NUMBER 11

$2.50

ANALOG COMPUTING™

THE MAGAZINE FOR ATARI® COMPUTER OWNERS

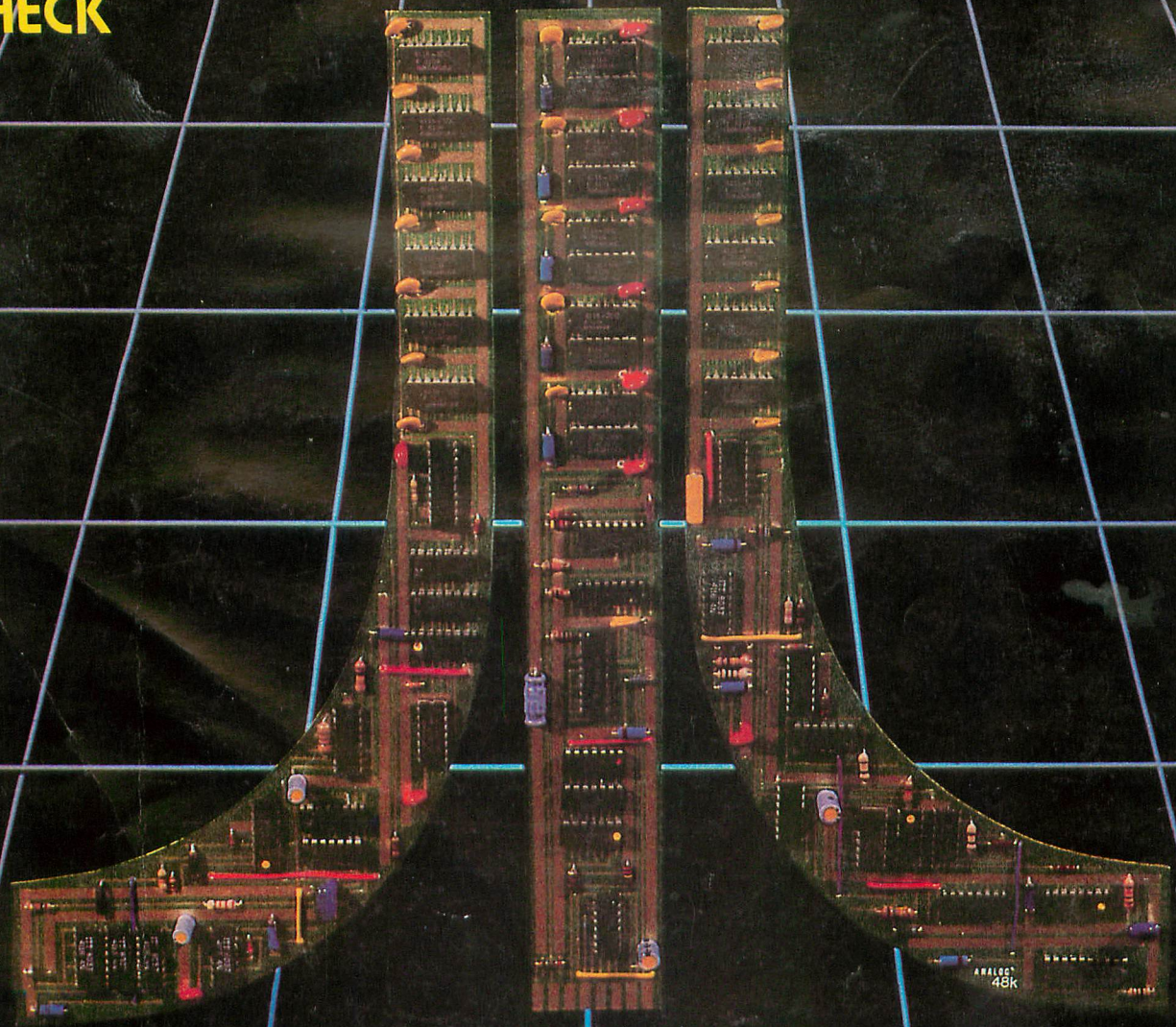
ADVENTURE IN THE 5th DIMENSION!

BACKUP PROGRAMS REVIEWED

MOVING MISSILES IN BASIC

COPYRIGHT BASICS

C:CHECK



0 74470 12385 11

MOSAIC™

64K RAM SELECT™

The only Compatible 64K RAM for

ATARI® 400 and 800

The Mosaic 64K RAM SELECT* will make your Atari† computer more powerful than ever before. No other memory board equals the power, dependability, flexibility, compatibility, documentation and guarantee of the Mosaic 64K RAM SELECT.

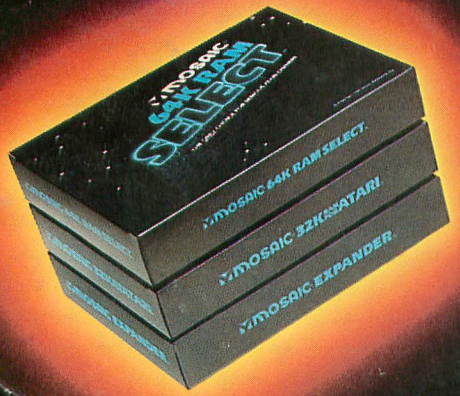
This Mosaic design takes full advantage of Atari's internal architecture. The result, for example, means in comparison with any other 48K or 64K RAM system you'll have 30% more workspace with the Atari word processor and 17% more workspace with Visicalc — a true advantage for any disc drive system. For program writers it means 16K RAM of special risk-free storage. No more "page 6" dangers. It's impossible with the Select System for your routines to be violated by the DOS, OS, BASIC or any other existing software. It's safety you can bank on!

Now you'll have 48K RAM hard wired with 4 banks of 4K RAM waiting, addressed above the normal RAM limit. This gives you 52K continuous RAM and 64K RAM total . . . plus complete compatibility with all Atari software and peripherals.

Each Mosaic 64K RAM SELECT comes with step by step picture guided instructions. The best in the industry.

FEATURES:

- 4 year guarantee.
- Complete instructions.
- Easy, no solder installation.
- Complete compatibility with all Atari† products.
- Test cycled 24 hours for reliability.
- Gold edge connectors for better reliability.
- Designed to take advantage of Atari computer's superior architecture.
- Designed for inter-board communication in Atari 800.†
- Always the best components used for superior screen clarity and reliability.
- Low power design for safety and reliability.



CALL FOR YOUR NEAREST MOSAIC DEALER 1-800-547-2807.

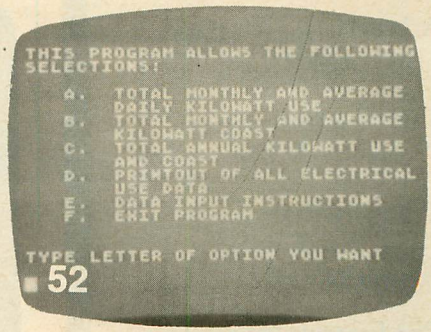
MOSAIC™ ELECTRONICS, INC.

*Trademark of MOSAIC ELECTRONICS, INC.
†Atari is a registered trademark of Atari, Inc. Mosaic is not affiliated with Atari.

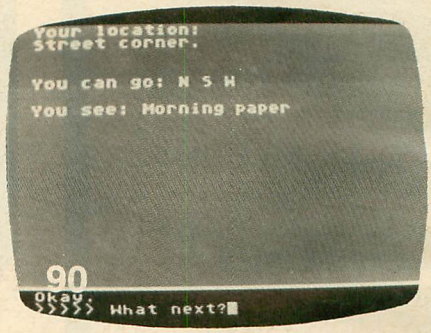
ANALOG COMPUTING™
ISSUE NO. 11 APRIL/MAY

FEATURES/ARTICLES

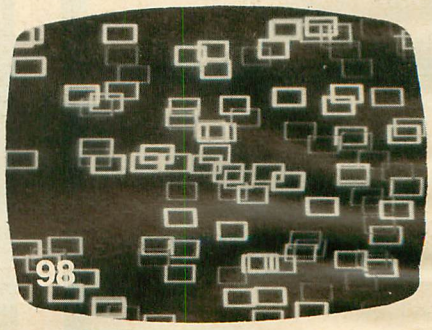
REVIEWS/PROFILES



PROGRAM LISTINGS



COLUMNS



A.N.A.L.O.G. STAFF

Editors/Publishers
MICHAEL DESCHENES
LEE H. PAPPAS

Managing Editor
JON A. BELL

Contributing Editors
TONY MESSINA
DR. GARY ROSE
MARCIA ROSE

Art Director
BOB DESI

Technical Division
CHARLES BACHAND
TOM HUDSON
BRIAN MORIARTY

Advertising Manager
MICHAEL DESCHENES

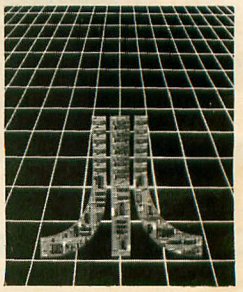
Distribution
PATRICK J. KELLEY

Typography
HENDRICKX &
LARRIVEE Co., Inc.

Production/Distribution
LORELL PRESS, INC.

Contributors
THOMAS RAY HAMEL
JOSEPH HARB
THOMAS KRISCHAN
DICK KUSHNER
RICHARD G. LYONS
JESSIE STEPHENS
DICK TEDESCHI
RAYMOND T. TILLMAN
BERT WILLIAMS

ABOUT THE COVER



The cover to this issue was produced as follows: the ATARI circuit board logo was constructed as a scratch-built model, using sheet styrene, old memory board chips, resistors and wire, and was painted green with gold traces. It was then mounted over a sheet of plexiglas on which a grid pattern had been painted. The plexiglass was lit from behind; the ATARI logo was front-lit; and here you have it — the cover to issue no. 11.

SUBSCRIPTIONS

All subscription orders and other correspondence related to subscriptions (except for Tape and Disk subscriptions), should be addressed to: **A.N.A.L.O.G. COMPUTING**
P.O. Box 615
Holmes, PA 19043

Foreign subscriptions must be made in U.S. currency.

Subscription prices:	U.S.	Canada	Foreign
1 yr .	$14.00	$18.00	$24.00
2 yrs.	$27.00	$31.00	$37.00
3 yrs.	$38.00	$42.00	$48.00

Airmail delivery on foreign subscriptions is available for an additional $36.00 per year.

Please allow six to eight weeks for subscriptions to begin. Allow eight weeks for change of address.

A.N.A.L.O.G. COMPUTING (ISSN 0744-9917) is published bi-monthly for $14 per year by **A.N.A.L.O.G.** Magazine Corp., 565 Main Street, Cherry Valley, MA 01611, Tel. (617) 892-3488. Second-class postage paid at Worcester, MA and additional mailing offices. POSTMASTER: Send address changes to **A.N.A.L.O.G. COMPUTING**, P.O. Box 615, Holmes, PA 19043. Program listings should be provided in printed form. Articles should be furnished as typed copy in upper and lower case with double spacing. By submitting articles to **A.N.A.L.O.G. COMPUTING**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **A.N.A.L.O.G.** If not accepted for publication, the articles and/or programs will remain the property of the author. If submissions are to be returned, please supply self-addressed, stamped envelope. U.S.A. Newstand distribution by Eastern News Distributors, Inc., 111 Eighth Ave., New York, NY 10011.

80 COLUMN COLOR VIDEO BOARD FOR THE ATARI 800™

IF YOU USE YOUR ATARI 800™ FOR:

- PROGRAMMING
- TERMINAL COMMUNICATIONS
- WORD PROCESSING
- DATA BASES

THE AUSTIN 80™ COLOR VIDEO BOARD GIVES YOU THE OPTIMUM FEATURES...

—80 * 25 Character Screen
—256 Character set with line and box graphics
—Underline, blink, reverse, half intensity
—Smooth scrolling
—16 color RGBI or RGB outputs + composite video
—Uses medium resolution B&W or color monitor
—Fits in last slot of 800
—Light pen input

LIST PRICE $279.95

SOFTWARE TO MAKE OPTIMUM USE OF THE AUSTIN 80™

SMART TERMINAL — EMMULATES DEC VT100™ — $19.95

WORD PROCESSOR with CONFIGURABLE PRINTERS
SUMMER '83

OPTIONAL RGBI OUTPUT CABLE — $34.95

AUSTIN 80™ CARRIES A 1-YEAR PARTS AND LABOR WARRANTY

Ask for AUSTIN BOARDS™ at your local dealer or call

617-772-0352

AUSTIN FRANKLIN ASSOCIATES

43 GROVE STREET, AYER, MA 01432

M/C, VISA & COD ACCEPTED

DEALER INQUIRIES INVITED

*ATARI IS A REGISTERED TRADEMARK OF ATARI, INC.

EDITORIAL

by Jon Bell

In the first weeks prior to last Christmas, I had the opportunity to discuss ATARI computer sales with a local computer store owner. His business was booming. There was a mad rush for joysticks, games, educational software and computers. I watched a salesman make several pitches for both the 400 and the 800. His customers were usually young couples, shopping for their first home computer. Many of these couples wanted their children to learn how to use a computer and to help prepare them for the world of the 1990s, when it is estimated that one out of every four American households will have personal computers and computer literacy will be regarded as a job necessity. However, most of these couples did not have a lot of money to spend, and a 1982 Christmas package consisting of an ATARI 800, 810 disk drive and software cost, at that time, over a thousand dollars. Therefore, the couples were interested in a cheaper system, one that could expand as their children — and assets — grew. They looked at the 400. The salesman made his pitches for it. I heard this response:

"— but this can't be expanded to more memory, can it?"

I shook my head in dismay as the salesman explained to several people that that belief was a common fallacy. I asked the store owner if this was a typical question. He told me that he had heard over a dozen people ask that same question in the past two weeks.

Nowhere in ATARI's literature or advertising for the 400 does ATARI mention that the 16K 400 can be upgraded to 48K. Consequently, in comparison ads pitting the ATARI 400 against such other small home computers as the TI-99/4A, the VIC 20 or the Commodore 64, such a personal computer authority as Bill Cosby can get away with implying that the 400 is limited to only 16K. Whenever I see that ad, I get annoyed. Not at Bill Cosby, certainly, or even Texas Instruments, the makers of the commercial, but at ATARI for their neglect of third-party hardware and software. They are hurting themselves with their "ATARI is an island" attitude.

A company should work with its customers. By extension, people who manufacture goods used in conjunction with your company's products are your customers too, and should be treated with respect. They are helping to promote your products. It is not wrong for others to make money off of your success if you do not have the ambition or the ability to "*fill* in the gaps" present in your products. The only way to stop these other companies would be to discontinue your own product — and if that isn't killing the goose that laid the golden egg, I don't know what is. I consider ATARI's inexplicable "isolationism" to be quite harmful, and ultimately alienating to many people. □

WINNERS!

In issue number 9 of **A.N.A.L.O.G. Computing** we inserted a readers poll card. As an incentive to send in the card, we did a random drawing and picked three people. They will be receiving a free disk or cassette subscription. *Here are* the winners:

STEPHEN W. YIP
Concord, CA

ALISTAIRE B. CALLENDER
Norman, OK

JAN BANICKI
South Bend, IN

A.N.A.L.O.G. would like to thank everyone who took the time to send in their card.

ISSUE #10 CONTAINED 1 ERROR. HERE IS THE CORRECTION:

D:CHECK2 (PG. 26)

```
130 CLOSE #1:? "ENTER FILENAME";:INPUT
 FI$
```

ComputAbility™ presents the Newest in Software Products for ATARI

ATARI 800 48K C
ATARI 810 DISK DRIVE..... C
ATARI 850 INTERFACE..... A
ATARI 410 PROGRAM..... A
ATARI 1010 PROGRAM RECORDER..... L
EDUCATOR KIT..... L
BOOKKEEPER KIT L
PROGRAMMER KIT .....

ENTERTAINER KIT C
COMMUNICATOR KIT..... A
HOME MANAGER'S KIT L
EPSON FX-80 L

MOSAIC
64K RAM..... 149.00
32K RAM 87.95

48K RAM ..... CALL
16/32 EXPANDER..... 74.95

B-KEY 400..... 84.95
SUNCOM JOYSENSOR 34.95
Touch Sensitive Joystick

We Carry The Complete PERCOM Line.
Call For Items and Prices.

Call For Our Special EPSON Printer Package

SUPER SPECIALS

Prices effective May 1 through May 31, 1983

Centipede	Galaxian	Missle Command	Paint	Protector II
$29.95	$29.95	$24.95	$29.95	$22.95
Cartridge	Cartridge	Cartridge	Disk	Disk/Tape

FREE* SOFTWARE AND ACCESSORIES

*** COUPON PROGRAM**

The purchase of each program (with the exception of Super Specials, Atari, and APX) will earn you 1 COMPUTABILITY DIVIDEND COUPON (CDC). Save 10 coupons and redeem them for your choice of any program we sell for $24.00 or less. Less than 10 coupons may be redeemed for premium items as indicated. You pay only a $2.50 shipping & handling charge.

ATARI

Conversational Languages - T ... 43.95
Programming 2 & 3 - T 21.95
Music Composer - C 32.95
My First Alphabet - D 26.95
Touch Typing - T 19.95
Home Filing Manager - D 37.95
Mailing List - T 19.95
Asteroids - C 26.95
Caverns Of Mars - D 28.95
Computer Chess - C 26.95
Missle Command - C 26.95
Super Breakout - C 26.95
Star Raiders - C 32.95
Assembly Editor - C 44.95
Basic - C 44.95
Macroassembler - D 65.95
Microsoft Basic - D 65.95
Pilot (Home Package) - C 58.95
Invitation To Programming I - T ... 18.95
Speed Reading - T 55.95
Basketball - C 26.95
Graph-It - T 15.95
Juggles House - D/T 22.95
Pilot (Educator) - C 97.95
Video Easel - C 26.95
Defender - C 32.95
Galaxian - C 32.95
Qix - C 32.95
Dig Dug - C 32.95
ET - C 38.95
Timewise - D 23.95
Atariwriter - C 61.95

ADVENTURE INTERNATIONAL

Preppie - D/T 23.95
S.A.G.A Adventures - D 31.95
Sea Dragon - D/T 27.95
Stratos - D/T 27.95
Bugoff - D/T 23.95
Preppie II - D/C 27.95
Stone of Sisyphus - D 27.95
Eliminator - D/T 20.95

ANALOG

Race In Space - D/T 20.95
Sunday Driver - D/T 23.95
Crash Dive! - D/T 23.95
Buried Bucks - D/T 23.95
Titan - D/T 23.95
Star Sentry - D/T 23.95

DATASOFT

Shooting Arcade - D/T 23.95
Pacific Coast Highway - D/T 23.95
Micropainter - D 27.95
Canyon Climber - D/T 23.95
Fathoms Forty - D 27.95
O'Riley's Mine - D/T 27.95
Rosen's Bridgade - D/T 27.95
Sands of Egypt - D 31.95
Zaxxon - D/T 31.95
Moon Shuttle - D/T 31.95

ON-LINE

Jawbreaker - D/T 23.95
Luna Leeper - D 23.95
Ultima I - D 31.95
Threshold - D 31.95
Ultima II - D 44.95
Mouskattack - D 27.95
Frogger - D/T 27.95

COMPUTABILITY DIVIDEND COUPON PREMIUMS

3 M Blank Disk - Box of 10 ..... 27.00 or 10 CDC
Wabash Blank Disks - Box of 10 ..... 21.00 or 8 CDC
Flip 'n File Disk Holder ..... 20.95 or 6 CDC
Flip 'n File Cartridge Holder ..... 20.95 or 6 CDC
Monitor Stand ..... 27.95 or 10 CDC
VU - Case Disk Holder W/Lock ..... 24.00 or 10 CDC
Heavy Plastic Dust Cover 800 or 810 ..... 12.95 or 5 CDC
First Book of Atari Software 1983 ..... 17.95 or 6 CDC
Kids & The Atari ..... 17.95 or 6 CDC
The Atari Assembler ..... 12.95 or 5 CDC
Atari Games & Recreation ..... 14.95 or 6 CDC
Atari Pilot for Beginners ..... 14.95 or 6 CDC
The Visicalc Book Atari ..... 14.95 or 6 CDC
Atari Basic ..... 12.95 or 5 CDC
Atari Sound & Graphics ..... 9.95 or 3 CDC
Starfighter ..... 12.99 or 5 CDC
Slik Stik ..... 9.95 or 3 CDC
Extension Cable - 6 ft. ..... 6.95 or 3 CDC
Lefty Adaptor ..... 9.95 or 3 CDC
Stik Stand ..... 6.99 or 3 CDC
Suncom Tack II ..... 19.95 or 8 CDC
Suncom Game Switch ..... 6.95 or 3 CDC
TI Adaptor ..... 12.95 or 5 CDC
Kraft Switch Hitter Joystick ..... 17.95 or 7 CDC
Kraft Joystick ..... 15.95 or 6 CDC

FIRST STAR

Cosmic Squeeze - D/T 23.95
Astro Chase - D/T 23.95

BIG FIVE

Miner 2049'er - Cart 39.95

I.D.S.I.

Pool 1.5 - D 27.95
Pool 400 - Cart 31.95
Speedway Blast - Cart 31.95
Juggler - D 23.95

SIRIUS

Bandits - D 27.95
Way Out - D 31.95
Repton - D 31.95
Twerps - D 23.95
Wavy Navy - D 27.95
Blade of Blackpoole - D 31.95

SENTIENT

Cyborg - D 27.95
Goldrush - D/T 27.95

BRODERBUND

Apple Panic - D/T 23.95
Stellar Shuttle - D/T 23.95
David's Midnight Magic - D 27.95
Star Blazer - D 25.50
Track Attack - D 23.95
Labyrinth - D/T 23.95
Serpentine - D 27.95
Dueling Digits - D 23.95
Choplifter - D 27.95
Genetic Drift - D/T 23.95
AE - D 27.95

EDU-FUN

Call For Items and Prices

MISCELLANEOUS

Night Mission - D/T 23.95
Jumpman - D 31.95
Airstrike - D 31.95
Castle Wolfenstein - D 23.95
Financial Wizard - D 49.95
Pinball Construction Kit - D 31.95
Pig Pen - D 23.95
Baja Buggies - D/T 25.50
Starbowl Football - D/T 25.50
Master Type - D 31.95
Ali Baba - D 26.50
Jeepers Creepers - D 23.95
Paint - D 33.95
Kid Grid - D/T 23.95

D - Disk T - Cassette
C - Cartridge

We Carry Hundreds of Items for ATARI 400/800, Ask for Our FREE CATALOG.

ATARI is a trademark of ATARI, Inc.

Mastercard/VISA Order Toll Free MasterCard **800-558-0003** VISA **In Wisc. Call 414/351-2007**

No surcharge for credit cards

ORDERING INFORMATION: To order by main, send money order, certified check, or personal check (allow 14 days to clear) to COMPUTABILITY. Include $2.00 shipping on all software orders. Include 3% shipping on all hardware orders, minimum $2.50. Mastercard and Visa please include card number and expiration date. WI residents please add 5% sales tax. Canadian, APO & FPO software orders include 5% shipping, minimum $5.00. All other foreign software, please add 15% shipping, minimum $7.00 (US Funds only). Prices subject ot change without notice.

COMPUTABILITY
P.O. Box 17882
Milwaukee, WI 53217

READER COMMENT

Dear Editor,

In your recent A.N.A.L.O.G. Computing issue #9 a review of our Letter Perfect Version 3 was written by Tony Messina.

The article had many complimentary things to say about Letter Perfect and the new version. However, it did contain some inaccuracies in regard to our update policy. Anyone who has purchased and registered an earlier version of Letter Perfect may upgrade to the new version 3 by sending us their old diskette and $30. In return, they will receive a new diskette, a new manual, and a new command sheet. Any individual who has purchased a backup diskette may hold the backup diskette until they have received the new version, and then send us the backup to receive an update of their backup. There is no policy that allows an individual who returns their old version to get a free backup of the new version.

Sincerely yours,
Kenneth L. Berry
LJK Enterprises, Inc.
St. Louis, MO

Dear Sirs:

We just bought an ATARI 800 computer and were glad to find there was a magazine for ATARI owners. But we sat for hours programming your programs *An Adventure Game, The Halls of the Leprechaun King, and Fill'er*, not one of the programs work. I have a friend that has a computer she bought before we did and she said they had the same problem. We have since talked to some others and they all say the same thing. It is very maddening to sit for four or five hours putting programs into the computer and not having it work. We were going to subscribe to your magazine but not now. We feel that you should check your programs first before you put them in your magazine. We have checked and checked them and we did not make an error in typing them into the computer. If in the future we hear that your programs work we will consider buying your magazine, in the mean time we will not reccommend your magazine to anyone.

Mr. Randall Beemer
Flint, Michigan

Dear **A.N.A.L.O.G.:**

This letter is sent to thank Tom Hudson for giving us what I feel is the best game ever to appear in the pages of a magazine. It tops a lot of games I've seen on sale, too! I am speaking, of course, about *Fill'er Up!* (Issue #10).

I recently moved from a large city to a small resort town. Unlike the city there are no software dealers here, so I rely **very** strongly upon the printed page. (I had to subscribe to even get that here!) So I really appreciate when an exceptional program such as *Fill'er Up* comes along. Needless to say I am eagerly awaiting Mr. Hudson's next contribution to your fine publication, which by the way gets better with each issue.

Very gratefully yours,
Rudy C. Houghton
Myrtle Beach, SC

Dear Editor:

I am glad to hear that the results of your readers poll generally agreed with my ideas and opinions. **A.N.A.L.O.G.** is the best magazine on the market that deals with ATARI computers in whole or in part, and it keeps getting better. I like the assembly-language games, and really enjoyed *Fill'er Up!*. However, I found it very boring to sit through the beginning of the program waiting for the computer to check the lines every time I wanted to play the game. I tried making a cassette-maker program (ala Maniac or Harvey Wallbanger) to no avail, — as I don't know very much about assembly programming. However, the following changes will speed up the wait from 2 1/2 minutes to 1 1/2 minutes. First type in the program as in **A.N.A.L.O.G.** #10 (to check for errors), then, when correct, make the following changes.

```
2 GRAPHICS 18:POSITION 3,5:? #6;"PLEAS
E WAIT...":FOR DE=1 TO 50:NEXT DE
5 Y=PEEK(559):POKE 559,0
25 LINE=LINE+10:READ DAT$
28 REM (delete this line)
40 PROG$(PNTR)=CHR$(BYTE):PNTR=PNTR+1:
TOTAL=TOTAL+BYTE:NEXT X:READ CHKSUM:IF
 LINE=1710 THEN 70
50 GOTO 25
70 POKE 559,Y:? "PRESS START TO BEGIN"
80 IF PEEK(53279)<>6 THEN 80
85 A=USR(ADR(PROG$),ADR(PROG$)):END
```

I hope the readers find this useful (unless someone has figured a way to make a cassette-maker program.) I really enjoyed *Fill'er Up!*, but it seems to me that once a game is over, the next game should start back on level 1, not on the level where the last game left off. This can be frustrating, especially since pressing RESET results in "ATARI lock-up." Other than that, thanks, and keep **A.N.A.L.O.G.** coming!

