5205 RETURN
5210 '

```

```

5220 FOR P=1 TO 7 'PRNWK
5230 N=TM(C+P)
5240 IF N=0 THEN LPRINT" ";;GOTO 5270
5250 ' IF N<10 THEN LPRINT" ";
5260 LPRINT USING "###"; N;
5270 NEXT P
5280 LPRINT" ";
5290 RETURN
5300 '
5310 LPRINT STRING$(17," "); 'PR3M
5320 FOR I=1 TO 2
5325 K=K+1
5330 LPRINT MO$(K);STRING$(18," ");
5350 NEXT I
5360 K=K+1
5370 LPRINT MO$(K);
5380 LPRINT
5390 RETURN
5400 '
5410 LPRINT STRING$(8," "); 'PR3W
5415 FOR I=1 TO 3
5420 LPRINT WE$;
5430 NEXT I
5440 LPRINT
5450 RETURN
5997 '
5998 ' INSTRUCTIONS ROUTINE
5999 '
6000 CLS
6010 PRINT STRING$(12," ");"*** CALENDAR PROGRAM ***"
6020 PRINT:PRINT
6030 PRINT"THIS PROGRAM WILL FIRST ASK YOU FOR THE YEAR YOU"
6040 PRINT"WANT THE CALENDAR FOR. IT WILL THEN GENERATE
RANDOM"
6050 PRINT"PATTERNS ON THE SCREEN. IF YOU PRESS THE <SPACE>"
6060 PRINT"BAR, THE PATTERN WILL CHANGE. IF YOU PRESS THE
<P>"
6070 PRINT"KEY, THE SCREEN WILL BE PRINTED ON THE PRINTER
AND"
6080 PRINT"THEN THE REST OF THE CALENDAR WILL ALSO BE
PRINTED."
6090 PRINT"AT THE END OF THE PRINTOUT, YOU WILL BE ASKED IF"
6100 PRINT"YOU WANT ANOTHER CALENDAR."
6110 PRINT:PRINT"PRESS <SPACE> BAR TO CONTINUE > ";
6120 GOSUB 7000
6130 GOSUB 8000
6140 RETURN
6997 '
6998 ' INKEY ROUTINE
6999 '
7000 Z$=INKEY$:IF Z$<>" " AND Z$<>"P" THEN 7000 ELSE RETURN
7997 '
7998 ' QUERY ROUTINE
7999 '
8000 CLS
8010 INPUT "WHAT YEAR DO YOU WANT THE CALENDAR FOR";YE
8020 RETURN
8997 '
8998 'FINAL ROUTINE
8999 '
9000 CLS
9010 INPUT "DO YOU WISH TO DO ANOTHER CALENDAR (Y/N)";Z$
9020 IF Z$="N" THEN CLS:END
9030 IF Z$<>"Y" THEN 9000
9040 GOSUB 8000
9050 GOTO 350
Anthony T. Scarpelli
98 Foxcroft Drive
Scarborough, ME 04074 ■

```

KOPYKAT

Model III Features for the Model I

Joe W. Rocke

Love my Model I with its independent keyboard! Hate the not so sly innuendos of those that look down their noses at it. "You call that a cursor?" "My Purple Plum has key repeat; does your Trash-80?" "When you goin' to graduate to a Model III?"

Many of us Model I owners have put up with slights like that for far too long. Throughout the past three years I have borne the love-hate affair with my Model I in silence. But the needling about upgrading to a Model III was too much! I vowed to show the skeptics that the Model I has more potential than most of them can tap.

Programming is the secret to tapping the power of any computer. With this in mind, I set about trying to emulate the more obvious features of the competition, namely the blinking cursor and key repeat features. Alas, the goal exceeded my amateur talents until a friendly Genie provided the needed clues to success.

PROGRAM CRITERIA

My primary goal was to develop a program that would provide a block cursor and automatic key repeat. Response speed, disk operating system (DOS) compatibility, and low memory overhead were major considerations for a program of this type. The program also had to be compatible with Level II BASIC and tape operations. Memory protection would be required to achieve maximum utilization of the features. All of which could be accomplished only by an assembly language program.

Sure, a blinking cursor can be programmed in BASIC. By using a variation of a key debounce program, a form of auto key repeat can be achieved. But alas, neither would survive in the real world of serious computer use.

THE BIRTH OF KOPYKAT

The KOPYKAT program source code of Listing 1 is the proof of the pudding, so to speak. A key debounce routine was modified to provide the auto key repeat feature. A block cursor routine was developed using a BASIC equivalent as a guide. The major problem at this point was in linking these two routines into a single workable program. But alas, the ROM routines took first priority, nullifying the routines so laboriously developed. It was at this point that my friendly Genie came to the rescue with "hook lines."

The KOPYKAT program listing reflects enhancements that were added once the "hook line" approach was mastered. The results are far beyond the original expectations. The program includes the following Model III "work alike" features:

- Automatically sets memory size protection for DOS.
- Provides a blinking or no-blink block cursor that can be changed via a keyboard POKE statement.
- Auto key repeat with key debounce as an added benefit.
 - Video line print mode that includes a graphics print capability and "escape" feature.
 - A "call Debug" mode that can be modified for

compatibility with most DOS's.

HOOK LINES

The use of "hook lines" paved the road to sweet success. Not that I can lay claim to originating this approach. The kudos go to my "Genie" who is an expert at programming for the TRS-80. It would be very difficult to achieve a workable program of this type without benefit of the hook line approach.

No, Charlie, I'm not talking about goin' fishin'! The term comes from "hooking" an address for a routine into the normal keyboard scan. The hook lines allow the ROM to perform its normal functions up to the point where control must be diverted to a routine stored in RAM. By using a machine code routine, all operations occur so rapidly that the external RAM routines are performed as fast as those in ROM.

The blinking cursor and key repeat routines are actually two separate routines. To make them work together and in conjunction with the normal ROM routines, the address of these routines must be "hooked" into the program upon its initialization. This, plus the video screen dump feature, is done in lines 450 through 550 of the source code listing. Once these addresses are loaded to memory, control can revert to ROM by the jump to DOS or BASIC as the case may be.

In normal operation, ROM is constantly scanning addresses 4016H and 401EH which are the control blocks to keyboard and video. There is one byte at address 4016H and another at 4017H. Normally these bytes will just return control to the ROM scan. So the key to diverting the scan to an external routine is to "hook in" on these address bytes.

Key repeat hooks in at 4016H because it is the keyboard scan routine. This address is pointed to at the start of a normal ROM scan. By hooking into this address the scan is looped through the external routine, bypassing the normal ROM routine. Key repeat also adds extra code to repeat the key if it is held down past the delay count provided in KOPYKAT. If not, the scan simply jumps to the bytes that allow ROM to continue its normal scan. The external program actually duplicates some of the ROM instructions in addition to those which provide the key repeat feature.

The block cursor operates in much the same manner. Addresses 401EH and 401FH point to the routine in ROM. In this case we tie in the cursor routine at that point but instead of bypassing any of the ROM code, the block cursor routine is called. The call is pushed to the stack and executed after which it eventually comes to a RETurn instruction, popping the routine return address from the stack. At this point the block cursor is substituted for the normal cursor.

In summary, initialization is rather simple. The address of the key repeat routine is loaded into register HL and transferred to 4016H, which completes that loop. The same thing is accomplished for the block cursor routine by loading that address into HL register and then transferring it to 401EH. Upon completion of

these operations, program control reverts to normal via a jump to DOS or BASIC.

THE PROGRAM LISTING

Keying in the program source code is not as formidable a task as it may first appear. Readers familiar with machine code need only key in portions of the program that suit their needs. The remarks are included for those interested in learning how the program works. Keying in the program sans the remarks will ease the task considerably.

The use of EQUates in line 60 through 380 eliminate the need to remember frequently used addresses during program development. The list also provides a ready reference of ROM addresses for novice machine code programmers such as *myself*.

The operating routines are hooked into specific locations during program initialization (lines 470 through 670). These lines can be overwritten after program initialization. The working portion of the program begins with the cursor routine at line 780. Remark statements accompanying each line explain the program flow.

The routines in the program were developed as individual modules. This approach eases the chore of checking and debugging a routine during its development. After each routine proved to be workable, it was added to the main source listing. ORG and END lines were deleted after merging the listings, as a program can have only one of each.

The reader can customize the program to include only the features desired by making minor changes when keying in the source code. For example, the video line print routine beginning at line 2340 can be used as a stand-alone program. To do so, it will be necessary to key in the EQUates used in the listing, add a program ORGin statement, and "hook" the VIDPNT address into the initialization block. The same general procedure can be applied to the other major routines.

The keys used to initiate a video dump to printer can be changed by substituting lines 2410 and 2420 for lines 2360 and 2370. Lines 2760 through 2810 allow a graphics screen display printout on Epson printers; these lines can be left out for printers that have standard TRS-80 graphics capability.

A DOS Debug call function is provided in lines 2520 through 2550. These were added after I failed to take into consideration that the key repeat feature would nullify DOS routines called by pressing a specific combination of keys. As my DOS happens to use the "1" and "0" keys for the Debug call, line 2530 was written accordingly. The compare (CP) value will have to be changed to accommodate a different call combination.

Key repeat time can be altered by changing the delay factor in line 2150. Decimal 2000 provides a delay time that appears to be satisfactory to both one-finger and touch typists. Repeat delay time may be decreased by using a lower number, or increased by using a higher decimal value. Use caution in changing the value as too little delay will cause what appears to be key bounce. Too long a delay makes touch typing awkward, and characters may be lost in the process.

USING THE PROGRAM

Assemble the program as a command module for DOS use. After calling the program, a blinking cursor will appear after the READY prompt. The blink feature may

be turned off or on at will by typing a shift/left arrow. Additionally, the cursor block may be changed to an ASCII character of your choice by typing the following from BASIC command mode: POKE 16419,nnn . . . where nnn equals the ASCII equivalent of the desired character.

Any key will automatically repeat when held down longer than a fraction of a second. As the program is compatible with most editor/assemblers, this is a handy feature in formatting machine code listings. The feature is also available in BASIC modes of operation.

Contents of the video screen may be dumped to the line printer by pressing the "V" and "P" keys simultaneously (or "ZX" keys, depending on which lines are typed in). The printout may be aborted at any time by pressing the SPACEBAR. The latter feature gives the user full control over the video line print function.

POSSIBLE PROBLEMS

Several versions of KOPYKAT were written, assembled and tested before arriving at the debugged version presented here. Although the program works flawlessly on my system, that is no assurance it will work with all systems. Slight differences in the ROMs used in some versions of the Model I may create what appears to be a bug in the program. One such anomaly may show up as an "Out of Memory" error message after the program is loaded. If this should occur, pressing the ENTER key once or twice will usually solve the problem.

TRSDOS™ users who have the lower case modification installed will find that the program overlays the lower case driver. In such case it will be necessary to originate KOPYKAT at a lower address, and call it after the driver is loaded.

It will be necessary to set Memory Size in the usual manner if you assemble the listing as a Level II tape-based system program. Attempting automatic memory size protection in Level II BASIC is tricky business at best due to variations in ROM sets. In addition, line 690 containing the jump to BASIC will have to be used in place of line 680.

The program may be used in 16K and 32K memory size systems by changing the originating address. To arrive at the proper value for program origin, subtract 380 decimal from the top of memory decimal equivalent. The resulting value represents program origin in decimal form.

KOPYKAT provides most all the features found in "the other models" plus a few that some do not have. Once you have the program up and running, you can display your Model I's capabilities to the scoffers with pride! This is your opportunity to say "Can your system do this?"

```
00010 ;*****
00020 ;** KOPYKAT ... MODEL I UPGRADE TO MODEL III FEATURES **
00030 ;** VERSION 1.06 ... BY: JOE W. ROCKE 7/82 **
00040 ;*****
00050 ;
00060 SHFTUP EQU 09H ;'SHIFT/UP ARROW' VALUE
00070 GRPBLK EQU 0B0H ;GRAPHIC BLOCK - CURSOR
00080 CRLF EQU 0DH ;CARRIAGE RETURN
00090 SHFTLT EQU 21H ;'SHIFT/LEFT ARROW' VALUE
00100 NORCUR EQU 5FH ;NORMAL 'ROM' CURSOR
00110 DELAY EQU 0060H ;ROM DELAY ROUTINE
00120 ROMROW EQU 03FBH ;ROM ROW COUNTER
00130 BASIC EQU 06CCH ;BASIC EXIT
```

```

00140 LPORT EQU 37E8H ;LINE PRINTER PORT
00150 KEYST EQU 3801H ;START OF KEYBOARD
00160 VPROW EQU 3804H ;'VP' ROW ADDRESS
00170 ZXROW EQU 3808H ;'ZX' KEYS ROW
00180 NUMROW EQU 3810H ;0 - 7 KEYS ROW
00190 SPCBAR EQU 3840H ;'SPACEBAR' ROW
00200 SHIFT EQU 3880H ;'SHIFT KEY' ROW
00210 SHFCNT EQU 38C0H ;LOOKS FOR 'SHIFT/ARROWS'
00220 VIDST EQU 3C00H ;START OF VIDEO RAM
00230 KBDBLK EQU 4016H ;KEYBOARD CONTROL BLOCK
00240 CURS1 EQU 4019H ;USED BY SOME 'DOS'
00250 VDBLK1 EQU 401AH ;PART OF 'CONTROL' BLOCK
00260 VDBLK EQU 401EH ;VIDEO CONTROL BLOCK
00270 CURP EQU 4020H ;CURSOR POSITION
00280 CURCHR EQU 4023H ;CURSOR CHARACTER STORE
00290 DOS EQU 402DH ;'DOS' EXIT
00300 KBWRK EQU 4036H ;KEYBOARD WORK AREA
00310 DOSM EQU 4049H ;'DOS' MEMORY SIZE?
00320 BASM1 EQU 40A0H ;BASIC 'STRING' SPACE
00330 ;* ;CLEAR 50+2
00340 BLINE EQU 40A2H ;BASIC CURRENT LINE
00350 BASM2 EQU 40B1H ;TOP OF 'BASIC' MEMORY
00360 BASM3 EQU 40D6H ;BASIC MEMORY SIZE?
00370 BASM4 EQU 40E8H ;STACK POINTER, POINTER
00380 ;* ;54 BYTES
00390 ;
00400 ;*****
00410 ;** INITIALIZATION 'BEGINS' HERE AND CAN BE WRITTEN **
00420 ;** OVER BY BASIC OR OTHER PROGRAMS - ONLY USED ONCE **
00430 ;*****
00440 ;
00450 ORG 0FE87H ;TOP OF 48K MEMORY
00460 ;
00470 BEGIN LD HL,BLINK ;GET 'BLINK ROUTINE'
00480 LD (KBDBLK),HL ;INTO KEYBOARD CONT. BLK.
00490 LD HL,(VDBLK) ;GET OLD VIDEO CALL
00500 LD (NWCALL),HL ;OUR ROUTINE
00510 LD HL,CURSOR ;NOW TRANS. ROUTINE TO
00520 LD (VDBLK),HL ;VIDEO CONTROL BLOCK
00530 LD HL,CURSOR ;SET 'DOS' MEMORY SIZE
00540 LD (DOSM),HL ;POINTER WITH ENTRY ADDR.
00550 LD HL,CURSOR-2 ;NOW TO GET IT FOR BASIC
00560 LD (BASM2),HL ;DISK BASIC MEMORY SIZE
00570 LD (BASM3),HL ;HERE TOO - ALL BASIC
00580 LD HL,CURSOR-52 ;CLEAR (50 BYTES)
00590 LD (BASM1),HL ;FOR 'DISK BASIC' STRINGS
00600 DEC HL ;NEED (2) MORE
00610 DEC HL ;THAT DOES IT
00620 LD (BASM4),HL ;FOR 'DISK BASIC' STACK
00630 LD A,GRPBLK ;GET THE GRAPHIC BLOCK
00640 LD (CURCHR),A ;INTO POKE AREA OF RAM
00650 LD (CURS1),A ;FOR 'SOME' DOS
00660 LD A,01H ;TURN 'ON' BLINK ROUTINE
00670 LD (FLAG),A ;AT OUR FLAG CHECK
00680 JP DOS ;GOTO 'DOS'
00690 ;* JP BASIC ;USE IN PLACE OF 'DOS'
00700 ;IF 'LEVEL II' ONLY
00710 ;
00720 ;*****
00730 ;** MAIN PROGRAM STARTS HERE 'AFTER' INITIALIZATION **
00740 ;** MEMORY SIZE? IS 'AUTO SET' FOR DISK BASIC - **
00750 ;** START HERE = 65216 FOR 48K 'LEVEL II BASIC' **
00760 ;*****
00770 ;
00780 CURSOR CALL 0000H ;BECOMES CALL VIDEO
00790 NWCALL EQU $-2 ;DRIVER ROUTINE
00800 PUSH AF ;SAVE 'AF'
00810 PUSH DE ;SAVE 'DE'
00820 LD DE,(CURP) ;GET CURSOR POS'N
00830 LD A,(DE) ;SWITCH 'CHAR.' TO 'A'
00840 CP NORCUR ;IS 'ROM' TRYING TO LOAD
00850 JR NZ,OK ;THE 'OLD' CURSOR CHAR.?
00860 LD A,(CURCHR) ;YES = GET THE NEW ONE
00870 LD (DE),A ;PUT IT IN PLACE
00880 OK POP DE ;RESTORE 'DE'
00890 POP AF ;RESTORE 'AF'
00900 CALL VIDPNT ;CHECK ON PRINTER
00910 RET
00920 ;
00930 ;*****
00940 ;** ROM CHECKS THE MODEL I KEYBOARD AT 03E3H BY THE **
00950 ;** WAY OF 4016H (KEYBOARD CONT. BLOCK) - WE WILL TIE **
00960 ;** IN THE 'BLINK & KEY REPEAT' THEN LOOP BACK TO ROM **
00970 ;** AT 03FBH FOR A 'SOFTWARE ROM PATCH' **
00980 ;*****
00990 ;
01000 BLINK EXX ;SWAP REGISTERS
01010 EX AF,AF' ;ALL OF THEM
01020 LD A,(SHIFT) ;SHIFT KEY ROW
01030 AND A ;SHIFT PRESSED
01040 JR Z,NOPE ;NOT YET
01050 LD A,(SHFCNT) ;CHECK ON 'SHIFT/ARROW'
01060 CP SHFTLT ;'SHIFT/LEFT ARROW' ??
01070 JR Z,TOGGLE ;CHECK CURSOR ON 'OR' OFF
01080 ;
01090 ;** THE FOLLOWING CODE CHECKS TO SEE IF A BASIC PROGRAM
01100 ;** IS BEING RUN AND WILL TURN THE CURSOR 'OFF'. CAN BE
01110 ;** DELETED IF 'BLINKING CURSOR' ON INPUT STATEMENTS IS
01120 ;** DESIRED. IF DELETED THEN USE 'NOPE' IN LINE 1200
01130 ;
01140 ;=====
01150 NOPE LD A,(BLINE) ;BASIC MAYBE IN A LOOP
01160 CP 0FFH ;= NO - OK TO 'BLINK'
01170 JR NZ,OUT ;ELSE 'STOP BLINKING'
01180 ;=====
01190 ;
01200 LD A,(FLAG) ;OTHERWISE KEEP CHECKING
01210 CP 01H ;= 'ON'
01220 JR NZ,OUT ;GO IF 'OFF'
01230 CONT LD A,(DCOUNT) ;USE AS 'DELAY' COUNTER
01240 CP 0FFH ;=255 DEC. ?
01250 JR Z,SWAP ;START DELAY AGAIN IF YES
01260 INC A ;NO = ADD (1)
01270 LD (DCOUNT),A ;SAVE UP DATED COUNT
01280 JR OUT ;MAKE A FAST EXIT
01290 ;
01300 SWAP XOR A ;ZERO 'A'
01310 LD (DCOUNT),A ;ZERO 'DELAY' COUNTER
01320 LD A,(LCOUNT) ;LOOP COUNTER
01330 OR A ;CHECK IT
01340 JR Z,SWAP2 ;=0 ? MAKE IT A (1)
01350 PUSH HL ;SAVE 'HL' TO GET CURSOR
01360 LD HL,(CURP) ;POSITION FROM VID. BLOCK
01370 LD (HL), ' ' ;=SPACE EVERY OTHER COUNT
01380 POP HL ;RESTORE 'HL'
01390 XOR A ;ZERO 'A'
01400 LD (LCOUNT),A ;ZERO LOOP COUNTER
01410 JR OUT ;OUT IF DONE
01420 ;
01430 SWAP2 LD A,01H ;CHANGE TO A (1)
01440 LD (LCOUNT),A ;MAKE LOOP COUNT = (1)
01450 PUSH HL ;SAVE 'HL'
01460 LD A,(CURCHR) ;GET THE CURSOR CHAR.
01470 LD HL,(CURP) ;GET CURSOR POSITION
01480 LD (HL),A ;INSERT CURSOR CHARACTER
01490 LD (CURS1),A ;FOR SOME 'DOS' SYSTEMS
01500 POP HL ;RESTORE 'HL'
01510 ;

```