Sincerely yours,
Jim Singer
New Castle, Penn.

Tom Hudson:

I loved your P/M mover subroutine in issue 10 of **A.N.A.L.O.G.** How can it be changed to include MISSILES as well as PLAYERS?

Jeff Stefanski
Melrose Park, IL

Your wish is our command. See page 29. — Ed.

Dear **A.N.A.L.O.G.**:

Issue number 10 is fantastic!! Your article on the ATARI *1200XL* is the only full length discription I have seen so far. I had heard some rumors about it, but nothing conclusive. I have an ATARI 800 and don't plan on getting a 1200XL in the near future, but I was excited to find out about the 800's big brother!

My friends and I entered *Fill' Er Up!* several days after getting the magazine. The time was worth it! It's the best game I've ever gotten for free. Tom Hudson, **A.N.A.L.O.G.** and ATARI are a great team!! There was only one thing I didn't like about it and that was the fact that you couldn't restart it by pressing the trigger. This is a feature any game that does not use the keyboard during the actual play should have. So after looking at the listing, I thought how nice it would be if I could change it all by myself...Being a beginning assembler programmger, that's just what I set out to do! I succeeded too! I'm including the needed changes.

```
Old Code:

CKSTRT LDA CONSOL
       AND #1
       BNE CKSTRT
RELEAS LDA CONSOL
       AND #1
       BEQ RELEAS


New Code:

;fill extra space

       NOP
       NOP
       NOP
       NOP

;wait for release

RELEAS LDA STRIG
       BEQ RELEAS

;trigger pressed?

CKSTRG LDA STRIG
       BNE CKSTRG
```

By adding two lines to the basic program the new code can be installed.

```
5 RESTORE 10:DATA 234,234,
234,234,173,132,2,240,251,
173,132,2,208,251
65 RESTORE:FOR X=1225 TO 1
238:READ N:PROG$(X,X)=CHR$
(N):NEXT X
```

Now you can have the freedom of sitting anywhere your joystick cord will reach AND restart the game without getting up!

I really enjoy reading **A.N.A.L.O.G.**. It's a great magazine! I have every issue! The first issue I got on my subscription was number 5, but when the reprints of the back issues came out I ordered numbers one through 4. And I'm glad I did, since they have been very helpful in getting the most out of my ATARI **and** because they are no longer available!!

There is one addition I would like to see in **A.N.A.L.O.G.** and that's a book review section. There are a lot of books out for ATARI's and we need help in picking the good ones from the bad.

This is the very first letter I have ever written to a magazine. And I'm glad to have a magazine I care enough about to take the time to write to it (would THEM have been a better choice, since **A.N.A.L.O.G.** *IS* made up of people . . . Well gotta sign off now.

Well gotta sign off now,
Chris Gruenler
S. Hamilton, MA

We were glad you enjoyed our article on the 1200XL. The response from our readers was overwhelming. We were very fortunate to have the 1200XL in our offices months before it was available to other publications. Look for more information on the 1200XL in future issues — and remember, when you want the "scoop" on new ATARI products, you'll see it first in the pages of A.N.A.L.O.G. Computing.

Many people have been requesting back issues of A.N.A.L.O.G. Computing. I'm sorry to report that we have no remaining copies of any back issues. However, the forthcoming A.N.A.L.O.G. COMPENDIUM (see Editorial, issue 10) will contain the best games, utilities and educational programs from the first ten issues of A.N.A.L.O.G. Computing. The A.N.A.L.O.G. COMPENDIUM will also feature programs not published in earlier issues of A.N.A.L.O.G. Computing. It will be available to computer stores and bookstores the first week in July.

Speaking of books, we are planning to include an up-to-date list and capsule review of all ATARI-related books in our next issue. —ED

Attention Programmers!

A.N.A.L.O.G. Computing is interested in programs, articles, and software review submissions dealing with the ATARI 400 and 800 home computers. If you feel that you can write as well as you can program, then submit those articles and reviews that have been floating around in your head, awaiting publication. This is your opportunity to share your knowledge with the growing family of ATARI computer owners. **A.N.A.L.O.G.** pays between $30.00-$360.00 for all articles. All submissions for publication must be typed, upper and lower case with double spacing. Program listings should be provided in printed form, and on cassette or disk. By submitting articles to **A.N.A.L.O.G. Computing**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **A.N.A.L.O.G.** If not accepted for publication, the articles and/or programs will remain the property of the author. If submissions are to be returned, please supply a self-addressed, stamped envelope. All submissions of any kind must be accompanied by the author's full address and telephone number. Send programs to: Editor, **A.N.A.L.O.G. Computing**, P.O. Box 23, Worcester, MA 01603.

Gentlemen:

I have been an avid reader of your magazine since the first issue. It has been interesting, informative, and amusing.

As of now, "Whither ATARI" is obvious. ATARI "whithered" to Taiwan. In these days of high labor costs, it became necessary to move production facilities to a location with lower labor costs. We all know that labor is the single most expensive component in the manufacturing process, right? WRONG! 50 days of network advertising prior to Christmas was the single most expensive component!

ATARI (read Warner Communications) was concerned with how to cut costs, that was obvious. However, did they stop to consider how many of the 1700 production workers would, if employed, but their computers? What about the people who do business with those 1700 former employees? How many ATARI computers will they be able to buy now? This kind of action has a "ripple effect" throughout the entire community. But it's still cheaper to build overseas, and you end up with a better product because the American worker just doesn't care to work as hard to do a good job. That's why SONY builds all their Trinitron color TVs for the U.S. in the U.S. It's also why HONDA builds all of the cars and motorcycles (above 900cc) destined for the States over in Ohio. SONY and HONDA seem to know what American manufacturers have forgotten — we can build it here cheaper and better if we put our minds to it. It makes me wonder whether anyone in marketing or manufacturing at Warner or ATARI has a mind left to work with.

Keep up the good work at **A.N.A.L.O.G.** — you're the best magazine around.

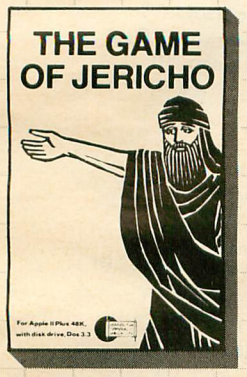
Sincerely,
Richard B. Roche

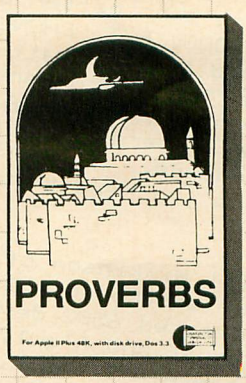
In Issue No. 10 of **A.N.A.L.O.G.** Computing, we published an article by Greg Peck entitled "Magic Keypad." This article was originally intended for publication in ANTIC Magazine, and is the sole property of ANTIC. We regret our error and apologize for any inconvenience.

ANTIC is a magazine published for ATARI computer owners. For information, write to ANTIC, 600 18th Street, San Francisco, CA 94107.

New family fun and a new learning experience!

JEWISH SOFTWARE FOR ATARI





Davka Corporation introduces new software programs and games designed to give you more understanding of Jewish history and traditions. Outstanding scholars have combined their teaching methods with state-of-the-art computer techniques and graphics, to bring you and your family an exciting and important new dimension in Judaica for the home. If you have an Atari 800 computer with one Disk Drive and 48K Memory, you're ready to start on a whole new adventure!

Here are a few of the many programs and games which are available for your Atari computer. Others are listed in our free catalog.

HEBREW READING This individualized program is designed to teach the reading of Hebrew to those who have no prior knowledge of the Hebrew alphabet and/or those who are just building their basic reading skills. **Number 101-I $50**

THE GAME OF JERICHO In this action-packed arcade game, the player directs the blasts from Joshua's shofar to destroy the walls of ancient Jericho. His shots must be quick and accurate to penetrate enemy lines. One must be careful, though—the enemy is constantly raining down clusters of arrows. "Jericho" combines outstanding sound and colorful graphics—but to win, one needs to be cunning, clever, and daring. "Jericho" is for children or adults. **Number 1400 $29.95**

ISRAEL GEOGRAPHY GAME As an "Israel Sherlock Holmes," the player uses historical and geographical "clues" to "deduce" the correct names of sites on the map of Israel. The game begins with a map of Israel drawn on the screen and divided into four regions. The player selects a region and an enlarged map of that section is drawn on the screen. A flashing cursor represents the "target" and the puzzle solving is set in motion. **Number 200-I $25**

PROVERBS Five hundred carefully selected Jewish Proverbs and Folk Sayings, taken from the Talmud, Medieval Hebrew Literature, and Yiddish and Hasidic sources, are presented in this program, indexed by specific categories. To view all the proverbs on any given topic, the user needs simply to type in the category he wishes to see. **Number 600-I $25**

JEWISH I.Q. BASEBALL "Jewish I.Q. Baseball" educates students in many areas of Jewish knowledge in an enjoyable, entertaining fashion. Players get singles, doubles, triples and home runs by answering questions correctly. The runners, the baseball field, and the scoreboard are all portrayed vividly to make the action realistic. "Jewish I.Q. Baseball" is suitable for one or two players of all ages. **Number 301-I $25**

Please add $2 for each shipment for handling.

WRITE FOR FREE CATALOG and to order any of the above software programs, MasterCard or Visa cardholders

CALL TOLL-FREE 1-800-621-8227

(In Illinois, call 1-312-944-4070)

Davka CORPORATION 845 North Michigan Avenue, Suite 843, Chicago, Illinois 60611

Atari is a Trademark of Warner Communications

THE WEST COAST COMPUTER FAIRE

On March 18, 19 and 20, the 8th Annual West Coast Computer Faire was held in the Brooks Convention Hall and Civic Center in San Francisco. The Faire was attended by over fifty thousand computer enthusiasts and hundreds of companies, representing an incredible diversity of products and services.

Among the many publishers exhibiting at the show was A.N.A.L.O.G. Computing. Editor Lee Pappas and technical staff members Tom Hudson and Brian Moriarty jostled around for a peek at the latest software and hardware releases for our favorite computer(s). The "payoff" for this jostling is one of the first profiles on the West Coast Computer Faire to appear in print.

There's no better place to start our profile than with ATARI. DIG DUG, E.T. PHONE HOME and the recently-released QIX were among the main attractions at the ATARI Home Computers exhibit. The "mother company" has also prepared a new word-processing cartridge called ATARIWRITER (see review in this issue), and a very interesting 16K implementation of the LOGO programming language. ATARIWRITER should be on your dealers' shelves by the time you read this. Look for LOGO in August.

Bill Wilkinson of Optimized Systems Software was demonstrating yet another new language for the ATARI. Called ACTION!, this advanced 6502 development system has a structure similar to Pascal and Algol, and is said to run at least 100 times faster than ATARI BASIC. The 16K cartridge includes a 128-column program editor and a built-in compiler, and will sell for somewhere around $100. Also coming from OSS are enhanced cartridge versions of BASIC A+ and MAC/65, and a powerful new mail/merge package.

Rana Systems, a hardware company best known for their Apple-compatible disk drives, is entering the ATARI market in a big way with their announcement of two 5 1/4 inch drives. The extraordinarily compact size and high-tech cosmetics of these products are almost as exciting as the prices: $399 for single-density and $549 for the double-density model. Rana drives have a reputation for outstanding reliability and performance. They should be available before the end of June.

The friendly folks at Adventure International set up their famous "castle" display for the introduction of PREPPIE II, a sequel to their best-selling arcade game. Datasoft was packing them in with the long-awaited home version of Sega's ZAXXON. It offers impressive high-res graphics and extended fine scrolling. The 32K ZAXXON disk faithfully recreates all of the action of the original; the cassette version sacrifices a few features to fit in 16K. Also on display at the Datasoft booth was an underground digging game called O'REILLY'S MINE and the first spelling check program for the ATARI, SPELLWIZARD.

Hayden Software is expanding their ATARI product line with a new educational program, MONKEY SEE MONKEY SPELL. Its amusing animated display should provide plenty of incentive for young learners. Broderbund is continuing the tradition of their hit program CHOPLIFTER with A.E., a high-res shoot-'em-up with multiple screens and lots of quick action. Broderbund has also put CHOPLIFTER and SERPENTINE on cartridges, and is handling the distribution of BANK STREET WRITER, a $70.00 word processing package that was recently lauded in the computer section of Time magazine for its ease of use.

Sirius Software will be following up their phenomenally successful WAYOUT with another three-dimensional arcade maze game called CAPTURE THE FLAG! and a DEFENDER-type program called REPTON. They are also putting the finishing touches on a pair of illustrated adventure games, BLADE OF BLACKPOOL and CRITICAL MASS. Still more adventure for the ATARI is forthcoming from ScreenPlay, whose "Warrior of RAS" role-playing series will include DUNZHIN, KAIV, and THE WILDE. Each adventure in the RAS series includes a randomizing function which makes every game different.

Hofacker-Elcomp had a broad line of ATARI books and software on display, including a very usable implementation of FORTH for less than $40. Axlon has revamped their RAMDISK memory system for better software compatibility, with an attractive new package and a new name: RAMPOWER 128.

Finally, Exersoft Corporation was demonstrating two very unusual products: a foot-operated game controller and an exercise bicycle attachment which allows you to "pilot" an imaginary bike over a scrolling color TV screen. Both products are expected to retail for less than $60.

There you have it -- a quick report of new products from the 8th Annual West Coast Computer Faire. More extensive coverage of these and other new products will appear in future issues of A.N.A.L.O.G Computing.

THE WEST COAST COMPUTER FAIRE

Arcade fans had to wait in line to play Datasoft's Atari version of ZAXXON.

The impressive Synapose Software display featured a separate TV monitor for every product.

The new Rana disk drives for the ATARI attracted lots of attention.

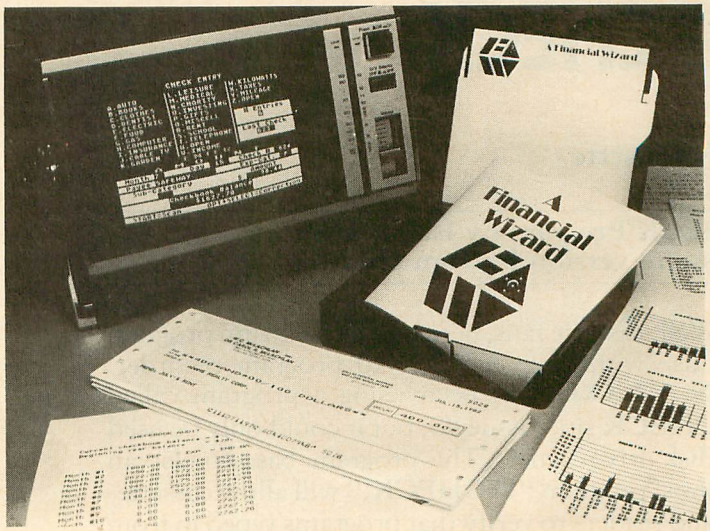
The ATARI exhibit was mobbed with fans eager to sample the new 16K game cartridges.

Business was brisk at the Educational Software booth.

Game fanatics peered over shoulders to check out the coming attractions from Sirius Software.

If you are serious about personal finance...

- **Budget Forecast** – 26 expense categories
- **Check Entry** – easy data entry – scan & modify 26 major & 36 sub-categories – information block
- **Check Search** – single or multiple parameters – (up to seven) to search entries
- **Tabulations** – detailed expense vs. budget comparisons by month, year-to-date, category
- **Bar Graphs** – screen displays in graph form expenses vs. budget – by month or category – printing with graphic capable printers
- **Check Reconciliation** – fast clearing of resident checks & deposits, complete summary report
- **Checkwriter** – prints your custom checks
- ***Print*outs** – most popular printers
- **Multi-Colored Graphics • Audio Enhancements**
- **7 Utility Programs • User-Friendly Operation**
- **Easy To Use Instruction Manual • Audit Report**
- **Handsome Tinted Plastic Storage Case**



COMPUTE!
The Leading Magazine Of Home, Educational, And Recreational Computing

In a feature editorial.

"If you want to use a finance system, but don't want to spend several days trying to learn how to use one, then A Financial Wizard by Computari may be just what you need."

"The illustrated manual that comes with this program is clear, direct, and very thorough."

"It appears that this finance system was designed to achieve the best and most comfortable working relationship between the user and the program."

"The check entry routine is the most attractive feature of this finance system. Data prompts are very clear and the category item names are displayed at all times during data entry for your convenience."

"The file search capabilities of this program are superior. You are offered seven ways to look up the checks."

"The system is disk intensive. All data is saved automatically and immediately following all routines that either enter data or modify it."

"Scanning your entries is made possible by pressing START. You can see records very quickly this way."

"This is an excellent finance system – entertaining, accurate, and fun to use."

ANALOG COMPUTING
THE MAGAZINE FOR ATARI COMPUTER OWNERS

Analog Magazine in a comprehensive study of personal finance systems for Atari* computers.

"A Financial Wizard from Computari is by far the best of these programs and will be the standard of comparison for the others."

"The check entry mode is easy to use..."

"The way a Financial Wizard handles your tabulations is excellent. You can chart your actual expenses vs. your budget by month, by category or year to date."

"...where it really outshines the rest is in the check reconciliation."

"In effect it gives you your bank statement on the screen, a complete list by month of all your checks and deposits."

"A Financial Wizard has one disk that does everything..."

"Graphics, while really not a factor in the quality of programs of this type, do make your budgeting chores a little more pleasant. Again A Financial Wizard comes out on top."

"Everything about this program is excellent..."

Antic
The ATARI Resource

In a Report from Antic.

"Like most Atarians, I am captivated by the graphic, color and sound capabilities of my machine. Nothing quite discourages me more than to boot up an applications program (personal, business, etc.) and to be presented with the standard graphic 'o' white characters on a blue screen.

Of course the usefulness and effectiveness of a program is of primary importance. However, enhancing the *dullest of applications* programs with some of Atari's charms, is a great asset. A Financial Wizard, a personal finance program by Computari's Bill McLachlan, is an excellent example of an applications program that integrates many of the Atari's features into a well conceived and executed program."

"The use of color and sound in the data input prompts and error checking routines are so well done that it's quite simple to boot up the disk, follow along with the very clear documentation, and be 'up and running' in short order."

"I give A Financial Wizard high marks in ease of use, documentation and performance. If a disk-based home finance package is in your future, The Wizard should get serious consideration."

Computari's
A Financial Wizard 1.5
The logical choice.

The system is designed for Atari computers having a minimum of 32K and operating from a disk drive. The cost is only $59.95 plus $3 for handling/postage.

If your dealer does not have **A Financial Wizard** . . . Telephone orders are accepted on Mastercharge or Visa credit cards. Mail order must be accompanied by check or money-order or credit card #.

Dealer inquiries invited.

*trademark of Atari Inc.

OK residents ad 4% Tax.

Exclusively thru **ON LINE** COMPUTER CENTERS of OKC (405)751-7925

10944 North May, Oklahoma City, OK 73120

☐ CHECK ENCLOSED ☐ VISA ☐ MASTERCHARGE

NAME (Print)_____

Address _____

City _____

State _____ Zip _____

Card # _____ Exp._____

Signature_____

C:CHECK

16K Cassette

CHECK Program by Istvan Mohos
Cassette version by Tom Hudson

Well, cassette fans, here it is! a checksum program for use with your ATARI 410 program recorder. **A.N.A.L.O.G.** has been using a program called **D:CHECK** since issue #8, but it could only be used by disk drive owners. This program, based on Istvan Mohos' program **"D:CHECK,"** will allow cassette-bound readers to verify that programs typed in from A.N.A.L.O.G. were entered correctly.

The Program

When D:CHECK was submitted to **A.N.A.L.O.G.** for publication, it was intended to be used only with the ATARI 810 disk drive. It could not be used with cassette because it required three separate "passes" through the BASIC program being checked. This is fine when using a disk drive, which is a "random access" device. A disk drive can read a file of stored information any number of times without the user even noticing. A program recorder, on the other hand, is a "sequential" device. If a file must be read several times, the user must rewind the tape to the beginning of the file each time it is needed. This is, to say the least, a cumbersome operation.

In order to adapt **D:CHECK** for cassette use, a prime consideration was to make the check process a one-pass operation. **C:CHECK** will read through a program which has been LISTed to cassette one time and produce the checksum data on either the screen or a printer.

To make life easier for the **A.N.A.L.O.G.** staff, **C:CHECK** will produce the same checksum data as **D:CHECK**, eliminating the need for separate checksum tables.

Type the **C:CHECK** program into your computer and LIST it to tape. Check data has been provided so that you can use **C:CHECK** to check itself. Follow the instructions below.

Using C:Check

When entering programs from A.N.A.L.O.G., you can use **C:CHECK** to make sure you typed them in without mistakes. The procedure is as follows:

1) After the program is typed in, LIST it to tape by typing LIST "C".
2) *LOAD* and RUN **C:CHECK**.
3) **C:CHECK** will ask if you want the output to go to the screen or to a printer. Type "S" for screen or "P" for printer and press RETURN.
4) Enter the issue number in which the program to be checked appeared, and press RETURN.
5) Position the tape to the beginning of the program that is to be checked and press PLAY on the program recorder. Press RETURN.
6) **C:CHECK** will begin reading the program from tape and generate a checksum table. This table should match the C:CHECK/D:CHECK data listed after the program in the magazine. Each line of the checksum table is a DATA statement representing 15 lines of the program being checked, plus a total of the 15 lines. The line number of each DATA statement is the number of the first line in the group of 15 lines. For example, in the following check data line:

100 DATA 12,3,200,126,60,45,344,455,452,54, 889,344,10,1,56,3051

The line number 100 tells that the first line in this group is 100. The value 12 represents the checksum of line 100, 3 is the checksum of the first line after line 100, 200 is the checksum of the second line after 100, etc. 3051 is the total of the 15 lines in the group. If the value 889 does not match the magazine's corresponding value, the tenth line after line 100 was typed incorrectly. Note all such errors. When **C:CHECK** ends, type NEW, ENTER your program from tape and correct the lines that are in error (if any). When these are corrected, you have a perfect, debugged copy of the program!

That's all there is to it! **C:CHECK** should make finding those bothersome typos much easier, and allow you to enjoy **A.N.A.L.O.G.** programs much faster. □

```
100 REM CHECK DEBUGGING AID
    BY  ISTVAN MOHOS
110 REM VERSION 2 MODS AND CASSETTE
120 REM VERSION BY TOM HUDSON
130 GRAPHICS 0:? :? "This run will LIS
T data statements    to the screen or
printer."
140 ? :? "This DATA is created by eval
uating    each character of a user pro
gram,     LISTed to tape.":?
150 DIM OUT$(1),I$(128),CR$(1)
160 ? "OUTPUT TO SCREEN OR PRINTER";:I
NPUT OUT$:IF OUT$()"S" AND OUT$()"P" T
HEN 160
170 IF OUT$="S" THEN OPEN #2,8,0,"E:":
GOTO 200
180 CLOSE #2:? "READY PRINTER AND PRE
SS RETURN";:INPUT CR$
190 TRAP 180:OPEN #2,8,0,"P:"
200 ? :? "ENTER ISSUE NUMBER";:TRAP 20
0:INPUT ISSUE
210 ? :? "READY TAPE AND PRESS RETURN"
;:OPEN #1,4,0,"C:":? :?
```

```
220 Z=0:LINECOUNT=Z:PLIN=Z:X=2
230 TRAP 340:INPUT #1,I$:LINECOUNT=LIN
ECOUNT+1:LINUM=VAL(I$(1,5))
240 NLCK=NLCK+1:IF NLCK>1 AND NLCK<16
THEN 290
250 IF LINECOUNT=1 THEN 280
260 ? #2;TOTAL:NLCK=1
270 IF OUT$="S" THEN PLIN=PLIN+1:IF PL
IN=10 THEN ? "PRESS RETURN TO CONTINUE
";:INPUT CR$:PLIN=0
280 TOTAL=Z:? #2;LINUM;" DATA ";
290 CHKSUM=Z:IF ISSUE>9 THEN X=2
300 FOR I=1 TO LEN(I$):PRODUCT=X*ASC(I
$(I,I)):CHKSUM=CHKSUM+PRODUCT:X=X+1:IF
 X=4 THEN X=1
310 NEXT I:CHKSUM=CHKSUM+X*155:X=X+1:I
F X=4 THEN X=1
320 CHKSUM=CHKSUM-1000*INT(CHKSUM/1000
)
330 ? #2;CHKSUM;",";:TOTAL=TOTAL+CHKSU
M:GOTO 230
340 CLOSE #1:IF LINECOUNT=Z THEN 370
350 ? #2;TOTAL
360 CLOSE #2:END
370 ? "■";? "Your typed-in program wa
s not properlyLISTed to tape."
380 ? :? "Please LIST your program to
tape, thenRUN ";CHR$(34);"CHECK";CHR$(
34);" again.":CLOSE #2:CLR :END
```

CHECKSUM DATA

(See D:CHECK/C:CHECK,p.26)

```
100 DATA 198,759,11,135,191,594,198,80
6,763,467,931,100,465,572,107,6297
250 DATA 764,922,11,168,375,783,304,25
9,534,890,875,136,732,361,7114
```

We have over 1000 Educational Programs for use on your Atari 400/800 Computers!

You and your family will enjoy learning in your home when you

Stop Playing with Your Atari

All along you've heard there isn't much of a selection of low cost microcomputer courseware. Well, do you want to learn Physics, Psychology, or Philosophy? Accounting or Auto Mechanics? Sociology, Supervision or Statistics? Economics or Electronics? How about English as a Second Language? Would you like to teach your children Math or Reading? If you do, then we have the educational programs for you—and at affordable prices.

It's true there aren't many companies that offer full-length courses in subjects other than reading and arithmetic, and what is offered seems to be drills, tests, games, or simulations. What you really want is a course that covers the subject with, say, 16 full-length lessons called tutorial programs, where you interact with an expert programmer backed by a staff of experts. That's exactly what we have.

ASK THESE QUESTIONS

Why haven't you heard of us before? We're a public company that's been trading over the counter for 16 years. We've been developing interactive learning systems since 1957. We sell hardware and software for interactive film, slide, video and computer learning to the educational field, vo-techs, industries, military, and several state and federal agencies. Our programs are used in all 50 states and even in some foreign countries! In 1975, we started to convert to the Talk & Teach Computer-Assisted Instruction (CAI) System, licensed Atari to use it in 1977, and in 1981 licensed Radio Shack to use our Talk/Tutor System. So, why haven't you heard of us? Frankly, we've kept a low profile. Until now.

Is there something wrong with our software? Well, we don't ask you to enter your name so we can drop it into some later text. We don't ask you to type in your answer and refuse to accept it if it's not spelled just right. And, we don't branch around a lot when you make an error. Our programs simply let you know if you're wrong by proceeding only when you select the right multiple-choice answer.

This proven learning-by-positive-reinforcement method lets you proceed quickly and smoothly through the programs, without a lot of cute tricks. But, if you're a devoted computerist or game freak you may be disappointed at the lack of motion in most of the pictures, or our special visual effects. We have some dandies, like the mushroom cloud that rises over Hiroshima in our History series, but your learning is not distracted by needless special effects.

OVER 1000 PROGRAMS

We have 64 courses of 16 half-hour programs: 1024 programs! All are easy-reading, upper-and-lower-case. All are in Atari color. All are illustrated by frequent graphics, composed of special and regular characters. And, best of all, every frame of every program is accompanied by high-quality, full-time audio narration by professional voice talent. David Stanton, James Mathers, Pam Barrymore. Recorded and played back, not synthesized or digitized!

- If your child is having trouble with Reading or Math, or if he or she is exceptional, and could benefit from professionally programmed lessons or courses, then you need our educational programs for your Atari 400 or 800 to help your child.
- If you want to build your math skills, we have programs on Numbers and their meanings, Addition, Subtraction, Multiplication, Division, Fractions, Decimals, Percents, Angles, Graphs, Word Problems, Algebra, Statistics. There are ten series in all, 160 math programs. Again, all with color, pictures, and a friendly tutor's voice—all the time.
- If your verbal skills need polishing, try our Talk & Teach programs on the alphabet, spelling skills, and every level of vocabulary; Sight Words or Learn by Phonics; develop Reading Comprehension skills from stories and articles. There's even a series on the Great Classics.
- For self-development, try a 16-program course in Economics or Psychology. Or Supervision, Sociology, Counseling (Personal, School, or Employee), Writing, Business, Philosophy, Government, World History, U.S. History, or Accounting.
- For vocational skills, study our 16-program courses in Electronics, Fluid Power, Auto Mechanics, Shop, Carpentry, Construction, Meat Processing, Military Skills (64 programs), and there are many more.

YOU HAVE WHAT IT TAKES

You'll need your Atari 400 or 800, and the Atari Cassette recorder. And to present the Talk & Teach programs you need the Educational System Master cartridge: it's **$25** from us, if you don't have one.

We say these courses, which are recorded one program per side of C20 to C30 (half hour) cassettes, are the equivalent of a chapter, or perhaps an hours lecture. We've been told that by studying our Economics or Psychology course, a student could pass a secondary or college test for credit. Many who have such credits never learned all that's in these courses. And while some of our courses have 1980 copyrights, most are 1981 and 1982. In U.S. History, for example, you'll learn about America's foreign policy during the Falklands crisis as well as that in the Mideast in 1982. And we're putting this information at your fingertips.

Not sure whether you agree more with Rationalists or Existentialists? Do you commit logical fallacies? Try our Philosophy course. If you're not sure about neurosis or psychosis, classical or instrumental conditioning, then you need our Psychology course. Just how independent is the Federal Reserve? Get Economics, Program 12. Confused about the changes in traditional roles of political parties? It's Government 5. And let our tutor tell you patiently and clearly about measures of dispersion and central tendency. Our Statistics goes through F tests. We even have 16 full-length audio-computer programs with Spanish words and phrases.

Remember, these are all for your Atari! Yes, we've done programs for TRS80 Model I, III, Color, for the Apple and T. I. But that's another story.

Would you like to do something more than play games on your Atari? Tired of squeaks and robotics and want to hear a human voice? Do you want to further your education or help your children along in theirs? Now you can. For just **$9.90**, we'll send you one cassette with 2 programs from the course of your choice, 100% guaranteed to work in your Atari, and you can try us out. Better yet, get one full course of 16 programs on 8 cassettes for only **$59.90**. For full documentation, 32 pre-post tests, fancy binder, 8 cassettes with 16 programs of the course you want, send **$79.00**. That's less than books and tuition for most college classes. And we offer a 10 day, 100% exchange allowance. Does your alma mater? Send us your check, or call us at (405) 288-2301 with your Visa/Master Card number. We'll send your educational software pre-paid. Please allow 15 days delivery. Or see your Atari dealer. He may have some of our courses in stock. You've got nothing to lose but your games!

DORSETT
Educational Systems, Inc.

P.O. Box 1226, Norman, OK 73070
(405) 288-2301

VISA MasterCard

IN A GALAXY OF SOFTWARE STARS, WE'RE THE ONE TO WATCH.

THE CORPORATION

Planetary Machine Intelligence is based on the concept that computer programs should accomplish the job they are intended for in the most technically sophisticated manner possible. But injected here must be the ergonomic equation, where man and machine function together, or as most programmers call it, "user friendliness".

THE GUARANTEE

PMI backs all its products with a 1 year limited warranty against defects in material and workmanship under normal use. Should any disk fail due to such defects within the warranty period, it will be replaced or repaired upon receipt of the damaged disk. After the 1 year warranty PMI will replace any of its disks that fail due to such defects for a service charge for an additional year. (For any disk returned, a warranty card must be on file at PMI.)

THE REPLICATOR SERIES FOR THE ATARI

If you're tired of promises and programs that tell you they'll copy everything, (and can't even copy themselves) then PMI has a pair of programs for you. The *Replicator Series* can copy any data from any disk perfectly...everytime. It gives you the ability to start at any sector you wish and stop at any sector you wish. If it encounters a bad sector it displays the bad sector number and the error code on the display screen. It also leaves that sector blank when writing to the duplication disk.

The *Replicator Series* encompasses the PMI philosophy that a computer program should work for you, not against you. Simplicity in operation and technical sophistication are fused into a program designed for everyone. Although technically different, both programs will meet or exceed your expectations of their ability to duplicate every disk in your library.

DEALER INQUIRIES INVITED

PMI

BRINGING COMPUTERS DOWN TO EARTH.

Replicator One requires 32K basic cartridge and one disk drive.
Replicator Two requires 32K basic cartridge and two disk drives.

~~$49.95~~ **$34.95**

THRU 8/1/83 ORDER NOW!

PROGRAMMERS!

If you have written a program that is technically sophisticated and also "user-friendly", PMI would be interested in publishing your work. For information on how to submit your material write to: PMI, P.O. Box 2895, Winter Park, FL 32790-2895, Or phone (305) 644-3822.

PMI, 2500 Lee Road, Suite 210 P.O. Box 2895, Winter Park, FL 32790-2895

☐ Replicator One ☐ Replicator Two
Quantity _____ at $34.95 plus $2.00 S.&H. (Florida residents add 5% sales tax)
☐ CHECK ENCLOSED OR M.O. ☐ C.O.D.

NAME (Print) _____
Address _____
City _____ State _____ Zip _____

Allow 3 weeks for personal checks to clear.

BASIC PROGRAMMING TECHNIQUES

16K cassette 24K disk

by Thomas M. Krischan

There are two fundamental principles in good programming technique: function and cosmetic. The functional principle is the first to be mastered by the novice. It involves the duty portion of programming, which covers these questions:

Does the program work?
Is it error free?
Is it comprehensive?

The cosmetic principle is usually overlooked by the novice programmer, and too often by even the advanced programmer. It addresses a less visible, yet equally important portion of the program:

Is the code structured?
Is it readable?

Are sufficient comments embedded into the listing to document the work properly?

The failings of the programmer to address this second principle are not necessarily the fault of the programmer alone. Most programming courses and text books neglect this subject area. The intent of this article is to outline the cosmetic principle.

A good program contains the following cosmetic elements: structure, readability, and internal documentation. A program listing, like a magazine article, is subdivided into smaller logical units of composition. Articles refer to logical units as paragraphs, programs refer to logical units as subroutines. Paragraphs are not magically created by indented openings, likewise, subroutines are not magically created by adding a RETURN statement. Your subroutine should therefore mark the logical subdivision of the function of the whole program. A subroutine must be complete in itself. Ask these questions of each subroutine:

Is the subroutine unified?
Does it hang together and read smoothly?
Is the function of the subroutine adequately developed?

Another element to consider is if the flow of the program is readable. Unlike a magazine article, logical subdivisions do not have to be read sequentially. Furthermore, some subdivisions may be read from other subdivisions. Proper programming technique requires that the vast majority of the program is read sequentially. This technique is referred to as the "top-down" approach. The alternative approach to this is entitled "spaghetti" programming. Spaghetti programming has nothing to do with the nationality of the programmer. Rather, it describes the flow of the program; a bowl of spaghetti. An excellent exercise for mastering structure and readability is to outline a program listing. Draw a bracket around each logical subdivision, connect the brackets with a line to indicate flow, and sketch an arrow on the line to designate direction. Figure 1. illustrates a program which has good structure and readability. Figure 2. illustrates a program with little structure and poor readability; spaghetti programming.

Figure 1. Good structure and readability.

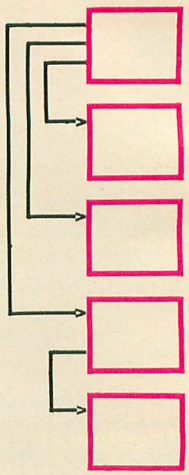
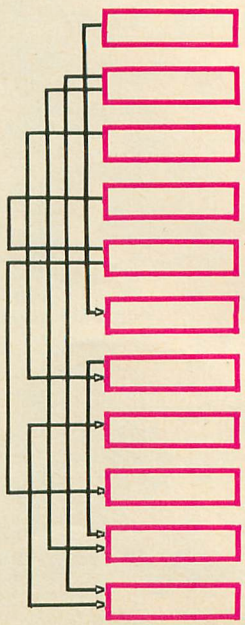


Figure 2. Poor structure and readability.



The last element in our outline is internal documentation. Documentation can be in the form of remark statements or self defining code. Since the BASIC language features English-like statements, I much prefer the latter option. Figure 3. is a listing of BASIC programming statements which use variable names so appropriately that further remarks from the programmer would seem redundant.

Figure 3. Self defining BASIC statements.

```
POKE CURSOR, OUT
SETCOLOR, BACKGRND, GREEN, DARK
WRONG=TOTAL-RIGHT
GOSUB NOISE
GOTO ERROR
```

The fundamental principles of functional and cosmetic programming technique are not without their conflicts. For example:

Assigning numeric constants in GOSUB, GOTO, SETCOLOR, PEEK and POKE commands will quickly use up the 128 available variable names offered by ATARI BASIC.

Long descriptive variable names require more space than do single letter variable names.

Modular design may waste time; GOSUB commands are slower than GOTO commands.

Frequently used subroutines are addressed most quickly if they reside closer to the top of the program.

Despite the conflicts, compromise can usually be reached. The good programmer can balance the benefits of functional programming with the desire to create an internally appealing program listing.

The sample program illustrates such a balanced approach. Figures 4-7 are the external documentation which accompany the program. The program functions as an education tutorial about natural events which occur in our native enviornment during the spring of the year. Because of the structure and readability of the program, questions can be easily added or changed to suit your needs. One word of advice; after running this program with its original contents, if you have not managed to get at least 50% of the questions correct, perhaps you should get up out of your chair and view the world from the other side of your window! □

Figure 4. String Variables.

```
ASK1$    - Questions.
ASK2$
TELL1$   - Supplemental information.
TELL2$
TELL3$
GUESS1$ - Possible answer.
GUESS2$
GUESS3$
```

Figure 5. Numeric Variables.

```
RIGHT   - The number of questions that were
          answered correctly.
WRONG   - The number of questions that were
          answered incorrectly.
TOTAL   - The total number of questions asked.
PITCH   - The pitch value for a sound.
TIME    - The time delay of a programming loop.
```

Figure 6. Numeric Constants.

```
LIGHT       - Color luminance.
MEDIUM
DARK
GREEN       - 3 Color hue.
RED
YELLOW
BLACK
BLUE
WINDOW      - Color register.
TEXT
TEXTCAPS
BACKGRND
BACKTEXT
VOICE0      - Sound generation.
PURE
NORMAL
SMALL       - Graphics mode.
AVERAGE
LARGE
CURSOR      - Hardware memory location.
ATTRACT
UNTOUCH
RESPONSE
KEYBOARD
OFF         - Reset memory locations.
OUT
PAUSE       - Subroutine location.
NOISE
BUZZ
TWINKLE
WAIT
QUIZ
ERROR
SCORE
INTRO
```

Figure 7. Major Program Subdivisions.

1 -7	General information.
11 -99	Main program; calls all supplemental subroutines.
100 -199	Quiz subroutine; generates questions, possible answers and supplemental information.
200 -299	Score subroutine; displays number of correct and incorrect responses, and remarks about the score.
500 -560	Data for closing remarks.
1000-2490	Data for questions, possible answers, and supplemental information.
5000	Noise subroutine; used for correct responses (up tone).
5050	Buzz subroutine; used for incorrect responses (down tone).

5100	Pause subroutine; pauses display for a few seconds before continuing on.
5150-5170	Wait subroutine; waits display for a keystroke.
5200-5299	Twinkle subroutine; flashes text for additional attraction.
5300	Error subroutine; when a wrong key is pressed.
6000-6099	Intro subroutine; draws first display screen.

```
1 REM .     DATE: DECEMBER 1982
2 REM .   AUTHOR: THOMAS M. KRISCHAN
3 REM .           TECHNIMETRICS CONSL.
4 REM .           646 S. 92nd St.
5 REM .           WEST ALLIS, WI,53214
6 REM .           1-(414)-476-4511
7 REM .
11 REM .
12 REM .
20 REM . *** MAIN PROGRAM ***
22 DIM GUESS1$(60),GUESS2$(40),GUESS3$
(40),ASK1$(40),ASK2$(80),TELL1$(40),TE
LL2$(40),TELL3$(120)
23 RIGHT=0:WRONG=0:TOTAL=0
24 LIGHT=10:MEDIUM=6:DARK=0:GREEN=12:R
ED=4:YELLOW=1:BLACK=0:BLUE=7
25 PAUSE=5100:NOISE=5000:BUZZ=5050:TWI
NKLE=5200:WAIT=5150:QUIZ=100:ERROR=530
0:SCORE=200:INTRO=6000:START=23
26 CURSOR=752:OUT=1:ATTRACT=77:UNTOUCH
=255:RESPONSE=764:OFF=0:KEYBOARD=53775
:LMARGIN=82
27 WINDOW=2:TEXT=1:TEXTCAPS=0:BACKGRND
=4:BACKTEXT=2:SMALL=0:AVERAGE=1:LARGE=
2
28 VOICE0=0:PURE=10:NORMAL=8
29 TRAP 80:GOSUB INTRO
30 FOR QUESTION=1000 TO 2400 STEP 100:
TOTAL=INT(QUESTION/100-9):GOSUB QUIZ:P
OKE ATTRACT,OFF
40 NEXT QUESTION
50 GOSUB SCORE
60 GOTO START
80 TOTAL=TOTAL-1:GOSUB SCORE
99 GOTO START
100 REM . *** QUIZ ***
110 GRAPHICS SMALL:POKE CURSOR,OUT:SET
COLOR BACKTEXT,BLACK,DARK:SETCOLOR TEX
T,BLACK,LIGHT:POKE LMARGIN,0
115 RESTORE QUESTION:READ ASK1$,ASK2$
118 POSITION 0,0:? #6;"QUESTION NUMBER
 ";TOTAL;"     NUMBER CORRECT ";RIGHT
120 POSITION 0,1:? #6;ASK1$
121 POSITION 0,2:? #6;ASK2$
125 RESTORE QUESTION+30:READ GUESS1$,G
UESS2$,GUESS3$,CORRECT,TELL1$,TELL2$,T
ELL3$
130 POSITION 0,5:? #6;GUESS1$
140 POSITION 0,7:? #6;GUESS2$
150 POSITION 0,9:? #6;GUESS3$
160 KEY=PEEK(KEYBOARD):IF KEY=UNTOUCH
THEN 160
161 IF PEEK(RESPONSE)=31 THEN ANSWER=1
:GOTO 170
162 IF PEEK(RESPONSE)=30 THEN ANSWER=2
:GOTO 170
163 IF PEEK(RESPONSE)=26 THEN ANSWER=3
:GOTO 170
166 IF PEEK(RESPONSE)=39 THEN QUESTION
=QUESTION-100:TOTAL=TOTAL-1:GOTO SCORE
169 GOTO ERROR
170 POSITION 15,12:IF ANSWER=CORRECT T
HEN ? #6;" YES ":GOSUB NOISE:RIGHT=RIG
HT+1:GOTO 180
175 ? " NO ":GOSUB BUZZ
180 POSITION 0,15:? #6;TELL1$
181 POSITION 0,16:? #6;TELL2$
182 POSITION 0,17:? #6;TELL3$
190 POSITION 0,20:? #6;"    PRESS ANY
KEY TO CONTINUE"
191 POSITION 0,21:? #6;"    PRESS )|(
KEY TO QUIT"
199 GOSUB WAIT:RETURN
200 REM . *** SCORE ***
210 GRAPHICS AVERAGE:POKE CURSOR,OUT:S
ETCOLOR BACKGRND,GREEN,DARK:SETCOLOR W
INDOW,GREEN,DARK
211 SETCOLOR TEXT,YELLOW,MEDIUM
215 WRONG=TOTAL-RIGHT
220 POSITION 0,0:? #6;"YOU ANSWERED "
230 POSITION 0,1:? #6;TOTAL;" QUESTION
S"
240 POSITION 0,3:? #6;RIGHT;" CORRECTL
Y"
250 POSITION 0,4:? #6;WRONG;" INCORREC
TLY"
260 POSITION 0,6:? #6;INT((RIGHT/TOTAL
)*100);" PERCENT!!!!"
265 REM . SELECT APPROPRIATE RESPONSE
270 RESTORE INT(RIGHT/TOTAL*6)*10+500:
READ GUESS1$
275 REM . PUT RESPONSE INTO A STRING
280 POSITION 0,9:? #6;GUESS1$
290 ? "    PRESS ANY KEY TO CONTINUE"
299 GOSUB WAIT:RETURN
500 DATA BETTER LUCK NEXT    TIME
510 DATA NOT TOO GOOD, BUT    YOU'RE L
EARNING!"
520 DATA NICE TRY!
530 DATA GOOD JOB!
540 DATA VERY GOOD!!
550 DATA EXCELLENT!! THAT'S   ALMOST A
 PERFECT   SCORE.
560 DATA TREMENDOUS!! THAT'S A PERFECT
 SCORE!!
1000 REM .** QUESTION NO.1 **
1010 DATA What are spring peepers?
1020 DATA
1030 DATA 1.  FLOWERS
1040 DATA 2.  FROGS
1050 DATA 3.  HUMANS
1060 DATA 2
1070 DATA  FROGS : a small amphibian t
hat sings
1080 DATA (ie.peeps) during the first
warm
1090 DATA weeks of spring.
1100 REM .** QUESTION NO.2 **
1110 DATA Name the wetland plant that
blooms
1120 DATA the earliest?
1130 DATA 1.  PURPLE LOOSESTRIFE
1140 DATA 2.  JOE-PYE WEED
1150 DATA 3.  SKUNK CABBAGE
1160 DATA 3
1170 DATA  SKUNK CABBAGE : it often bl
ooms so
1180 DATA early that it pokes up throu
gh the
1190 DATA snow.
1200 REM .** QUESTION NO.3 **
1210 DATA What bird begins it's nestin
g season
1220 DATA in February?
1230 DATA 1.  GREAT HORNED OWL
1240 DATA 2.  CARDINAL
1250 DATA 3.  PHEASANT
1260 DATA 1
1270 DATA  GREAT HORNED OWL : often be
fore the
1280 DATA eggs are laid you can hear t
he
1290 DATA courtship hooting of the owl
s.
1300 REM .** QUESTION NO.4 **
1310 DATA What mammal is the legendary
1320 DATA forecaster of spring?
1330 DATA 1.  CHIPMUNK
1340 DATA 2.  WOODCHUCK
1350 DATA 3.  MUSKRAT
1360 DATA 2
1370 DATA  WOODCHUCK : sometimes known
 as a
1380 DATA groundhog; if it wakes too e
arly and
```

48K AUSTIN BOARD™

$114.95 SUGG. LIST

GIVE YOUR ATARI 400* COMPUTER

THE MEMORY POWER OF AN ATARI 800*

AUSTIN BOARD™ FEATURES

- Lifetime warranty.
- Works with all hardware and software including all cartridges.
- Custom components for highest reliability.
- Superior CAD/CAM generated P.C. design.
- Complete bank and board de-coupling for cleanest signal possible.
- Gold edge connectors.
- P.C. template provided for easy ATARI 400* installation.
- Fully tested.
- No system lock-up due to overheating. Uses 1/2 the power of other 48K boards on the market.
- ATARI 800* compatibility with optional loop-back card (available for $5.00 additional).

$99.95 with 16K board trade-in

NOW AVAILABLE

AUSTIN 80™ — An 80 Column color video display board for the ATARI 800*

Ask for AUSTIN BOARDS™ at your local dealer or call

617-772-0352

AUSTIN FRANKLIN ASSOCIATES

43 GROVE STREET, AYER, MA 01432

M/C, VISA & COD ACCEPTED

DEALER INQUIRIES INVITED

*ATARI IS A REGISTERED TRADEMARK OF ATARI, INC.

```
1390 DATA food is scarce it can return
 to          hibernation for a few more
 weeks.
1400 REM .** QUESTION NO.5 **
1410 DATA Which bird nests later than
other
1420 DATA small birds?
1430 DATA 1.  GOLDFINCH
1440 DATA 2.  SPARROW
1450 DATA 3.  STARLING
1460 DATA 1
1470 DATA GOLDFINCH: waits until sum
mer because
1480 DATA it uses thistle down to line
 it's
1490 DATA nest. Thistle mature in summ
er.
1500 REM .** QUESTION NO.6 **
1510 DATA What NATURAL DISASTER can he
lp the
1520 DATA prairie to be more fertile?
1530 DATA 1.  FLOODING
1540 DATA 2.  FIRE
1550 DATA 3.  WIND
1560 DATA 2
1570 DATA FIRE: burns off the old dr
y plant
1580 DATA stalks and helps some seeds
to
1590 DATA germinate.
1600 REM .** QUESTION NO.7 **
1610 DATA Which native bird lays it's
eggs in
1620 DATA the nests of other birds?
1630 DATA 1.  STARLING
1640 DATA 2.  BLACKBIRD
1650 DATA 3.  COWBIRD
1660 DATA 3
1670 DATA COWBIRD: used to follow th
e buffalo
1680 DATA for food and did not have ti
me to
1690 DATA care for their young.
1700 REM .** QUESTION NO.8 **
1710 DATA What part of a maple tree ca
n you eat
1720 DATA with your breakfast?
1730 DATA 1.  LEAVES
1740 DATA 2.  SAP
1750 DATA 3.  BARK
1760 DATA 2
1770 DATA SAP: used to make maple sy
rup.
1780 DATA
1790 DATA
1800 REM .** QUESTION NO.9 **
1810 DATA What favorite spring flower
in the
1820 DATA Lily family has 3 petals - 3
 sepals       and 3 leaves?
1830 DATA 1.  BUTTERCUP
1840 DATA 2.  TRILLIUM
1850 DATA 3.  ROSE
1860 DATA 2
1870 DATA TRILLIUM: one of our showi
est
1880 DATA woodland flowers.
1890 DATA
1900 REM .** QUESTION NO.10 **
1910 DATA What insect is a major polli
nator of
1920 DATA the flowers of fruit trees?
1930 DATA 1.  HONEY BEE
1940 DATA 2.  BUTTERFLY
1950 DATA 3.  WASP
1960 DATA 1
1970 DATA HONEY BEE: workers use the
 flowers of
1980 DATA early blooming trees to buil
d up the
1990 DATA honey supply and feed new be
es.
2000 REM .** QUESTION NO.11 **
2010 DATA What colorful crustacean swi
ms upside
2020 DATA down in temporary spring pon
ds?
2030 DATA 1.  CYCLOPS
2040 DATA 2.  FAIRY SHRIMP
2050 DATA 3.  CRAYFISH
2060 DATA 2
2070 DATA FAIRY SHRIMP: they mate in
 spring -
2080 DATA lay their eggs quickly - and
2090 DATA disappear until next spring.
2100 REM .** QUESTION NO.12 **
2110 DATA What bird has a speckled bre
ast in
2120 DATA winter that changes to black
 in spring?
2130 DATA 1.  STARLING
2140 DATA 2.  ROBIN
2150 DATA 3.  JAY
2160 DATA 1
2170 DATA STARLING: the beak also ch
anges to
2180 DATA bright yellow during spring
mating
2190 DATA season.
2200 REM .** QUESTION NO.13 **
2210 DATA What insect comes to Canada
from
2220 DATA Mexico to mate and lay eggs?
2230 DATA 1.  DRAGONFLY
2240 DATA 2.  TARANTULA
2250 DATA 3.  MONARCH BUTTERFLY
2260 DATA 3
2270 DATA MONARCH BUTTERFLY: lays eg
gs on
2280 DATA Milkweed plants.
2290 DATA
2300 REM .** QUESTION NO.14 **
2310 DATA What wetland bird does an ae
rial sky
2320 DATA dance each spring?
2330 DATA 1.  WOODCOCK
2340 DATA 2.  HERON
2350 DATA 3.  WOOD DUCK
2360 DATA 1
2370 DATA WOODCOCK: at sunrise and s
unset male
2380 DATA woodcocks woo females with t
his
2390 DATA spectacular display.
2400 REM .** QUESTION NO.15 **
2410 DATA What spring mushroom is a fa
vorite of
2420 DATA gourmet cooks?
2430 DATA 1.  HEDGEHOG
2440 DATA 2.  SHAGGY MANE
2450 DATA 3.  MOREL
2460 DATA 3
2470 DATA MOREL: considered by many
to have
2480 DATA the best flavor of any wild
mushroom.
2490 DATA But some people are allergic
 to them.
5000 FOR PITCH=200 TO 2 STEP -3:SOUND
VOICE0,PITCH,PURE,NORMAL:NEXT PITCH:SO
UND VOICE0,OFF,OFF,OFF:RETURN
5050 FOR PITCH=50 TO 255 STEP 3:SOUND
VOICE0,PITCH,PURE,NORMAL:NEXT PITCH:SO
UND VOICE0,OFF,OFF,OFF:RETURN
5100 FOR TIME=0 TO 300:NEXT TIME:RETUR
N
5150 FOR TIME=0 TO 50:NEXT TIME
5160 KEY=PEEK(KEYBOARD):IF KEY=UNTOUCH
 THEN 5160
5170 FOR TIME=1 TO 3:NEXT TIME:RETURN
5200 FOR FLASH=1 TO 100:SOUND VOICE0,4
0,PURE,NORMAL:HUE=RND(0)*16:LUMIN=HUE:
SETCOLOR TEXTCAPS,HUE,LUMIN
5299 SOUND VOICE0,15,PURE,NORMAL:NEXT
FLASH:SOUND VOICE0,OFF,OFF,OFF:RETURN
5300 POSITION 5,12:? "OOPS.....THAT'S
THE WRONG KEY":GOSUB NOISE:GOTO QUIZ
6000 REM . ***INTRODUCTION***
6003 GRAPHICS AVERAGE:POKE CURSOR,OUT:
SETCOLOR BACKGRND,YELLOW,DARK:SETCOLOR
 WINDOW,YELLOW,DARK
6004 SETCOLOR TEXT,YELLOW,MEDIUM:SETCO
LOR TEXTCAPS,GREEN,MEDIUM
6005 POSITION 0,1:? #6;"self teaching
```

```
series"
6010 POSITION 0,7:? #6;"       A SPRING
"
6020 POSITION 0,9:? #6;"        NATURE"
6030 POSITION 0,11:? #6;"         QUIZ
"
6060 ? :? :? "    (C) 1983, Thomas M.
Krischan";
6070 GOSUB PAUSE:GOSUB PAUSE:? :? :? :
? "        PRESS ANY KEY TO BEGIN."
6080 GOSUB TWINKLE
6099 GOSUB WAIT:RETURN
```

•

CHECKSUM DATA
(See D:CHECK/C:CHECK,p.26)

```
1 DATA 636,747,676,413,268,265,246,82,
85,577,110,712,53,4,4,4878
27 DATA 743,187,341,522,623,116,162,29
0,195,718,634,424,720,458,464,6597
125 DATA 986,677,686,695,453,309,312,3
19,973,135,170,130,623,629,635,7732
190 DATA 3,163,493,556,108,838,223,990
,549,471,731,200,375,566,16,6282
280 DATA 702,955,495,421,67,566,498,62
7,400,473,562,93,528,263,300,6950
1050 DATA 758,372,875,41,338,568,11,76
5,607,66,482,378,885,90,463,6699
1200 DATA 574,423,772,754,474,473,375,
899,88,219,580,636,824,512,932,8535
1350 DATA 281,381,62,518,904,586,925,8
52,899,274,495,381,6,906,263,7733
1500 DATA 592,831,775,484,32,40,387,99
8,184,277,598,817,745,493,866,8119
1650 DATA 288,393,976,458,605,604,952,
432,710,517,28,393,277,555,556,7744
1800 DATA 610,882,78,32,537,72,396,158
,902,559,782,453,654,820,51,6986
1950 DATA 63,396,805,430,73,758,696,43
4,255,64,467,374,761,320,621,6517
2100 DATA 762,599,176,480,264,523,374,
66,23,684,766,255,207,962,952,7093
2250 DATA 111,383,593,441,543,770,385,
9,505,294,856,380,566,395,438,6669
2400 DATA 774,579,338,430,641,312,389,
292,110,182,438,560,612,449,581,6687
5170 DATA 98,236,493,885,683,473,255,9
36,591,360,423,140,822,603,173,7171
```

•

WHAT IS D:CHECK/C:CHECK

Most program listings in **A.N.A.L.O.G.** are followed by a table of numbers appearing as DATA statements, called "CHECKSUM DATA." These numbers are to be used in conjunction with D:CHECK, which appeared in issue no. 10, and C:CHECK, in this issue, p. 16.