```

01520 OUT EXX ;SWAP REGISTERS BACK
01530 EX AF,AF' ;ALL OF THEM
01540 JR KEYREP ;GOTO 'KEY REPEAT'
01550 ;
01560 TOGGLE LD A,(FLAG) ;GET OUR FLAG
01570 CP 01H ;IS IT 'ON'
01580 JR NZ,CURON ;THEN TURN IT 'ON'
01590 ;
01600 CUROFF LD A,00H ;ZERO 'A'
01610 LD (FLAG),A ;MAKE IT 'OFF'
01620 LD HL,(CURP) ;IN CASE IT IS NOT ON
01630 LD A,(CURCHR) ;GET CURSOR CHARACTER
01640 LD (HL),A ;BACK IN POS'N
01650 JR OUT ;THEN EXIT
01660 ;
01670 CURON LD A,01H ;GET A (1)
01680 LD (FLAG),A ;AND SHOW IT 'ON'
01690 LD HL,(CURP) ;GET CURSOR POS'N
01700 LD A,(CURCHR) ;AND RE-DRAW CURSOR
01710 LD (HL),A
01720 JR CONT ;THEN CONTINUE
01730 ;
01740 DCOUNT NOP ;DELAY COUNTER
01750 LCOUNT NOP ;LOOP COUNTER
01760 FLAG NOP ;STORE FOR ON/OFF FLAG
01770 ;
01780 KEYREP LD HL,KBDWRK ;GET KEYBOARD WORK AREA
01790 LD BC,KEYST ;START OF KEYBOARD ROWS
01800 LD D,00H ;D=ROW COUNTER (0 - 6)
01810 LP2 LD A,(BC) ;GET THE KEY PRESSED
01820 LD E,A ;TRANS. TO 'E'
01830 AND E ;CHECK IT
01840 JR NZ,LP ;GO IF NOT ZERO
01850 LD (HL),A ;ELSE PUT IN 'WORK AREA'
01860 LP4 INC D ;ADD ONE TO 'D' ROW COUNT
01870 INC L ;MAKE 'MSB' HIGHER
01880 RLC C ;ROTATE 'C' LEFT 1 'BIT'
01890 LD A,C ;TRANS. RESULT TO 'A'
01900 SUB 80H ;ADJUST FOR 'ASCII'
01910 JR NZ,LP2 ;GO IF NOT
01920 LD A,(HL) ;GET ADDRESS
01930 LD B,07H ;LOAD 'B' WITH SEVEN
01940 LP3 DEC L ;REDUCE 'LSB'
01950 ADD A,(HL) ;ADD BYTE TO 'A'
01960 DJNZ LP3 ;GO FOR COUNT OF '7'
01970 CP 00H ;IS IT ZERO
01980 LD A,00H ;MAKE 'A' = ZERO
01990 RET NZ
02000 LD (VDBLK1),A ;PART OF KEYBOARD CNTL.
02010 RET
02020 ;
02030 LP AND (HL) ;LP2 MAY HAVE GOTTEN HERE
02040 JR Z,LP5 ;GO IS RESULT IS ZERO
02050 LD A,(VDBLK1) ;ELSE GET OUT STORE
02060 INC A ;ADD (1) TO IT
02070 LD (VDBLK1),A ;AND STICK IT BACK
02080 CP 0FFH ;CHECK FOR 255 DEC.
02090 JR NZ,LP4 ;NO = OK TO CARRY ON
02100 DEC A ;ELSE REDUCE COUNT
02110 LD (VDBLK1),A ;AND STICK IT!
02120 LD A,E ;GET OUR KEY BACK AGAIN
02130 LP5 LD (HL),E ;AND TRANS. TO ADDRESS
02140 PUSH BC ;SAVE 'BC'
02150 LD BC,2000 ;KEY REPEAT COUNT
02160 ;
02170 ;** INCREASE OR REDUCE THE ABOVE COUNT TO SUIT YOUR NEED
02180 ;** ALSO WILL ACT AS 'KEY DEBOUNCE'
02190 ;
02200 CALL DELAY ;ROM DELAY ROUTINE
02210 POP BC ;RESTORE 'BC'
02220 LD A,(BC) ;'BC' HAD THE KEY PRESSED
02230 AND E ;CHECK 'E'
02240 RET Z
02250 JP ROMROW ;BACK TO ROM 'ROW' ADDR.
02260 ;
02270 ;*****
02280 ;** LINE PRINTER ROUTINE STARTS HERE - PLEASE NOTE: **
02290 ;** A 'CALL' TO ROM AT 003BH WILL OUTPUT A 'BYTE' IN **
02300 ;** THE 'A' REGISTER BUT WE WILL OUTPUT IT DIRECTLY **
02310 ;** TO SAVE TIME. **
02320 ;*****
02330 ;
02340 VIDPNT EXX ;SWAP REGISTERS
02350 EX AF,AF' ;ALL OF THEM
02360 LD A,(VPROW) ;'VP' KEYS ROW
02370 CP 41H ;='VP' PRESSED
02380 ;
02390 ;** USE THE NEXT TWO LINES IF 'ZX' IS PREFERRED
02400 ;
02410 LD A,(ZXROW) ;'ZX' KEYS ROW
02420 CP 05H ;='ZX' PRESSED
02430 ;
02440 JR Z,DUMP ;GO IF PRESSED
02450 ;*****
02460 ;** THE NEXT FOUR LINES ARE USED WITH 'DISK' TO CALL THE
02470 ;** SYSTEM 'DEBUG' ROUTINE BY PRESSING '1' AND '0' AT
02480 ;** THE SAME TIME. KEY AND ROW VALUES MUST BE CHANGED
02490 ;** TO PROVIDE COMPATIBILITY WITH OTHER DOS.
02500 ;** DELETE THE FOUR LINES FOR NON-DISK SYSTEMS.
02510 ;
02520 LD A,(NUMROW) ;KEYS 0-7
02530 CP 03H ;'1' & '0' PRESSED?
02540 JR NZ,EXIT ;NO = GO!!
02550 RST 30H ;DEBUG ENTRY IN ROM
02560 ;*****
02570 ;
02580 JR EXIT ;SET 'PC' FOR RETURN
02590 DUMP LD HL,(CURP) ;GET CURSOR POS'N
02600 LD (HL),' ' ;'V' OR 'P' MAYBE ON
02610 DEC HL ;THE VIDEO SO, WIPE IT
02620 LD (HL),' ' ;OUT WITH SPACES
02630 LD HL,VIDST ;YES = GET VIDEO
02640 LD E,10H ;EQUALS 16 LINES
02650 LOOP LD C,40H ;EQUALS 64 CHARS/LINE
02660 LP1 CALL LPTST ;CHECK ON PRINTER
02670 LD A,(HL) ;FIRST CHARACTER
02680 ;
02690 ;*****
02700 ;** THE NEXT ROUTINE IS FOR THE 'EPSON' PRINTERS & **
02710 ;** IS MARKED WITH DASHES TO 'DELETE' FOR OTHER **
02720 ;** PRINTERS THAT USE 'NORMAL' TRS-80 GRAPHICS **
02730 ;*****
02740 ;
02750 ;-----
02760 MX80 CP 20H ;SUB-ROUTINE TO CHECK FOR
02770 JR NC,CHK ;SPACE COMPRESSION AND
02780 ADD A,40H ;GRAPHICS CODES TO
02790 CHK CP 80H ;OUTPUT TO 'EPSON'
02800 JR C,LPRINT ;PRINTERS.
02810 ADD A,20H ;ADD 32 DEC. FOR GRAPHICS
02820 ;-----
02830 ;
02840 LPRINT LD (LPORT),A ;OUTPUT BYTE TO PRINTER
02850 LD A,(SPCBAR) ;CHECK FOR 'SPACEBAR'
02860 CP 80H ;WAS IT PRESSED ?

```

continued on page 58

COBOL PRIMER #2

H. C. Goodrich

Text Abbreviations:

CUG: *Cobol Users Guide*

CRM: *RSCOBOL Reference Manual*

CEDIT: *Source Program Editor*

This is the second lesson on RSCOBOL. Last time you learned about the four major Divisions, The Program-ID, and the Configuration Section. You used two COBOL expressions DISPLAY and STOP RUN. You also were introduced to the CEDIT commands I(nput), E(dit), W(rite), Q(uit) & L(oad). Finally, you learned to compile your source code using RSCOBOL, and then run your object code using RUNCOBOL with the T(ube) or P(rint) switches. Now you will take a few more steps.

Remember, the idea is for me to use minimal explanations of COBOL programming—you can get that in any textbook—and a maximum of exemplifying working programs.

For this lesson, we are going to add one step to each of the four Divisions and give you some more practice with CEDIT. Where appropriate, I will give you page number references to one or more of the RSCOBOL manuals.

Before entering Program #2, look at line ##1 below. It is permissible to write up to 80 characters on one line and more than one expression; however, it is common practice to keep it easy to read. The following are allowed:

```
##1 DISPLAY "HELLO" LINE 1 POSITION 10 ERASE. ACCEPT HELLO  
LINE 1 POSITION 17 TAB.
```

or

```
##1 DISPLAY "HELLO"  
##2 LINE 1  
##3 POSITION 10  
##4 ERASE. (the period ends the statement)  
##5 ACCEPT HELLO  
##6 LINE 1  
##7 POSITION 17  
##8 TAB.
```

or variations between. The compiler will accept all. It is a question of style and readability. I will use multiple expressions with the first two divisions but rarely in the DATA or PROCEDURE DIVISIONS. If the line is long, I generally split the statement and place logical elements in succeeding lines. The commas or semi-colons are not required. They used for reader legibility and for no other reason.

Load your RSCOBOL and TRSDOS System disk and type CEDIT <ENTER>; press I(nput) <ENTER>; and type the following program (numbers are shortened):

```
100 IDENTIFICATION DIVISION.  
110 PROGRAM-ID. PROGRAM2.  
120 AUTHOR. HC GOODRICH.  
130 ENVIRONMENT DIVISION.  
140 CONFIGURATION SECTION.  
150 SOURCE-COMPUTER. TRS803.  
160 OBJECT-COMPUTER. TRS803.  
170 DATA DIVISION.  
180 WORKING-STORAGE SECTION.
```

```
190 01 NAME-WHATEVER PICTURE X(15) VALUE SPACES.  
200 01 YES-OR-NO PICTURE X VALUE SPACES.  
210 01 AGE PICTURE 99 VALUE ZEROS.  
220 PROCEDURE DIVISION.  
230 START-PROGRAM.  
240 DISPLAY "WHAT IS YOUR NAME?",  
250 LINE 1, POSITION 10, ERASE.  
260 ACCEPT NAME-WHATEVER,  
270 LINE 1, POSITION 30, TAB.  
280 DISPLAY "HOW OLD ARE YOU?",  
290 LINE 2, POSITION 10.  
300 ACCEPT AGE,  
310 LINE 2, POSITION 30, TAB.  
320 DISPLAY "YOUR NAME IS. " NAME-WHATEVER.  
330 DISPLAY "ARE YOU REALLY "AGE" YEARS OLD (Y/N)?".  
340 ACCEPT YES-OR-NO,  
350 LINE 4, POSITION 40.  
360 IF YES-OR-NO EQUALS "N"  
370 GO TO START-PROGRAM.  
380 STOP RUN.
```

COMMENTS:

Lines 100,110,130-170, & 220 are required and were introduced in lesson #1.

Line 120 (CRM p.28-9): Commentary & optional. What would happen if I used periods after the initials?

Line 180 (CRM p.54): This section allows you to define record descriptions or "variables." It will become increasingly more important as we increase our use of files.

Line 190 (CRM p.60) #01: The use of numbers with data names or variables will be covered later. The data name may be from 1 to 30 characters long but must be connected (no space). As in BASIC, a DATA name must not use a reserved word (CRM 238-41). DAY is a reserved word but by modification (DAY-1, DAY-BIRTH, DDAY, etc) it becomes unique and may be used. PICTURE (or PIC) (p.64-65) is COBOL's way of describing the data or entry that is to be placed into it. Data that is alphanumeric uses "X;" data which is numeric uses a "9;" and "A" data (seldom used in practice) indicates the use of non-numeric characters only. The number in parentheses indicates the number of characters allowed. The following are examples:

Data	Written as:	or as:
SMITH	AAAAA	A(5)
	XXXXX	X(5)
06/15/82	XXXXXXXXX	X(8)
12345	99999	9(5)

(CRM p.18,84) The VALUE SPACES places the literal "spaces" into the initial value. This is used for alphanumeric values. VALUE ZEROS places zeros in numeric data items. (This would be similar to writing A\$="" or B=0 in BASIC.)

Line 230: A paragraph name to mark the start of a routine.

Line 250 LINE (CRM p159): counting from the top of the screen, this indicates on which line the data is to be displayed. If not used, the next line will display the

continued on page 51

ASK RICHARD

Richard Kaplan

What is a database management system?

A database management system (DBMS) is a program which stores, sorts, and prints any type of information. Essentially it is a computerized index card system.

In a sense, a DBMS system is a mailing list, general ledger, check register, and filing system all in one. This is because most DBMS systems allow the user to design exactly what the computer will ask you for on the screen, what should be stored on disk, and the exact format of the program's printed report. In essence, you can design the program to do any type of job which involves inputting, storing, sorting, or printing information.

If a database management system is so versatile, why should I buy individual programs, such as a mailing list or a general ledger?

For starters, operating a DBMS to the extent of simulating a mailing list or general ledger package usually requires a good deal of computer experience. Many people find it more convenient to purchase a program which has already been designed for their purpose and which can simply be "booted and run."

Another advantage of purchasing a specific program is that the program will speak to you in your own terms. For example, with a mailing list there might be a choice on a menu to "PRINT ALPHABETIZED 4-UP LABELS." However, with a database management system, which is designed to be very general, the equivalent command might be something to the effect of "OUTPUT FILE MAIL ON SORT KEY A\$ WITH FORMAT 4." Both programs will achieve the same result; however, the "pure" mailing list would obviously be more appropriate for a computer novice.

I've heard many conflicting stories about programs not being compatible on various computers. In very brief terms, what is it that makes a program for one computer not work on another computer?

The most common difference between computers is simply the difference in language between two machines. Every computer manufacturer (except those whose computers use CP/M—see previous installments of "Ask Richard") designs a language for his computer similar to, but different from, every other language. In other words, most manufacturers use the BASIC language, but each one adds in special features to make the computer better. The result is that these special features will not work on any computer except that for which the program was originally written.

Why doesn't someone write a program to convert programs between different computers automatically?

This seems to be an obvious solution. However, as far as I know, there never has been a program which can translate programs. The reason for this is simple: translating programs between computers is not an *objective* task. That is, there is no way to define step-by-step exactly how to convert a program. There is a substantial element of creativity and judgment involved in translating programs, and no one has yet been able to teach a computer creativity or judgment.

For example, a common difficulty in translating

programs involves the width of a computer's screen. Often it is necessary to translate from a computer which has 64 columns on its screen (such as the TRS-80 Model III) to one which has fewer characters (such as the TRS-80 Color Computer, with 32 columns per line). There is absolutely no way to tell the computer how to decide what can be eliminated from the screen or how to design a nice screen with 32 columns. Only a person can do this.

When using disk storage, I've always used MKI\$, MKS\$, and MKD\$ because I know they can reduce the space needed to store numbers on disk. How is this done?

Our number system uses the digits 0-9 to represent numbers. There are 100 possible 1- or 2-digit combinations of these digits, which we refer to as the numbers 0-99. Remember, using the digits 0-9 gives us 10 possible symbols from which we can make numbers. If we were to use more than 10 symbols to represent digits of a number, there would be more possible combinations of these digits. When the computer stores information on disk, it does just that—it uses more than 10 digits to represent numbers, so it can store a very large number with very little memory.

Let's examine in detail the MKI\$ function, which is used to store integers. Since our number system has 10 symbols to represent digits, it is called a *base 10* system. Since the highest possible value of the 10's digit is 90 (a 9 in this digit represents $90 = 10 * 9$) and the highest possible value of the units digit is 9, the highest possible 2-digit number is $99 (10 * 9 + 9)$.

Similarly, the highest possible 2-digit number for base 16, which uses the letters A-F in addition to the numerals 0-9, is $16 * 15 + 15$, or 255.

MKI\$ uses base 256. (Base 256 is used because there are 256 different characters which can be stored in 1 byte. There are 8 bits to one byte, and there are 2 alternatives to each bit. Hence, $2^8 = 256$ possible characters, and there are 256 characters in base 256). Using the technique described in the preceding paragraphs, it can be seen that in base 256 a 2-digit number can be as high as $256 * 255 + 255$, or 65535.

Let's say we wish to store positive and negative numbers; we'll divide this number in half, allowing for an equal range of positive and negative numbers. Now, $65535/2 = 32767.5$, or 32767 remainder 1 (the extra number is used for -32768). Wait a minute, isn't that a familiar number? MKI\$ uses 2 characters, and it can store integers as high as 32767. This is no coincidence.

MKS\$ and MKD\$, which can store single-precision and double-precision numbers, do not store numbers exactly as does MKI\$, but the essential principle is the same. By using more symbols than the 10 digits 0-9, it is possible to generate more possible combinations of digits, thereby storing a large number in a smaller amount of memory.

What is the difference between a programmable calculator and a portable computer?

There is a very fine line between these two, and sometimes there is no definitive classification for a given instrument. In general, a calculator is considered to be

programmable if it can automatically perform calculations based upon instructions entered into it. A portable computer, as compared to a programmable calculator, can display letters as well as numbers and usually uses BASIC or another high-level language, instead of a language of numbers and cryptic symbols, which most programmable calculators use.

Speaking of calculators, what does it mean when a calculator has Reverse Polish Notation?

Reverse Polish Notation (RPN) refers to a specific procedure for entering numbers into a calculator. An alternative is known as an Algebraic Operating Systems (AOS). Finally, the simplest calculators use a "standard" entry procedure.

Let's start with the "standard" entry method. How does that work?

Most simple calculators perform calculations immediately when entered on the keyboard. In other words, every time an operation (+, -, multiplication, or division) key is pressed the calculator displays the current subtotal of the calculations. To add 2 to the product of 3 times 4, you would first enter 3 X 4. The 4 would then be in the display. After pressing the + key, a 12 would appear. Finally, you would type 2 and then =, and 14 would appear. The reason that we multiplied 3 by 4 before adding 2 is that if we had entered 2 + 3 X 4, the calculator would have first added 2 to 3 and then multiplied the sum by 4, for a total of 20.

How does the AOS method differ from the standard entry method?

To answer this question requires a quick review of one of the fundamental rules of algebra, the Order of Operations. Put simply, when a mathematical equation is written on paper, multiplication and division are meant to be done first (from left to right) and then addition and subtraction. To refer to our previous example, 2 + 3 X 4 equals 14. However, as explained above, if "2 + 3 X 4" were entered on a standard calculator, the result would be 20.

The solution to this discrepancy (between the order in which a calculator performs calculations and the order in which they mathematically should be carried out) was to develop a calculator which "knew" the correct order of operations. The most well-known of these calculators are manufactured by Texas Instruments, which refers to this entry method under the trademark AOS (Algebraic Operating System). A calculator with AOS logic automatically performs calculations in the order in which they should be done.

With an AOS calculator, to calculate 2 + 3 X 4, you would enter the numbers and symbols exactly as they are written. After typing 2 + 3 X, the 3 would still appear in the display. A 5 would *not* appear under AOS (although it would with a standard entry procedure) because the calculator would know that it first has to wait for the number to multiply by three and *then* add 2. After typing "2 + 3 X 4 =", the final result of 14 would appear.

Finally, how does Reverse Polish Notation work?

RPN calculators, which are manufactured almost exclusively by Hewlett-Packard, operate essentially like an adding machine. While they do not follow order of operations, as does AOS, they are considered by many to be the easiest to use.

Operating a calculator with RPN consists of typing numbers to be worked with, always immediately followed by an operation (+, -, multiplication, or division). For example, to compute 2 + 3 * 4, as in our previous example, you would enter "3 + 4 * 2 +." After each number and operation were pressed, the current subtotal would be displayed.

Although an RPN calculator does not follow order of operations, many people consider this system to be the easiest to use because they say it follows the natural process one goes through when solving a mathematical problem.

Which is the best system—a standard calculator, an AOS calculator, or an RPN calculator?

Obviously, the answer to this question is no more clear cut than is the answer to the classic question, "Which is the best computer?" Each system is best for a specific audience; that is why each has survived in the industry.

The standard calculator is best for someone without a strong math background who simply needs a calculator for basic arithmetic. The "average" person is most likely to "think" like this type of calculator.

The AOS system is my personal favorite, although by no means does this mean I recommend it for everyone. I prefer to have a calculator which follows the same rules of math which I use. It seems to me that when one is doing a great deal of arithmetic, it is much easier if the calculator follows the same rules as the equations on paper.

RPN is often preferred by scientific people. Again, many feel this is the easiest system to use because this is how people naturally think. The disadvantage is that it does not follow order of operations.

Questions not attributed to a specific person represent a typical computer user, a composite "character" drawn from the author's personal experience speaking with customers of H & E Computronics.

Questions from readers on all aspects of personal computing are welcomed. Please enclose a self-addressed, stamped envelope with your request.

Richard Kaplan
H & E Computronics
50 North Pascack Road
Spring Valley, NY 10977 ■

PIPS

continued from page 16