D:CHECK and C:CHECK are programs by Istvan Mohos and Tom Hudson. They are designed to find and correct typing errors when entering programs from the magazine. For those readers who do not have a copy of issue no. 10, send a pre-addressed, stamped, business-sized envelope to:

D:CHECK ARTICLE
P.O. BOX 23
WORCESTER, MA 01603

CONTROL CHARACTERS

Some program listings reproduced in **A.N.A.L.O.G.** may contain "strange" characters not shown on the ATARI keyboard. These are special characters which use the CTRL, ESC and "ATARI LOGO" (INVERSE) keys. Shown below is a list of these characters and the keystrokes used to get them. □

--- CTRL ,
--- CTRL A
--- CTRL B
--- CTRL C
--- CTRL D
--- CTRL E
--- CTRL F
--- CTRL G
--- CTRL H
--- CTRL I
--- CTRL J
--- CTRL K
--- CTRL L
--- CTRL M
--- CTRL N
--- CTRL O
--- CTRL P
--- CTRL Q
--- CTRL R
--- CTRL S
--- CTRL T
--- CTRL U
--- CTRL V
--- CTRL W
--- CTRL X
--- CTRL Y

--- CTRL Z
--- ESC ESC
--- ESC CTRL UP-ARROW
--- ESC CTRL DOWN-ARROW
--- ESC CTRL LEFT-ARROW
--- ESC CTRL RIGHT-ARROW
--- CTRL .
--- CTRL ;
--- ESC SHIFT CLEAR
--- ESC BACK S
--- ESC TAB
--- INVERSE CTRL ,
--- INVERSE CTRL A
--- INVERSE CTRL B
--- INVERSE CTRL C
--- INVERSE CTRL D
--- INVERSE CTRL E
--- INVERSE CTRL F
--- INVERSE CTRL G
--- INVERSE CTRL H
--- INVERSE CTRL I
--- INVERSE CTRL J
--- INVERSE CTRL K
--- INVERSE CTRL L

--- INVERSE CTRL M
--- INVERSE CTRL N
--- INVERSE CTRL O
--- INVERSE CTRL P
--- INVERSE CTRL Q
--- INVERSE CTRL R
--- INVERSE CTRL S
--- INVERSE CTRL T
--- INVERSE CTRL U
--- INVERSE CTRL V
--- INVERSE CTRL W
--- INVERSE CTRL X
--- INVERSE CTRL Y
--- INVERSE CTRL Z
--- ESC DELETE
--- ESC INSERT
--- ESC CTRL TAB (CLR)
--- ESC SHIFT TAB (SET)
--- INVERSE SPACE
--- INVERSE _
--- INVERSE CTRL .
--- INVERSE CTRL ;
--- INVERSE |
--- ESC CTRL 2
--- ESC CTRL BACK S
--- ESC CTRL INSERT


Castle Wolfenstein™
by MUSE™
FOR APPLE & ATARI
THE MOST INTERACTIVE GAME EVER!
The wildest, most addictive adventure game to date–a genuine merger of fantasy and real-time arcade elements.
"A virtuoso feat of programming..."–New York Magazine.
For the Apple II 48 K, the Atari 400 or 800, 32 K. $29.95
By Silas Warner.
™MUSE
347 N. Charles Street
Baltimore, MD 21201 (301) 659-7212
Call or write for information and the name of your nearest MUSE dealer. Apple is a TM of Apple Computer Corp. Atari is a TM of Atari Inc.
ARCADE AWARDS
1983
ELECTRONIC GAMES MAGAZINE
Certificate of Merit
For Outstanding Achievement
WALDREP

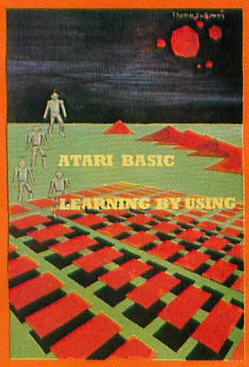
HOFACKER

BOOKS + SOFTWARE

FOR ATARI 400/800® · VIC-20® · SINCLAIR/TIMEX® · OSI · APPLE® PET/CBM

ELCOMP PUBLISHING, INC.
53 Redrock Lane, Pomona CA 91766 / USA
Phone (714) 623-8314

Payment: Check, money order, VISA, Master Charge, Eurocheck
Orders from outside USA: add 15 % shipping.
CA-residents add 6.5 % tax.



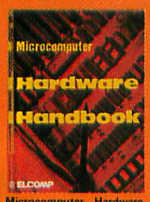
ATARI BASIC – Learning by Using
An excellent book for the beginner. Many short programs and learning exercises.
Order-No. 164 $7.95

Games for the ATARI Computer
This book describes advanced programming techniques like player-missile-graphics and use of the hardware-registers. Contains many ready-to-run programs in BASIC and one called GUNFIGHT in machine language.
Order-No. 162 $7.95

How to program your ATARI in 6502 machine language
The subject of this book is to teach you how to program your ATARI computer in 6502 machine language. Contains a large collection of programs.
Order-No. 169 $9.95

Program Descriptions (ATARI)
Order-No. 173 $5.00

FORTH on the ATARI–Learning by Using, Order-No. 170 $7.95



Microcomputer Hardware Handbook (845 pages)
Descriptions, pinouts and specifications of the most popular microprocessors and support chips.
A MUST for the hardware buff.
Order-No. 29 $14.95

Care and Feeding of the Commodore PET
Eight chapters exploring PET hardware. Includes repair and interfacing information. Programming tricks and schematics.
Order-No. 150 $9.95

Complex Sound Generation
New revised applications manual for the Texas Instruments SN 76477 Complex Sound Generator.
Order-No. 154 $6.95

8K Microsoft BASIC Reference Manual
Order-No. 161 $9.95

Expansion Handbook for 6502 and 6802
Order-No. 152 $9.95

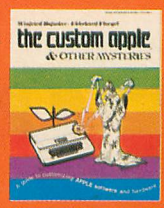
Books for OSI

The First Book of OHIO	#157	$7.95
The Second Book of OHIO	#158	$7.95
The Third Book of OHIO	#159	$7.95
The Fourth Book of OHIO	#160	$9.95
The Fifth Book of OHIO	#161	$7.95

Bare boards from the Custom APPLE Book

6522 I/O Board	#605	$39.00
EPROM Burner	#607	$49.00
8K EPROM/RAM Board	#609	$29.00
Prototyping board	#604	$29.00
Slot repeater board	#606	$49.00

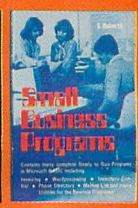
Order two boards and get the book FREE!



The Custom Apple + Other Mysteries – A complete guide to customizing the Apple Software and Hardware.
Order-No. 680 $24.95

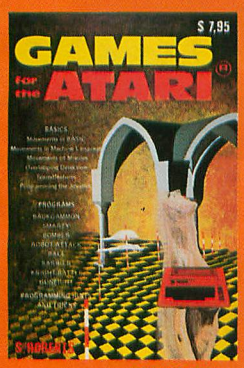
We also stock the boards which are used in the book "The Custom Apple..." (bare boards).

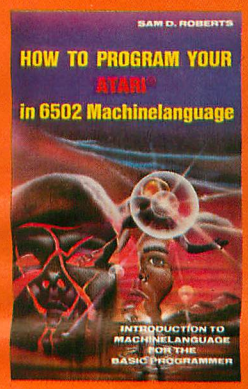
Programming in 6502 Machine Language on your PET + CBM
2 complete Editor/Assemblers (Source code 3 hexdump + description plus a powerful machine language monitor (hexdump).
Order-No. 166 $19.95
Above assemblers on cassette (Book No. 166 included).
Order-No. 4812 $39.95



Programming in BASIC and machine language with the ZX-81 (82) or TIMEX 1000.
Order-No. 174 (book) $9.95

Small Business Programs
Complete listings for the business user. Inventory, invoice writing, mailing list and much more.
Order-No. 156 $14.90





Books + Software for VIC-20 (requires 3K RAM Exp.)

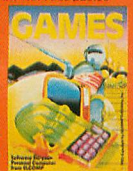
Wordprocessor,8K RAM	#4870	$19.95
Mailing List, 16K RAM	#4883	$14.95
Tricks for VICs (book)	#176	$ 9.95
TIC TAC VIC	#4880"	$ 9.95
GAMEPACK I(3games)	#4881	$14.95
Dual Joystick Instruct.	#4885	$ 9.95
INPUT/OUTPUT Progr	#4886	$ 9.95
Miniassembler	#4896	$19.95
Tennis, Squash,Break	#4881	$ 9.95
Runfil for VIC	#4894	$ 9.95

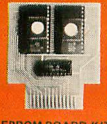
Universal Experimenter Board for the VIC-20 – (Save money with this great board). This board plugs right into the expansion slot of the VIC-20. The board contains a large prototyping area for your own circuit design and expansion. The construction article shows you how to build your own 3k RAM expander and ROM-board.
Order-No. 4844 $18.95

Software for SINCLAIR ZX-81 and TIMEX 1000
Machine Language Monitor
Order-No. 2399 $9.95
Mailing List #2398 $19.95

COMING SOON! ORDER NOW!
A Look in the future with your ATARI (Astrology and how to do your own horoscope on the ATARI 800).
Order-No. 171 $9.95
Astrology and Biorhythm for ATARI (cass. or disk).
Order-No. 7223 $29.95
Birth control(Knaus Ogino). Cass. or disk.
Order-No. 7222 $29.95

Programs from Book No. 164
The programs from book No. 164 on disk (book included).
Order-No. 7100 $29.00
Programs from Book No. 162 on disk (book included)
Order-No. 7221 $29.95
GUNFIGHT
This game needs two joysticks. Animation+ sound. Comes on a bootable cassette.
Order-No. 7207 $19.95





Hardware-ADD-ONs for ATARI

EPROM BOARD (Cartridge) Holds two 4k EPROMs (2532). EPROMs not included.
7043 $29.95

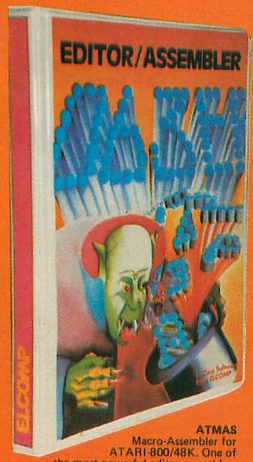
EPROM BOARD KIT
Same as above but bare board only with description.
Order-No. 7224 $14.95

Printer Interface
This construction article comes with printed circuit board + software. You can use The EPSON printer without the ATARI printer interface. (Gameport 3 and 4).
Order-No. 7211 $19.95

RS-232 Interface for your ATARI 400/800
Software + connector + constr. article.
Order-No. 8291 $19.95

EPROM BURNER for ATARI 400/800
Works with gameport. No additional power supply.Comes compl. assembled with software (2716,2732,2532).
Order-No. 7042 $179.00

EPROM BURNER for ATARI 400/800 Kit
Printed circuit board incl. software and extensive construction article.
Order-No. 7292 $49.00



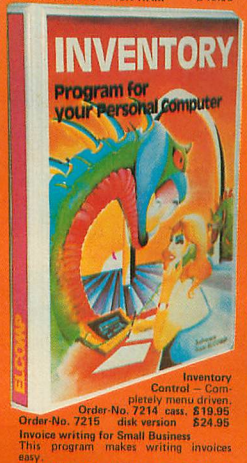
ATMAS
Macro-Assembler for ATARI-800/48K. One of the most powerful editor assemblers on the market. Versatile editor with scrolling. Up to 17k of source-code. Very fast, translates 5k source-code in about 5 sekonds. Source code can be saved on disk or cassette.(IncludesATMONA-1).
Order-No. 7099 disk version $89.00
Order-No. 7999 cartridge version $129.00

ATAS
Same as ATMAS but without macro-capability. Cassette-based.
Order-No. 7098 32K RAM $49.95
Order-No. 7998 48K RAM $49.95

ATMONA-2
This is a tracer debugger) that lets you explore the ATARI RAM/ROM area. You can stop at previously selected address, opcode, or operand. Also very valuable in understanding the microprocessor.Includes ATMONA-1.
Order-No. 7049 cassette version $49.95
Order-No. 7050 disk version $54.00

ATMONA-1
A powerful machine language monitor. Disassemble, dump (hex and ASCII), change memory location, block transfer, fill memory block, save and load machine-language programs, start programs. Printer option via three different interfaces.
Order-No. 7022 cassette version $19.95
Order-No. 7023 disk version $24.95
Order-No. 7024 cartridge version $59.00

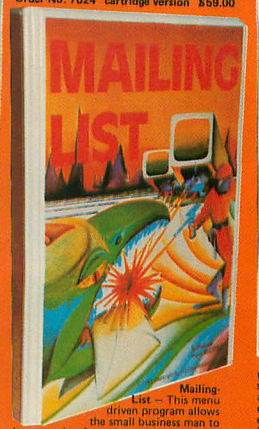
FORTH from Elcomp Publishing,Inc. is an extended Fig-Forth-version, Editor and I/O package included. Utility package includes decompiler, sector copy, Hex-dump (ASCII), ATARI Filehandling, total graphic and sound, joystick program and player missile. Extremely powerful!
Order-No. 7055 disk $39.95
Floating point package with trigonometric functions (0–90°).
Order-No. 7230 disk $29.95
Learn-FORTH from Elcomp Publishing, Inc.
A subset of Fig-Forth for the beginner. On disk (32K RAM) or on cassette (16k RAM).
Order-No. 7053 $19.95



Inventory Control – Completely menu driven.
Order-No. 7214 cass. $19.95
Order-No. 7215 disk version $24.95

Invoice writing for Small Business
This program makes writing invoices easy.
Order-No. 7201 cassette version $29.95
Order-No. 7200 disk version $39.95

ATAMEMO – This is the date-block you always wanted. Makes effective time planing easy!
Order-No. 7310 disk only $29.95



Mailing-List – This menu driven program allows the small business man to keep track of vendors and customers. You can search for a name or address of a certain town or for an address with a certain note. 50 addresses are put into one file.
Order-No. 7212 cassette version $19.95
Order-No. 7213 disk version $24.95

ATEXT-1
This wordprocessor is an excellent buy for your money. It features screen oriented editing, scrolling, string search (even nested), left and right margin justification. Over 30 commands. Text can be saved on disk or cassette.
Order-No. 7210 cassette version $29.95
Order-No. 7216 disk version $34.95
Order-No. 7217 cartridge version $69.00
A version for the BIT3-80 col. card is avail.
ATCASH-1 – Cash register for ATARI
Order-No. 7307 disk only $49.95

MOVING MISSILES IN BASIC

16K Cassette or Disk

by Tom Hudson

Last issue, I presented a machine-language subroutine which enabled BASIC programmers to move players around on the screen quickly. Shortly after the issue was sent to readers, I received a letter from Jeff Stefanski (see the Reader Comment section of this issue). Jeff asked for a modification to allow the subroutine to move missiles as well.

Rather than modify the existing player movement subroutine, I decided to write a new, stand-alone missile movement subroutine. It can be used by itself if only missiles are desired, or can be used in conjunction with the player mover from A.N.A.L.O.G. #10 if both players and missiles are needed.

Following this article are two listings. The first is a BASIC program which demonstrates the use of the missile movement subroutine. The second listing is the fully documented assembler source code for the subroutine.

What Are Missiles?

Hidden deep inside each ATARI computer is a mysterious graphics ability known as player-missile graphics. These graphics work with any graphics mode, and can be moved around on the screen without disturbing any other graphics.

Why are these graphics called players and missiles? PLAYERS are eight pixels (picture elements) wide and therefore can be used to create fairly detailed images, such as spaceships, cars, or other animated figures representing the player.

MISSILES, on the other hand, are only TWO pixels wide. They were designed to be used as simple projectiles, because of their limited resolution.

The Demonstration Program

Enter Listing 1 into your computer. Before running it, be sure to SAVE it, as a mistake in typing the assembly-language code could "lock-up" your computer, making it necessary to re-enter the program. When RUN, this program will place the numbers 1-4 on the screen using the four missiles and move them around randomly. Let's walk through the program and see what each line does.

Line 200 — This line loads the machine-language missile movement subroutine into a string called "MISMOV$." This subroutine will be called whenever we want to move a missile on the screen.

Line 240 — This line sets up four string variables, M0$ through M3$. These strings will hold missile shape data. These strings are currently set up as 6 bytes long, which limits the missile graphics images to 6 pixels in height. You can change this length to up to 128 bytes, making the missile image 128 pixels high.

Lines 250-280 — These lines READ the DATA in lines 690-750 into the missile shape strings set up in line 240. Once again, note that each line reads 6 bytes into the appropriate string. To make missile images of different height, change the 6 to the desired value.

Lines 290-320 — These lines set up the player-missile area in memory and activate them. Line 320 sets the player-missile priority to 1. This causes the players and missiles to appear "in front" of other graphics. These lines should not be changed.

Lines 330-360 — Since there is no SETCOLOR command for player-missile graphics, we must POKE the appropriate color values into the P/M color registers. To get the color number, use the formula:

```
COLOR POKE VALUE = (COLOR NUMBER *16)
                   + BRIGHTNESS
```

Line 370 — Sets the background color to black.

Line 410 — Dimensions two arrays, X and Y. These arrays will be used to hold the X and Y coordinates of each missile.

Line 420 — This line initializes all the missiles' X coordinates to 128 and the Y coordinates to 64. These coordinates will place the missiles at the center of the screen.

Line 430 — This line starts a FOR-NEXT loop which will process each missile, from missile 0 to missile 3.

Lines 440-490 — These lines randomly change the X and Y coordinates of the missiles, which will make them wander around on the screen.

Line 500 - Depending on the missile number (I) this line transfers control to the appropriate USR

statement in order to move the desired missile.

Lines 510-540 — These lines send the X and Y coordinate information to the missile movement subroutine. Line 510 moves missile 0, 520 moves missile 1, etc. Let's look at line 510:

```
A=USR(MISL,0,PMB,ADR(M0$),X(0),Y(0),6)
:GOTO 550
```

This statement has 7 parameters inside the USR parentheses.

"MISL" is set up in line 200. Do not change this variable. It is the address of the missile mover subroutine.

"0" means we want to move missile zero. Note that line 520 has a "1" here, since it moves missile 1. This value can range from 0 to 3, and will move the appropriate missile.

"PMB" is the player-missile base address, which was calculated in line 300. Do not change this variable.

"ADR(M0$)" tells the missile mover where to get the missile image data. In this case, we want the subroutine to use the information in the string variable M0$. Try changing this to "ADR(M1$)" and RUN the program. You will see two "2's" moving on the screen. This is because the string M1$ contains the data for the number 2, and it is now used in both missile 0 and missile 1.

X(0) is the X coordinate where we want to place missile 0. You can place any number or variable here, ranging from 0-255.

Y(0) is the Y coordinate where we want to place missile 0. You can place any number or variable here, ranging from 0-127.

"6" tells the subroutine how many bytes are used for the player image. In this case our missiles are 6 bytes long (see lines 240, 250, and 690). If you want a different number of bytes (resulting in a different missile height), change these lines accordingly.

Line 550 — This line completes the FOR-NEXT loop set up in line 430.

Line 560 — This line passes control to line 430, causing the program to loop forever, moving the missiles randomly until you press the BREAK key.

Lines 600-640 — These lines contain the DATA for the missile mover routine. Do not change these values.

Line 690 — DATA for M0$, the number "1."

Line 710 — DATA for M1$, the number "2."

Line 730 — DATA for M2$, the number "3."

Line 750 — DATA for M3$, the number "4."

Creating Your Own Missile Images

Now, that we have walked through the program and studied what each line does, let's design our own missile image. Since we're limited to 2 pixels in width, the image will have to be very simple. Of course, it can be any height up to 128 pixels.

The numbers that the demonstration program moved around on the screen are very simple. Figure 1 shows how the number 2 was turned into DATA in line 710.

```
2+1  =3
  1  =1
2+1  =3
2    =2
2    =2
2+1  =3
```

Figure 1

The shape we will make is shown in Figure 2.

```
  1  =1
  1  =1
2+1  =3
     =0
2+1  =3
     =0
2+1  =3
2    =2
2    =2
```

Figure 2

Now let's put our shape into demonstration program. We'll use missile number 0 to show the image, so replace line 690 with the following:

```
690 DATA 1,1,3,0,3,0,3,2,2
```

This missile image is 9 bytes long, so it will be necessary to change lines 240, 250 and 510 as follows:

```
240 DIM M0$(9),M1$(6),M2$(6),M3$(6)
250 FOR I=1 TO 9:READ N:M0$(I)=CHR$(N)
:NEXT I:REM *** MISSILE 0 ***
510 A=USR(MISL,0,PMB,ADR(M0$),X(0),Y(0
),9):GOTO550
```

After the program is changed, RUN it. You will see the numbers 2, 3, and 4 on the screen, along with the shape we just defined. It's that simple!

Summary

Using players and missiles in BASIC can be very fast and easy when a machine-language subroutine is used to perform time-consuming operations. This demonstration program may be used as a framework for more complex programs. Simply replace lines 410-560 with your own program code, and you're all set to begin exploring the wonders of player-missile graphics!

If you have any questions or suggestions about this article write me care of **A.N.A.L.O.G.** Be sure to include a pre-addressed, stamped envelope if you would like a reply.