```
407 T1=((X*A(2,I))+(X*C4))
408 T2=((PA(X)*A(1,I))*C3)
409 TC(X)=T1+T2
410 IF U=5 AND A=1 THEN GOSUB 7500:IF PA(X)>0 THEN 420 ELSE
FOR Z=1 TO 800:NEXT Z:GOTO 5000
411 IF X<=1 THEN 420
412 IF U=5 AND A=2 ANDTC(X-1)<TC(X) AND TC(X-2)>TC(X-1) THEN
2000 ELSE 415
415 IF TC(X-1)<TC(X) AND TC(X-2)>TC(X-1) THEN GOTO 2000
420 NEXT X
425 S(I)=(X-1):C(I)=TC(X-1)
430 NEXT I
500 PRINT@896,"PRESS ENTER TO CONTINUE.....";:INPUT A$
510 IF U=5 THEN GOTO 5000 ELSE 3000
```

```

600 G4=.3010
610 FOR B=3 TO J
620 Y8=LOG(B)/LOG(10)
630 G4=G4+Y8
640 NEXT B
650 RETURN
1000 CLS:PRINT TAB(15)"PARTS INVENTORY PLANNING SIMULATION"
1010 PRINT STRING$(63,"-")
1015 PRINT:PRINTTAB(17)"DISPLAY OPTIMUM STOCKING LEVEL ":
PRINT TAB(17)"(1) CRT or (2) PRINTER";: INPUT F: IF F<1 OR
F>2 THEN 1015
1017 IF F=2 THEN 1045
1018 CLS: PRINT TAB(15)"PARTS INVENTORY PLANNING SIMULATION":
PRINT STRING$(63,"-")
1020 PRINT TAB(14)"PARTS";TAB(23)"OPTIMUM";TAB(33)"REORDER";
TAB(43)"OPTIMUM";TAB(52)"PROBABILITY"
1030 PRINT"PART NUMBER";TAB(14)"USAGE";TAB(22)"STOCK LV";
TAB(34)"POINT";TAB(43)"TL COST";TAB(53)"STOCK-OUT"
1040 PRINT STRING$(63,"-"):RETURN
1045 LPRINT TAB(22)"PARTS INVENTORY PLANNING SIMULATION":
LPRINT STRING$(80,"-")
1050 LPRINT TAB(19)"PARTS";TAB(26)"LEAD";TAB(34)"OPTIMUM";
TAB(44)"REORDER ";TAB(54)"OPTIMUM";TAB(63)"PROBABILITY"
1055 LPRINT TAB(2)"PART NUMBER";TAB(14);"TIME";TAB(19);
"USAGE";TAB(26);"TIME";TAB(32);"STOCK LEVEL";TAB(45);"POINT";
TAB(54)"TL COST"; TAB(64); "STOCK-OUT"
1060 LPRINT STRING$(80,"-")
1070 RETURN
1500 IF A(1,I)<1 AND Z$="DAY" THEN A(1,I)=A(1,I)*5:
U$(I)="WEEK": GOTO 1530
1510 IF A(1,I)<1 AND Z$="WEEK" THEN A(1,I)=A(1,I)*4:
U$(I)="MONTH": GOTO 1530
1520 IF A(1,I)<1 AND Z$="MONTH" THEN A(1,I)=A(1,I)*12:
U$(I)="YEAR":GOTO 1530
1525 IF A(1,I)>1 THEN U$(I)=Z$
1530 RETURN
1700 IF A(4,I)>A(1,I) AND U$(I)="DAY" THEN A(4,I)=A(4,I)/5:
A(1,I)=A(1,I)*5:U$(I)="WEEK":GOTO 1730
1710 IF A(4,I)>A(1,I) AND U$(I)="WEEK" THEN A(4,I)=A(4,I)/4:
A(1,I)=A(1,I)*4:U$(I)="MONTH":GOTO 1730
1720 IF A(4,I)>A(1,I) AND U$(I)="MONTH" THEN A(4,I)=A(4,I)/12:
A(1,I)=A(1,I)*12:U$(I)="YEAR":GOTO 1730
1725 IF A(4,I)<A(1,I) THEN U$(I)=Z$
1730 RETURN
2000 IF F=2 THEN 2020
2010 PRINT USING E$;P$(I);A(1,I);U$(I);X-1;INT(A(4,I)*
A(1,I));TC(X-1);PA(X-1): GOTO 425
2020 LPRINT TAB(2);:LPRINT USING E2$;P$(I);U$(I);A(1,I);
A(4,I); X-1;INT(A(4,I)*A(1,I));TC(X-1);PA(X-1):GOTO 425
3000 IF F=2 THEN 3130
3010 CLS:PRINT TAB(13)"PARTS INVENTORY PLANNING SIMULATION"
3015 PRINT STRING$(63,"-")
3030 PRINT TAB(15)"PRODUCT";TAB(25)"OPTIMUM";TAB(36)"OPTIMUM"
3040 PRINT TAB(15)"GROUP";TAB(25)"STOCK LV";TAB(36)"TL COST"
3050 PRINT STRING$(63,"-")
3060 FOR G=1 TO JJ: I(G)=0:I1(G)=0
3070 FOR I=1 TO E
3080 IF A(3,I)<>G THEN 3090
3085 I(G)=S(I)+I(G): I1(G)=C(I)+I1(G)
3090 NEXT I
3095 IF F=2 THEN 3105
3100 PRINT USING E1$;G;I(G);I1(G):GOTO 3110
3105 LPRINT USING E3$;G;I(G);I1(G)
3110 NEXT G
3115 PRINT@896,"PRESS ENTER TO CONTINUE....."::INPUT A$:
GOTO 5000

```

```

3130 LPRINT:LPRINT:LPRINTTAB(22)"PARTS INVENTORY PLANNING
SIMULATION"
3135 LPRINT STRING$(80,"-")
3140 LPRINT TAB(25)"PRODUCT";TAB(38)"OPTIMUM";TAB(50)"OPTIMUM"
3150 LPRINT TAB(25)"GROUP";TAB(38)"STOCK LV";TAB(50)"TL COST"
3155 LPRINT STRING$(80,"-")
3160 GOTO 3060
5000 U=0:CLS:PRINT TAB(15)"PARTS INVENTORY PLANNING
SIMULATION"
5005 PRINT TAB(25)"( P I P S )"
5010 PRINT STRING$(63,"=")
5020 PRINT:PRINT TAB(25)"*** M E N U ***"
5030 PRINT:PRINT TAB(15)"(1) BUILD NEW PARTS DATA FILE"
5040 PRINT TAB(15)"(2) LOAD EXISTING PARTS DATA FILE"
5050 PRINT TAB(15)"(3) SAVE NEW PARTS DATA FILE
5060 PRINT TAB(15)"(4) DISPLAY OPTIMUM STOCKING LEVEL"
5070 PRINT TAB(15)"(5) DISPLAY SELECTED PARTS "
5080 PRINT:PRINT TAB(20)"YOUR SELECTION";:INPUT U:IF U<1 OR
U>5 THEN 5080
5085 IF U=1 THEN PRINT: PRINT TAB(18)"HOW MANY PRODUCT
GROUPS"; :INPUT JJ
5090 ON U GOTO 100,6000,7000,305,8000
6000 CLS:PRINT:PRINTTAB(15)"LOAD EXISTING PARTS DATA FILE"
6020 PRINT TAB(18)"ENTER NAME OF DATA FILE": PRINT
TAB(18)" ";:INPUT J$
6030 OPEN"1",1,J$
6040 INPUT#1,C1,C2,C3,C4,E
6050 FOR I = 1 TO E
6055 INPUT #1,P$(I),U$(I)
6056 INPUT #1,A(1,I),A(2,I),A(3,I),A(4,I)
6060 NEXT I
6070 PRINT:PRINT TAB(10)"DATA FILE LOADED ...": FOR Z=1
TO 300:NEXT :GOTO 5000
7000 CLS:PRINT:PRINT TAB(20)"SAVE NEW PARTS DATA FILE"
7010 PRINT TAB(21)"ENTER NAME OF DATA FILE": PRINT TAB(23);
" ";:INPUT J$
7020 OPEN"0",1,J$
7030 PRINT#1,C1;C2;C3;C4;E
7040 FOR I= 1 TO E
7050 PRINT#1,P$(I);" ";U$(I);" ";A(1,I);A(2,I);A(3,I);A(4,I)
7055 NEXT I
7060 PRINT:PRINT TAB(20)"DATA FILE SAVED ON DISK":FOR Z=1
TO 300:NEXT Z:GOTO 5000
7500 IF F=2 THEN 7520
7510 PRINT USING X$;X;T1;T2;TC(X);PA(X):GOTO 7530
7520 LPRINT USING X1$;X;T1;T2;TC(X);PA(X)
7530 RETURN
8000 CLS:PRINT TAB(15)"PARTS INVENTORY PLANNING SIMULATION"
8005 PRINT STRING$(63,"-")
8010 PRINT:PRINTTAB(18)"ENTER SELECTED PART NUMBER"
8020 PRINT@281,,:INPUT SP$
8030 FOR I=1 TO E
8040 IF P$(I)<>SP$ THEN 8050
8041 GOSUB 8500
8045 GOTO 320
8050 NEXT I
8060 PRINT:PRINT TAB(15)"PART NUMBER ";SP$;" NOT FOUND IN
FILE"
8070 FOR Z=1 TO 500:NEXT Z:GOTO 5000
8100 CLS:IF F=2 THEN 8130
8110 PRINT TAB(18)"PART NUMBER ";SP$:PRINT STRING$(63,"-")
8115 PRINT TAB(5)"STOCK";TAB(15)"INVENTORY";TAB(27);
"STOCK-OUT";TAB(38)"TOTAL INV";TAB(49)"PROBABILITY"
8120 PRINT TAB(5)"LEVEL";TAB(18)"COST";TAB(30)"COST";
TAB(40)"COST";TAB(50)"STOCK-OUT"
8125 PRINT STRING$(63,"-"):GOTO 8180

```

```

8130 LPRINT TAB(22)"PARTS INVENTORY PLANNING SIMULATION"
8140 LPRINT TAB(25)"PART NUMBER ";SP$:LPRINT STRING$(80,"-")
8150 LPRINT TAB(10)"STOCK";TAB(20)"INVENTORY"; TAB(33);
"STOCK-OUT";TAB(46)" TOTAL INV.";TAB(58)"PROBABILITY"
8160 LPRINT TAB(10)"LEVEL";TAB(23)"COST";TAB(36)"COST";
TAB(49)"COST";TAB(59)"STOCK-OUT"
8170 LPRINT STRING$(80,"-")
8180 RETURN
8500 PRINT:PRINT TAB(12)"PARTS USAGE PER ";U$(I);
" ";A(1,I)
8510 PRINT TAB(12)"PART COSTS..... ";A(2,I)
8520 PRINT TAB(12)"PRODUCT GROUP..... ";A(3,I)
8530 PRINT TAB(12)"LEAD TIME TO REPLENISH PART.....";A(4,I)
8540 PRINT@896,"ANY CHANGES (Y/N)";:INPUT W$:IF W$<"Y" AND
W$<"N" THEN 8540
8550 IF W$="N" THEN 8580
8560 PRINT@915,"CHANGE ENTRY #";:INPUT B:IF B<1 OR B>4 THEN
8560
8565 PRINT@938,"TO ";:INPUT K5
8570 A(B,I)=K5
8580 PRINT@896,STRING$(63," ");:PRINT@896,"DISPLAY (1)
CALCULATIONS OR (2) OPTIMUM STOCK LEVEL ";:INPUT A:IF A<1 OR
A>2 THEN 8580
8590 PRINT@896,STRING$(63," ");:PRINT@896,"DISPLAY VIA (1)
CRT OR (2) PRINTER ";:INPUT F:IF F<1 OR F>2 THN 8590
8595 CLS:RETURN
9000 IF Z$="D" THEN Z$="DAY"
9010 IF Z$="W" THEN Z$="WEEK"
9020 IF Z$="M" THEN Z$="MONTH"
9030 IF Z$="Y" THEN Z$="YEAR"
9040 RETURN

```

Dennis P. Avola
20 Prescott Street
Rutland, MA 01543 ■

COBOL PRIMER #2

continued from page 47

material. POSITION (CRM p159): Counting from the left, this indicates the column on which the data is to start being displayed. (Column 1 if not used.) ERASE clears the screen prior to printing the message. (Commas are optional and used for legibility—p.6)

Line 260: ACCEPT takes the data entered (via the keyboard in this case) and transfers it to the variable data name "NAME-WHATEVER." The cursor is positioned on line 1, column 30. TAB causes the computer to wait for the <ENTER> key to resume operation. It will not allow you to enter more characters than permitted by the PICTURE clause. Use the backspace to correct input.

Line 320 DISPLAY can display messages bound by quotes or information held in data names.

Line 340 If TAB is not used, action will resume as soon as the PICTURE field is complete (1 in this case). (It is similar in use to INKEY\$.)

Line 360 This is one example of the IF conditional (CRM p. 115,167,288). Many variations similar to BASIC exist. The EQUAL and "=" are interchangeable.

Line 370: The GO TO (CRM p.166) must have a space between the two words. It performs the same function as in BASIC; it transfers the activity to the Paragraph Name specified—not a line number. (The Paragraph Name could be a number; however, you might lose the ability to understand what is happening.) For example, look at the difference between the

following:

```

BASIC: GOTO 500
COBOL: GO TO START-PROGRAM

```

Now press <BREAK> to get out of I(nput) and go to the top of the program. There are three ways to do this. You can enter T(op) <ENTER> to go immediately to the first line (B for bottom); you can use the up arrow to single-step backwards to the top (what do you think down arrow accomplishes?); or you can P(rint)100.

P duplicates the LIST in BASIC. P 100:200 will display the the screen lines 100 to 200 inclusive. (Notice the colon in place of the dash.) P 200 will display the one line. Enter P with no line number and the next current 14 lines will be displayed; P#=top; P*=bottom; and P.=current line. (Be sure not to use L in place of P. The computer will ask you if you wish to L(oad) another program. If you say yes, you'll wipe out what you have. Use <BREAK> to return to >READY mode.) Once at the top, you may single step down or display one group of lines at a time to check for errors.

Use E(dit) to make changes and D(DELETE) or I(nsert) to make required modifications. When all spelling, spacing and punctuation is considered correct, enter W(rite) PROGRAM2/CBL:1 <ENTER> to write the source code to disk #1. When the > prompt appears, type Q(uit) <ENTER> to exit CEDIT.

Now, from TRSDOS Ready mode, type:

```
RSCOBOL PROGRAM2/CBL:1 (options*)
```

*Options: You have a number of options to consider. Lesson #1 mentioned T(ube) for displaying the source code and error messages to the CRT and P(rint) to send the same information to your printer. O(utput)=d (CUG p.4) ie O=1, means write the object code to disk #1. The default is to write to the first free disk. Option E(rrors) will print syntax errors. You can use more than one option in some cases: (O=1 P E) will write the object code to disk #1 and will write errors only to your printer (this will use up less paper during your inevitable debugging activities). Examples and explanations of error messages are listed in CUG p8-14.

When all is correct, from TRSDOS Ready type:

```
RUNCOBOL PROGRAM2/COB:1
```

and wait for your program to clear the screen and run.

In this lesson, we have looked at the WORKING-STORAGE SECTION, level number 01, PICTURE, VALUE and identifying the type and length of data. We have also introduced the ACCEPT, GO TO, and IF expressions with LINE#, POSITION#, TAB, and ERASE. Under CEDIT, we looked at the single letter commands B, T, D, and P. Finally, we offered O(utput) and E(rror) as options for RSCOBOL compiling.

While waiting for lesson #3, go over the manuals again. Do the same using your textbook. Also take some time to experiment with variations of Programs #1 & #2 (but use a different program name or you'll wipe these out). Try to change the data names in WORKING-STORAGE SECTION. Try to change the location of the displayed material using DISPLAY and ACCEPT, LINE & POSITION, and with or without TAB. Now is the time to familiarize yourself with these elementary tools of entering and editing your programs.

Lesson #3 will introduce some of the arithmetic functions. Until then, don't forget to practice.

Hubbard C. Goodrich
South Harpswell, ME 04079 ■

USING NEWDOS/80 DISK FILES (PART II)

John L. Gross

Since I wrote my last article, Apparat, Inc. has released NEWDOS80 Version 2 (for both the Model I and III). If you have not already gotten version 2, I strongly recommend that you do so immediately. The new version has added many new features. One of the new features is a new file position designation. An example of the new file position designation is PUT1,&&. If you refer back to my last article you will see that version 1 contained a similar looking file position designator. An example would be PUT1,&. This file positioner is still retained in version 2. The function of PUT1,& is to force a write to disk. The file positioner PUT1,&& does the same thing but also updates the disk directory. When I set up a program I include a counter and on every 5th record added I have the program execute a PUT1,&& that forces all the data to be written to disk and updates the directory. By doing that it will assure me that I will not lose more than 5 records in the event of a power or equipment failure. The program that I used in the last article could be changed to incorporate this feature by changing line 100 to the following:

```
100 PUT1,!%,80: PUT1,&: CN=CN+1: IF CN=5 THEN PUT 1,&&:CN=0:
    GOTO 20 ELSE 20
```

The first PUT in line 100 positions the file for adding to the end of the file. The second PUT actually forces the writing of the data onto the disk. The next part of line 100 increases our counter CN by one. Then if CN=5 the PUT1,&& forces the directory to be updated with the proper end of file, and sets CN=0. The program then continues to line 20. If CN doesn't equal zero the program ignores the PUT1,&& and instead goes to line 20 to continue execution. In this and all future articles I will assume that you have updated your NEWDOS80 version 1 to version 2.

I will now examine the different methods of accessing a file. Create a data file using the mail list program listed in my last article. There are several ways to search the mail list for a certain name. You can use a LEFT\$ type of search. In this type of search the data file is searched for a name equal to or greater in length than the desired name with an exact letter for letter match starting the search with the first letter on the left of the name. A program listing for this type of search would look like this:

```
10 CLEAR 3000: OPEN "R",1,"MAIL/DAT","FF",75: GOTO 20
15 (25)NAS$, (25)ST$, (15)CIS$, (2)SA$,ZC#:
20 GOSUB 100
30 X=X+1:GET1,X,15
40 IF A$= LEFT$(NAS$,LEN(A$)) THEN 200
50 IF LOC(1)$ THEN 20 ELSE 30
100 X=0:CLS: PRINT @ 522,"SEARCH FOR NAME = ";: LINEINPUT A$
110 IF A$="" THEN CLOSE:END ELSE RETURN
200 CLS:PRINT"NAME -----> ";NAS$:TAB(50)"RECORD #";X
210 PRINT"ADDRESS -----> ";ST$
220 PRINT"CITY -----> ";CIS$
230 PRINT"STATE -----> ";ST$
240 PRINT"ZIP CODE -----> ";ZC#
250 PRINT @ 466,"IS THIS THE CORRECT RECORD";: INPUT V$
260 IF V$<>"Y" AND V$<>"N" THEN 250
270 IF V$="N" THEN 50
280 PRINT @ 458,CHR$(31);: INPUT "DO YOU WANT TO SEARCH
```

```
ANOTHER RECORD";Z$
290 IF Z$<>"Y" AND Z$<>"N" THEN 280
300 IF Z$="Y" THEN GOTO 20 ELSE CLOSE:END
```

Line 10 clears all variables, reserves string space, opens the file "MAIL/DAT" as an "FF" type file with a record length of 75 and then continues at line 20. Line 15 is the IGEL. It specifies the length and type (i.e. string, integer, etc.) of each variable. Line 20 is a gosub to line 100. Line 100 then sets the the record counter (the variable X) equal to zero and asks what name you want to search for in the data file and assigns it to variable A\$. Line 110 then checks A\$. If A\$ is a null string the file is closed and the program ends. Otherwise the program continues back at line 30. This search is a sequential search of the data file starting with record 1 and ending with the last record. Every time line 30 is executed X is increased by +1. The GET statement in line 30 instructs the computer to use buffer number 1 to get record number X and using the IGEL on line 15. Line 40 then checks to see if the name we are searching for contained in A\$ is equal to, using the LEFT\$ function, the name in the record we have just read. If the names do not match in line 40 the program then continues on line 50. The LOC(1)\$, in line 50, checks to see if this is the last record in the file. If it is the last record the program then continues at line 20. If it is not the last record the program then continues the search at line 30. If in line 40 the names do match the program then continues at line 200. Line 200 clears the screen and prints, to the screen, the name and record number of the current record. Line 210 through 240 print, to the screen, the street address, city, state, and zip code for this record. Line 250 asks if the record display is the correct record. Line 260 checks the response to make sure that V\$ equals either "Y" or "N." If it does not the program then goes to line 250 again and asks the same question. If the response is "N" the program then continues the search at line 50, otherwise the program continues at line 280. In line 280 the CHR\$(31) clears the screen from the PRINT@458 to the bottom of the screen. This effectively erases the question in line 250. Line 280 then continues by asking if you want to search for another record. Line 290 checks to make sure that you have answered with either a "Y" or a "N." Line 300 checks your response, if you answered with a "Y" the program then continues with line 20. Otherwise, it closes the file and ends the program.

The LEFT\$ function in line 40 can cause this program to execute slowly because LEFT\$ must be interpreted for every record searched. This can be eliminated by using some of NEWDOS80's special file commands. By changing lines 30, 40, 100, 110 and 200 to the following the program speed can be increased.

```
30 GET1,,, (A)NAS$;
40 IF A$=NAS$ THEN GET 1,#,15: GOTO 200
100 GET1,!$0:CLS: PRINT @ 522,"SEARCH FOR NAME =";
    : LINE INPUT A$
110 A=LEN(A$): IF A$="" THEN CLOSE:END ELSE RETURN
200 CLS:PRINT"NAME -----> ";NAS$:TAB(50)
    "RECORD #";LOC(1)
```

continued on page 60

MAZE

James Gallagher

This program creates a maze of user-determined size, which has only one solution. The program was developed from an algorithm by David Matuszek presented in "How to Build a Maze" (Byte, December 1981). The maximum size of the maze is determined by the printer used to print the maze and the size of the computer's memory. Memory is likely to be a greater limiting factor than the printer, because the printer only limits the maze's width, while memory limits the maximum number of cells the maze may contain. The formula for required memory is:

$$(\text{length of maze}+2) \times (\text{width of the maze} + 1) \times 4 = \text{number of bytes required}$$

The program is reasonably efficient. However, it takes almost an hour to build a maze of 120 by 60 cells.

A traditional maze, as Mr. Matuszek calls it, must satisfy three conditions: it must have one entrance and one exit; all parts of the maze must be accessible from the entrance; and there must be only one path connecting the entrance to the exit. One way to meet these requirements is to build the maze using a data structure called a spanning tree. In such a structure, each node of the tree connects to exactly one other node. However, a given cell may indirectly connect with all of its adjacent cells; the program makes no distinction as to which cell connects with another cell, only that the barrier between the two is erased. Thus, between any two cells there exists one direct, unique path, and all cell locations, including the exit, are accessible from the entrance.

The program works in the following way: On line 100, a cell is randomly chosen and put into the spanning tree. The four cells surrounding this cell (fewer if it is on a side or a corner) are marked as "frontier cells." These are cells that are next to a cell that is in the spanning tree. The program then enters a loop which builds a spanning tree out of a two dimensional array. One of the two newest entries to the frontier cell list is removed from this list and added to the spanning tree. This spanning tree cell is then connected with a cell already in the spanning tree, and any untouched cells surrounding it are added to the frontier cell list. The program stays in this loop until the frontier cell list has no more elements. When this happens, the frontier cell list element counter will equal zero. When no cells remain in the frontier cell list, then the spanning tree is complete. The program then chooses an entrance to the maze by randomly choosing a cell in the upper left corner of the maze and erasing that section of the maze's boundary. A second cell is chosen in the lower right section of the maze and made into the exit. The maze is then printed, and the user has the option of saving the maze on a floppy disk or ending the program.

Note: during the program's execution, a number representing the current length of the frontier cell list will appear at the center of the video display. This counter starts at 5 or 6 (after the first set of entries) and continues to grow until the maze is about halfway completed; then it decrements to zero, and the maze is printed. This counter has two important functions: it

tells the user that the computer hasn't crashed (important when building large mazes), and it gives the user an idea of how much more time must elapse before the maze can be printed.

This program lends itself to adaptation to a number of system configurations and uses. The program was intended to be run on a system that includes floppy disks and the Radio Shack Line Printer VII. The remarks in the program give all the information needed to modify the program if your system does not have either the disks or a line printer. If the user wants to print the maze using the graphics mode of a printer other than the Line Printer VII, it should be easy to modify the first print-out routine. The print-out routine treats each cell as if it only has two walls (the upper and left ones), and therefore must test for four conditions: (1) both the upper and left walls removed, (2) left wall only removed, (3) Upper wall only removed, and (4) no walls removed. Line 870 prints condition 1, line 880 condition 2, line 900 condition 3, and line 910 condition 4. If the user doesn't have access to a printer with graphics capabilities, then the second print-out routine should be used.

```
1 REM                                     MAZE
2 REM                                     by James Gallagher
3 REM                                     12250 6th ST. E.
4 REM                                     Treasure island
5 REM                                     Florida 33706
6 REM                                     June 1982
7 REM
8 REM                                     THIS PROGRAM WILL BUILD A RECTANGULAR MAZE
9 REM                                     ON THE TRS-80 MODEL I COMPUTER. MAXIMUM
10 REM                                    DIMENSIONS OF THE MAZE ARE DETERMINED BY
11 REM                                    THE AVAILABLE MEMORY AND/OR THE PRINTER.
12 REM
13 REM                                    NOTE: THE PROGRAM WAS DESIGNED TO RUN ON
14 REM                                    A SYSTEM WITH AT LEAST ONE DISK DRIVE.
15 REM                                    FOR A NON-DISK SYSTEM DELETE LINES
16 REM                                    50-70, 440-670, AND 1050-1100.
17 REM                                    ALSO NOTE THAT A ROUTINE IS INCLUDED
18 REM                                    AT THE END OF THE PROGRAM THAT WILL
19 REM                                    PRINT THE MAZE ON A LINE PRINTER, USING
20 REM                                    STANDARD ASCII CHARACTERS ( + - ! ).
21 REM                                    TO USE THIS SUBROUTINE CHANGE LINES 430
22 REM                                    AND 660 TO "GOSUB 1110".
23 REM
30 CLS:RANDOM
40 DEFINT A-M,R-Z
50 INPUT"CREATE A NEW MAZE OR PRINT AN OLD ONE (C/P)";Q$
60 IF Q$="P" THEN 580 'Branch to read maze from disk
70 T1$=RIGHT$(TIMES,8) 'Get time at start of maze building
80 INPUT"MAZE DIMENSIONS (L/W)";L,W
90 IF (L+2)*(W+1)*4>MEM THEN PRINT"MAZE DIMENSIONS TOO LARGE":
GOTO 80
95 IF L<5 OR W<5 PRINT"MAZE DIMENSIONS TOO SMALL":GOTO 80
100 DIM M(W+1,L+1),R(L*W/2),C(L*W/2)
110 S=RND(W):T=RND(L) 'Load spanning tree w/random cell
120 M(S,T)=M(S,T)+2
130 GOSUB 680 'Call frontier cell subroutine
131 REM
132 REM Choose one of the two most recent entries to the frontier
133 REM cell list and add it to the spanning tree. If less
```

```

134 REM than two cells are in the list then choose from
135 REM those remaining. Once an element has been removed from
136 REM the list, it must be shortened so that the list doesn't
137 REM have any holes.
138 REM
140 IF U<2 THEN 170
150 X=RND(2)+U-2
160 GOTO 180
170 X=1
180 S=R(X):T=C(X)
190 M(S,T)=2
200 FOR I=XTOU-1
210 R(I)=R(I+1):C(I)=C(I+1)
220 NEXT I
230 U=U-1
231 REM
232 REM Connect the newest entry to the spanning tree with
233 REM a cell already in the spanning tree.
234 REM
240 I=RND(4)
250 IF A(I)=1 THEN 240
260 ON I GOTO 270 ,300 ,330 ,360
270 IF M(S+1,T)<1 THEN A(1)=1:GOTO 240
280 M(S+1,T)=M(S+1,T)+8
290 GOTO 380
300 IF M(S-1,T)<1 THEN A(2)=1:GOTO 240
310 M(S,T)=M(S,T)+8
320 GOTO 380
330 IF M(S,T-1)<1 THEN A(3)=1:GOTO 240
340 M(S,T)=M(S,T)+4
350 GOTO 380
360 IF M(S,T+1)<1 THEN A(4)=1:GOTO 240
370 M(S,T+1)=M(S,T+1)+4
380 FOR I=1TO4:A(I)=0:NEXT I:GOSUB 680
390 PRINT@542,U
391 REM
392 REM If any cells are still in the frontier cell list
393 REM (ie. not in the spanning tree) select another cell
394 REM from the frontier cell list and add it to the spanning
395 REM tree.If no more cells are left in the frontier cell
396 REM list then select an entrance to the maze and print
397 REM the maze.
398 REM
400 IF U<=0 THEN 140
401 REM
402 REM Choose the entrance and exit of the maze.
403 REM
410 X=RND(W/3)+W/3:M(X,1)=M(X,1)+4
420 Y=RND(W/3)+W/3:M(Y,L)=M(Y,L)+1
430 GOSUB 810 'Call print subroutine
440 T2$=RIGHT$(TIME$,8) 'Record time at completion of maze
450 GOSUB 1050 'Call time subroutine
460 LPRINT"TOTAL COMPUTER TIME REQUIRED TO CREATE THIS MAZE ";
470 LPRINT"WAS ";S3;": ";S2;": ";S1; (HH:MM:SS)"
480 LPRINT:LPRINT:LPRINT
490 INPUT"SAVE THIS MAZE ON DISK OR END PROGRAM (S/E)";Q$
500 IF Q$<="S" THEN END
501 REM
502 REM Write maze to disk
503 REM
510 INPUT"NAME OF MAZE";N$
520 OPEN"O",1,N$
530 PRINT#1,L;W;X;Y;
540 FOR I=1TOL
550 FOR J=1TOW
560 PRINT#1,M(J,I);
570 NEXT J,I:END
571 REM
572 REM Read maze from disk

```

```

573 REM
580 INPUT"NAME OF MAZE";N$
590 OPEN"O",1,N$
600 INPUT#1,L,W,X,Y
610 DIM M(W,L)
620 FOR I=1TOL
630 FOR J=1TOW
640 INPUT#1,M(J,I)
650 NEXT J,I
660 GOSUB 810 'Call print-out subroutine
670 END
671 REM
672 REM Subroutine to mark previously untouched cells as
673 REM Frontier cells.
674 REM
680 IF M(S+1,T)<=0 OR S=W THEN 710
690 M(S+1,T)=-1
700 U=U+1:R(U)=S+1:C(U)=T
710 IF M(S-1,T)<=0 OR S=1 THEN 740
720 M(S-1,T)=-1
730 U=U+1:R(U)=S-1:C(U)=T
740 IF M(S,T+1)<=0 OR T=L THEN 770
750 M(S,T+1)=-1
760 U=U+1:R(U)=S:C(U)=T+1
770 IF M(S,T-1)<=0 OR T=1 THEN RETURN
780 M(S,T-1)=-1
790 U=U+1:R(U)=S:C(U)=T-1
800 RETURN
801 REM
802 REM Subroutine to printout maze on a Radio Shack
803 REM line printer VII, using the printer's graphics.
804 REM
810 IF W<7 THEN LPRINT"START" ELSE LPRINT TAB(INT(X*7/6-3));
"START"
820 LPRINT CHR$(18) 'Switch to graphics mode of printer
830 FOR I=1TOL
840 FOR J=1TOW
850 IF M(J,I)-8<0 THEN 890
860 IF M(J,I)-12<0 THEN 880
870 LPRINT CHR$(28);CHR$(7);CHR$(128)::GOTO 920
880 LPRINT CHR$(28);CHR$(7);CHR$(129)::GOTO 920
890 IF M(J,I)-4<0 THEN 910
900 LPRINT CHR$(255);CHR$(28);CHR$(6);CHR$(128)::GOTO 920
910 LPRINT CHR$(255);CHR$(28);CHR$(6);CHR$(129);
920 NEXT J
930 LPRINT CHR$(255)
940 NEXT I
950 FOR I=1TOW
960 IF FIX(M(I,L)/2)<=M(I,L)/2 THEN 990
970 LPRINT CHR$(28);CHR$(7);CHR$(129);
980 GOTO 1000
990 LPRINT CHR$(28);CHR$(7);CHR$(128);
1000 NEXT I
1010 LPRINT CHR$(30) 'Switch to character mode of printer
1020 IF W<7 THEN LPRINT"FINISH" ELSE LPRINT
TAB(INT(Y*7/6-3));"FINISH"
1030 LPRINT
1040 RETURN
1041 REM
1042 REM Subroutine to calculate the time required by the
1043 REM computer to create and print the maze.
1044 REM
1050 S1=VAL(RIGHT$(T2$,2))-VAL(RIGHT$(T1$,2))
1060 S2=VAL(RIGHT$(T2$,5))-VAL(RIGHT$(T1$,5))
1070 S3=VAL(T2$)-VAL(T1$)
1080 IF S1<0 THEN S1=S1+60:S2=S2-1
1090 IF S2<0 THEN S2=S2+60:S3=S3-1
1100 RETURN
1101 REM

```


COLOR COMPUTER CORNER

HOT COCO

HOT CoCo is a new magazine for Color Computer owners, published by Wayne Green Publications (who are the owners of several other successful microcomputer magazines, including *80-Micro*, *Desktop Computing*, *Kilobaud*, and *InCider*).

This is at least the third publication for Color Computer owners to appear this year, joining the *Color Computer Weekly* and *The Color Computer Magazine*, which both appeared in January. These magazines reflect the fact that the Color Computer is quickly growing in popularity, and more manufacturers are now trying to tap this market.

The first issue of *HOT CoCo* is full of good programs and articles that any C.C. owner will get a lot out of, and the magazine is highly recommended (as are the other two publications). The initial issue includes a project for adding a Model III-style keyboard to the C.C., a text processor/program generator, a 3D graphics program, a guided tour through the system RAM, program merging maneuvers, speech for the Color Computer, two program documentation utilities, a BASIC word processor, a physics demonstration program, and three games: Cavehunt (adventure-type game), Draw Poker, and Color Backgammon. For more info., contact *HOT CoCo*, Subscription Department, P.O. Box 975, Farmingdale, NY 11737. Subscription rates in the U.S. are \$25 per year, \$38 for two years, and \$53 for three years.

The *Color Computer Magazine* is another very worthwhile publication. It's published by New England Publications, Inc., costs \$24 per year, and can be ordered from *The Color Computer Magazine*, P.O. Box 468, Hasbrouck Heights, NJ 07604.

EXPANSION BUS

Basic Technology has introduced a five-slot bus extender for the Color Computer that will allow the addition of serial ports, parallel ports, disk controller and other cartridges to the Color Computer—all connected at the same time. The BT-1000 Expansion Interface Unit uses a 40-wire cable and buffer cartridge, with its own internal power supply, memory decoder, gold edge connectors, and four 24-pin RAM/ROM sockets. 8K of factory-installed RAM is an option. The Expansion unit is available for \$270 (\$300 with 8K RAM installed). For further information, contact Basic Technology, P.O. Box 511, Dept. S, Ortonville, MI 48462;

(313) 627-2002.

MORE C.C. SOFTWARE

Nelson Software Systems is a good company to watch—they have released some impressive software for the Color Computer and are working on releasing even more. Some of their more noteworthy selections are:

Super "Color" Writer II, an advanced word processor for the Color Computer, is a full-featured machine language word processor that provides C.C. owners with just about every feature you could want—from full screen editing to text file chaining, from right justification to global search and replace. It is available in tape, ROMpack or disk versions, costing only \$49.95, \$74.95 and \$99.95 respectively.

Super "Color" Terminal, also available in tape, ROM or disk versions, is a good smart terminal system for the C.C. Hook up to Dow Jones, Compuserve and The Source, or communicate with just about any other computer that has a similar communications utility. Price: \$39.95 for tape, \$49.95 for ROMpack, \$69.95 for disk.

Super "Color" Mailer is a multi-purpose file merging program that uses files created by the Super "Color" Writer II. It will produce form letters, including special phrases inserted into the body of the letter. The program can also be used for printing labels, invoices, addressing envelopes, and producing "boiler plate" legal documents out of many different paragraphs. Features include the ability to selectively print mailing lists by any of up to 10 user-defined fields, automatic printing of the date, address, salutation, closing, P.S., etc. The program will print any ASCII file, and will justify text. Tape version costs \$39.95, the disk version goes for \$59.95.

Super "Color" Calc is going to be available soon, and it's the first example we've seen of a full-featured VISICALC-type spreadsheet program for the Color Computer. The files it creates will be compatible with Super "Color" Writer II, so that you can combine spreadsheet tables with your documents to create ledgers, projections, statistical and financial reports. (We don't know the price of this one yet.) For more information, contact Nelson Software Systems, 9072 Lyndale Avenue So., Minneapolis, MN 55420; (612) 881-2777. ■