I would like to thank Jeff Stefanski for his suggestion that a missile movement subroutine be written for A.N.A.L.O.G. Computing. If you have any suggestions for articles, just write. □

```
100 REM ********************************
110 REM *  MISSILE SUBROUTINE DEMO  *
120 REM *                           *
130 REM *        BY TOM HUDSON      *
140 REM *                           *
150 REM *  A.N.A.L.O.G. COMPUTING   *
160 REM ********************************
170 REM
180 REM ************ SETUP ************
190 REM
200 DIM MISMOV$(114):MISL=ADR(MISMOV$)
:FOR X=1 TO 114:READ N:MISMOV$(X)=CHR$
(N):NEXT X:REM *READ ML DATA*
210 REM
220 REM *** LOAD MISSILE IMAGES ***
230 REM
240 DIM M0$(6),M1$(6),M2$(6),M3$(6)
250 FOR I=1 TO 6:READ N:M0$(I)=CHR$(N)
:NEXT I:REM *** MISSILE 0 ***
260 FOR I=1 TO 6:READ N:M1$(I)=CHR$(N)
:NEXT I:REM *** MISSILE 1 ***
270 FOR I=1 TO 6:READ N:M2$(I)=CHR$(N)
:NEXT I:REM *** MISSILE 2 ***
280 FOR I=1 TO 6:READ N:M3$(I)=CHR$(N)
:NEXT I:REM *** MISSILE 3 ***
290 PMBASE=INT((PEEK(145)+3)/4)*4:POKE
 54279,PMBASE:REM *** SET UP P/M AREA
***
300 PMB=PMBASE*256
310 POKE 559,46:POKE 53277,3:REM *** P
/M DMA ***
320 POKE 623,1:REM *** P/M PRIORITY **
*
330 POKE 704,134:REM *** P/M 0 COLOR *
**
340 POKE 705,136:REM *** P/M 1 COLOR *
**
350 POKE 706,138:REM *** P/M 2 COLOR *
**
360 POKE 707,142:REM *** P/M 3 COLOR *
**
370 SETCOLOR 2,0,0:REM *** BACKGROUND
IS BLACK ***
380 REM
390 REM **** YOUR PROGRAM HERE! ****
400 REM
410 DIM X(3),Y(3)
420 FOR I=0 TO 3:X(I)=128:Y(I)=64:NEXT
 I
430 FOR I=0 TO 3
440 XI=2-INT(RND(0)*5):YI=2-INT(RND(0)
*5)
450 X(I)=X(I)+XI:Y(I)=Y(I)+YI
460 IF X(I)<50 THEN X(I)=50:GOTO 480
470 IF X(I)>190 THEN X(I)=190
480 IF Y(I)<20 THEN Y(I)=20:GOTO 500
490 IF Y(I)>110 THEN Y(I)=110
500 ON I+1 GOTO 510,520,530,540
510 A=USR(MISL,0,PMB,ADR(M0$),X(0),Y(0
),6):GOTO 550
520 A=USR(MISL,1,PMB,ADR(M1$),X(1),Y(1
),6):GOTO 550
530 A=USR(MISL,2,PMB,ADR(M2$),X(2),Y(2
),6):GOTO 550
540 A=USR(MISL,3,PMB,ADR(M3$),X(3),Y(3
),6)
550 NEXT I
560 GOTO 430
570 REM
580 REM *** MISSILE MOVER DATA ***
590 REM
600 DATA 216,104,104,104,133,213,104,1
33,206,104,24,105,128,133,205,165,206,
105,1,133,206,104,133,204,104
610 DATA 133,203,104,104,133,208,104,1
04,133,209,104,104,24,101,209,133,207,
160,0,162,0,134,212,169,252
620 DATA 166,213,240,7,10,10,9,3,202,2
08,249,166,212,49,205,145,205,196,209,
144,30,196,207,176,26
630 DATA 132,212,138,168,177,203,164,2
13,240,5,10,10,136,208,251,164,212,17,
205,145,205,232,169,0,240
640 DATA 0,200,192,128,208,196,166,213
,165,208,157,4,208,96
650 REM
660 REM *** MISSILE IMAGE DATA ***
670 REM
680 REM "1"
690 DATA 1,1,1,1,1,1
700 REM "2"
710 DATA 3,1,3,2,2,3
720 REM "3"
730 DATA 3,1,3,1,1,3
740 REM "4"
750 DATA 1,3,3,1,1,1
```

●

CHECKSUM DATA

(See D:CHECK/C:CHECK,p.26)

```
100 DATA 778,122,244,236,250,743,796,9
5,82,101,420,79,336,85,653,5020
250 DATA 303,310,317,324,918,525,764,9
56,925,934,943,937,145,102,581,8984
400 DATA 80,27,287,293,985,719,583,574
,556,545,765,333,343,353,205,6648
550 DATA 751,724,103,504,109,674,637,1
46,251,237,99,405,105,342,282,5369
700 DATA 322,279,330,282,338,280,1831
```

●

Assembly Language Listing

```
;       -------------------------
;       MISSILE MOVER SUBROUTINE
;       -------------------------

;       -------------------------
;       BY TOM HUDSON
;       A.N.A.L.O.G. COMPUTING #11
;       -------------------------

;       -------------------------
;       PAGE ZERO USAGE
;       -------------------------

PMSTR   =   $CB         ;P/M BASIC STRING
MADR    =   $CD         ;MISSILE ADDRESS
PMEND   =   $CF         ;MISSILE IMAGE END
XPOS    =   $D0         ;X POSITION
YPOS    =   $D1         ;Y POSITION
HOLD    =   $D4         ;HOLD AREA
MNUM    =   $D5         ;MISSILE # TO MOVE

;       -------------------------
;       OPERATING SYSTEM EQUATES
;       -------------------------

HPOSM0  =   $D004

;       -------------------------
;       SUBROUTINE STARTS HERE!
;       -------------------------

        ORG $6000       ;ANY ADDRESS

START   CLD             ;CLEAR DECIMAL MODE
        PLA             ;DISCARD
        PLA             ;DISCARD # HI
        PLA             ;PULL MISSILE # LO
        STA MNUM        ;AND SAVE IT!
```

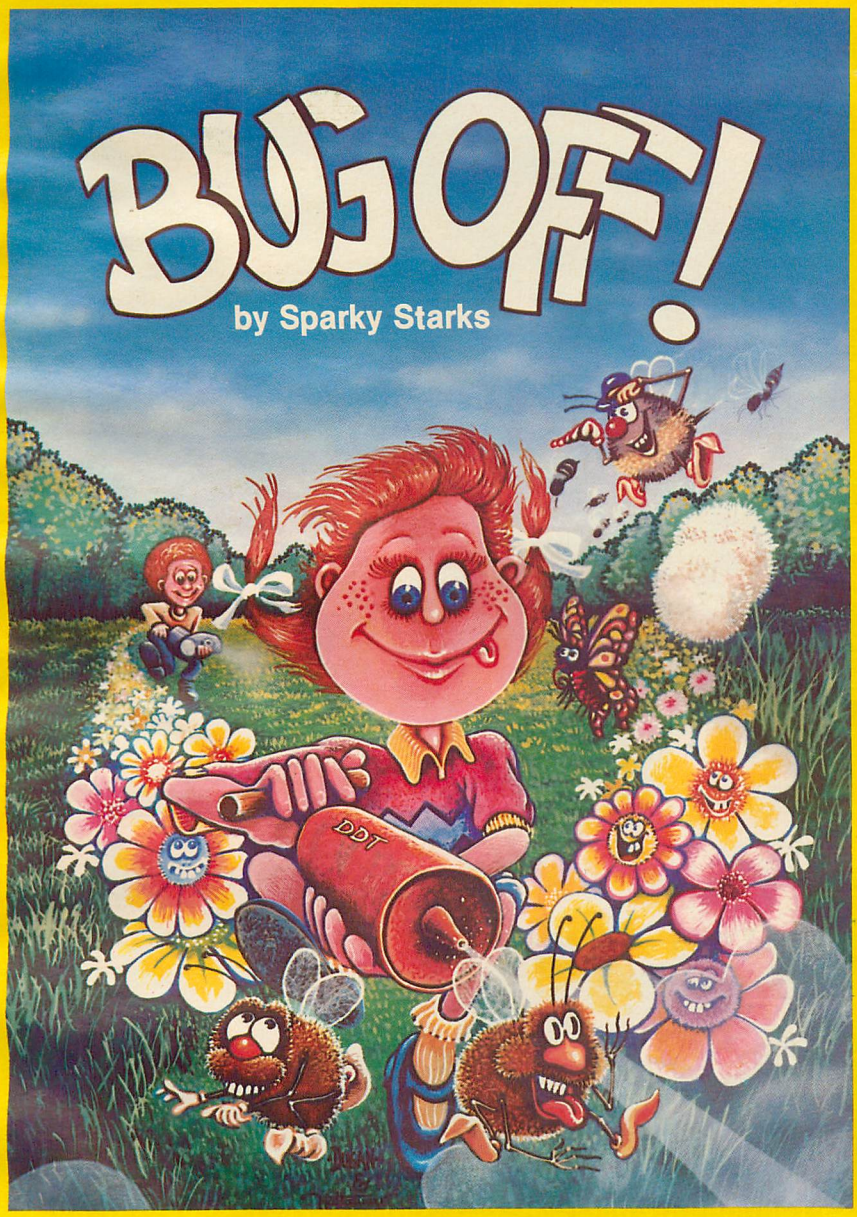
```
        PLA             ;PULL P/M BASE HI
        STA MADR+1      ;AND SAVE!
        PLA             ;PULL P/M BASE LO
        CLC             ;OFFSET INTO
        ADC #128        ;MISSILE AREA
        STA MADR        ;AND SAVE!
        LDA MADR+1      ;OFFSET MISSILE
        ADC #1          ;ADDR HI
        STA MADR+1
        PLA             ;PULL STRING HI
        STA PMSTR+1     ;AND SAVE!
        PLA             ;PULL STRING LO
        STA PMSTR       ;AND SAVE!
        PLA             ;DISCARD X HI
        PLA             ;PULL X LO
        STA XPOS        ;AND SAVE IT!
        PLA             ;DISCARD Y HI
        PLA             ;PULL Y LO
        STA YPOS        ;AND SAVE IT!
        PLA             ;DISCARD LENGTH HI
        PLA             ;PULL LENGTH LO
        CLC             ;ADD Y POSITION
        ADC YPOS        ;TO GET END
        STA PMEND       ;AND SAVE IT!
        LDY #0          ;ZERO P/M COUNT
        LDX #0          ;ZERO STRING COUNT
COPYLP  STX HOLD        ;SAVE X REG
        LDA #$FC        ;SET HIGH 6 BITS
        LDX MNUM        ;GET MISSILE #
        BEQ ZEROIT      ;IF 0, DON'T SHIFT
ZERSHF  ASL A           ;SHIFT LEFT...
        ASL A           ;TWO BITS
        ORA #3          ;SET LOWER 2 BITS
        DEX             ;DONE SHIFTING?
        BNE ZERSHF      ;NO!

ZEROIT  LDX HOLD        ;RESTORE X REG
        AND (MADR),Y    ;ZERO OUT...
        STA (MADR),Y    ;MISSILE BYTE!
        CPY YPOS        ;COPYING DATA YET?
        BCC NEXT        ;NO!
        CPY PMEND       ;FINISHED COPYING?
        BCS NEXT        ;YES!
        STY HOLD        ;SAVE Y REG
        TXA             ;MOVE X REG...
        TAY             ;TO Y REGISTER
        LDA (PMSTR),Y   ;GET P/M BYTE
        LDY MNUM        ;SHIFT BIT IMAGE
        BEQ ENDBS       ;IF NOT MISSILE 0
BYTSHF  ASL A           ;SHIFT LEFT...
        ASL A           ;2 BITS
        DEY             ;MORE TO SHIFT?
        BNE BYTSHF      ;YES!
ENDBS   LDY HOLD        ;GET P/M OFFSET
        ORA (MADR),Y    ;'OR' BITS
        STA (MADR),Y    ;CHANGE MISSILE!
        INX             ;NEXT STRING BYTE.
        LDA #0          ;FORCE BRANCH
        BEQ NEXT        ;TO NEXT BYTE!
NEXT    INY             ;NEXT P/M BYTE
        CPY #128        ;DONE W/COPY?
        BNE COPYLP      ;NOT DONE YET!
        LDX MNUM        ;GET MISSILE #
        LDA XPOS        ;NOW JUST SET
        STA HPOSM0,X    ;X LOCATION!
        RTS             ;FINIS!

        .END
```

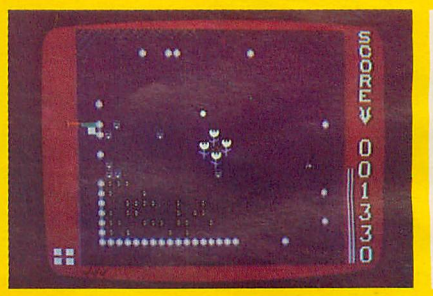
●

YOU'RE GOING TO HAVE FUN WITH YOUR ATARI!!



AND THE FUN GETS ROLLING WITH BUG OFF!

Yikes! The bugs are swarming here, there and everywhere and only a strong whiff of DDT can put 'em away. The object of the game is to control the seven different kinds of pests that are running helter-skelter over everything. The Army can airlift in more DDT to fill your bug sprayer . . . but will they make it in time?



The action builds to a furious frenzy as an awesome assortment of insects attack anything and everything in sight. A definite case of "spray first and ask questions later." All this and hi-res graphics, too!

16K TAPE	050-0167	$29.95
32K DISK	052-0167	$29.95

TUTTI FRUTTI — by Alan Newman

"GRAPE FUN" FOR THE WHOLE FAMILY!

Somewhere between Never Never Land and next Thursday, there's a wild and wacky place where your joystick-controlled "Hungry" snacks on an orchard of goodies. But there are some nasty ol' bugs who'd just as soon snack on YOU, so look sharp and beware. **Fun**tastic action for all ages!

TUTTI FRUTTI . . . Wow! A crazy cast of characters in vivid hi-res color all performing in their natural habitat, otherwise known as TUTTI FRUTTI Land. Grape fun for ages 6 and up!

16K TAPE	050-0160	$24.95
32K DISK	052-0160	$24.95

Adventure INTERNATIONAL

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) 862-6917
PRICES SUBJECT TO CHANGE

THE PLACE: a remote outpost on an ice world light years from earth.

Suddenly the small planet THULE is surrounded by alien life orbs destined to change the atmosphere to suit themselves. However, this spells certain death for your base as the frozen mountains start to melt!

You take the command of your single seat fighter to destroy the aliens before it's too late. But you didn't know the aliens had help . . .

STAR SENTRY is an arcade-type space action game written entirely in machine language for one player. Cassette or disk, 24K. $29.95.

A.N.A.L.O.G.

SOFTWARE

P.O. BOX 23
WORCESTER, MA 01603
(617) 892-3488

STRINGS IN ATARI BASIC

The Republic of Letters

> **They that dally nicely with words may quickly make them wanton.**
> ***(Shakespeare: Twelfth Night III)***

by Richard G. Lyons

The ability to easily manipulate alphanumeric characters was a major innovation in computer software. For any program to be considered "user friendly," it must communicate with, and accept from the user, English words. Being able to reply to computer inquiries with "Yes" and "No" gives a user the impression of having a conversation with the machine. Furthermore, it is much more efficient for a computer to "interpret" our human language than it is for us to interpret the computer's numerical language. String operations that delete, change, or insert groups of characters in a manuscript is the primary activity of Text Editors. All BASIC programmers soon realize the necessity of learning and understanding string operations. Although numerous articles have been published describing the ATARI 400/800 computer, they have dealt primarily with graphics capabilities. This article provides an expanded description of the ATARI BASIC String operations.

Strings

A string is an array of alphanumeric characters. These characters can consist of letters, numbers, punctuation marks, or even the special ATARI keyboard symbols. A string which contains no characters is called a "null string." Examples of strings are:

```
"ABCD"
"X"
"007"
"$IGI!"
""          (null string)
```

Note that each string was contained within quotation marks. The quotation marks inform the BASIC interpreter of the beginning and ending characters in a string. Consequently, quotation marks are illegal as string characters. Carriage Return (CR) is also invalid as a string character. Although most versions of BASIC restrict string lengths to 256 characters, ATARI BASIC permits strings to contain up to 32767 characters.

String Names

Since strings can be manipulated as variables they must have variable names. There are several conventions that must be followed when defining string variable names. A string name must be from 1 to 120 characters in length, begin with a letter, and end with a dollar sign ($). String names may not contain punctuation marks or ATARI special characters. Examples of string variable names and the direct definition of the contents of the strings are:

```
A$="280Z"   ....legal
X0T4$="ABCD" ..legal
EX$="!!!"   ....legal
B:$="BNUM"  ...illegal string name
RST$=""RESET"".illegal string characters
```

Dimensioning Strings (DIM)

Although A$="ABCD" is a valid BASIC statement, it cannot be used alone. All strings must be "dimensioned" before they can be defined or manipulated. Strings are dimensioned using the DIM statement. DIM statements allocate memory storage locations, and establish the string names for string variables. For example:

```
10 DIM T$(12)
```

permits the programmer, at some later time, to define the contents of string T$ with up to twelve characters. Note that the DIM statement does not define the contents of string T$, but merely reserves twelve memory locations. Twelve 8-bit bytes of RAM memory are allocated in the above example. Consider the following program executed on an ATARI 800 with 48K bytes of memory:

```
10 DIM A$(32767)
20 DIM B$(32767)
30 END
```

This is a valid program. Lines 10 and 20 dimension the strings A$ and B$ by giving them the ability to consist of a maximum of 32767 characters. However, strings A$ and B$ are null strings since they each contain no characters. Their dimensioned lengths are 32767, but their "character lengths" (number of characters in a string) are zero. The above program can be executed, but if an attempt is made to fill the strings with characters, an error will result because of insufficient memory space.

ATARI BASIC permits several strings to be dimensioned in one statement. For example:

```
10 DIM A$(100),B$(200),C$(300)
```

It is common practice to put DIM statements at the beginning of a program, and to dimension string variables with a number that is larger than necessary.

Defining Strings

There are several ways of defining a string in ATARI BASIC. The most direct method is a statement which indentifies the characters in a string. Such as:

```
10 A$="ABCD"
```

This statement merely defines string A$ as the string ABCD. Another definition technique envolves the INPUT statement. In this technique, the user of the program is prompted to define a string. Consider the following program:

```
10 DIM NAME$(10)
20 PRINT "WHAT IS YOUR NAME"
30 INPUT NAME$
40 PRINT "YOUR NAME IS_";NAME$
50 END
```

In this case, line 20 prompts the user to define the string NAME$. Line 30 performs the definition. Should the user key in more than 10 characters, only the first 10 characters would be used to define NAME$. Throughout this article, blank spaces will be identified with the symbol _ as shown in line 40.

READ statements can be used to define strings. For example:

```
10 DIM A$(11),B$(11),C$(11),D$(11)
20 READ A$,B$,C$,D$
30 PRINT A$;B$;C$;D$
40 DATA STRING_,DEFINITION_,USING_,READ
50 END
```

Line 20 defines the four strings A$,B$,C$, and D$. Read statements help minimize programming effort by defining several strings using only one statement. Note that quotation marks must not be used in DATA statements containing strings. The most flexible (and complex) way of defining strings concerns the use of subscripts and substrings.

Subscripts and Substrings

Subscripts are numbers, or variables, used to identify portions of a string. Substrings are strings of characters that are contained in a larger string. A single character can be considered a substring. Substrings are defined by applying subscripts to a larger string.

String names can have zero, one, or two subscripts. First, let's consider string definitions without the use of subscripts:

Statement **Printed Results**

```
10 DIM A$(10),B$(10),C$(4)
20 A$="ABCDEF":PRINT A$     ..... ABCDEF
30 B$=A$:PRINT B$           ..... ABCDEF
40 C$=A$:PRINT C$ ............... ABCD
```

Line 20 directly defines string A$. Although A$ has a dimensioned length of 10, its character length is 6 since it contains only 6 characters. A$ occupies 6 bytes of memory. Line 30 defines string B$ by setting string B$ equal to the string A$. We'll refer to string B$ as the destination string and string A$ as the source string. Line 40 illustrates an interesting characteristic of ATARI BASIC. Since string C$ has a dimensioned length of 4, only the first 4 characters of string A$ are used to define C$. Attempting to define a destination string with more characters than it can contain does not result in a software error!

Next, let's consider string definition using a single subscript. String statements containing a single subscript take the form STRINGNAME$(s1), where sl is the subscript. For example:

Statement **Printed Results**

```
10 DIM A$(10),B$(10)
20 A$="12345678":PRINT A$ ..... 12345678
30 B$=A$(4):PRINT B$      ..... 45678
40 B$=A$(5):PRINT B$      ..... 5678
50 B$(5)=A$(5):PRINT B$   ..... 56785678
60 B$=A$(0):PRINT B$      ..... ERROR
70 B$=A$(9):PRINT B$      ..... ERROR
```

The first subscript encountered in this program is the (4) in line 30. The term A$(4) is a substring of the larger string A$. Substring A$(4) is the string of characters starting with the 4th and extending to the last character of string A$. Line 30 defines string B$ to be equal to the substring A$(4), namely "45678". Line 40 shows a similar definition with a subscript of 5. Line 50 illustrates the definition of a destination substring, B$(5), with a source substring A$(5). This operation combines the two substrings to define string B$. Combining strings, or substrings, is known as concatenation. In order to concatenate two strings without losing any characters, the subscript of the destination substring must be equal to one plus the character length of the current destination string. This principle is illustrated in line 50. Since the character length of B$ is 4 (before line 50 was executed), the destination subscript must be 5. Line 60 illustrates that zero is not a valid subscript. If a subscript exceeds the character length of a string, an error occurs as shown in line 70.

The use of double subscripts permits the execution of additional string operations. String statements containing double subscripts take the form STRINGNAME$(s1,s2). Consider the following statements:

Statement **Printed Results**

```
10 DIM S$(15),T$(15),Q$(15)
20 S$="ATASIC":PRINT S$  .. ATASIC
30 Q$=S$(3,5):PRINT Q$  ... ASI
40 T$="RI_BA":PRINT T$  ... RI_BA
50 S$(7,13)=T$:PRINT S$  . ATASICRI_BA
60 S$=S$(1,6):PRINT S$  ... ATASIC
70 T$(6,9)=S$(4,6):PRINT T$ RI_BASIC
80 S$(4)=T$:PRINT S$  ..... ATARI_BASIC
90 Q$=S$(1):PRINT Q$  ..... ATARI_BASIC
100 Q$=S$(1,1):PRINT Q$  .. A
```

Line 30 defines string Q$ to be equal to the substring S$(3,5). Substring S$(3,5) is the string of characters starting with the 3rd character and ending with the 5th character of string S$, namely "ASI". Line 50 shows the concatenation of strings S$ and T$. An example of string truncation is illustrated in line 60. A technique for isolating the first character of a string using double subscripts is shown in line 100. This double subscript method is useful for examining a user response. For example:

```
10 DIM A$(5)
20 PRINT "DO YOU WANT TO CONTINUE? YES
OR NO"
30 INPUT A$
40 IF A$(1,1)="Y" THEN GOTO 60
50 END
```

This routine examines the first character of the user's response to line 20. Any character in a string can be isolated by double subscripts when both subscripts are set equal to the appropriate character number.

It would be impractical (not to mention tedious) to demonstrate all possible single and double subscript combinations for defining strings in this article. As with any programming technique, experience is the best teacher. Therefore, the reader is encouraged to experiment on his/her own.

ATARI String Functions

String operations are greatly enhanced by the use of String Functions. Some of the brief descriptions of String Functions given in Chapter 7 of the ATARI BASIC Reference Manual require further explanation.

Variables in ATARI BASIC are either numbers or strings of characters. Often times it is convenient to treat a numeric variable as a string or to treat a string variable as a number. Numeric variables can be converted to strings and string variables can be converted to a number by two String Functions, STR$ and VAL. The following statements show how the function STR$ converts a number to a string:

Statement **Printed Results**

```
10 DIM S$(20)
20 S$=STR$(17):PRINT S$       17
30 P=2*S$:PRINT P        .... ERROR
40 S$=STR$(10/3):PRINT S$     3.33333333
50 S$=STR$(22E17):PRINT S$    2.2E+18
60 PRINT S$(4,4)       ....... E
```

Line 20 defines string S$ to be the two-character string "17". Although S$ is equal to "17", it is a string variable and, as line 30 shows, it is illegal to attempt to perform an arithmetic operation on a string. Lines 40 and 50 show two more examples of converting a number to a string.

The VAL function converts a string into a number. For example:

Statement **Printed Results**

```
10 DIM S$(8):S$="25":PRINT S$       25
20 PRINT SQR(S$)       ............ ERROR
30 PRINT SQR(VAL(S$))     ........ 5
40 S$="36TT":PRINT SQR(VAL(S$))    6
50 S$="X3.6TT":PRINT SQR(VAL(S$)) ERROR
60 PRINT 2*VAL(S$(2))       ....... 7.2
```

Line 10 defines S$ as a two-character string, namely "25". Line 20 is an illegal statement because you cannot perform an arithmetic operation on a string. The VAL function in line 30 converts string S$ to the numerical value of 25. Line 30 also performs a square root operation. Lines 40, 50, and 60 show that string S$ can contain non-numerical characters but the VAL function can only be applied to numerical characters.

There are two additional String Functions that convert variables from string to numeric and vice versa. These two String Functions are ASC and CHR$. They deal primarily with obtaining the ATASCII decimal code of a character and obtaining the ATASCII character corresponding to a decimal number.

Let's consider ASC(sexp) first:

Statement **Printed Results**

```
10 DIM A$(5):A$="VWXYZ"
20 N=ASC(A$):PRINT N ..............86
30 N=ASC(A$(4)):PRINT N ...........89
```

Note that if the string expression (sexp) is a string name, A$ in line 20, the ASC(A$) function returns the decimal ATASCII code for the first character in the string.

Line 20 sets a numeric variable N equal to the decimal code for the first character in string A$. The decimal code for any character in a string can be obtained if subscripts are used, as shown in line 30. The corresponding decimal code for ATASCII characters can be found in Appendix C of the ATARI BASIC Reference Manual.

The String Function CHR$ performs the opposite operation of ASC. CHR$ is used to obtain the

ATASCII character whose corresponding code number is an integer from 0 to 255. CHR$ has the format: CHR$(aexp). The argument (aexp) can range from 0 to 65535. This range corresponds to values that can be contained in a 16-bit word. However, the CHR$ function only operates on the least significant 8 bits of the value (aexp). Consider the following examples:

```
10 PRINT CHR$(65)
15 REM PRINTS A$ "A"
20 PRINT CHR$(577)
25 REM PRINTS A$ "A"
30 PRINT CHR$(65.49)
35 REM PRINTS A$ "A"
40 PRINT CHR$(65.5)
45 REM PRINTS A$ "B"
50 PRINT CHR$(2.33)
55 REM PRINTS A$ "B"
60 PRINT CHR$(-65)
65 REM ERROR (negative aexp)
70 PRINT CHR$(65,66)
75 REM ERROR (one character only
```

Line 10 shows the most common form of CHR$; i.e., (aexp) is normally in the range of 0 to 255. If (aexp) is greater than 255, the BASIC interpreter substracts some integer multiple of 256 from (aexp) to obtain a number in the range of 0 to 255. Line 20 shows that CHR$(577) is equivalent to CHR$(65), since 577-2x256=65. Lines 30 and 40 show how (aexp) is rounded to an integer. Lines 60 and 70 show two illegal forms of (aexp).

Often times, for emphasis, it is advantageous to display a message in ATARI's Inverse Video. (On a printer, the Inverse Video characters would appear as underlined alphanumeric characters.) A string can be converted to Inverse Video with the ASC and CHR$ Functions. The reader is encouraged to execute the following program:

```
10 DIM MSG$(11)
20 MSG$="_ATTENTION_"
30 FOR X=1 TO LEN(MSG$)
40 MSG$(X,X)=CHR$(ASC(MSG$(X,X))+128)
50 NEXT X
60 PRINT MSG$
```

The loop, from lines 30 to 50, obtains the decimal code for each character in string MSG$, adds 128 to the code value, and then converts the new code back to an Inverse Video character.

Perhaps the most useful String Function is LEN. The format of this function is LEN(STRINGNAME$). LEN is used to obtain the character length of the string STRINGNAME$. For example:

```
10 DIM A$(10)
20 X=LEN(A$):PRINT X
25 REM PRINTS 0 (A$ is a null string)
30 A$="ABCD":PRINT LEN(A$)
35 REM PRINTS 4
40 A$(LEN(A$)+1)=A$:PRINT A$
45 REM PRINTS A$ "ABCDABCD"
50 X=LEN(A$):PRINT X
55 REM PRINTS 8
```

Line 20 shows that the character length of the undefined string A$ is zero. String A$ is defined as LEN(A$) is printed in line 30. A straightforward technique for concatenation is shown in line 40. The subscript (LEN(A$)+1) will always point to the character position just beyond the last character in string A$. A good example of using LEN for concatenation is given on page 39 of the ATARI BASIC Reference Manual.

Although the Logical Operators NOT, AND, and OR cannot be applied to strings directly, they can be used with the LEN function. For example:

```
10 DIM A$(10),B$(10)
20 X=NOT LEN(A$):PRINT X
25 REM PRINTS 1 (LEN(A$)=0)
30 A$="AB":PRINT A$
35 REM PRINTS A$ "AB"
40 X=NOT LEN(A$):PRINT X
45 REM PRINTS 0
50 B$=NOT A$:PRINT B$
55 REM ERROR illegal logical operation
```

The ATARI BASIC Memory Management cannot concatenate strings that have a character length of some integer multiple of 256 (i.e. 256, 512, 768, etc.). The following routine uses the LEN function to guard against this problem:

```
10 REM * STRING LENGTH CHECK ROUTINE
20 DIM SPACE$(1)
30 SPACE$="_"
40 FOR I=1 TO 127
50 IF LEN(A$)=I*256 THEN A$(LEN(A$)+1)
=SPACE$
60 IF LEN(B$)=I*256 THEN B$(LEN(B$)+1)
=SPACE$
70 NEXT I
80 -continue-
```

The routine checks the character lengths of the two (previously defined) strings A$ and B$. The loop, from lines 40 to 70, checks both strings to see if either has a length which is an exact multiple of 256. If either string does, lines 50 or 60 will add a space character to the string, enabling correct string manipulations later in the program.

Basic String Functions Not Available in ATARI BASIC

There are several useful String Functions, found in other BASIC Interpreters, which are not available in ATARI BASIC.

String Function	Explanation
LEFT$(A$,I)	Returns the LEFTmost I characters in string A$.
RIGHT$(A$,I)	Returns the RIGHTmost I characters in string A$.
MID$(A$,I,J).....	Returns J characters, starting with the Ith character, of string A$.
POS(A$,B$).....	Determines the POSition of string B$ in string A$ and returns the POSition number.

These additional String Functions can be implemented by ATARI BASIC String Functions as shown below:

Statement	Printed Results

```
10 DIM A$(10),B$(10)
20 A$="ABCDE":PRINT A$      ..... ABCDE

30 REM ** LEFT$ FUNCTION **
40 LET I=3
50 DIM X$(LEN(A$))
60 X$=A$(1,I):PRINT X$      ....... ABC

70 REM ** RIGHT$ FUNCTION **
80 X$=A$(LEN(A$)-I+1,LEN(A$))
85 PRINT X$      ................... CDE

90 REM ** MID$ FUNCTION **
100 LET J=2
110 X$=A$(I,I+J-1):PRINT X$      ... CD

120 REM ** POS FUNCTION **
130 LET B$="DE":PRINT B$      ...... DE
140 FOR I=1 TO LEN(A$)
150 IF A$(I,I+LEN(B$)-1)=B$ THEN 180
160 NEXT I
170 I=0
180 PRINT I      ..................... 4
```

Lines 50 and 60 implement the LEFT$ function, line 80 performs the RIGHT$ function, line 110 is the MID$ function, and lines 140-180 are the POS function. The above program is not as complex as it looks. All operations are based on previously discussed principles. This program provides a good test to see how much the reader has learned thus far.

String Comparisons

As previously noted, the logical (or Boolean) operators NOT, AND, and OR cannot be applied directly to string variables. However, the following Relational Operators can be applied to string variables:

Relational Operator	Explanation

```
 <   .......   less than
 >   .......   greater than
 =   .......   equal to
<=   .......   less than or equal to
>=   .......   greater than or equal to
<>   .......   not equal to
```

When Relational Operators are applied to strings, the BASIC Interpreter converts the string's characters to ATASCII decimal code numbers and then compares these numbers. Therefore, a character's position in the ATASCII chart (Appendix C of the ATARI BASIC Reference Manual) will indicate its "relation" to any other character.

The execution of the following program will familiarize the reader with relational string comparisons.

```
10 DIM A$(20),B$(20)
20 A$="ABCDE"
30 PRINT "A$=";A$
40 PRINT "WHAT IS B$"
50 INPUT B$
60 PRINT :PRINT
70 PRINT "A$=";A$
80 PRINT "B$=";B$
90 IF A$<>B$ THEN PRINT "A$<>B$"
100 IF A$<B$ THEN PRINT "A$<B$"
110 IF A$=B$ THEN PRINT "A$=B$"
120 IF A$>B$ THEN PRINT "A$>B$"
130 PRINT :PRINT :GOTO 30
```

Line 40 prompts the user to define string B$, and then lines 90-120 apply Relational Operators to strings A$ and B$. The result of the string comparisons are then printed. The user is encouraged to input various string characters for B$, such as; B$="ABCD" and B$="ABCDEF".

Since the decimal codes for the ATASCII alphabet are in numerical order, the Relational Operators are useful for sorting names in alphabetical order. In the above program, if string A$ is set equal to A$="JONES", the user can define (Input) string B$ to be various last names and verify the alphabetical sorting.

Any Relational Operator expression, A$=B$ for example, will return a value of 1 (if the expression is True) or a 0 (if the expression is False). This allows a Logical Operator to be applied to string comparison expressions. Consider the following:

```
10 DIM A$(10),B$(20)
20 A$="ABCD":B$="BBCD"
30 IF NOT (A$=B$) THEN PRINT "A$<>B$"
40 IF NOT A$=B$ THEN PRINT "A$<>B$"
```

The result of the string comparison (A$=B$) in line 30 returns the value 0. Therefore, the expression NOT (A$=B$) is equal to 1 which initiates the print operation. Line 40 illustrates that the string comparison expression need not be contained in parentheses.

String Locations in Memory

The Special Purpose Function ADR(String-name$) permits a programmer to ascertain, and control, where strings are stored in RAM memory. Consider the following statements:

Statement	Printed Results

```
10 DIM A$(22),B$(10)
20 A$="ABCDE"
30 PRINT ADR(A$)          ......... 2164
40 PRINT ADR(B$)          ......... 2186
50 PRINT ADR(A$(3))         ...... 2166
```

Line 30 shows that the block of 22 memory locations, reserved for string A$, starts at location 2164. Line 40 shows that string B$ starts at memory location 2186. Note that: ADR(B$)-ADR(A$)= 'dimensioned length' of A$. The memory location of a single character of a string, ADR(A$(3)) for example, can be obtained by the use of subscripts, as shown in line 50. The use of subscripts with ADR is only legal if the string has been previously defined. Delete line 20 and execute the above program to verify this restriction. The BASIC Memory Management will change the memory locations of strings dependent on the number of statements in a

program. Add the following statements to the above program and note the new values for ADR(A$) and ADR(B$) when the program is executed:

```
60 PRINT
70 PRINT
```

The Special Function ADR is useful if the starting location of a string is needed in a USR (User) machine language subroutine.

There are occasions when the programmer must control the memory location of a string. This can be accomplished with a 'filler string'. Assume that string B$ must start at memory location 3000 and consider the following routine:

Statement **Printed Results**

```
10 DIM A$(1)
20 PRINT ADR(A$)          ......... 2185
30 DIM F$(3000-ADR(A$)-1)
40 DIM B$(10)
50 PRINT ADR(F$)          ......... 2186
60 PRINT ADR(B$)          ......... 3000
```

Line 10 establishes the memory location of the first dimensioned string A$. Line 30 establishes the dimensioned length of the filler string F$. The expression (3000-ADR(A$)-1) sets the length of F$ to 814, so that F$ starts at location 2186 and extends to location 2999. String memory locations are established in the order that the strings are dimensioned. So, in the above example, the filler string F$ had to be dimensioned prior to B$.

Conclusions

Although ATARI BASIC is not the most powerful BASIC available, it is sufficiently flexible to implement all typical string operations. ATARI BASIC is certainly more powerful than the ATARI BASIC Reference Manual indicates. Any reader willing to experiment with string functions and operations, will readily become proficient in programming string manipulations in BASIC. □

EXPLORE A NEW DIMENSION IN SOFTWARE

When you reach the stage where you need more sophisticated software to help you with business and other applications, explore the programming power and flexibility available to you with systems software from OSS.

C/65—Another Powerful First from OSS

C/65 is the first commercially-available C compiler for both the Atari and Apple machines which actually produces assembly language output.

C/65 supports a very usable subset of the extremely powerful and popular C language. Just as C is used by the most sophisticated programmers from the professional and academic communities, so shall C/65 prove to be a powerful and much-needed tool for 6502 software developers.

C/65 supports INTegers and CHARacters, arrays thereof and pointers thereto. Naturally, it also features full recursion, easy assembler interface, #INCLUDE, and a non-macro version of #DEFINE. AUTOmatic, global and EXTERNal variables are also available. When used with our MAC/65 assembler, C/65 is a powerful and flexible tool...$80.00

A Strong Software Family

Other major systems software products from OSS include:

BASIC A+	the only logical upgrade to Atari BASIC with extra features for games and business programs....$80.00
MAC/65	the finest and fastest complete 6502 macro assembler/editor package you can buy....$80.00
TINY C	for structured programming, an easy-to-use interpreter, a learning tool....$99.95
BUG/65	a powerful, self-relocatable debugger. FREE with MAC/65....$34.95

And More...

OS/A+, the first and finest operating system for BOTH Atari and Apple II computers, is NOW included FREE as a part of every OSS systems software package. OS/A+ features a keyboard-driven, easy-to-use command processor, several simple resident commands, and logical and readable requests for even the most sophisticated utility commands. Versions of OS/A+ for some higher capacity drives are available at extra cost.

NOTE: Unless otherwise noted, all OSS products require 48K and at least one disk drive.

ASK YOUR DEALER, or call or write for our brochure.

OSS **Optimized Systems Software, Inc.** 10379 Lansdale Avenue • Cupertino • California • 95014 • (408) 446-3099

ATARI PASCAL - A GOOD PRODUCT?

by Raymond T. Tillman

In March, 1982, after many promises and much delay, ATARI quietly released their ATARI PASCAL Language System. However, they didn't release it through the expected channels. PASCAL was released through the ATARI Program Exchange as unsupported software. Adding to that disappointment, it required two ATARI 810 disk drives. ATARI also said that the program "was not suited for learning PASCAL or the ATARI 800 computer." Furthermore, the absence of a compatible assembler (one which generates relocatable code), limited debugging tools, and incomplete documentation are considered to have reduced the usefulness of ATARI's PASCAL. Finally, only part of the PASCAL language system was released. Since then, almost nothing has been said by ATARI or anyone else about the product.

PASCAL: A Standardized Language

Just what is ATARI's PASCAL? ATARI PASCAL is a powerful, structured programming language built on the core of a real computer language standard: the International Standards Organization's draft standard (DPS/7185). This standard has been adopted for almost all large computer versions of PASCAL. The language system is significantly expanded to provide graphics and sound and to eliminate some of the shortcomings of standard PASCAL. As such, it is a very nice package at an extremely attractive price.

Language Description

The ATARI PASCAL was written for a double density disk drive. In order to run it with the 810 drive and the current ATARI 800 computer, it was necessary to place the files on two separate disks and make some other changes. These changes cause some small problems, especially if compilation is aborted. The compiler expects the system monitor to be on the same disk as the compiler. It isn't; there isn't room for it. Some consider this a drawback and have said that this problem has driven them away. They, like ATARI's top brass, just don't know what they have here.

ATARI PASCAL is a well-written version of the standard language with a select few extensions to the base and a good complement of library routines. It contains all of the standard scalar and structured data types associated with PASCAL, as well as three additional scalars: BYTE, WORD, STRING. It allows absolute variables (defined to a particular memory location), external procedures, modular compilation (similar to UCSD units, but easier to use), bitwise operations, heap management (including pointer variables and garbage collection), and a host of other nice things. It even provides an ELSE on case statements.

My sources at ATARI tell me that only two bugs, both associated with the ABSOLUTE variables, have been reported. Some claim to have found others, but no supporting documentation exists. ATARI also tells me that other parts of the language, including random disk access (segmented files) and possibly a compatible relocatable code assembler have been withheld, presumably because of bugs. Maybe they simply won't run on a 48K ATARI.

Speed and Programming Effort

PASCAL by nature is not as easy to work with as BASIC. You must write your programs with an editor. The PROGRAM TEXT EDITOR was designed for use with the PASCAL and is an excellent product. I use TEXT WIZARD. The ATARI Word Processor and Letter Perfect are probably not compatible. After writing the code you must load the PASCAL interpreter and call up the compiler (switching disks when you do). Then you can sit and wait. The PASCAL compiler is slow and performs three passes. This is not standard for PASCAL but it does allow for an easier, more flexible compilation. After successfully compiling the program you must again switch disks and link the program. This can be tricky. After linking the program modules you may run the program. If it blows up, you may not know why — the debugger was not released.

While the compile and link steps are very time consuming, the run time is normally very fast. A graphics demonstration using player graphics smoothly and quickly scrolls in both vertical and horizontal directions, quite unlike a similar demonstration program written in BASIC.

When I ran a test using an algorithm which generates Archimedes Spiral, I was rather disappointed. It took more than three and a half hours to compute and plot the spiral; longer than ATARI BASIC! This is certainly because of the slowness of the transcendental functions and the required conversions between integer and real numbers. (Strong data typing is maintained in ATARI PASCAL.)

Memory Map

There is not a lot of hard information available about the ATARI PASCAL memory requirements or where each portion of the system resides in

memory. ATARI representatives provided me with a tentative memory map. It is both incomplete and not guaranteed to be accurate.

ATARI PASCAL Language System
Tentative Memory Map

Page Zero	0-007FH
Interpreter	0080-00FFH
Concatenation Buffer	048-057FH
Evaluation Stack	0600-0700H
Interpreter Vector Table	1D00-1F00H
Interpreter Jump Table	1F00-IF80H
Parameter Compiler Comms Area	1F80-2000H
Assorted Program Related Operations & Working Space	2000-9400H
Monitor	9400-BC00H
Screen Area	BC00-BFFFH

Although page zero is used by PASCAL, the main operations, parameter passing and evaluation operations are performed in page six and elsewhere. This can be devastating to those accustomed to using page six for other things. In PASCAL, it is not available to the user.

Machine language modules may be added to PASCAL programs as inline code. These must be fully relocatable. Since parameters are passed on the evaluation stack (page six), programmers must perform the extra task of maintaining that stack when they use machine language subroutines.

Comparisons to Other Versions of PASCAL — UCSD PASCAL

ATARI PASCAL and UCSD PASCAL are very similar. The minor differences which exist between the ATARI PASCAL and the UCSD version should cause few problems; in almost every instance, the ATARI version is more flexible. The major differences between the USCD PASCAL and ATARI's PASCAL is in design philosophy, speed of compilation, and the hardware differences between the ATARI and other microcomputers.

UCSD PASCAL is not simply a computer language, nor does it comply wholly with the ISO draft standard. It is a complete software system including a filer (DOS), compiler, linker, assembler, program editor, and a pseudo-code (P-code) interpreter. These make up the operating system and are not part of the programming language. The language core requires 40K on the APPLE II computer. It includes both RAM and ROM (optional) segments and apparently replaces the APPLE II's base operating system. APPLE PASCAL programmers must learn a new operating system and file manager to use that product. Benchmark tests reported by John Sommer in MICROCOMPUTING magazine (April, 1982) showed that the APPLE II PASCAL was only twice as fast as the APPLESOFT BASIC.

The ATARI PASCAL is smaller, uses the ATARI filemanager and operating system, and is entirely RAM resident. This concept is in full accord with the ISO standard. It appears that the runtime environment may leave as much as 28K for programs. Program compilation is much slower than the UCSD PASCAL and is restricted to about 300 program lines per module but multiple modules are allowed. Linkage of modules is also restricted, but it is possible to chain modules (passing parameters between modules.)

ATARI PASCAL has been shown to execute as much as seven times faster than the APPLE PASCAL in some benchmark tests. But, since ATARI's PASCAL uses the internal floating point routines and BCD REAL numbers, any operation using REAL numbers is terribly slow.

As with the UCSD product, ATARI PASCAL uses a P-code interpreter. However, the P-code has been optimized for the 6502 CPU. Similar compilation and link time error checking schemes are employed. Modular compilation is possible, actually even easier to perform, with the ATARI PASCAL. One significant advantage for the UCSD PASCAL is segmented files (random access files). The ATARI PASCAL segmented files library has not been released, although it exists.

PASCAL/MT+, an ATARI PASCAL Look-Alike

ATARI PASCAL is not alone in its design. A review of the CP/M PASCAL/MT+ compiler marketed by Digital Research is revealing. Comparison of the documentation for the ATARI PASCAL and the PASCAL/MT+ showed that page after page of the language descriptions are identical or nearly so. Examples used are in almost all instances identical. Special functions, not part of the standard PASCAL, are the same. In one case, an odd comment is found in the ATARI document, in the description of the ADDR function, on page 41. It reads, "Output is system dependent." That doesn't make sense in the ATARI manual, but in the CP/M document, it makes a lot of sense, since the PASCAL MT+ is designed to run on a variety of microcomputer systems. It and the ATARI version produce relocatable MICROSOFT compatible code.

Additional surprises come when looking at the system monitor and at descriptions of the runtime link modules. Both products have the same user interface and most of the PASCAL/MT+ link modules have counterparts in the ATARI version. There are differences. The PASCAL/MT+ documentation is more complete and the language is more powerful. It also requires 64K of RAM.

People at ATARI tell me that MT Microsystems, a division of Digital Research, the authors of PASCAL/MT+, wrote ATARI PASCAL. However, it was written for a super ATARI with 128K of RAM — a product which has not seen the light of day, and for the 815 disk drive which has been shelved. ATARI wasn't able to get this super machine to work correctly so it and PASCAL were shelved. Also, the story is that ATARI wanted a UCSD PASCAL, but couldn't get it, so they decided that nothing was better than what they had. Pressure from ATARI owners and from some people within the company caused ATARI to release the PASCAL. But, it was either APX or nothing.

Incompatibilities with ATARI PASCAL

I have learned of a few problems with the ATARI PASCAL and its usefulness with certain third source hardware and software. First, the compiler and linker modules will not work with the Corvus Hard Disk. Also, the RAM DISK is not compatible. You cannot compile a program using the PERCOM Disk Drive in the double density mode or using the modified DOS. However, programs can be compiled in single density and then transferred to double density for execution. The ATR 8000 will work with the PASCAL package exactly in the same way as the PERCOM drives. It is possible that the RAM PAGE by MAXXAM Dataware Corp. will be compatible because the several bytes of memory which it reserves for its use are F7C0-F7C3H, outside the PASCAL usage area.

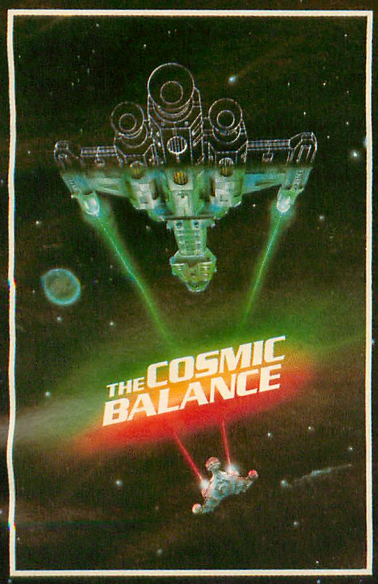
Summary

ATARI PASCAL is an extremely well designed, powerful programming language for the ATARI Personal computer. Although it compiles programs much slower than does USCD PASCAL, it more closely follows the language standard. Execution speed is generally faster.

PASCAL is not getting the support it richly deserves from ATARI. With the limitations placed on it by release of only part of the system, by the total memory available to it, and with the lack of adequate documentation, ATARI PASCAL is greatly handicapped. Still, until people begin using it and start screaming for better documentation, ATARI will not provide any support, nor will the missing parts of the system be released. I for one, think that ATARI PASCAL has the potential to become one of the best, if not the best product available for the ATARI program developer, but only if ATARI's top brass will wake up to its potential. □

The ATARI Pascal Language System can be purchased from the ATARI Program Exchange, P.O. Box 3705, Santa Clara, CA 95055. $49.95 Disk. Order number APX-20102.

THE GALAXY AWAITS YOUR COMMAND.



When SSI introduced THE COSMIC BALANCE™, it was hailed as one of the finest **tactical** space game ever made: It not only gave you starship combat that was fun, fast and furious, it also let you design your ships. You became both starfleet commander and starship architect.

Now we are proud to present its **strategic-level** sequel — THE COSMIC BALANCE II™. It allows all you aspiring Galactic Emperors out there to plot the growth of your space kingdom — from a few, paltry planets to the entire Galaxy! You discover and colonize planets, establish commerce nets, organize production of necessities, and send starships out on missions. There are five scenarios prepared for you, but you are free to create your own.

No matter how you play it, THE COSMIC BALANCE II™ is a game of interstellar conquest. And the only way you're going to enlarge your share of the cosmic pie is to win starship battles against your opponent (which can be a human or the computer).

When actual combat occurs, you can let the computer resolve it instantly. Or you can slug it out in all its blazing glory by using THE COSMIC BALANCE™. The battle outcome can then be incorporated into the strategic game.

Space may be what these games are all about, but there isn't enough of it here to adequately describe them. But why read when the Universe beckons? Plot a course to the nearest computer/game store and get these games today! You have a destiny to fulfill — a destiny that lies out there among the stars.

ON DISC FOR THE APPLE® AND ATARI®.

THE COSMIC BALANCE & THE COSMIC BALANCE II ($39.95 each) are on 48K diskette for the Apple II+ or Apple II with Applesoft ROM Card. Also on 48K disk for the Atari 400/800.

RapidFire
GAMES FROM SSI

Apple is a registered trademark of Apple Computer, Inc.

Atari is a registered trademark of Atari Inc.

If there are no convenient stores near you, VISA and MASTERCARD holders can order direct by calling **800-227-1617, x335 (toll free)**. In California, call 800-772-3545, x335.

To order by mail, send your check to: **Strategic Simulations Inc**, 465 Fairchild Drive, Suite 108, Mountain View, CA 94043. California residents, add 6½% sales tax.

WRITE FOR A FREE COLOR CATALOG OF ALL OUR GAMES.

WHAT THE SOFTWARE REVIEWERS ARE SAYING ABOUT YOUR DOCUMENTATION

by Jessie Gunn Stephens

Software reviewers are, by and large, articulate folk. When a program pleases them, a warm, friendly chuckle seems to reverberate beneath their published prose. But when a program frustrates them, you can practically hear the grinding of their teeth. And the package component most likely to activate that crunching, damning sound against the programs you're trying to market seems to be the user documentation.

Reviewers like to write about documentation, and an informal survey I've been conducting over the past six months indicates that they know a lot, in general terms, about what's wrong with your documentation and what you should do to correct it. If you're interested in producing documentation which will impress reviewers as favorably as your software does, perhaps you'd like to know what they have to say.

Identifying Users and Their Needs

The most serious complaint reviewers lodge against user documents is that they often don't properly identify the readers and set out to meet those readers' needs. Or, do you know *who* is going to want to use your software to accomplish *what* in the real world? If you don't, your user guides won't be able to fill users' needs, because you won't know what they are.

"The guide tells me how the program works," writes the reviewer, "but what I want to know is what I can make it do for me." Only other programmers are likely to have much interest in how your program works. Users want to know how to make it accomplish the specific tasks they need done.

To improve your user documents, take the time to form a clear picture, by research if necessary, or personal interviews with prospective users, of who those people are and what they want your software to do. Then, concentrate on instructing them in how to do those very things. Once you've clearly identified the users and their needs, it's much easier to plan and write documents from the user's perspective, rather than from the programmer's.

Respecting the User

Reviewers become particularly incensed, as I believe most users would, by documentation which fails to take itself seriously. All too often, documentation is a last minute effort, a few pages slapped together just before the package goes out the door. Careless preparation shows; poor organization, inaccurate descriptions, inappropriate humor (at the reader's expense), typographical and spelling errors, and bad reproduction are all marks of the documentor's lack of respect for the user.

If you don't respect your users, don't expect them to respect you or your product.

The corrective for such poorly conceived and executed documents is, of course, time and effort. Read that as "money," if your time and effort are worth anything. But remember, a slipshod product condemns itself in the user's hand.

Honesty in Packaging

Another source of user discontent is the hackwork product which dishonestly disguises itself in plush packaging. You've never heard moral indignation until you've heard a user gripe about being deceived into judging a user guide by its cover. Of course, we all know better than to do

such a naive thing, consciously, at least. But our culture teaches us to consider appearance a valid criterion of quality, in everything from choosing a mate to buying an automobile. Appearances are important in our consumer advertising-oriented society. But no amount of plush packaging can compensate for an inferior product between the covers. And finding trash bound in embossed leather tends to raise reviewer's blood pressure because it leaves them feeling that someone has deliberately tried to hornswoggle them, and they don't like it. Can you blame them?

Of course, no one is suggesting that you shouldn't package your documents as attractively as possible, merely that you ensure that the contents merit their covers. Packaging cannot substitute for quality.

Providing Adequate Examples & Illustrations

Statistically, the complaint which most often crops up in reviews concerns the lack or inadequacy of examples and illustrations. Even the experienced computerist needs examples when learning to operate a new program, and for the computer novice, examples and practice exercises can mean the difference between learning to use your program at all or shelving it with a curse of frustration and swearing never to buy another product with your name on it.

There's no such thing as too many examples, too many "for instance"s, "let's do this together"s, "let me show you how"s, "this is how it will look"s, and "such and such will appear on your screen"s. You don't have to be an artist to provide your readers with an image of their display screens. A simple box will do; they'll catch on to the convention at once. The more you can reinforce prose with illustrations, instructions with examples, and usage with practice exercises, the stronger your documents will be.