```
1102 REM Subroutine to print the maze using standard ASCII
1103 REM symbols (+ - ! ). This routine should be used if
1104 REM you don't have the line printer VII. Change
1105 REM lines 430 and 660 to "GOSUB 1110".
1106 REM
1110 IF X=1 THEN LPRINT"START" ELSE LPRINT TAB(2*X-3)"START"
1120 FOR I=1TOL
1130 FOR J=1TOW
1140 IF M(J,I)-12>=0 THEN LPRINT"+ ";GOTO 1180
1150 IF M(J,I)-8>=0 THEN 1170
1160 IF M(J,I)-4>=0 THEN LPRINT"+ ";GOTO 1180
1170 LPRINT"+-";
1180 NEXT J
1190 LPRINT"+"
```

```
1200 FOR J=1TOW
1210 IF M(J,I)-8<0 THEN LPRINT"! ";GOTO 1230
1220 LPRINT" ";
1230 NEXT J
1240 LPRINT"! "
1250 NEXT I
1260 FOR I=1TOW
1270 IF FIX(M(I,L)/2)<M(I,L)/2 THEN 1290
1280 LPRINT"+-";GOTO 1300
1290 LPRINT"+ ";
1300 NEXT I
1310 LPRINT "+ "
1320 IF Y=1 THEN LPRINT"FINISH" ELSE LPRINT TAB(2*Y-3)"FINISH"
1330 RETURN
1340 END ■
```

HARDWARE REVIEW

MDX-2 INTERFACE EXPANSION BOARD

K. I. Brown

The MDX-2 Interface Expansion Board, made by Micro-Design, P. O. Box 748, Manchaca TX 78652, is an alternative interface for the Radio Shack TRS-80 Model I Microcomputer. In my readings of *80 Micro* and other related magazines, not much press is given to this board.

My discussion of this interface will cover what I ran across in building the semi-kit version (they supply the board—you retrieve the hundreds of parts).

The MDX-2 is manufactured by Micro-Design of Manchaca, Texas. The bare board and user's manual retail for \$74.95. As of this writing, a fully assembled version is offered for \$399.95. Also, a data separator is available, yet I am not able to comment on the device because I have not used it.

Several manufacturers now offer custom enclosures for these boards. For a complete list of prices and other expansion offerings, see the advertisers' index in your most recent issue of *80 Micro*.

The MDX-2 is fully hardware and software compatible with the Radio Shack Model I Microcomputer. It has all of the features of the standard interface plus:

- A direct-connect crystal controlled modem

- A 2K or 4K EPROM option

- Super-quiet memory design

- Contact-less RS-232-C Interface

The power supply is on board with the exception of the power transformer and two diodes. This transformer could be either the standard computer power supply (#4000007) or a stand alone transformer and diodes (I will cease to list part numbers since my manual is an earlier version).

I mounted a separate transformer inside my monitor (atop the one for my CPU, in the area where the tuner normally would reside), brought out the cable through the bottom of the monitor, and terminated it with a plug to mate a similar cable from the J1 power connection of the main board. The +5, -5, +12, -12 volt regulators are all on board. IC regulators are provided except for the -5V source, which delivers minimal current. The 78H05 regulator for the +5V source must be heat sunk, for when all options are installed it get "HOT." This seems to pose no problem, for my unit has run several days without trouble, yet a fan would not hurt. Jumpers are provided on the outputs of all these supplies to aid in testing and debugging.

This unit will expand the computer to a full 48K of memory. The memory control signals MUX, CAS, TAS, and WR are all terminated at the expansion connector. The address lines of all the standard 16K x 1 RAMs are terminated with series resistors for a more reliable memory. Memory tests and programs without disk access run flawlessly at 3.56MHz.

The floppy disk controller circuitry runs along the same vein as does the standard interface. With my unit, the disk drive booted up normally the first time out—well, maybe the second time. The only problem there was a 4MHz crystal that would not oscillate. In the circuit there is an update that U14 should be or need to be a 74SO4, not just an 74LS04, the former being the

faster of the two. Since all of the integrated circuits are in sockets, I tested them both and the 74SO4 was the only one to oscillate. Please be aware also that some cuts of crystals will not work in some configurations, so try a different type.

The serial interface again is standard in that UART addresses are the same. The main difference lies in the fact that this serial interface is an integral part of the main circuit board. The result is NEVER having to worry about adjusting the position of the board in order to obtain a good electrical contact, heat or warpage, or any other related malady. One unfortunate omission is that of a programmable baud rate generator. If a terminal program reads the baud rate switches, it thinks the baud rate is set for 9600. There are NO baud rate SENSE switches there, so that an IN E9 (read sense switches) generated by the terminal program always sees data bits D0, D1, and D2 as being high. The user may try to change the baud rate from within his or her program, and confirmation will appear on their crt, but the OUT E9 (set baud rate) is not decoded in the MDX-2. The baud rate is set by dip switches that pass the clock on to the UART. It should not be too difficult to replace these switches with electronically programmable ones. If you want to alternately use the RS-232-C port and your on-board modem, an external switch should be installed to select between the two.

The on-board modem uses a MC14412 chip that can operate in the originate or answer modes. It is crystal controlled so that drift needn't be a worry. A 600 ohm isolation transformer is required to match the telephone line to the output of the modem. With two external modular telephone jacks and a DATA/TELEPHONE switch mounted on the enclosure, a unique modem is yours.

The line printer bus is the least complicated to build, yet there are signals from other circuits that it requires in order to operate. This holds true for most of the interface.

The EPROM option is unique in that the user is allowed to store his firmware in the upper 2/4K of RAM. Thus non-volatile drivers, subroutines, or programs are ready for use. If the EPROM option is jumpered in and the upper 2/4K of memory are addressed, RAM will deselect EPROM will select, and no bus contention will occur.

There are diagrams available to construct a dual cassette line, which mainly involves external hardware. With critical volume settings, infinitesimal loading times, and a host of other related disorders, I saw no reason for two cassette recorders.

CONSTRUCTION

The first step in the construction phase is to purchase the board. Mine arrived in about 4 days. A careful inspection of the board revealed no visible damage or defects. One is allowed to purchase the manual first to judge the worth of the product. The cost of the manual is deducted from the full purchase price if you then decide to obtain the board.

The assembly begins with the owner deciding what sections are to be implemented into the board. There is a master parts list of everything needed to complete the entire interface. A separate parts list is given for each individual section. For each section that will be built, you check off any part on the individual list from the master. This way a duplication of inventory is not developed for sections requiring the same parts.

Now we begin by sorting all of the parts into groups. Next install all the needed sockets and solder into place. If you bend the diagonal end leads before soldering, the sockets won't fall out when you turn the board over.

This continues with the remaining parts in a systematic order governed by the check list in the assembly instructions. As always, be careful of solder bridges and poor (cold) solder joints. For a more professional look, deflux the board after all parts are installed. Not only does this look good, but it aids in locating shorts and poor solder joints. The last instruction step is very important, even though you know that a flawless assembly was performed, and that is to not insert any of the IC's.

Now that everything is soldered in place, we can run some tests. First, we check all Four power sources. The +5, -5, +12, and -12 volt supplies must be verified unloaded. Next, the outside world jumpers and soldered in place, and this test is once again repeated. These two steps show the ease in the isolation on of any supply voltage problem, either the supply or the board. If all is not well, we can check our work over to isolate the problem (solder shorts, capacitor polarities, etc.).

The IC's are next. Carefully check orientation and pin alignment. Power down the interface and install the necessary chips. Restore power and check the four supply voltages. If any are down or non-existent, correct the problem before proceeding. One will get you twenty that if you now have a problem, a chip is probably in upside down.

A cable now has to be had for the connection to the computer. Two choices are available: buy one or build it. 40 conductor ribbon cable and the proper insulation displacement connectors are available from Radio Shack, or you may opt for an assembled unit from the computer center. A vice is the best method for assembly of your cable if you plan to do it yourself. I noticed that the numbers on the connectors did not match the board numbers on the computer or interface, yet the pattern was the same for the computer/interface. This may distort your sense of well being, but the pins of the computer connect to the identical pins of the interface. A pictorial is given in the manual to show the right orientation of the assembled cable. If standoffs are mounted on the board, the component side will be facing up, and a fold is needed in the cable. The disk and printer cables attach to the interface with the cable exiting below the connector. With an enclosure for the board (at least mine), the board is mounted upside down, so arranged that the cpu and interface connectors face each other. The cable from the cpu/interface now attaches without a bend. The disk and printer cables now have the cable exiting above the connector. I can't compare this to the Radio Shack interface since I have never owned one. If you purchase a joystick, and the label on the connector reads "THIS SIDE UP," then place this side down.

Testing of the system in actual operation begins with the power up sequence. First the monitor, then the interface, and last the computer. You will find it easier to just have everything plugged into the same strip. If you have the disk controller circuitry, but no disk, hold the BREAK key down on power up or reset. MEM SIZE twice will appear on the screen. Press ENTER to get the READY prompt. The next logical step is to find out if your computer recognizes the new memory installed in the interface. Typing ?MEM should show results greater than 48000 for the full 48K system or a number larger than 31000 with 32K. Later, a memory test such as TEST1, available on TRSDOS 2.3, would be an excellent idea (no, you can't run it now, because we haven't tested the disk yet).

The serial port can be tested by tying the output to the input. The separate receive and transmit baud rates must be set. In our test we can select any two values, provided they are the same. One switch only for the receive and transmit clocks can be turned on. Next we write a byte to the serial port, then read the byte looped back. If the two bytes match, all is well. All of this is easily performed with the serial driver program provided with the manual or any terminal program of your choice that uses the standard port addresses. Jumpers are offered between the serial port in/out lines and the DB25 connector to be either an input or output, thus the proper match for DCE (data communications equipment) or DTE (data terminal equipment).

The modem is driven by the serial port. To test, we connect the correct jumpers from the serial port. The MC14412 chip has an internal test/loopback procedure that is outlined in the manual. I was unable to obtain any results with this test, yet the modem worked fine when connected on line to a bulletin board. Never being able to find a data sheet on this chip, I cannot explain why.

The printer port is simply tested by the LPRINT statement in basic.

The floppy disk controller is tested by booting a diskette. If this works, try backing up a diskette to ensure that the WRITE functions of the controller are in order.

I won't cover the EPROM and cassette tests, since I did not build either. The manual is rather explicit on these two areas.

TROUBLESHOOTING

First the power up the system and check to see if the computer functions (hold down break if disk is not present). If the computer is locked up, power down, remove the interface cable, and retest. If the interface is causing a problem, check the interface cable for shorts and proper orientation. Check supply voltage in the interface as outlined earlier. Now, reconnect the interface and try again. If the problem still persists, remove all IC's and power from the interface, connect interface cable and try again. An inoperative computer now tends to point to a short on the circuit board. In my case, I located two addresses lines shorted together. After clearing the short, the computer operated with the complete interface.

continued on page 61

YOUNG PERSON'S MATH PROGRAM

Robert V. Pritula

As a parent of small children, I am constantly being asked by them for math problems. After a while, I got quite tired of giving them problems, but on the other hand, I felt guilty if I told them I didn't want to do it.

I wrote this small program, which solved these problems, and it allowed them to play with daddy's toy.

The program is designed for children learning elementary math. It will give children a series of 10 problems, plus a summary of how well they did. If they miss a problem, they will have two other chances to answer the problem before the program tells them the correct answer.

The child (or parent) can determine the difficulty of the math to be done. The division problems will only generate problems that will have answers in whole numbers (no fractions).

The program was written on a Model I, Level II computer.

```
4 ' YOUNG PERSON'S MATH PROGRAM BY ROBERT PRITULA, 144
TRICKOVIC, BATTLE CREEK, MI 49017
5 CLEAR(200):CLS:PRINT CHR$(23):T$="###,###,###":RANDOM
10 PRINT @448,"WHO IS GOING TO LEARN TODAY?":PRINT :INPUT"TYPE
NAME ";N$:P=0
11 CLS:PRINT CHR$(23):PRINT N$,"":PRINT "ENTER NUMBER FROM 10 TO
10000":INPUT XX
15 IF P=1 THEN GOSUB 1000
20 CLS:PRINT CHR$(23):PRINT "HELLO "N$:PRINT :PRINT :PRINT
"PRESS ' 1 ' TO DO ADDS":PRINT "PRESS ' 2 ' FOR TAKE
AWAYS":PRINT "PRESS ' 3 ' TO DO TIMES":PRINT "PRESS ' 4 ' TO
DO DIVISION":H=0:Q=0
30 A$=INKEY$:IF A$="" THEN 30
40 ON VAL(A$) GOTO 50,200,250,300
50 W$="ADD":GOSUB1100
60 IF Z=X+Y THEN GOSUB 400 ELSE GOTO 80
70 IF H>9 THEN GOTO 15 ELSE 50
80 GOSUB 500
90 GOTO 60
200 W$="TAKE AWAY":GOSUB 1100
210 IF Z=X-Y THEN GOSUB 400 ELSE GOTO 230
220 IF H>9 THEN GOTO 15 ELSE 200
230 GOSUB 500
240 GOTO 210
250 W$="TIMES":GOSUB1100
255 IF Z=X*Y THEN GOSUB 400 ELSE GOTO 265
260 IF H>9 THEN GOTO 15 ELSE 250
265 GOSUB 500
270 GOTO 255
300 W$="DIVIDE BY":GOSUB1100
305 IF Z=X/Y THEN GOSUB 400 ELSE GOTO 315
310 IF H>9 THEN GOTO 15 ELSE 300
315 GOSUB 500
320 GOTO 305
350 V=0: PRINT N$ - PLEASE "W$: X=RND(XX): Y=RND(XX):IF X<Y
GOSUB 700
355 IF VAL(A$)=4 GOSUB 1200
360 PRINT @ 286,"":PRINT TAB(20):PRINT USING T$;X:PRINT @384,W$:
361 PRINT TAB(20):PRINT USING T$;Y
362 PRINT @ 512,"ANSWER IS ";:PRINT @546,"";:INPUT" ";Z:F=F+1
370 RETURN
400 PRINT @768,"GOOD ANSWER, "N$:P=1:IF V<1 THEN Q=Q+1
405 GOSUB 600
410 RETURN
500 IF F>2.5,GOSUB 900
```

```
501 V=1:PRINT @768,"THE ANSWER IS WRONG. TRY AGAIN":GOSUB 600
510 PRINT @768,STRINGS(63," "):PRINT @512,STRINGS(63," " )
520 PRINT @512,"NEW ANSWER IS ";:PRINT @546,"";:INPUT" ";Z:F=F+1
530 RETURN
600 FOR I=1TO750:NEXT:RETURN
700 IF VAL(A$)=1 OR VAL(A$)=3 THEN GOTO 720
710 C=X:X=Y=C
720 RETURN
900 CLS: PRINT CHR$(23): PRINT "YOU HAVE MISSED 3 TIMES!!!":
PRINT :PRINT "CORRECT ANSWER IS "
910 IF VAL(A$)=1 THEN Z=X+Y
911 IF VAL(A$)=2 THEN Z=X-Y
912 IF VAL(A$)=3 THEN Z=X*Y
913 IF VAL(A$)=4 THEN Z=X/Y
915 PRINT :PRINT X" "W$" "Y" = "Z
920 PRINT : PRINT "PLEASE REMEMBER": PRINT : PRINT "PRESS ' A
' TO CONTINUE"
930 G$=INKEY$:IF G$="" THEN 930
940 ON VAL(A$) GOTO 50,200,250,300
950 RETURN
1000 CLS:PRINT CHR$(23):PRINT N$:"
1010 PRINT :PRINT "YOU GOT "Q" "W$"S RIGHT":PRINT " OUT OF 10
POSSIBLE.":PRINT :IF Q=10 THEN PRINT "EXCELLENT, "N$
1020 PRINT :PRINT "PRESS ' 1 ' TO CONTINUE"
1021 B$=INKEY$:IF B$="" THEN 1021
1030 RETURN
1100 CLS:PRINT CHR$(23):F=0:H=H+1:GOSUB 350:RETURN
1200 IF INT(X/Y)=X/Y THEN RETURN
1210 X=RND(XX):Y=RND(XX):IF X<Y GOSUB 700
1220 GOTO 1200
1230 RETURN
```

Robert V. Pritula

144 Trickovic

Battle Creek, MI 49017 ■

KOPYKAT

continued from page 46

02870	JR	Z,EXIT1	;YES = EXIT FAST	
02880	INC	HL	;ELSE GET NEXT CHAR.	
02890	DEC	C	;SUBTRACT ONE FROM COUNT	
02900	JR	NZ,LP1	;CONTINUE FOR 64 CHARS.	
02910	CALL	LPTST	;CHECK PRINTER	
02920	LD	A,CRLF	;CARRIAGE RETURN	
02930	LD	(LPORT),A	;OUTPUT IT TO PRINTER	
02940	NEXT	DEC	E	;GET NEXT LINE
02950	JR	NZ,LOOP	;GO FOR ALL 16 LINES	
02960	JR	EXIT1	;EXIT IF DONE	
02970	LPTST	LD	A,(LPORT)	;GET PRINTER STATUS
02980	CP	3FH	;EQUALS PRINTER READY	
02990	JR	NZ,LPTST	;LOOP UNTIL READY	
03000	RET		;THEN RETURN TO CALLER	
03010	EXIT1	CALL	LPTST	;CHECK PRINTER
03020	LD	A,CRLF	;DO A CARRIAGE RETURN	
03030	LD	(LPORT),A	;OUTPUT TO PRINTER	
03040	EXIT	EXX	;SWAP REGISTERS	
03050	EX	AF,AF'	;ALL OF THEM	
03060	RET			
03070				
03080	END	BEGIN		

Joe W. Rocke

224 W. Benson

Ridgecrest, CA 93555 ■

EVERLASTINGLY AT IT

Video's Visible Future: The Computer Connection

"We ain't seen Nothin' yet!"

Mike Shadick

We've seen the leading edge of this emerging technology, already. When Atari and others introduced the first arcade video games way back (!) in the mid-70's, they represented the first direct application of microprocessor technology (i.e., computery) to live, viewer-controlled video. The video console and the computer had, in effect, merged. Yet this was—and is—only the beginning!

Today's videonic microprocessors will grow into tomorrow's full-fledged video computers, growing ever smaller (except for the viewscreen!), ever more powerful, and ever more affordable by the rank-and-file videophile.

What, then, are some of the ways in which the ongoing video/computer marriage will affect the shape of video to come, and thus affect our future lifestyles? Let's take a conservatively futuristic look at the probable state of the science and art and practice of video computery, at the dawn of the Third Millennium A.D. (Why, that's less than two decades away!) Let's also consider how profoundly will video be influenced by that little box of electronic micromiracles which is coming to be known as the videonic computer, or simply *videocom*.

Talk about GAMES!

By the year 2000 (and probably long before), video gaming will have reached such a sophisticated state that many of the available games will be vying with their respective real McCoy's!

Let me be more specific. Take video tennis, as but one example. It will be playable at a nearly infinite number of "challenge" levels, the most difficult of which will tax the abilities of future Tracy Austins and Bjorn Borgs! *How much* of a challenge will they find the video version to be? *Every bit as much as the game itself*. Indeed, the videonic equivalents to football, soccer, golf, you name it, will bear uncannily realistic resemblances to their real-life counterparts.

How will such be made possible? What will make video games so enthrallingly realistic? It will be the application of an entirely revolutionary visual technology to video: the technology of *holography*.

"Is it live, or is it ALIVE?"

As you may have heard—or, better yet, have seen with your own eyes—holography is the application of laser technology to the projection of three-dimensional images. The projected *hologram* is so realistic in appearance, that what your eyes tell you you're seeing is not merely an *image*, but rather, the actual people, places, and things being projected. Holograms are—visually, at least—*identical* to the real things themselves.

Moreover, the holographic illusion is heightened by the fact that when your visual vantage point changes, so does the *hologram*! Thus, you can, in fact, walk completely around a holographic image, and its

appearance to your eyes will change exactly as if you were walking around the person, object, and/or scene itself.

Uncanny? Only when your mind *tries* to tell you that what you are witnessing is nothing but an illusion!

Holographic video, then, will be able to flawlessly simulate *just about anything*. In football, for example, the field and players and crowd will appear to actually be present—in a miniaturized form, of course, owing to the fact that most living rooms are somewhat smaller than a football stadium!

The Digital Revolution

DVR.

You might as well get used to those three letters, for you'll be seeing a lot of them in the years to come. Though *Digital Video Replication* is, as yet, still in its infancy, its predecessor—digital *audio* replication—is, of course, well on its way into the home, having been already developed and (nearly) perfected in recording studios throughout the country. Indeed, if you are anything of an audiophile, you doubtless already own many record albums recorded with digital equipment.

So why all the fuss over digital? What makes it so

BUSINESS OPPORTUNITY

Exclusive franchise in America's most profitable and dynamic industry is being offered for the first time in your area. International company will place qualified individual in "Turn Key" business, train key people, provide inventory, finance your customers, and pay you thousands of dollars "up front" on orders where your customers pay only on future energy savings. Existing customers of our franchise reads like "Who's Who" of *Fortune* 500.

If you qualify, you will be flown to Los Angeles for a tour of installations and personal interview. Minimum investment of \$29,500 cash required. Call president at 1-800-323-6556, ext R-137.

FEDERAL ENERGY SYSTEMS, INC.

Suite 200, 336 N. Foothill Road,
Beverly Hills, CA 90210

THIS IS NOT AN OFFERING TO SELL

superior to conventional analog recording? Many, many things!

For starters, distortion levels of the digital recording process are, for all practical and most measurable purposes, *non-existent*. Which is to say that there is no, repeat *no*, quality loss between an actual performance—be it Barry Manilow or the neighbor's dog—and the digital playback of same. Distortion, you see, *just can't happen*, simply because a digital recording isn't really a recording at all. Rather, it's an electronic duplicate or replica of the original sounds. Indeed, a digital audio replica is a perfect electronic *clone*, identical to its sonic source in every way.

The originally-generated sounds—music, voice, whatever—are captured and stored electronically, via a microprocessor/computer, in a series of on/off (digital) impulses. Any given sound is replified into a huge number of impulses (thousands per second), each of which is either an *on* or *off* impulse. There's nothing in between. That is, the computerized replication is capable of only on's and off's. Thus, it is literally *incapable* of distortion.

...and the Rest is HISStory

You're undoubtedly familiar with the background tape "hiss" annoyingly audible in even the finest analog recordings, especially at high playback volume levels. Well, the tape hiss is completely *absent* from the digital replicating process. Yes, even at ultra-high playback levels. All you hear, then—quite literally—is digitally-replicated sounds themselves. Indeed, that's all the digital replicator is capable of producing! Nothing less—and, even more significantly, nothing *more*.

The bottom line? The reproduction quality inherent in digital replication is—well, *perfection*.

To take advantage of it, of course, one would need true digital *playback* equipment as well, instead of the conventional stereophonic analog equipment in most homes today. Thus, when we lay claim to owning the latest digital records, we are only *half* correct. Granted, the original sounds were (and are!) replicated digitally. But, unless we're among the super-rich who can dish out five- and six-figure sums being asked for digital playback devices today, we must settle for digital recordings—that is, digital replicas "analogued" into conventional stereo records for playback on standard home equipment.

It's a little like *smelling* a sirloin steak, without *tasting* it.

Digital Video? Not Tommorrow!

As well you might imagine, it'll be quite a ways down the electronic pike before we'll be tasting Digital Video Replication in our homes! Or *will* it? The advantages of DVR over analog video will, of course, be everything that digital audio is—and much, much more! With full DVR capacity—that is, in replication *and* playback—we will enjoy, not only a video picture superior in every way to even the very best available today; we will also be able to make as many video "dubs" or "reps" as desired, *with no quality loss whatsoever*.

Imagine, for example, that you have a friend who has a video tape of *her* friend's video tape of *his* friend's video tape of a certain TV show, and you want a copy for yourself. So you borrow the first friend's tape and make a copy of it. What will you have? *Today*, of course,

you'd have a sixth-generation dub, six degrees removed from the quality of the original tape. But with DVR technologies of the future, you'd have a copy *indistinguishable in any way from the original*.

Indeed, what you would in fact have is an identical electronic *twin* to the DVR original. You dub would, in effect, *be* its source, as would be all of the dubs in between!

DVR Holograms, and Beyond?

The future possibilities of *combining* the two emerging technologies already touched upon—holography and Digital Video Replication—are not merely visionary in nature. *They are already on the drawing boards*. For example:

Imagine, if you will, having the capacity to "copy" (in sight and in sound and perhaps in other sensory parameters as well) people, places, and things in a totally *lifelike* (what a wholly inadequate word, in describing the wonders of holographic video!) manner. And being able to make them reappear before you, at the push of a button!

And yet, that's only the beginning. Indeed, at this point the possibilities come close to boggling the mind of even the most far-sighted futurist.

Suffice it to say that, in terms of video's truly fantastic future potential, we've *only just begun*.

Mike Shadick
Cedar Square West, Apt. E-414
1515 South Fourth Street
Minneapolis, MN 55454 ■

USING NEWDOS/80 DISK FILES, PART II

continued from page 52

Two things have been accomplished by the above changes. First, a record counter, the variable X, has been eliminated. Second, the BASIC function LEFT\$ has been eliminated. In Line 100 we replaced the X=0 with GET1,!\$0. X was previously our record counter and we had to initialize it equal to zero. By using GET1,!\$0 we are positioning to the zero byte from the beginning of the file. That is to say we are positioning at the beginning of the file. Line 110 is setting the variable A equal to the length of A\$. Since no record number or positioning marker is specified the next record is selected. The GET statement in line 30 contains its own IGETL. If A=10 then according to the IGETL in the GET statement only the first 10 bytes of each record will be gotten from the disk file and used in the comparison on line 40. In line 40 if A\$=NA\$ then a second GET is executed. The # symbol in the GET statement tells BASIC to access the same record number again. This time however we use the IGETL in line 15 which will access the whole record. The IGETL in line 30 is a partial reading of the record. It only accesses the number of characters equal to the variable A, which is equal to the length of A\$. If A=10 then instead of accessing all 75 bytes of the record for each record, it only accesses 10 bytes for each record until a match is found. That means that if you have 100 records and used the original program you would access 7,500 bytes to find the last record. Using the revised program and assuming the search name is equal to the maximum length of 25 characters you

would access 2,500 characters to find a match and then a whole record reading of 75. This will result in a total accessing of 2,575 characters instead of 7,500 characters.

In my next article I will discuss other methods that can be used to search or access a file as well as methods to edit or update a file.

John L. Gross
208 Main Street
Akron, PA 17501 ■

HARDWARE REVIEW

continued from page 57

Problems in the memory area fall generally into two areas: all or a block of memory not being selected, or memory errors. If you have 48K or 32K of memory in the interface, yet ?MEM does not verify this, one or more blocks aren't being selected. A statement such as 10 POKE-1,5: PRINT PEEK(-1):GOTO 10 provides a loop of read/write activity in 32K section. While this program is running, you can use a logic probe to test for activity on the RAS, W, CAS, and address lines of the memory chip. The CAS signal is gated through by the address decode signal. By changing the value -1 in the program line to, say, -30000, the upper block of memory can be tested in the same manner.

Memory errors are best detected with a memory test program. I used the program on TRSDOS, TEST1, to locate a defective chip. Will running the program, a chip went flaky. The program told me the chip's location. I exchanged this chip with another on the board, ran the test again, and the program pointed to the new location. I exchanged this chip with another on the board, ran the test again, and the program pointed to the new location. The only problem was that the program used the chip designation of the Radio Shack expansion interface. This was not a problem for me, since I own The Radio Shack Interface Service Manual. It was just a matter of picking the chip in my interface that was at the same location of the Radio Shack chip. Random errors at the bit location most likely will be the chip. If errors are produced for every address at that bit position, even with a different chip, check your soldering and circuit continuity in that area. A shorted chip is easily found by touching for excessive heat. If one of the memory pulls down a supply voltage (three for the RAM), remove one at a time until the voltage returns. The last one removed is defective.

This simple type of troubleshooting can be used throughout the rest of the interface. Just type in a simple loop that reads and writes to the address of the area at fault. Check for address decoding and other activity necessary. A TTL cookbook or data sheets, a logic probe and/or scope, volt/ohm meter, and the manual are about all that is needed to debug the interface. Don't hesitate to contact MICRO DESIGN for assistance.

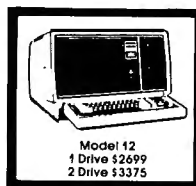
CONCLUSION

I am satisfied with my purchase of the MDX-2 board.

continued on page 68

From Computer Plus to YOU ...

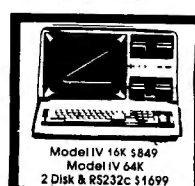
PLUS after PLUS after PLUS



Model 12
1 Drive \$2699
2 Drive \$3375



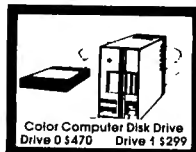
Color Computer 16K \$175
w/16K Ext. Basic \$255
w/32K Ext. Basic \$345



Model IV 16K \$849
Model IV 64K
2 Disk & RS232c \$1699



Okidata 80 \$320
Okidata 82A \$399
Okidata 92 \$510



Color Computer Disk Drive
Drive 0 \$470 Drive 1 \$299



Smith Corona TPI
Daisy Wheel \$495

BUY DIRECT

Here are just a few of our line offers ... call TOLL FREE for full information.

COMPUTERS		R S Acoustic Coupler AC-3	129	DISK DRIVES	
Model 12 64K 1 Drive	52699	R S Modem I D C	129	R S Model IV 151-Drive	505
Model 12 64K 2 Drive	3375	R S Modem II D C	199	Tandon 40 Track MI	289
Model IV 16K	849	PRINTERS		Color Computer Drive 1	299
Model IV 64K		Daisy Wheel II	1715	Color Computer Drive 0	470
2 Disk & RS232 c	1699	DWP-410	1320	Primary Hard Disk Mill	3099
Color Computer 16K	175	Smith Corona TPI Daisy Wheel	495	Primary Hard Disk Mill	2199
Color Computer 16K		Epson		ETC.	
w/extended basic	255	GGP-115	199	CCR-81 recorder	52
1 Color Computer 32K		DMP-100	315	C C Joysticks	22
w/extended basic	345	DMP-120	410	16K RAM chips	25
Packet Computer 2	165	DMP-200	599	64K Ram Chips	75
Packet Computer 4	59	DMP-400	1010	Coco FHL Flex D O S	69 95
Model 16 1DR 128K	4199	DMP-500	1539	32K Microbuffer Inline	229
Model 16 2DR 128K	4799	DMP-2100	1779	SOFTWARE	
Model 100 8K	679	Okidata 82A	369	Brand Name Software *	
Model 100 24K	835	Okidata 83A	655	Send for listing	
MODEMS		Okidata 84 Parallel	999	R S Software 10% off list	
Lynx Direct Connect MI/MIII	235	Okidata 92	510	Parallel Printer Cables are	
Hayes Smart Modem II	235	Okidata 93	859	available for most computer	
Hayes Smart Modem 1200	565	Gemini 10	319		
Novation Smartmodem 1200	459	Prowriter	375		
Novation J-CAT	125	P C Plotter Printer	180		

We have the lowest possible Fully Warranted Prices AND a full complement of Radio Shack Software.

Prices subject to change without notice. Not responsible for typographical errors. 195-80 is a registered trademark of Tandy Corp.



TOLL FREE
1-800-343-8124
computer plus
P O Box 926
480 King Street
Littleton, MA 01460
617-486-3193
Write for your free catalog

RIBBONS

Low Price • FREE Shipping