Testing Documents

"The manual has lots of examples, but at least half of them didn't work," gripes the reviewer, his tone resentful and tinged with incredulity. The reader and prospective buyer is very aware of the indictment, even if it remains unwritten: "This guy didn't even bother to test out the examples in the user's guide. Think twice before deciding to trust either his document or his software."

A full-out beta test of your user documents is the only way to assure their accuracy and completeness. To make this test mean anything, you must approach it as carefully and with as much integrity as you do the testing, debugging, and retesting of your software.

The purpose of testing software is not to prove that it works, but to discover the places where it doesn't work. Test documents with the same goal in mind, and correct the flaws your testing uncovers just as assiduously as you correct software flaws.

Providing Usage Tools

The best documents are those which aid the reader. They provide information in logical units and in conventional order. Indexes grace them. Tables of contents map them clearly, guiding readers straight to the information they're looking for. They frequently isolate technical or other special material into appendices or even separate manuals. Also, such documents are clearly designed for the specific package version the customer holds in his hand.

Maintaining Documents

If your package exists in more than one version, give some careful thought to the need for more than one version of your user documents. It's not uncommon for an irate reviewer to lambast a package upon finding that the documentation didn't reflect the version of the software he had in hand. If you modify your package through updates available to customers, be sure you provide updates for their documentation, as well.

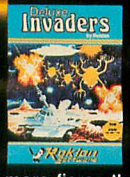
Software reviews appearing in computer magazines do impact your sales. You know that they're written for the most part by computer-literate people, not novice users. Experienced in the use of a wide range of software tools and applications, such writers aren't likely to think themselves dumb, clumsy, or otherwise at fault when they encounter difficulties in the use of your product, particularly difficulties which might have been precluded by more effective documentation. They know what they're talking about. You might do worse than to listen to them. □

FOR THE FUN OF IT

Arcade Excitement for Your Atari® Home Computer

Midway's GORF
"Faithful to the arcade version." —Book of Atari Software 1983
In the dark reaches of hyperspace, confront the fierce Gorfian Empire. Battle Gorfs, Droids, Lasers and Subquark Torpedoes in your attempt to survive. Multi-screen action for one or two players. ROM Cartridge or 24K disk.

Midway's WIZARD OF WOR
"An action-packed shoot-em-up; an outstanding job." —Book of Atari Software 1983
Battle hideous and deadly creatures of doom. Survive the changing mazes and defeat the fiendish Wizard of Wor. Multiple screens, simultaneous one or two-player action. ROM Cartridge or 32K disk.



DELUXE INVADERS
"By far the best Space Invaders program ever released for a personal computer." —Electronic Games
Be warned! The Invaders have broken arcade boundaries to bring their awesome challenge to your home computer. The better your defense, the more fierce their assault. Nine difficulty levels. One or two-player action. ROM Cartridge or 16K disk.

ANTI-SUB PATROL
NEW! Tactical Search and Destroy Mission
Anti-Sub Patrol puts you in command of a squadron of sub-hunting destroyer escorts. Your mission—rid the seas of two subs whose single goal is to blast you from the waters. Stand by for suspense! 32K disk, 24K cassette.

COMING SOON FOR EVEN MORE FUN: Da Fuzz™, Lifespan™, Rockball™, Eyes™, and Castle Hassle.

Ask for Roklan Software at leading software dealers nationwide.

Roklan Software
3335 N. Arlington Heights Road
Arlington Heights, IL 60004

Experience the MicroProse Challenge!!!

Now On Cassette!

CHOPPER RESCUE

Zip up your flying suit and strap yourself into your sleek assault chopper: CHOPPER RESCUE challenges your skill and daring. Pilot your nimble craft through the treacherous underground labyrinth. Search out and rescue the captives trapped inside. Danger approaches from all sides as defenders fire winged missiles and sparkling energy bursts at your fragile craft. But you are not defenseless, your explosive tipped rapid-fire missiles dart out to blast a path through the dense minefields. Thread your bomb directly into the heart of their pulsating nuclear furnace. And just when you think the path is clear, a swift patrol drone appears from behind a ledge, spitting missles as you scramble for a firing position! CHOPPER RESCUE features three scenarios, multiple skill levels, and competition in solo or team configurations for 1 to 4 players. Immerse yourself in the non-stop excitement of CHOPPER RESCUE. You'll keep coming back for more!

Requires ATARI® 32K, Disk or Cassette . . . $29.95

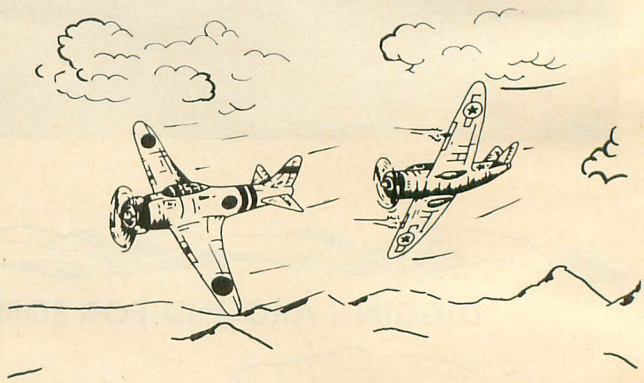
HELLCAT ACE

HELLCAT ACE is a superbly realistic re-creation of WWII Air-to-Air combat in the Pacific using a fully three-dimensional airspace. HELLCAT ACE puts you in the fighter cockpit with its unique pilot's eye view. To escape and destroy the tenacious enemy fighters, HELLCAT ACE allows you a full range of high-performance aerobatics; including loops, stalls, rolls, immellman turns, and spins. The fifteen well documented scenarios include combat air patrol, night action, and kamikazes --- and, if things get out of hand, you can bail out or attempt a water landing in your damaged fighter! HELLCAT ACE includes multiple skill levels and options for up to four players. Can you avoid being shot down and win five victories to become a HELLCAT ACE?

Requires ATARI® 40K, Disk or 32K Cassette . . . $29.95

For aerial combat challenge and excitement over Europe in WWII fly the SPITFIRE ACE!

Requires ATARI® 40K Disk or 32K Cassette . . . $29.95



FLOYD OF THE JUNGLE

Looking for a fun game you can play with your family and friends? FLOYD OF THE JUNGLE is a great game up to four people can play simultaneously! Can you be the first to guide Floyd through the jungle perils to save the lovely Janice? It isn't easy! Floyd must leap the poisonous snakes, climb the slippery vines, ride the elephants, and cross the raging rivers. FLOYD OF THE JUNGLE includes fully-animated wild animals, music, sound effects, and five different multi-color jungle adventures.

Requires ATARI® 32K, Disk or Cassette . . . $29.95

MicroProse Software is dedicated to bring you games that will challenge you far beyond the first few plays. We promise you hours of excitement and pleasure. Our games are created using "MicroProse", our own proprietary assembly language gaming system, and are available at select computer stores.

PHONE: (301) 357-4739

For Fast Delivery by C.O.D.,
MasterCard or Visa . . . Or
Send Check or Money Order . . .
Please add $2.50 shipping/handling.
Maryland residents add 5% tax.

MicroProse Software
One Caribou Court
Parkton, Maryland 21120

Dealer Inquiries Welcome!

48K

Expand your ATARI 400 16K to a full 48K by using our 48400 48K Ram upgrade expansion kit !!!!!*

*** Expand your 16K Ram board to 48K for the price you would pay for most 32K Ram expansions!!!!

***Easy to follow step by step installation instructions with detailed drawings.

****

for only $99.95

Free shipping in USA

SEND CHECK,MONEYORDER, or BANK CERTI-CHECK

allow 4-6 wks for per. check.

USA POLLY PRODUCTS TM

MAIL TO :
POLLY PRODUCTS
of Riverside
P.O. Box 8485
Riverside, California
92515-8485

Ca. Res. add 6% TAX.

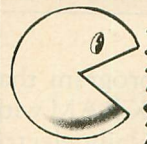
ATARI 400* is the trademark of **ATARI, INC.**

COMPUTER STORE

SOFTWARE GALLERY®

P.O. BOX 3354, CHERRY HILL, N.J. 08034
(609) 424-7106

ATARI 810 DISK DRIVE $ CALL $
ATARI 800 48K.....$ CALL $
ATARI 1200 64K.....$ CALL $



If you like me you will LOVE Defender

Defender
Pac-Man
Galaxian
Centipede
$32.95 ea.

ATARI®

Large selection of software for Atari 400/800 Call

INTEC RAM BOARD FOR ATARI 400/800
32K Ram Board ....$67.95 48K Ram Board $99.95
64K Ram Board $124.95

B KEY 400 KEYBOARD $94

EPSON PRINTERS $ CALL $

PHONE ORDERS CALL (609) 424-7106 FOR IMMEDIATE COD, SHIPMENT

VISA

CHARGE IT WITH YOUR VISA OR MASTER CARD
Add 3% for Master Card and Visa

MasterCard

THANK YOU FOR YOUR ORDER!

Atari® Users!!

NOW AVAILABLE...

MIGHTY BYTE™ Disk Copier. A backup copier that allows you to select sectors, identify and write bad format. Copy almost any disk. The most powerful copy system available today on Atari® 400/800 computers.

Only $29.95 + $2 shipping

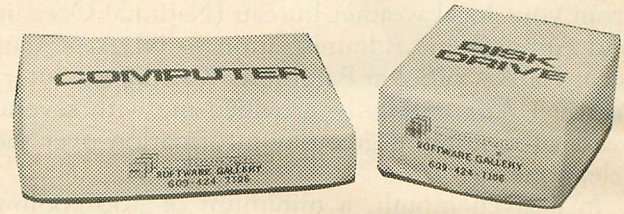
VISA & MASTERCARD call now
(815) 968-2228 for fast delivery

Software Authors and Users: call or write for further information on our complete line of Atari® compatible backup systems and services.

Can't slow that new drive below 250rpm?
We modify your drive $10.00
You modify with our parts $5.00

MIGHTY BYTE COMPUTER
828 Green Meadow Ave., Dept. AG
Rockford, Illinois 61107

PROTECT YOUR COMPUTER



MasterCard VISA **special $8.99**

COMPUTER COVER

Features heavy duty canvas with vinyl interior-waterproof.

ATARI-800/AC-8	$8.99	*EPSON/MX-80*	$8.99
ATARI-400/AC-4	$8.99	*EPSON/MX-80FT*	$8.99
ATARI-810/AD-D	$8.99	*IBM PC PRINTER*	$8.99
		OKIDATA/μ82A	$8.99

Mail Order
☆
SOFTWARE GALLERY
P.O. Box 3354
Cherry Hill, N.J. 08034
Call: (609) 424-7106
Add $2 for postage and handling.
N.J. residents add 5% sales tax.

Dealer & Distributor
Universal
TRAV-LER INC.
359 Wales Ave.
New York, N.Y. 10454
Call: (212) 993-7100

NOTE: Atari, Epson, IBM and Okidata are registered trademarks.

HOME ELECTRICITY CONSUMPTION ANALYSIS

16K cassette 24K disk

by Joseph E. Harb Jr.

"Kilowatts" is an ATARI BASIC program that requires 16K RAM with cassette or 24K RAM with disk. It analyzes yearly, monthly, and daily electricity consumption and cost for all-electric homes.

When we moved into our present house several years ago, I planned to make a number of energy conservation modifications. I decided that I would like to use my ATARI 800 to determine what impact those modifications had on our energy consumption and costs. That led to the writing of "Kilowatts," which makes provisions for yearly and monthly temperature fluctuations. Statistics generated by "Kilowatts" can be displayed on the screen or printed on a line printer.

Monthly and yearly temperature variations are taken into consideration by analyzing kilowatt consumption per cooling/heating degree day, as appropriate. A heating degree day is each degree that the average temperature drops below 65 degrees F. on a given day. A cooling degree day is each degree above 65 degrees F. The total number of cooling and heating degree days in each month can be obtained from your local weather bureau (National Oceanic and Atmospheric Administration — NDAA). Our local NDAA office at Baltimore Washington International Airport kindly provided me with several years of monthly degree day information over the telephone.

In a given month, a minimum of 100 cooling degree days is required before the program will calculate cooling degree day consumption for that month. A minimum of 200 heating degree days is required for heating degree day analysis. This was done because in months when the number of heating or cooling days is below the threshold, energy use for heating or cooling is so low that the data becomes heavily biased by other energy use. This bias makes it seem that consumption per degree day is abnormally high. To change the threshold for cooling degree days, change the value of MINCD in line 100. To change the threshold of heating degree days, change the value of MINHD on the same line.

In order to further minimize distortion by electricity consumption for uses other than heating and cooling, the program subtracts 400 kilowatts from each month's total electricity use before computing consumption per degree day. (This subtraction is not performed in computing any other statistics.) The variable used in the subtraction is FCTR, also in lines 100/110. It can be changed if you feel your nonheating/cooling electricity use is higher or lower.

All REM statements can be eliminated without requiring any line number changes. Additionally, if you feel the explanation of DATA statements given in the following paragraph is adequate, you can eliminate the instruction subroutine (line 2050 and 6999-7190). If you do not have a printer, you can also remove the printer subroutines (lines 2040 and 5999-6880).

One DATA line is required for each month of data. DATA lines must be numbered in increments of 1, beginning with line 1000; e.g.,

```
1000 DATA JAN,79,1329,29,56.10,30,29.88,984,0
1001 DATA FEB,79,1426,28,60.44,32,31.44,1100,0
1002 DATA MAR,79,520,31,50.98,11,20.33,520,15
```

DATA statements must contain: month (first three letters); year (last two digits); number of kilowatts used; number of days in billing period; cost of electricity (paid on time and including surcharges); heating degree days; and cooling degree days. All of the required information except heating and cooling degree days can be obtained from utility bills. As explained above, the information on heating and cooling degree days can be obtained from your local NOAA office.

If you have been looking for a relatively quick and easy way of neatly aligning columns of figures, particularly those with decimal fractions, you might want to consider using the technique I employed in this program, for example in lines 3170-3190. It can be done in four easy steps:

1. Decide the rightmost column for displaying a particular set of figures. Then add 1 to that value. In subroutine 3000, I wanted the last digit of the variable X to be printed in column 11. I then added 1 to that number, for a total of 12. If you are aligning figures with decimal fractions, use the column where the decimal point is to be printed, and do not add 1.

2. Measure the length of the variable by converting it to a string and using the LEN function. In line 3170, LEN(STR$(INT(X))) means calculate the

length (LEN) of the variable X after converting it to an integer (INT) and then to a string (STR$). The variable must be converted to a string because the LEN function can only measure the length of string variables. For this measurement, it is important to convert a numeric variable to an integer when the variable includes a decimal fraction. This is necessary because the ATARI eliminates final zeros after the decimal point. Thus, 3.50 is displayed as 3.5. Consequently, if you wished to align the numbers 3.5 and 4.27 and if you measured the whole length of the variable, the columnar alignment of the numbers would be:

3.5
4.27

3. Pick a variable name for the column where printing of the display variable is to begin. (I used CL1 in the example.) Then, use the algorithm in this paragraph to calculate the column where printing is to begin. The algorithm subtracts the length of the integer portion of the string from the value calculated in step 1. In other words, the column where printing is to begin equals the length of the integer portion of the variable subtracted from the column where printing is to end. That is expressed in BASIC as CL1=12-LEN(STR$(INT(X))). This means that the first digit of the variable X will be displayed at screen column 12 minus the length of the integer X.

4. Position the cursor at the column and row where printing is to begin. This is done with the POSITION statement. In line 3180, the cursor is positioned at column CL1,row PEEK(84). PEEK(84) is the memory location of the current cursor row. Finally, use the PRINT statement to display the variable on the screen. Once you get used to this process, it can be done fairly quickly. Of course, it can be further simplified by performing the whole operation at one time:

```
POSITION 12-LEN(STR$(INT(X))),PEEK(84):?X
```

During operation of "Kilowatts" do not depress the return key at any time when responding to a screen prompt. Simply type the letter(s) or numbers desired for input. The GET statement will determine which key(s) you depressed.□

VARIABLES USED IN "KILOWATTS"

A: Used with GET to determine last key depressed on keyboard.

ANET: Used to represent electricity cost (NET) whenever single subroutine must calculate either gas or electricity statistics.

AVG: Per kilowatt cost.

B: Used with A when more than one key input from keyboard is required.

C: Used with A & B when three-key input required from keyboard.

CAVG: Average monthly consumption of kilowatts per degree day.

CD: Cooling degree days in a given month.

CDAVG: Average annual consumption of kilowatts per cooling degree day.

CDDIV: Total number of kilowatts used when computing annual average consumption of kilowatts per cooling degree day.

CDTOT: Total number of cooling degree days per annum.

CL1: (Column 1); Column where printing of specified data begins. Used to right justify screen display.

CL2: (Column 2); Used with CL1 when more than 1 column cannot be right justified in some other way.

CL3: (Column 3); Used with CL1 & CL2 when more than two columns cannot be right justified in some other way.

CL4: (Column 4); Used with CL1, CL2, & CL3 when more than three columns cannot be right justified in some other way.

COST: Total annual cost of electricity.

DAYS: Number of days during billing period.

DD: Used to represent either cooling or heating degree days in subroutines where either can be used.

DDAVG: Average annual use of kilowatts per cooling or heating degree day.

DDN$: Used in subroutines 3000, 5000, & 6000 to represent words "HEAT" or "COOL" in column headings, depending on whether user has requested cooling or heating degree day information.

DDT: Total number of heating/cooling degree days in a given year.

DIV: Total number of energy units used when computing annual average consumption per degree day.

FCTR: Estimated minimum amount of electricity used monthly for uses other than heating or cooling. Subtracted from UNITS before computing consumption per degree day. Can be raised or lowered if estimated minimum is different.

HAVG: Average monthly consumption of kilowatts per heating degree day.

HD: Number of heating degree days in a given month.

HDAVG: Average annual consumption of kilowatts per heating degree day.

HDDIV: Total number of kilowatts used when computing average annual consumption of kilowatts per heating degree day.

HDTOT: Total number of heating degree days per annum.

HIYR: High year in data base.

HL: Nr. of lines to be printed on each page.

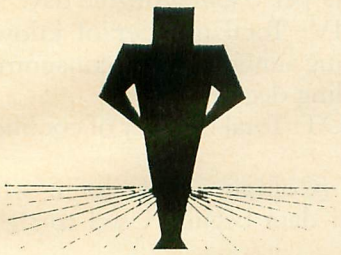
K$: Month for which data requested in menu options A, B, E, & F.

KPD: Average number of kilowatts per degree day.

KPD$: Used to represent either variable KPD or letters "N/A" when printing out results of kilowatts

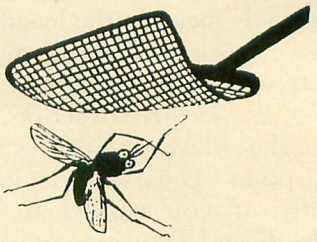
NEW FOR ATARI FROM MMG MICRO SOFTWARE

NOW — THE TWO MOST POWERFUL AIDS FOR ATARI BASIC!!



BASIC COMMANDER

- **Single key entry file commands**
  - — ENTER "D:
  - — LIST "D:
  - — LOAD "D:
  - — SAVE "D:
  - — RUN "D:
- **Single Key DOS functions from BASIC**
  - — FORMAT a disk
  - — RENAME a file
  - — DELETE a file
  - — LOCK a file
  - — UNLOCK a file
  - — DISK DIRECTORY
- **THREE PROGRAMMABLE KEYS!!**
  - — single keys programmed for your own use, even whole subroutines
- **AUTONUMBER**
  - — automatically generates line numbers for you — speeds program entry 25-75%
- **BLOCK DELETE**
  - — deletes any range of lines instantaneously!
- **RENUMBER**
  - — renumbers lines and all references
  - — extensive error trapping
  - — 3 seconds to renumber 500 lines



MMG BASIC DEBUGGER

- **TRACE through your Basic program**
  - — Single step
  - — TRACE while
  - — TRACE UNTIL
  - — change variables
  - — LIST line numbers executed
  - — examine variables' values
- **Full screen BASIC editing**
  - — scroll up or down by cursor
  - — edit your whole program easily
  - — no more LIST line number ranges
- **Split screen mode**
  - — view two parts of your BASIC program at once, and edit both!
  - — scroll each window independently
- **CROSS REFERENCE**
  - — provides a list of variables and the line numbers in which they are used in your program
- **SEARCH FOR PHRASE**
  - — search your BASIC program for any phrase, command or string of characters; let your computer do the searching for you!

EACH PRODUCT ALONE REQUIRES 16K, AND IS AVAILABLE ON DISK FOR *ONLY $34.95*

Now, the convenience of both powerful utilities together in your Atari *at once*

The Combined Basic Commander and MMG Basic Debugger requires 24K. $74.95

ADDITIONAL PRODUCTS

NECESSITIES

RAM TEST II - The fastest and most thorough memory test available for the ATARI has now been further improved! Tests not only all locations, but also tests the memory addressing system. This all machine language program takes 4 minutes to test 48K. It's the only program that tests the cartridge area of RAM. Good for new 400/800 computer owners, for testing new RAM boards and for use in computer stores to test and pinpoint bad memory locations. Bad memory locations are pinpointed so repair is as simple as replacing a chip!

Requires 8K, Disk or Cassette **$29.95**

DISK COMMANDER II - Just save this program on your BASIC disks and it will autoboat and automatically list all programs for the disk into your screen. Simply run any program by typing a single number.

Requires 16K, Disk Only **$29.95**

TUTORIALS

ASTEROID MINERS - This 50 page book and program provides for a unique intermediate to advanced tutorial. A 32K BASIC game utilizing over 25 players in player-missle graphics, machine language subroutines, a redefined character set, multiprocessing utilizing the vertical blank interrupt interval, and much more! The 50 page book included with the program documents each part of the entire program and contains the fully documented source code for both the BASIC and assembly language parts of the program. Use these routines in your own programs. These examples make it easy!

Requires 32K, Disk or Cassette **$34.95**

GAMES

CHOMPER - An all machine language arcade style game with intelligent monsters. Requires 16K Ram, 1 Joystick and nerves of steel.

Available on Disk or Cassette **$29.95**

BUSINESS/HOME

MAILING LIST - Extremely fast BASIC and machine language program. Each data disk holds over 500 files. Sort on any of 6 fields at machine langugage speed or search on any fragment of a field! Use any size labels or envelopes.

Require 40K, Disk Only **$39.95**

NEW

MMG DATA MANAGER - If you frequently find yourself looking for something, only to find it eventually right under your nose, then MMG DATA MANAGER is for you. Organize virtually anything into a computer-searchable format, and let your ATARI do the hunting for you. MMG DATA MANAGER is the first of a series of business applications from MMG MICRO SOFTWARE, all of which will share the ability to access files created by any of them. This flexible database manager will allow many fields, with machine language sorting, on any field. In addition, you have total control of the structure of your data, allowing you to design a database which you feel most comfortable. A special feature of MMG DATA MANAGER is its ability to select for a given value of any single field, or any combination of values from many fields. You could, for instance, determine who lived in Las Vegas, Nevada, and bought item #3145 from you, and whose last name began with SM, and whose telephone number began with (702) 873-4. You'll never lose track of information again! Multiple print options add to the versatility of MMG DATA MANAGER.

Requires 40K, Disk Only **$49.95**

Available At Your Favorite Computer Store

OR Send a Check or Money Order to:

MMG MICRO SOFTWARE

P.O. Box 131 • Marlboro, NJ 07746

OR CALL

(201) 431-3472

For MasterCard, Visa or COD Deliveries

(Please Add $3.00 For Postage & Handling)

NJ RESIDENTS ADD 5% FOR SALES TAX

per degree day computation.

LINE: Last line of DATA.

LOYR: Lowest year of data in data base.

M$: Month of data contained in DATA line.

MINCD: Minimum number of cooling degree days necessary for computing electricity consumption per cooling degree day.

MINHD: Minimum number of heating degree days necessary for computing electricity consumption per degree day.

NET: Cost of electricity without late charge.

NR: Used to calculate number of months in data base.

PRNT$: One PRNT$ string is created for each line of data to be printed with the line printer in subroutine 6000. Allows data to be aligned easily in columns without using TAB functions which vary from printer to printer.

R$: Represents month in subroutine 6460/6570 to compare same month of different years.

SET: Sets flag when high line of page print reached during loop.

T: A flag set at beginning of subroutines 3000 and 5000 to identify whether user has requested information on consumption per cooling or heating degree day.

TIME: Last line printed on printer.

UNITS: Kilowatts used during billing period.

UP: Average daily kilowatt consumption.

USE: Total annual consumption of electricity.

Y: Year of data on DATA line.

YR: Year of data being processed.