SATISFACTION GUARANTEED

RIBBON CARTRIDGES

top quality factory fresh

Cartridges for use on these printers:	price each in quantity of
	1-5 6-23 24-99 100 +
MX-70, MX-80, IBM PC	7.41 6.45 5.61 4.88
MX-100	19.96 17.36 15.09 13.13
Prowriter, PC 8023A-01	7.98 6.94 6.04 5.25
RS LP2, LP3, LP5	7.98 6.94 6.04 5.25

RIBBON LOOPS

top quality nylon refills for your old cartridge

Loops for use on these printers:	price each in quantity of
	1-5 6-23 24-99 100 +
MX-70, MX-80, IBM PC	3.56 3.09 2.69 2.34
MX-100	5.41 4.71 4.09 3.56
Prowriter, PC 8023A-01	2.46 2.14 1.86 1.62
RS DMP 400, LP6, LP8	2.04 1.77 1.54 1.34
RS DMP 200, DMP 500	3.56 3.09 2.69 2.34
RS LP2, LP3, LP5	2.46 2.14 1.86 1.62
Spinwriter (nylon)	2.46 2.14 1.86 1.62

Cartridges and loops may be mixed for quantity prices. Our FREE CATALOG includes loading instructions for loops. Discounts available for schools. Florida res. add 5% tax.

VISA **DATA SYSTEMS** MasterCard
(305) 788-2145 • Box 99 • Fern Park, FL 32730

COMPUTRONICS CLASSIFIEDS

CLASSIFIED ADVERTISING

Introductory Rates (per insertion)	1X	3X	6X	12X
Special discount Price (to 25 words)	\$20	\$15	\$12	\$10
Special Discount Add'l charge/word	\$.45	\$.40	\$.35	\$.30

To figure cost of ad, consider words like "a," "the," "etc." as one word each. Telephone number with area code counts as two words. Please type or print your ad and send along with payment in full (check, money order, Visa, Master Card, or American Express) to *H & E Computronics, Inc., Classified Advertising Department, 50 North Pascack Road, Spring Valley, NY 10977*. Your ad will begin in the next available issue.

YOUR AD CAN APPEAR HERE for as little as \$10 per month.

INNOVATIVE SIMULATION GAMES is our specialty—whether travel-, business-, or tennis-oriented. Write for information. Triangle Software, P.O. Box 58182, Raleigh, NC 27658.

Learn Radio Shack Cobol's built-in ISAM by studying source code for a 5-key mailing list and a 5-key article index, both with printing programs. Source code can be typed into TRS-80 I, II, III, 12, or 16 Cobol, then compiled. Receive all 24 pages of source code for \$10. Or send \$25 to get it on a Model III data disk. Check or money order to: Richard Bueche, 5704 Spring Valley #1056, Dallas TX 75240.

MIND SIDE OF COMPUTER KNOWLEDGE: thru Hypnosis complicated language/signs become second nature to you. Proven to work. Send \$19.95 H.T.I., Piney Point, MD 20674.

ASYNCHRONOUS SERIAL COMMUNICATIONS BOARD for IBM Personal Computer, never used, \$90.00. (914) 634-1821.

CHECKBOOK 2.2. Balances checkbook, Prints summary, Expenses for month, etc., Sorts tool Model I/III, \$14.95. Write to:

Gopher Software
4710 Valley View
Columbus, NE 68601

TRS-80 Model I, 2 drives, R.S. upper/lower case, doubler installed, green phosphor monitor, cassette recorder, dust covers, original manuals, excellent condition. Must sacrifice for \$1,295 or best offer.

Call (914) 634-1406 after 5 P.M.

DIABLO HyType I PRINTER, letter quality line printer with tractor feed attachment, full set of typewheels and ribbons, must sacrifice for \$1495 or best offer. Interfaces to expansion port of TRS-80 Model I. Word processing software included. Original cost over \$2300. Call (914) 634-1821 after 6:00 P.M.

PROGRAMS PUBLISHED IN Computronics on diskettes. Do you want to avoid typing in all those programs published each month in *Computronics*? For \$12.00, you can have a diskette containing all the programs published in any single issue with corrections. Specify the issue number you want and whether you want Model I or Model III media (not available for other models). Order from Box A, H & E Computronics, 50 North Pascack Road, Spring Valley, NY 10977, or call toll-free order number (800) 431-2818 outside New York state; inside New York, call (914) 425-1535. Add \$3.00 shipping and handling charge.

COMPUTRONICS Issue #42 (February 1982) program diskette contains the following programs: The Graphic Pie, Mortgage Comparisons, Statement of Income, Chainer, Windcrab, Horsepower, and Permute.

COMPUTRONICS Issue #43 (March 1982) program diskette contains the following programs: Interest Formulas, Depreciation under the 1981 Economic Recovery Tax Act, Electric, Sieve, and Horizon, and CLINTEST, a series of programs to help diabetics monitor their sugar levels, in-

cluding Menu, Clintest, Foodlog, Calcount, Master and Print.

COMPUTRONICS Issue #44 (April 1982) program diskette contains the following programs: Check writer, Castle Adventure, Graphic Combinations, the Ladder, National Debt, Box, Roots, and April Fool.

COMPUTRONICS Issue #46 (June 1982) program diskette contains the following programs: Break Even Analysis, Coin Inventory program, Chase, Reaction Time, USA, Selling Price, Loan Processor, No Star Baseball, and Metric Conversion Calculator.

COMPUTRONICS Issue #48 (August 1982) program diskette contains the following programs: Information Retrieval System, including KW, Sort, Merge, and List; the Transportation method of linear programming; the Beale Treasure, including Filer, Write1, Write2, Write3, Process/Let, Process/L1, and Process/L2; and the Versatile Peeker.

COMPUTRONICS Issue #49 (September 1982) program diskette contains the following programs: the Simplex method of linear programming; Records and Files, including Schedule, Schedule/Sub, Schedule/Seq, Schedule/Ran; Monthly Expenditure Information package, including Update, Mstex, Monthre, and Report; and Scramble.

COMPUTRONICS Issue #50 (October 1982) program diskette contains the following programs: Cash flow analysis for real estate and other investments; Epson MX-80 Graftrax Initialization program; Kaleidoscope, Gascost, Permile, and Balloon by Gordon Speer.

COMPUTRONICS Issue #51 (November 1982) program diskette contains the following programs: the Cardwriter, the Graphical Method of Linear Programming, MACSMAP, and Phone bill sorting program.

COMPUTRONICS Issue #52 (December 1982) program

diskette contains the following programs: Regression Analysis with Confidence and Prediction Limits, Change baud rates on system tapes, PERT—Program Evaluation and Review Technique, Grid, and Monogram.

COMPUTRONICS Issue #53 (January 1983) program diskette contains the following programs: General Ledger Menu, Alpha Program, Roulette, Serious EDTASM, Surname Conversion, Jack and Jill, and Disk Checkbook Maintenance System.

COMPUTRONICS Issue #54 (February 1983) program diskette contains the following programs: Startup routines for General Ledger package, CLOCK/BAS, Findit, Boxer.

COMPUTRONICS Issue #55 (March 1983) program diskette contains the following programs: TRANS program to input transactions, Michigan and Graphics for Epson, Type, Alphabet Puzzle, Fraction Calculator, Memory Display, two programs for bit-image graphics. (Note: Bowling Statistics Ledger not included.)

COMPUTRONICS Issue #56 (April 1983) program diskette contains the following programs: SORTING demonstration program, CPA Program (Month #4 of General Ledger series), Ball Bearings, Football, SPACEX Assembly-Language game, Rubik's cube, and Interval Program for time-shared condominiums.

COMPUTRONICS Issue #57 (May 1983) program diskette contains the following programs: Housekeeping programs UPDATE and YEAR (Month #5 of General Ledger package), 3-Across Mailing Labels, Service Territory and Manpower Planning Simulation program, Sword of Raschkil game, Escape, Sketchit, Concentration, Tmult, Magic Squares, and Word Chase.

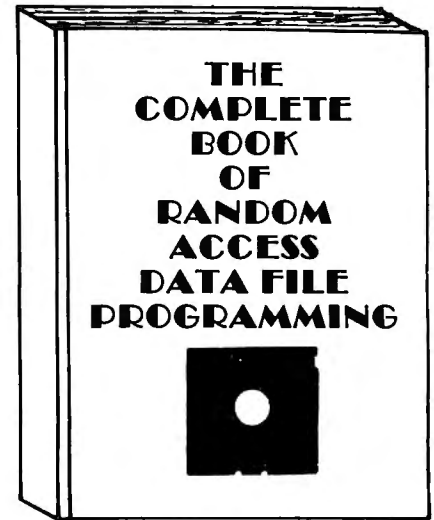
COMPUTRONICS Issue #58 (June 1983) program diskette contains the following programs: Trial Balance programs (Month #6 of General Ledger package), Chinese Zodiac, Easel program for drawing on screen in graphics.

The Complete Book Of Random Access Data File Programming

For TRS-80*, IBM Personal Computer*, Osborne*, and all Microsoft BASIC* computers

The last word on disk random access and file handling techniques, this series is intended for everyone — beginning programmers, businessmen and professionals will learn how to create custom programs to handle inventories, mailing lists, work scheduling, record keeping, and many other tasks, while more experienced programmers will learn advanced, professional programming techniques for faster, more efficient data storage and retrieval.

Although random access file handling is a matter of some complexity, the subject has been treated in a simple and down-to-earth fashion, so that anyone with some small familiarity with programming in Microsoft BASIC will be able to cope with the material. Each stage of learning uses a sample program as a starting point. The programs grow in capability and complexity as the books progress into all of the various aspects of file handling and record manipulation. An extensive effort has been made to keep the material coherent and every program line is explained in detail.



Volume I: Basic File Handling Techniques

- The writing of a Menu to Summarize program functions
- The writing of a screen format to accept data for records
- The creation of the basic record
- The FIELD and LSET routines for buffer preparation
- The writing of the record to disk in a random access mode
- The ability to change or edit a record
- The LPRINT capability from disk using three different formats
- Deleting a record from a random file
- Sorting the random file
- Searching the random file by name or key field
- The ability to search in a "NEXT or PRIOR" fashion
- The ability to purge deleted records from a disk file
- The ability to calculate with data from a disk file
- The provision for future expansion of the data fields
- The use of flags to prevent program crashes
- Date setting, printer on-line and many other routines to make a program run like a commercially-written program

Volume II: Advanced File Handling Techniques

- Blocking & de-blocking, Shell-Metzner sort, In-place screen editing, recovery of deleted record space
- Alpha-index record retrieval, fast machine/BASIC sort
- Linked list record structure and sort-merge, deleted record removal and file reorganization
- Multi-key file reorganization and record searching
- Relational database programming--comprehensive self-balancing accounting system with printouts
- Hashcoded data file manipulation--(probably the fastest method of data retrieval). Hashing the input key and recovery method explained
- Span-blocking techniques (allows; creation of records longer than 256 bytes without wasted space)

The Complete Book Of Random Access Data File Programming

Volume I: Basic File Handling Techniques	\$29.95
optional Vol. I Program Disk for Model I/III	\$28.50
optional Vol. I Program Disk for Model II	\$32.50
Volume II: Advanced File Handling Techniques	\$29.95
optional Vol. II Program Disks for Models I, II or III	\$49.95

COMPUTRONICS I N C.

50 N. PASCACK ROAD
SPRING VALLEY, NEW YORK 10977



24 HOUR
ORDER
LINE

(914) 425-1535



NEW TOLL-FREE
ORDER LINE
(OUTSIDE OF N.Y. STATE)

(800) 431-2818

ADD \$3.00 FOR SHIPPING IN UPS AREAS
ADD \$4.00 FOR C.O.D. OR NON-UPS AREAS
ADD \$5.00 TO CANADA AND MEXICO
ADD PROPER POSTAGE OUTSIDE OF U.S.
CANADA AND MEXICO

*** ALL PRICES AND SPECIFICATIONS SUBJECT TO CHANGE ***
DELIVERY SUBJECT TO AVAILABILITY

— 30-DAY MONEY BACK GUARANTEE —



H & E COMPUTRONICS INC.

•• EVERYTHING FOR YOUR TRS-80™ • ATARI™ • APPLE™ • PET™ • CP/M™ • XEROX™ • IBM™ • OSBORNE™ ••
 •• KAYPRO ••

* TRS-80 is a trademark of the Radio Shack Division of Tandy Corp. * ATARI is a trademark of Atari Inc. * APPLE is a trademark of Apple Corp. * PET is a trademark of Commodore
 * CP/M is a trademark of Digital Research * XEROX is a trademark of Xerox Corp. * IBM is a trademark of IBM Corp. * OSBORNE is a trademark of Osborne Corp.



BUSINESS PAC 100

100 Ready-To-Run

Business Programs

★ All orders processed within 24-Hours
 ★ 30-Day money back guarantee

(ON CASSETTE OR DISKETTE).....Includes 128 Page Users Manual.....

Inventory Control.....Payroll.....Bookkeeping System.....Stock Calculations.....

Checkbook Maintenance.....Accounts Receivable.....Accounts Payable.....

BUSINESS 100 PROGRAM LIST

NAME	DESCRIPTION
1 RULE78	Interest Apportionment by Rule of the 78's
2 ANNU1	Annuity computation program
3 DATE	Time between dates
4 DAYYEAR	Day of year a particular date falls on
5 LEASEINT	Interest rate on lease
6 BREAKEVN	Breakeven analysis
7 DEPRSL	Straightline depreciation
8 DEPRSY	Sum of the digits depreciation
9 DEPRDB	Declining balance depreciation
10 DEPRDDB	Double declining balance depreciation
11 TAXDEP	Cash flow vs. depreciation tables
12 CHECK2	Prints NEBS checks along with daily register
13 CHECKBK1	Checkbook maintenance program
14 MORTGAGE/A	Mortgage amortization table
15 MULTMON	Computes time needed for money to double, triple, etc.
16 SALVAGE	Determines salvage value of an investment
17 RRVARIN	Rate of return on investment with variable inflows
18 RRCONST	Rate of return on investment with constant inflows
19 EFFECT	Effective interest rate of a loan
20 FVAL	Future value of an investment (compound interest)
21 PVAL	Present value of a future amount
22 LOANPAY	Amount of payment on a loan
23 REGWITH	Equal withdrawals from investment to leave 0 over
24 SIMPDISK	Simple discount analysis
25 DATEVAL	Equivalent & nonequivalent dated values for oblig.
26 ANNUDEF	Present value of deferred annuities
27 MARKUP	% Markup analysis for items
28 SINKFUND	Sinking fund amortization program
29 BONDVAL	Value of a bond
30 DEplete	Depletion analysis
31 BLACKSH	Black Scholes options analysis
32 STOCVAL1	Expected return on stock via discounts dividends
33 WARVAL	Value of a warrant
34 BONDVAL2	Value of a bond
35 EPSEST	Estimate of future earnings per share for company
36 BETAALPH	Computes alpha and beta variables for stock
37 SHARPE1	Portfolio selection model-i.e. what stocks to hold
38 OPTWRITE	Option writing computations
39 RTVAL	Value of a right
40 EXPVAL	Expected value analysis
41 BAYES	Bayesian decisions
42 VALPRINF	Value of perfect information
43 VALADINF	Value of additional information
44 UTILITY	Derives utility function
45 SIMPLEX	Linear programming solution by simplex method
46 TRANS	Transportation method for linear programming
47 EOQ	Economic order quantity inventory model
48 QUEUE1	Single server queueing (waiting line) model
49 CVP	Cost-volume-profit analysis
50 CONDPFROF	Conditional profit tables
51 OPTLOSS	Opportunity loss tables
52 FQJQQ	Fixed quantity economic order quantity model
53 FQEQWSH	As above but with shortages permitted
54 FQEQQPB	As above but with quantity price breaks
55 QUEUECB	Cost-benefit waiting line analysis
56 NCFANAL	Net cash-flow analysis for simple investment
57 PROFIND	Profitability index of a project
58 CAP1	Cap. Asset Pr. Model analysis of project
59 WACC	Weighted average cost of capital
60 COMPBAL	True rate on loan with compensating bal. required
61 DISCBAL	True rate on discounted loan
62 MERGANAL	Merger analysis computations
63 FINRAT	Financial ratios for a firm
64 NPV	Net present value of project
65 PRINDLAS	Laspeyres price index
66 PRINDPA	Paasche price index
67 SEASIND	Constructs seasonal quantity indices for company
68 TIMETR	Time series analysis linear trend
69 TIMEMOV	Time series analysis moving average trend
70 FUPRINF	Future price estimation with inflation
71 MAILPAC	Mailing list system
72 LETWRT	Letter writing system-links with MAILPAC
73 SORT3	Sorts list of names
74 LABEL1	Shipping label maker
75 LABEL2	Name label maker
76 BUSBUID	HOME business bookkeeping system
77 TIMECLCK	Computes weeks total hours from timeclock info.
78 ACCTPAY	In memory accounts payable system-storage permitted
79 INVOICE	Generate invoice on screen and print on printer
80 INVENT2	In memory inventory control system
81 TELDIR	Computerized telephone directory
82 TIMUSAN	Time use analysis
83 ASSIGN	Use of assignment algorithm for optimal job assign.
84 ACCTREC	In memory accounts receivable system-storage ok
85 TERMSPAY	Compares 3 methods of repayment of loans
86 PAYNET	Computes gross pay required for given net
87 SELLPR	Computes selling price for given after tax amount
88 ARBCOMP	Arbitrage computations
89 DEPRSF	Sinking fund depreciation
90 UPSZONE	Finds UPS zones from zip code
91 ENVELOPE	Types envelope including return address
92 AUTOEXP	Automobile expense analysis
93 INSFIL	Insurance policy file
94 PAYROLL2	In memory payroll system
95 DILANAL	Dilution analysis
96 LOANAFD	Loan amount a borrower can afford
97 RENTPRCH	Purchase price for rental property
98 SALELEAS	Sale-leaseback analysis
99 RRCONVBD	Investor's rate of return on convertible bond
100 PORTVAL9	Stock market portfolio storage-valuation program

- TRS-80 Cassette Version \$99.95
- TRS-80 (Mod-I or III), Pet, Apple or Atari Versions \$99.95
- TRS-80 Mod-II, IBM, Osborne and CP/M Versions \$149.95

ADD \$3.00 FOR SHIPPING IN UPS AREAS
 ADD \$4.00 FOR C.O.D. OR NON-UPS AREAS
 ADD \$5.00 TO CANADA AND MEXICO
 ADD PROPER POSTAGE OUTSIDE OF U.S., CANADA AND MEXICO

COMPUTRONICS!
 MATHEMATICAL APPLICATIONS SERVICE

50 N. PASCACK ROAD
 SPRING VALLEY, NEW YORK 10977

ASK FOR OUR 64-PAGE CATALOG
 DEALER INQUIRIES INVITED

NEW TOLL-FREE
 ORDER LINE
 (OUTSIDE OF N.Y. STATE)
 (800) 431-2818

24 HOUR
 ORDER LINE
 (914) 425-1535



ALL PRICES & SPECIFICATIONS SUBJECT TO CHANGE
 DELIVERY SUBJECT TO AVAILABILITY

H & E COMPUTRONICS INC.

• EVERYTHING FOR YOUR TRS-80* • ATARI* • APPLE* • PET* • CP/M* •

* TRS-80 is a trademark of the Radio Division of Tandy Corp. - * ATARI is a trademark of Atari Inc. - * APPLE is a trademark of Apple Corp. - * PET is a trademark of Commodore -
 * CP/M is a trademark of Digital Research.

100 SUPER PROGRAMS

MASTER PAC 100 2nd EDITION (COMPLETELY REVISED)

FOR YOUR TRS-80™ LEVEL II MICROCOMPUTER

ALL ON CASSETTE OR DISKETTE

BUSINESS AND PERSONAL FINANCE

1. CHECKBOOK MAINTENANCE
2. TIME FOR MONEY TO DOUBLE
3. FEDERAL FICA & WITHHOLDING TAX COMPUTATIONS
4. HOME BUDGET ANALYSIS
5. ANNUITY COMPUTATION
6. UNIT-PRICING
7. CHANGE FROM PURCHASE
8. NEBS CHECK PRINTER
9. DAYS BETWEEN DATES
10. MORTGAGE AMORTIZATION TABLE
11. INVENTORY CONTROL
12. PORTFOLIO VALUE COMPUTATIONS
13. VALUE OF A SHARE OF STOCK
14. SALES RECORD KEEPING SYSTEM
15. FUTURE VALUE OF AN INVESTMENT
16. EFFECTIVE INTEREST RATE (LOAN)
17. PRESENT VALUE OF A FUTURE AMOUNT
18. RATE OF RETURN-VARIABLE INFLOW
19. RATE OF RETURN-CONSTANT INFLOW
20. REGULAR WITHDRAWAL FROM INVESTMENT
21. STRAIGHT LINE DEPRECIATION
22. SUM OF DIGITS DEPRECIATION
23. DECLINING BALANCE DEPRECIATION
24. BREAK EVEN ANALYSIS
25. SALVAGE VALUE OF INVESTMENT
26. PAYMENT ON A LOAN
27. FUTURE SALES PROJECTIONS
28. CREDIT CARD FILE
29. ECONOMIC ORDER QUANTITY (EOQ) INVENTORY MODEL
30. VALUE OF HOUSE CONTENTS
31. TEXT EDITOR
32. MONTHLY CALENDAR
33. DAY OF WEEK
34. CASH FLOW VS. DEPRECIATION
35. COMPLETE MAIL SYSTEM
36. INTEREST RATE ON A LEASE

BUSINESS

PERSONAL FINANCE

STATISTICS AND MATHEMATICS

37. RANDOM SAMPLE SELECTION
38. ANGLO-METIC CONVERSION
39. MEAN, STANDARD DEVIATION, MAXIMUM AND MINIMUM
40. SIMPLE LINEAR REGRESSION
41. MULTIPLE REGRESSION ANALYSIS
42. GEOMETRIC REGRESSION
43. EXPONENTIAL REGRESSION
44. SIMPLE MOVING AVERAGE
45. SIMPLE T-TEST
46. CHI-SQUARE TEST
47. NORMAL PROBABILITIES
48. BINOMIAL PROBABILITY
49. POISSON PROBABILITY
50. MATRIX ADDITION AND SUBTRACTION
51. MATRIX TRANSPOSE
52. MATRIX INVERSE
53. MATRIX MULTIPLICATION
54. SOLUTION OF SIMULTANEOUS EQUATIONS
55. QUADRATIC FORMULA
56. LINEAR EQUATION SOLUTIONS
57. ROOT HALF INTERVAL SEARCH
58. ROOTS OF POLYNOMIALS
59. ROOTS-NEWTON'S METHODS
60. PRIME FACTORS OF INTEGER
61. LEAST COMMON DENOMINATOR
62. RADIAN-DEGREE CONVERSION
63. NUMERICAL INTEGRATION

STATISTICS

MATH

UTILITIES

64. QUICK SORT ROUTINE
65. PROGRAM STORAGE INDEX
66. MULTIPLE CHOICE QUIZ BUILDER
67. FORM LETTER WRITER
68. SHELL SORT
69. CASSETTE LABEL MAKER
70. CODES MESSAGES
71. MERGE TWO FILES
72. SORT WITH REPLACEMENT

GRAPHICS

73. DRAWS BAR GRAPH
74. DRAWS HISTOGRAM
75. MOVING BANNER DISPLAY

GAMBLING AND GAMES

76. RANDOM SPORTS QUIZ
77. GOVERNMENT QUIZ
78. HORSE RACE
79. MAGIC SQUARE
80. ARITHMETIC TEACHER
81. HIGH LOW GAMBLE
82. UNSCRAMBLE LETTERS
83. HANGMAN
84. GAME OF NIM
85. RUSSIAN ROULETTE
86. ROULETTE GAME
87. ONE-ARMED BANDIT
88. HIT THE TARGET
89. WALKING DRUNK
90. STATE CAPITAL QUIZ
91. TIC-TAC-TOE
92. DICE GAME
93. LUNAR LANDAR GAME
94. BIORHYTHM
95. HORSE SELECTOR (CLASS CALCULATOR)
96. RANDOM DICE ROLL
97. RANDOM ROULETTE ROLL
98. RANDOM CARD DEALER
99. GUESS THE NUMBER
100. WHITE OUT SCREEN

GAMBLING

INCLUDES 110 PAGE USER MANUAL

**GUARANTEED SATISFACTION
30-DAY MONEY BACK GUARANTEE ON ALL SOFTWARE**

*** ALL PRICES AND SPECIFICATIONS SUBJECT TO CHANGE***

COMPUTRONICS

MATHEMATICAL APPLICATIONS SERVICE

50 N. PASCACK ROAD
SPRING VALLEY, NEW YORK 10977

PLEASE SEND ME:

- MASTER PAC 100 CASSETTE VERSION \$ 99.95
- MASTER PAC 100 DISKETTE VERSION \$ 99.95
- MASTER PAC 100 (MODEL II DISKETTE VERSION) ... \$149.95



24 HOUR ORDER LINE

(914) 425-1535



**NEW TOLL-FREE ORDER LINE (OUTSIDE OF N.Y. STATE)
(800) 431-2818**

★ All orders processed within 24-Hours
 ★ 30-Day money back guarantee on all Software

CREDIT CARD NUMBER EXP. DATE

SIGNATURE

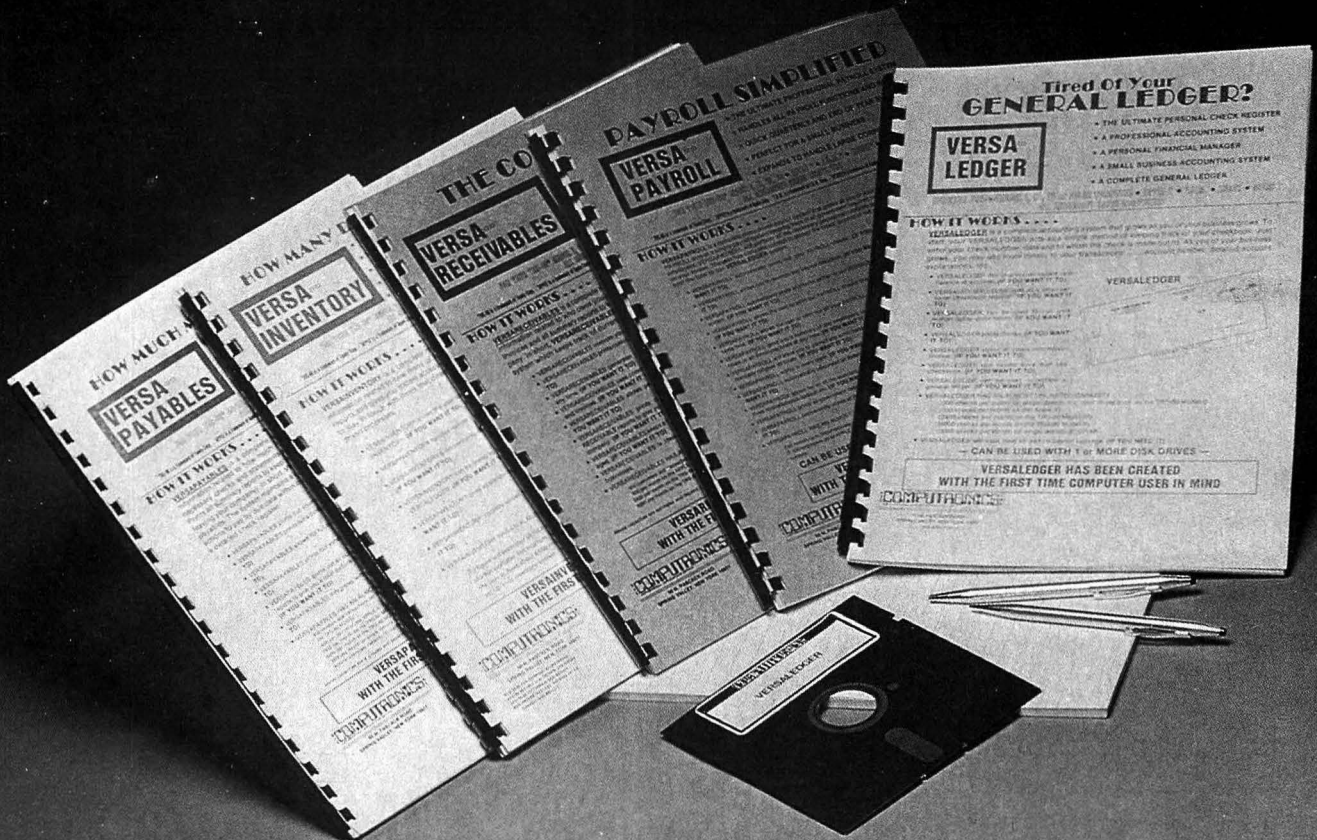
NAME

ADDRESS CITY STATE ZIP

*** ADD \$3 FOR POSTAGE & HANDLING ADD \$4 FOR C.O.D. OR NON UPS AREAS ADD \$5 CANADA & MEXICO EXACT POSTAGE ELSEWHERE ***

Introducing the Most Powerful Business Software Ever!

TRS-80™ (Model I, II, III, or 16) • APPLE™ • IBM™ • OSBORNE™ • CP/M™ • XEROX™



The VERSABUSINESS™ Series

Each VERSABUSINESS module can be purchased and used independently, or can be linked in any combination to form a complete, coordinated business system.

VERSARECEIVABLES™

\$99.95

VERSARECEIVABLES™ is a complete menu-driven accounts receivable, invoicing, and monthly statement-generating system. It keeps track of all information related to who owes you or your company money, and can provide automatic billing for past due accounts. VERSARECEIVABLES™ prints all necessary statements, invoices, and summary reports and can be linked with VERSALEDGER II™ and VERSAINVENTORY™.

VERSAPAYABLES™

\$99.95

VERSAPAYABLES™ is designed to keep track of current and aged payables, keeping you in touch with all information regarding how much money your company owes, and to whom. VERSAPAYABLES™ maintains a complete record on each vendor, prints checks, check registers, vouchers, transaction reports, aged payables reports, vendor reports, and more. With VERSAPAYABLES™, you can even let your computer automatically select which vouchers are to be paid.

VERSAPAYROLL™

\$99.95

VERSAPAYROLL™ is a powerful and sophisticated, but easy to use payroll system that keeps track of all government-required payroll information. Complete employee records are maintained, and all necessary payroll calculations are performed automatically, with totals displayed on screen for operator approval. A payroll can be run totally, automatically, or the operator can intervene to prevent a check from being printed, or to alter information on it. If desired, totals may be posted to the VERSALEDGER II™ system.

VERSAINVENTORY™

\$99.95

VERSAINVENTORY™ is a complete inventory control system that gives you instant access to data on any item. VERSAINVENTORY™ keeps track of all information related to what items are in stock, out of stock, on backorder, etc., stores sales and pricing data, alerts you when an item falls below a preset reorder point, and allows you to enter and print invoices directly or to link with the VERSARECEIVABLES™ system. VERSAINVENTORY™ prints all needed inventory listings, reports of items below reorder point, inventory value reports, period and year-to-date sales reports, price lists, inventory checklists, etc.

VERSALEDGER II™

\$149.95

VERSALEDGER II™ is a complete accounting system that grows as your business grows. VERSALEDGER II™ can be used as a simple personal checkbook register, expanded to a small business bookkeeping system or developed into a large corporate general ledger system **without any additional software.**

- VERSALEDGER II™ gives you almost unlimited storage capacity (300 to 10,000 entries per month, depending on the system),
- stores all check and general ledger information forever,
- prints tractor-feed checks,
- handles multiple checkbooks and general ledgers,
- prints 17 customized accounting reports including check registers, balance sheets, income statements, transaction reports, account listings, etc.

VERSALEDGER II™ comes with a professionally-written 160 page manual designed for first-time users. The VERSALEDGER II™ manual will help you become quickly familiar with VERSALEDGER II™, using complete sample data files supplied on diskette and more than 50 pages of sample printouts.

SATISFACTION GUARANTEED!

Every VERSABUSINESS™ module is guaranteed to outperform all other competitive systems, and at a fraction of their cost. If you are not satisfied with any VERSABUSINESS™ module, you may return it within 30 days for a refund. Manuals for any VERSABUSINESS™ module may be purchased for \$25 each, credited toward a later purchase of that module.

To Order:

Write or call Toll-free (800) 431-2818
(N.Y.S. residents call 914-425-1535)

- * add \$3 for shipping in UPS areas
- * add \$4 for C.O.D. or non-UPS areas

- * add \$5 to CANADA or MEXICO
- * add proper postage elsewhere

DEALER INQUIRIES WELCOME

All prices and specifications subject to change / Delivery subject to availability.



COMPUTRONICS

50 N. PASCACK ROAD, SPRING VALLEY, N.Y. 10977

* TRS-80 is a trademark of the Radio Shack Division of Tandy Corp. * APPLE is a trademark of Apple Corp. * IBM is a trademark of IBM Corp. * OSBORNE is a trademark of Osborne Corp. * CP/M is a trademark of Digital Research. * XEROX is a trademark of Xerox Corp.



THE MARKET PAC From COMPUTRONICS INC.

STOCK MARKET ANALYSIS PACKAGE

For TRS-80* Models I, II & III, IBM PC*, Apple* II & III, & CP/M* Computers

*TRS-80 is a trademark of Tandy Corp. - *IBM PC is a trademark of IBM Corp. - *Apple is a trademark of Apple Corp. - *CP/M is a trademark of Digital Research

Portfolio Valuation • Trend Analysis • Bond Calculations Money Market Analysis • Portfolio Bookkeeping • Future Projections

This collection of programs aids both financial professionals and individuals in the evaluation, selection, and management of investment portfolios. It features: coverage of stocks, bonds, convertible securities, options, warrants and annuities; realistic treatment of taxes and commissions; and portfolio selection methods. The clearly written user's manual makes it easy to quickly learn how to use all of the programs in the package, even if you've never used a computer before. With the STOCK MARKET ANALYSIS PACKAGE, you'll soon find that your microcomputer is an indispensable tool, performing all of these instant calculations:

- Annuity Analysis
- Computation of Alpha and Beta Values for Security
- Option Valuation and Hedge Ratio using the Black-Scholes Method
- Bond Valuation — Yield to Maturity & Other Values
- Future Net Worth and Present Value of Projected Investment Schedule
- Compound Interest Computations
- Estimate of Future Earnings Per Share
- Date Computations (Number of Days Between Any Two Dates)
- Option Writing Computation
- Portfolio Listings with Various Profit/Loss Analyses
- Portfolio Selection by Sharpe's Method
- Rate of Return — Variable Inflow
- Valuation of a Share of Stock
- Value of a Warrant
- Investor's Rate of Return on a Convertible Bond
- Dilution Analysis
- Arbitrage Computations
- Future Price Estimation with Inflation
- Seasonal Quantity Indices
- Financial Ratios
- Merger Analysis
- Value of a Right
- Depreciation vs. Cash Flow
- Time Needed for Money to Double, Triple, etc.
- Time Series Analysis — Linear Trend
- Time Series Analysis — Moving Average Trend
- Brokerage Commissions
- Margin Account Computations
- Advanced Option Strategies
- Money Market Computations
- Forecasting Cash Flows
- Leverage Analysis

\$99.95

*** ALL PRICES AND SPECIFICATIONS SUBJECT TO CHANGE ***
DELIVERY SUBJECT TO AVAILABILITY

DEALER INQUIRIES WELCOME



50 N. PASCACK ROAD
SPRING VALLEY, NEW YORK 10977

NEW TOLL-FREE
ORDER LINE
(OUTSIDE OF N.Y. STATE)
(800) 431-2818



24 HOUR
ORDER
LINE

(914) 425-1535



30 DAY MONEY-BACK GUARANTEE

- All orders processed within 24 hours
- 30-day money back guarantee
- Add \$3.00 for shipping in UPS areas
- Add \$4.00 for C.O.D. or Non-UPS areas
- Add \$5.00 to Canada or Mexico
- Add exact postage to all other countries

FREE business software directory

- Radio Shack's Model I, II, III.
- Heath's MBASIC and HDOS
- CPM: Xerox, Alto...
- IBM Personal Computer

"IDM2 is GREAT!" - publisher of 80-US

"(GL) superior to either the Osborne (SBSG & Taranto) or Radio Shack... MAIL-X has a greater capacity... more flexible than (R.S.)"
- columnist of 80-microcomputing

"imperceptively fast...(DBMS) is a good and reliable workhorse"
- publisher of Interface Age

Data base manager, integrated accounting package (AR, AP, GL & Payroll), inventory, word processing, and mailing list. Compare and be selective!



Micro Architect, Inc.
96 Dothan St., Arlington, MA 02174
(617)643-4713

ADVERTISING DIRECTORY

- 4 ABC Data Products..... 800-854-1555
- 7 Anitek Software Products..... 305-259-9397
- 17 Bealin Corp. 301-490-2744
- Cover 2 The Business Division 305-830-8194
- 61 Computer Plus 1-800-343-8124
- 6 Computer Services of Danbury . 203-743-1299
- 41 Computer Shopper..... 1-800-327-9920
- 61 Data Systems 305-788-2145
- 8 EAP Co..... 817-498-4242
- 59 Federal Energy Systems, Inc. . 1-800-323-6556
- 63-67 H & E Computronics 1-800-431-2818
- 4 J. E. S. Graphics 918-742-7104
- 33 JKR Engineering..... 408-263-7139
- Cover 4 Leading Edge Products, Inc... 1-800-343-6833
- 4 Mayday Software
- 68 Micro Architect 617-643-4713
- 9 Micro Images 212-445-7124
- 29 Micro-Labs, Inc..... 214-235-0915
- 3 Micro Systems Software 1-800-327-8724
- 8 Nodvill Software 203-431-6449
- 68 Powersoft..... 1-800-527-7432
- 12 S. D. C. & S. Co., Inc. 212-849-8600
- 12 Solutions, Inc.
- 6 Triple-D Software 801-546-2833
- 8 Virginia Micro Systems 703-491-6502

HARDWARE REVIEW

continued from page 61

TRS-80™ OWNERS... Enter the World of

POWERSOFT



Nationally Acclaimed Programs for the TRS-80 Computers

Now from the authors of **SUPER UTILITY+** a complete line of software to increase the use and capabilities of your system.

While **SUPER UTILITY+** won recent honors as Utility Program of the year by the 200,000 readers of *80 Micro*, the quality and consistency carries through the complete Powersoft line.

Professionally written and completely documented, Powersoft programs are accepted as industry standards among TRS-80 enthusiasts.

SUPER UTILITY PLUS	74.95	PowerMAIL+.....	150.00
SCRIPUS	39.95	PowerDRAW	39.95
TOOLBOX FOR LDOS	69.95	PowerDOT	49.95
MASTER MECHANIC SET FOR LDOS	39.95	PowerDRIVER.....	29.95
BASIC/S COMPILER SYSTEM	49.95	PowerTERM	29.95
MAKE/80	49.95	INSIDE SU+	19.95
SUPERMOVE XFER SYSTEM	700.00	SU+ TECH MANUAL	14.95
DOSPLUS II	250.00	SU+ SPECIAL EDITION	500.00

BOOT UP WITH POWERSOFT...
THE WORLD'S MOST POWERFUL SOFTWARE VENDOR!

WE SUPPORT LDOS! BUY ANY PRODUCT AND
GET LDOS FOR \$99!

SEND FOR OUR COMPLETE CATALOG TODAY!
DEALER INQUIRIES INVITED

AVAILABLE THROUGH SELECTED DEALERS EVERYWHERE

POWERSOFT

11500 Stemmons Fwy.
Suite 125

Dallas, Texas 75229

Info: (214) 484-2976

Orders Only 800-527-7432

PRODUCTS FROM BREEZE/QSD, INC.

The fact that I never have had any of the problems that I've read about the Radio Shack interface make all of the work involved in building the board worth the effort. The most difficulty in locating parts for the board were the resistor arrays and header.

Over all, I may have spent more for this kit than I would have had to pay for a new Radio Shack interface, probably arising the fact that four or five parts suppliers had to be used.

Unexplained to me, my disk system ran flawlessly for over a year, then I/O errors began to creep into the picture. I purchased an alignment diskette to align the drive, yet the problem still persisted. The installation of a Percom Data Separator cured the problem. Its installation, though, required that three parts had to be moved to the solder side of the board. I believe MICRO DESIGN now sells a separator, maybe a good investment from the start. I plan to go to double density soon, but it looks as if the heat sink from the five volt regulator will prevent this. Hopefully MICRO DESIGN or some other manufacturer will have come to my aid before I am forced to modify my board again.

Perfect or not, I recommend the MDX-2. Novices would probably prefer the assembled version. I prefer kits and the fun you have in getting the thing to work.

K. I. Brown
2320 Gaylord Drive
Washington, DC 20746 ■

Put 64K CP/M® 2.2 in your TRS-80 Model III and tap into 2,000 business programs.

Now you can run programs such as WordStar, dBASE II, SuperCalc, MailMerge and virtually thousands of other CP/M-based programs on your TRS-80 Model III.

CP/M 2.2 is the industry standard operating system that gives you access right now to over 2,000 off-the-shelf business programs.

Our plug-in Shuffleboard III comes with 16K of RAM, giving your Model III the power of full 64K CP/M 2.2 without interference of the ROM or video memory. In fact, the Shuffleboard will appear transparent in the TRS-80 mode and will not interfere with any DOS operation.

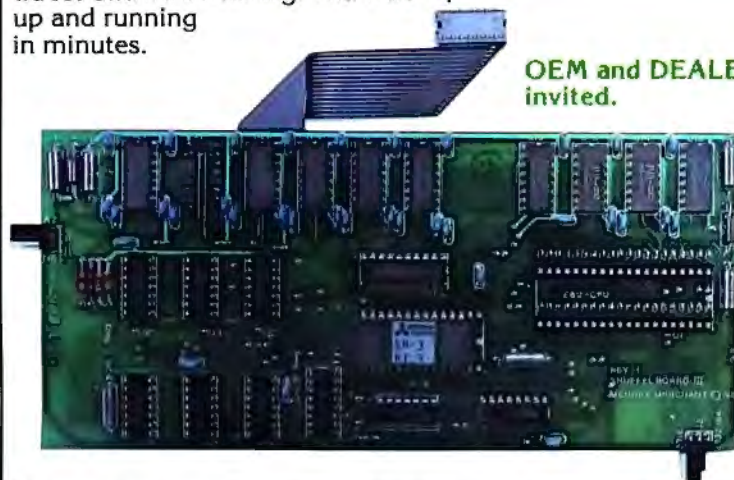
READ and WRITE Osborne, Xerox and IBM personal computer software plus many more popular formats.

Unfortunately, there is no standardized CP/M format for 5 1/4" diskettes. But we have developed a way to READ/WRITE and RUN standard programs under the following single-sided formats: Osborne 1 S/D, Xerox 820 S/D, IBM PC* D/D for CP/M 86 only, Superbrain D/D, Kapro II D/D, HP 125 D/D and TeleVideo D/D.

*Will Read and Write Only.

Easy plug-in installation.

It's so simple. The Shuffleboard III plugs into two existing sockets inside your Model III. There are no permanent modifications, no cut traces and no soldering. You'll be up and running in minutes.



OEM and DEALER inquiries invited.



New Products.

80 x 24 VIDEO BOARD: Features dual intensity screen, programmable cursor control for block, underline & blink rate, on-board bell with audible keyclick, battery-operated real time calendar/clock, full ASCII character set plus 256 special character graphics, dual RS-232 outputs and composite video output.

FLOPPY DISK CONTROLLER: Now you can access 5 1/4" and 8" floppy disk drives in any combination up to 4 drives of S/D density, S/D sided. Tap into a wealth of CP/M software which comes on 8" IBM 3740 format or Pickles & Trout CP/M for the Model II.

SOFTWARE: Additional CP/M software programs are available. Call or write for details.

Introductory price of

\$299.

The Shuffleboard III comes fully burned-in and tested complete with 64K CP/M 2.2 and MBASIC 80 interpreter, plus software manuals and a first class user's manual — with a 1-year limited warranty and 15-day no-risk free trial — for only \$299.

See the Shuffleboard III at your dealer's now.

Once you see what the Shuffleboard can do for your Model III you'll want one at once. If your dealer does not yet stock the Shuffleboard have him give us a call. Or send check, money order, VISA or MASTERCARD number (sorry, no COD's) plus \$5 shipping per board (\$17 outside the USA & Canada)* directly to the address below. Cal. residents please add sales tax. Credit card purchases can be phoned in directly and we'll ship from stock.

(415) 483-1008

*Air mail shipments to Canada & all other countries.

Memory™ Merchant

14666 Doolittle Drive San Leandro, CA 94577
(415) 483-1008

**BUY A BANANA.
SAVE A BUNCH.
MORE TO COME.**



Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021.
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.