Z: Index variable for loops, i.e., keeps track of nr. of times loop has occurred.

```
10 POKE 82,0
20 ? "K↓                    KILOWATTS"
30 ? "                   ELECTRICITY"
40 ? "                ANALYSIS PROGRAM"
50 ? "                  BY JOE HARB"
60 ? "↓↓DURING OPERATION OF THIS PROGR
AM,       DO NOT DEPRESS RETURN KEY AF
TER TYPING  ANSWERS TO PROMPTS."
70 OPEN #1,4,0,"K:":REM OPEN KEYBOARD
TO GET INPUTS LATER IN PROGRAM  WHEN G
ET STATEMENT IS USED
80 ? :? "DEPRESS ANY KEY TO CONTINUE."
:GET #1,A
90 DIM DDN$(4),M$(3),K$(3),KPD$(6),PRN
T$(65),R$(3)
100 MINCD=100:MINHD=200:FCTR=400:REM M
INCD=MINIMUM COOLING DAYS NECESSARY FO
RCOMPUTATION
110 REM MINHD=MINIMUM HEATING DEGREE D
AYS NECESSARY
120 REM FCTR=NR. OF KILOWATTS TO BE SU
BTRACTED FROM MONTHLY KILOWATT USE WHE
N COMPUTING DEGREE DAYS.
130 REM SUBTRACTING FCTR REDUCES EXTEN
T TO WHICH OTHER HOUSEHOLD ELECTRICITY
 USE BIASES HEATING AND COOLING STATS
200 REM CALCULATE: TOTAL MONTHS OF DAT
A (NR); LOW YEAR OF DATA (LOYR); AND H
IGH YEAR OF DATA (HIYR)
210 NR=0
220 READ M$,Y,UNITS,DAYS,NET,HD,CD
230 REM M$=MONTH, Y=YEAR, UNITS=KILOWA
TTS USED USED IN BILLING PERIOD
240 REM DAYS=NR. OF DAYS IN BILLING PE
RIOD
250 REM NET=COST OF ELECTRICITY WHEN B
ILL PAID ON TIME,DD=DEGREE DAYS DURING
 BILLING MONTH
260 REM HD=HEATING DEGREE DAYS
270 REM CD=COOLING DEGREE DAYS
280 LOYR=Y:RESTORE
290 READ M$,Y,UNITS,DAYS,NET,HD,CD
300 IF M$="END" THEN RESTORE :GOTO 200
0
310 NR=NR+1:HIYR=Y
320 GOTO 290
498 REM SUBROUTINE TO GET INPUT FOR ME
NU OPTIONS A & B; THEN CLEAR INPUT QUE
STIONS FROM SCREEN TO ALLOW DISPLAY
499 REM OF ADDITIONAL DATA
500 ? "KTYPE FIRST THREE LETTERS OF MO
NTH YOU   WANT.":GET #1,A:GET #1,B:GET
 #1,C
510 REM NEXT LINE CONVERTS ATASCI VALU
ES TYPED ON KEYBOARD TO A STRING
520 K$=CHR$(A):K$(LEN(K$)+1)=CHR$(B):K
$(LEN(K$)+1)=CHR$(C):GOSUB 530:RETURN
530 POKE 84,PEEK(84)-2:FOR Z=0 TO 1:?
"
  ":NEXT Z:REM 39 SPACES
540 POKE 84,PEEK(84)-2:RETURN
1000 REM YOUR DATA STATEMENTS GO HERE
1899 DATA END,999,0,0,0,0,0
1999 REM MENU OPTIONS
2000 ? "KTHIS PROGRAM ALLOWS THE FOLLO
WING       SELECTIONS:":?
2010 ? "     A. TOTAL MONTHLY AND AVER
AGE DAILY        KILOWATT USE"
2020 ? "     B. TOTAL MONTHLY AND AVER
AGE DAILY        KILOWATT COST"
2030 ? "     C. TOTAL ANNUAL KILOWATT
USE AND          COST"
2040 ? "     D. PRINTOUT OF ALL ELECTR
ICAL USE         DATA"
2050 ? "     E. DATA INPUT INSTRUCTION
S"
2060 ? "     F. EXIT PROGRAM":?
2070 ? "TYPE LETTER OF OPTION YOU WANT
.":GET #1,A
2080 REM GET IS USED TO DETERMINE LETT
ER TYPED ON KEYBOARD; A=ATASCI VALUE O
F LETTER TYPED
2090 IF A=65 THEN 3000
2100 IF A=66 THEN 4000
2110 IF A=67 THEN 5000
2120 IF A=68 THEN TRAP 2160:OPEN #4,8,
0,"P:":TRAP 10000:GOTO 6000
2130 IF A=69 THEN 7000
2140 IF A=70 THEN POKE 82,2:END
2150 GOTO 2070
2160 REM PRINTER ERROR MESSAGE
2170 CLOSE #4:? "PRINTER IS NOT ON-LIN
E!":TRAP 10000:GOTO 2070
2999 REM SUBROUTINE FOR MENU OPTION A
3000 GOSUB 500
3010 ? "DO YOU WANT TO INCLUDE INFORMA
TION ON   HEATING(H), COOLING(C) OR NE
ITHER(N)?":GET #1,A:GOSUB 530
3020 IF A=67 THEN DDN$="COOL":T=0
3030 IF A=72 THEN DDN$="HEAT":T=1
3040 IF A=78 THEN DDN$="    ":T=2:DD=0
3050 ? "MONTH   TOTAL AVG     TOTAL  A
VG KWU"
3060 ? "         KWU   DAILY   DGREE  P
ER ";DDN$:REM 8 SPACES BEFORE KWU
3070 ? "                KWU USE DAYS   D
GREE DAY":REM 14 SPACES BEFORE KWU
3080 FOR Z=1 TO NR
3090 READ M$,Y,UNITS,DAYS,NET,HD,CD
3100 IF M$<>K$ THEN 3200
3110 UP=INT(100*UNITS/DAYS)/100:REM CO
MPUTE UNITS PER DAY AND LIMIT DECIMAL
PLACES DISPLAYED
3120 IF T=0 THEN DD=CD:IF CD>MINCD THE
N GOTO 3150
3130 IF T=1 THEN DD=HD:IF HD>MINHD THE
N GOTO 3150
3140 IF T=2 OR CD<=MINCD OR HD<=MINHD
THEN KPD$="N/A":GOTO 3170
3150 KPD=INT(100*(UNITS-FCTR)/((DD/30)
*DAYS))/100:KPD$=STR$(KPD):REM COMPUTE
 UNITS PER DEGREE DAY
3160 REM LINES 3170-3190 USED TO ALIGN
```

```
   AND PRINT SCREEN DISPLAY
3170 CL1=12-LEN(STR$(INT(UNITS))):CL2=
17-LEN(STR$(INT(UP))):CL3=26-LEN(STR$(
DD)):CL4=32-LEN(STR$(INT(KPD)))
3180 ? M$;" ";Y;:POSITION CL1,PEEK(84)
:? UNITS;:POSITION CL2,PEEK(84):? UP;:
POSITION CL3,PEEK(84):? DD;
3190 POSITION CL4,PEEK(84):? KPD$
3200 NEXT Z:RESTORE
3210 ? :? "DO YOU WANT TO LOOK AT ANOT
HER MONTH?   TYPE Y OR N.":GET #1,A
3220 IF A=89 THEN GOSUB 530:GOSUB 500:
GOTO 3080
3230 ? "K":GOTO 2000
3999 REM SUBROUTINE FOR MENU OPTION B
4000 GOSUB 500
4010 ? "MONTH    TOTAL   TOTAL    UNIT"
4020 ? "         KWU     COST     COST"
:REM 9 SPACES BEFORE KWU
4030 FOR Z=1 TO NR
4040 READ M$,Y,UNITS,DAYS,NET,HD,CD
4050 IF M$<>K$ THEN 4090
4060 AVG=INT(10000*(NET/UNITS))/10000:
REM CALCULATE AVERAGE DAILY USE
4070 CL1=13-LEN(STR$(INT(UNITS))):CL2=
19-LEN(STR$(INT(NET)))
4080 ? M$;" ";Y;:POSITION CL1,PEEK(84)
:? UNITS;"   ";:POSITION CL2,PEEK(84):
? NET;:POSITION 25,PEEK(84):? AVG
4090 NEXT Z:RESTORE
4100 ? :? "DO YOU WANT TO LOOK AT ANOT
HER MONTH?   TYPE Y OR N.":GET #1,A
4110 IF A=89 THEN GOSUB 530:GOSUB 500:
GOTO 4030
4120 ? "K":GOTO 2000
4999 REM SUBROUTINE FOR MENU OPTION C
5000 ? "KDO YOU WANT TO INCLUDE INFORM
ATION ON   HEATING (H) OR COOLING (C)?
":GET #1,A:YR=LOYR
5010 IF A=67 THEN DDN$="COOL":T=0
5020 IF A=72 THEN DDN$="HEAT":T=1
5030 ? "                    ";DDN$;"
   AVG KWATT":REM 22 SPACES BEFORE DDN
$
5040 ? "YEAR  KWATTS          DGREE  P
ER DGREE":REM 10 SPACES BEFORE DGREE
5050 ? "      USED    COST    DAYS   D
AY"
5060 USE=0:COST=0:DDT=0:DIV=0
5070 FOR Z=1 TO NR
5080 READ M$,Y,UNITS,DAYS,NET,HD,CD
5090 IF Y<>YR THEN 5140
5100 IF T=0 THEN DD=CD:IF CD<=MINCD TH
EN DD=0:GOTO 5130
5110 IF T=1 THEN DD=HD:IF HD<=MINHD TH
EN DD=0:GOTO 5130
5120 DDT=DDT+DD:DIV=DIV+UNITS-FCTR
5130 USE=USE+UNITS:COST=COST+NET
5140 NEXT Z:RESTORE
5150 DDAVG=0:IF DDT>0 THEN DDAVG=INT(1
00*DIV/DDT)/100
5160 CL1=17-LEN(STR$(INT(COST))):CL2=2
6-LEN(STR$(INT(DDT))):CL3=30-LEN(STR$(
INT(DDAVG)))
5170 ? YR+1900;"  ";USE;"   ";:POSITIO
N CL1,PEEK(84):? COST;:POSITION CL2,PE
EK(84):? DDT;:POSITION CL3,PEEK(84)
5180 ? DDAVG
5190 YR=YR+1:IF YR<HIYR+1 THEN 5060
5200 RESTORE
5210 ? :? "DEPRESS ANY KEY TO RETURN T
O MENU.":GET #1,A
5220 GOTO 2000
5999 REM SUBROUTINE FOR MENU OPTION D
6000 TIME=0:SET=0:? "KTYPE NUMBER OF L
INES PER PAGE TO BE     PRINTED.":GET
#1,A:GET #1,B:HL=((A-48)*10)+(B-48)
6010 LPRINT CHR$(27);CHR$(56):REM DISA
BLE EPSON "END OF PAPER" FUNCTION
6020 ? #4;"                              TOT
AL   AVG      TOTAL   AVG":REM 24 SPACES
 BEFORE TOTAL
6030 ? #4;"                             HEA
T    KWATT    COOL    KWATT":REM 24 SPAC
ES BEFORE HEAT
6040 ? #4;"         KWATTS   TOTAL    DGR
     PER      DGR     PER":REM 8 SPACES
BEFORE KWATT
6050 ? #4;"YEAR     USED     COST     DAY
S     DGR DAY DAYS    DGR DAY":LPRINT
6060 TIME=TIME+5:YR=LOYR
6070 USE=0:COST=0:CDDIV=0:CDTOT=0:HDDI
V=0:HDTOT=0:CDAVG=0:HDAVG=0
6080 REM CDDIV & HDDIV ARE NUMBER OF A
NNUAL KILOWATTS FOR HEATING & COOLING.
 ONLY MONTHS WITH MORE THAN 100
6090 REM COOLING OR 200 HEATING DEGREE
DAYS ARE INCLUDED. 500 KWATTS PER MONT
H SUBTRACTED BY FCTR FOR OTHER ELECT.
6100 REM CDTOT & HDTOT ARE TOTAL HEATI
NG/COOLING DEGREES PER ANNUM FROM MONT
HS WITH SUFFICIENT DEGREE DAYS
6110 PRNT$="

 ":REM 65 SPACES
6120 FOR Z=1 TO NR:REM CALCULATE ANNUA
L CONSUMPTION AND COST
6130 READ M$,Y,UNITS,DAYS,NET,HD,CD
6140 IF Y<>YR THEN 6180
6150 IF CD>MINCD THEN CDTOT=CDTOT+CD:C
DDIV=CDDIV+UNITS-FCTR
6160 IF HD>MINHD THEN HDTOT=HDTOT+HD:H
DDIV=HDDIV+UNITS-FCTR
6170 USE=USE+UNITS:COST=COST+NET
6180 NEXT Z:RESTORE
6190 IF CDTOT>0 THEN CDAVG=INT(100*CDD
IV/CDTOT)/100
6200 IF HDTOT>0 THEN HDAVG=INT(100*HDD
IV/HDTOT)/100
6210 ? #4;YR+1900;:PRNT$(11-LEN(STR$(U
SE)),10)=STR$(USE)
6220 PRNT$(16-LEN(STR$(INT(COST))),18)
=STR$(COST)
6230 PRNT$(25-LEN(STR$(HDTOT)),24)=STR
$(HDTOT)
6240 PRNT$(30-LEN(STR$(INT(HDAVG))),32
)=STR$(HDAVG)
6250 PRNT$(41-LEN(STR$(CDTOT)),40)=STR
$(CDTOT)
6260 PRNT$(45-LEN(STR$(INT(CDAVG))),47
)=STR$(CDAVG)
6270 ? #4;PRNT$:TIME=TIME+1
6280 YR=YR+1:IF YR<HIYR+1 THEN 6070
6290 RESTORE :LPRINT :TIME=TIME+1
6399 REM CALCULATE AND PRINT MONTHLY
DATA. SUBROUTINE 6410 PRINTS COLUMN
HEADINGS ON EACH SHEET OF PAPER
6400 GOSUB 6410:GOTO 6460
6410 ? #4;"
                        KWAT            KWAT
":TIME=TIME+1:REM 47 & 10 SPACES
6420 ? #4;"MONTH    DAILY    MNTHLY   MNT
HLY   COST     HEAT    PER     COOL    PER"
:TIME=TIME+1
6430 ? #4;"          KWATT    KWATT    COS
T     PER      DGRE    DGRE    DGRE    DGRE
":TIME=TIME+1:REM 8 SPACES BE4 KWATT
6440 ? #4;"          USE      USE
     KWU      DAYS    DAY     DAYS    DAY"
:LPRINT :TIME=TIME+1:REM 8 & 13 SPCS
6450 RETURN
6460 R$="JAN":GOSUB 6600
6470 R$="FEB":GOSUB 6600
6480 R$="MAR":GOSUB 6600
6490 R$="APR":GOSUB 6600
6500 R$="MAY":GOSUB 6600
6510 R$="JUN":GOSUB 6600
6520 R$="JUL":GOSUB 6600
6530 R$="AUG":GOSUB 6600
6540 R$="SEP":GOSUB 6600
6550 R$="OCT":GOSUB 6600
6560 R$="NOV":GOSUB 6600
6570 R$="DEC":GOSUB 6600
6580 CLOSE #4:? "K":GOTO 2000
6600 FOR Z=1 TO NR:REM CALCULATE MONTH
LY CONSUMPTION AND COST
6610 READ M$,Y,UNITS,DAYS,NET,HD,CD
6620 HAVG=0:CAVG=0
6630 IF M$<>R$ THEN 6820
6640 ? #4;M$;" ";Y;
6650 PRNT$="

 ":REM 65 SPACES
6660 UP=INT(100*(UNITS/DAYS))/100
```

```
6670 PRNT$(6-LEN(STR$(INT(UP))),8)=STR
$(UP)
6680 PRNT$(15-LEN(STR$(UNITS)),14)=STR
$(UNITS)
6690 PRNT$(22-LEN(STR$(INT(NET))),24)=
STR$(NET)
6700 AVG=INT(1000*(NET/UNITS))/1000
6710 PRNT$(28-LEN(STR$(INT(AVG))),31)=
STR$(AVG)
6720 PRNT$(39-LEN(STR$(HD)),38)=STR$(H
D)
6730 IF HD>MINHD THEN HAVG=INT(100*((U
NITS-FCTR)/HD))/1000
6740 IF HAVG=0 THEN PRNT$(42,44)="N/A"
:GOTO 6760
6750 PRNT$(43-LEN(STR$(INT(HAVG))),46)
=STR$(HAVG)
6760 PRNT$(53-LEN(STR$(CD)),52)=STR$(C
D)
6770 IF CD>MINCD THEN CAVG=INT(100*((U
NITS-FCTR)/CD))/1000
6780 IF CAVG=0 THEN PRNT$(56,58)="N/A"
:GOTO 6800
6790 PRNT$(57-LEN(STR$(INT(CAVG))),60)
=STR$(CAVG)
6800 TIME=TIME+1:IF TIME=HL THEN SET=1
6810 ? #4;PRNT$
6820 NEXT Z:RESTORE
6830 IF SET=0 THEN 6870
6840 IF R$="DEC" THEN 6880
6850 ? "INSERT ANOTHER SHEET OF PAPER;
 THEN   DEPRESS ANY KEY":GET #1,A
6860 TIME=0:SET=0:GOSUB 6410
6870 LPRINT :TIME=TIME+1:IF TIME=HL TH
EN 6840
6880 RETURN
6999 REM INSTRUCTIONS FOR PREPARING DA
TA LINES
7000 LINE=NR+999
7010 ? "↖FOR EACH MONTH OF DATA YOU HA
VE, YOU    MUST TYPE ONE DATA LINE."
7020 ? "↓THE FIRST DATA LINE MUST BE N
UMBERED    1000."
7030 ? "↓AFTER THAT, EACH DATA LINE MU
ST BE      NUMBERED ONE HIGHER THAN TH
E LAST."
7040 ? "FOR EXAMPLE, 1000 MUST BE FOLL
OWED BY   1001, 1002, 1003, 1004, ETC.
"
7050 ? "↓DEPRESS ANY KEY WHEN READY FO
R NEXT     INSTRUCTIONS.":GET #1,A
7060 ? "↖THE FOLLOWING IS THE FORMAT F
OR A DATA  LINE:"
7070 ? "↓1000 DATA OCT,82,1350,30,79.2
5,495,0"
7080 ? "↓REQUIRED DATA SEQUENCE AND FO
RMAT:"
7090 ? "1. MONTH; MUST BE 3 LETTERS LO
NG"
7100 ? "2. YEAR; MUST BE 2 NUMBERS LON
G"
7110 ? "3. NUMBER OF KILOWATTS USED IN
 MONTH"
7120 ? "4. NUMBER OF DAYS IN BILLING P
ERIOD"
7130 ? "5. NET COST OF ELECTRICITY IN
BILLING      PERIOD"
7140 ? "6. HEATING DEGREE DAYS IN BILL
ING          PERIOD"
7150 ? "7. COOLING DEGREE DAYS IN BILL
ING          PERIOD"
7160 IF LINE<>999 THEN ? "↓LAST LINE O
F DATA YOU ENTERED WAS: ";LINE
7170 ? "↓NOW BEGIN TYPING NEW DATA LIN
ES."
7180 END
```

●

CHECKSUM DATA

(See D:CHECK/C:CHECK,p.26)

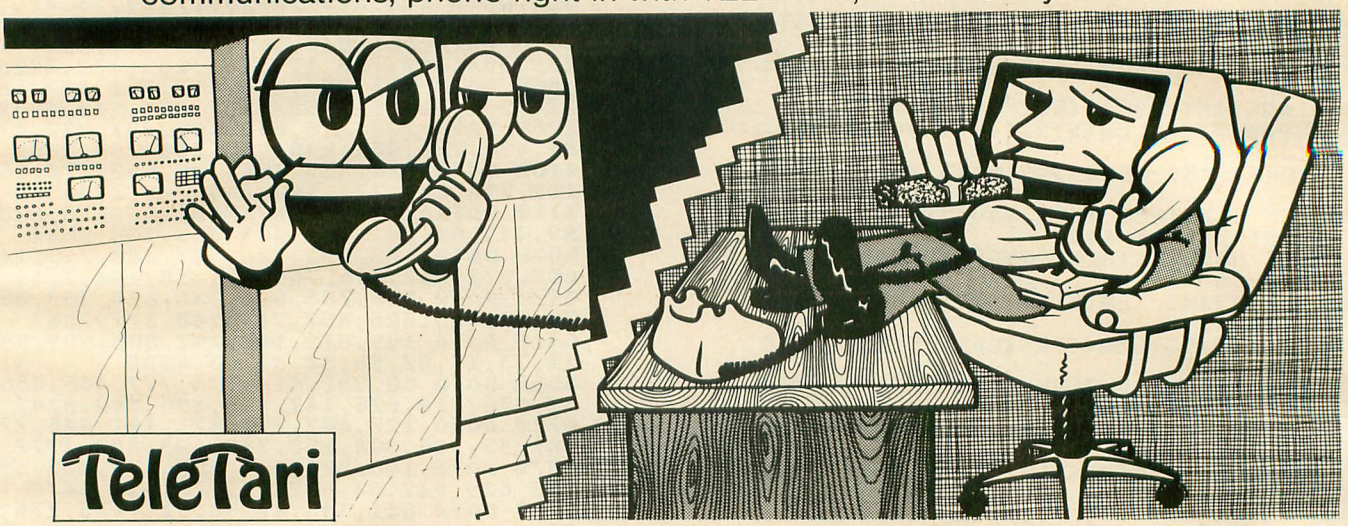
```
10 DATA 626,571,316,11,87,650,765,970,
462,317,312,349,844,885,96,7261
220 DATA 748,501,730,152,156,203,79,76
9,55,604,722,11,498,617,840,6685
520 DATA 258,376,999,591,189,73,420,43
5,728,140,203,330,681,639,405,6467
2090 DATA 851,848,852,604,860,824,723,
279,394,487,725,125,171,153,332,8228
3050 DATA 496,276,286,497,761,837,185,
274,293,885,611,584,357,351,249,6942
3200 DATA 778,316,848,308,491,727,470,
84,494,758,865,148,662,750,783,8482
4100 DATA 314,839,306,495,552,174,156,
129,793,294,777,500,764,740,77,6910
5110 DATA 101,50,652,783,713,236,823,6
69,448,51,341,715,499,355,108,6544
6020 DATA 373,553,240,527,488,719,149,
800,872,557,326,764,751,18,104,7241
6170 DATA 658,789,616,635,828,389,85,5
36,63,546,626,457,719,248,372,7567
6410 DATA 109,812,106,297,806,999,992,
11,33,13,57,56,34,32,23,4380
6560 DATA 60,992,414,734,777,465,896,3
52,576,210,625,142,869,295,840,8247
6720 DATA 127,482,778,321,101,446,779,
309,95,995,804,758,980,109,893,7977
6870 DATA 166,821,716,350,473,886,594,
695,429,247,84,642,143,82,144,6472
7120 DATA 531,305,47,98,613,435,276,23
05
```

●

Mr. Harb has also prepared a gas/electric version of this program called "Thermowatts." Space limitations prevented us from including it here, but the listing will appear in **The A.N.A.L.O.G. Compendium** *(to be published later this year).*

DON'T ASK PROVIDES THE MISSING LINKS

⟷ the link between your modem and the outside world. For hassle-free communications, phone right in with TELETARI, The Friendly Terminal.

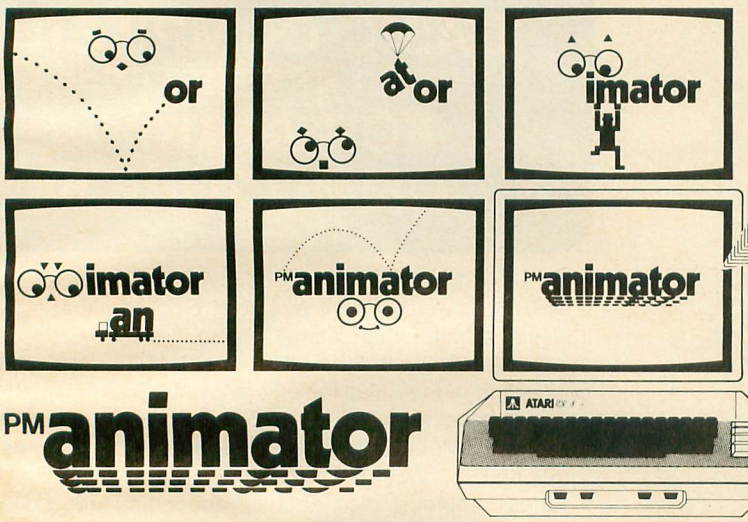


Your Atari has never had such easy access to the whole world of telecommunications – bulletin boards, news reports, large time-sharing computers, the works. Now it's a snap to tap into all these, and it's just as easy to transfer your program or text files to and from a remote computer. Meet TELETARI, The Friendly Terminal. It's just what your modem needs: a powerful, adaptable telecommunications package that's a cinch to use. With TELETARI, you simply choose the desired communications function from a menu. Commonly used terminal parameters are included in the program, but you can change them to suit your needs with a couple of keystrokes, using another handy menu, and store the ones you plan to use again. TELETARI's generous buffer stores up to 20K, so you can review, print, or save received information long after you've hung up the phone. You never knew using a modem could be so convenient. Because it's very flexible, TELETARI is compatible with most modems and a wide variety of computers. And because it works through the RS 232 port, TELETARI is not limited to modem/telephone uses. Put it to work in any RS232 application your imagination can devise – even operating a laser disk!

- buffer of up to 20K
- menu-driven
- highly adaptable
- supports all 850 options
- compatible with 1200 baud modems and BiT 3 Full-view 80™ board
- suitable for any RS232 application

$39.95 Requires Basic, 32K RAM, disk, 850 Interface

⟷ the link between BASIC and arcade-style graphics. Draw and animate pictures for your own BASIC games and other programs with pm ANIMATOR. Create running men, flying rockets, moving figures of all kinds.



Now available from DON'T ASK

BASIC programmers, pm ANIMATOR puts the power of Player-Missile Graphics at your fingertips.

$34.95 Requires 32K RAM, BASIC, disk.

PM animator

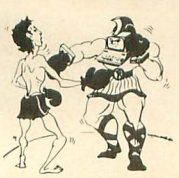
To order direct from Don't Ask, send a check or money order, or call to order COD. Add $2.00 for shipping and handling. California residents add 6% sales tax (6.5% if you reside in L.A. County).

⟷ the link between fast game action and verbal learning:

WORDRACE

Kids and adults, increase your vocabulary while you compete in this exciting word game.
Disk version:
3 levels of play – Beginner, Regular, Challenge
Requires 32K RAM, disk, BASIC. $24.95
Cassette version:
2 levels of play – Beginner, Intermediate
Requires 16K RAM, cassette, BASIC. $19.95
⟷ turn WORDRACE into a history game or a famous athletes game, and get more vocabulary words, with the WORDRACE accessory disk: CLAIM TO FAME/SPORTS DERBY. 3 new games in all.
Disk only. Requires WORDRACE disk. $19.95

Atari is a trademark of Atari, Inc. Full-view 80 is a trademark of BiT 3 Computer Corporation.



⟷ the link between you and what your Atari is really thinking:

ABUSE

the insult-exchange program.
Have you cursed out your computer? Now it can understand you and answer back!
Requires 40K RAM, BASIC, disk. $19.95
Release your aggressions! Inflict ABUSE on anyone who's got it coming!

DON'T ASK
⟷ the link between technical excellence and the fun of computing. Why do we give you so much? Don't Ask.

DON'T ASK COMPUTER SOFTWARE INC.

2265 Westwood Bl., Ste. B-150
Los Angeles, CA 90064
(213) 477-4514 or 397-8811

BASIC COMPILERS FOR THE ATARI

A Comparative Review

by Brian Moriarty

ABC by MONARCH DATA SYSTEMS
P.O. Box 207
Cochituate, Massachusetts 01778
40K Disk $69.95

THE BASIC COMPILER by DATASOFT
9421 Winnetka Avenue
Chatsworth, California 91311
48K Disk $99.95

BASM by COMPUTER ALLIANCE
21115 Devonshire Street
Suite 132
Chatsworth, California 91311
32K Disk $99.95

The world is full of ATARI BASIC programmers lusting for speed. They squirm with envy as the disciples of FORTH and C expound the virtues of those fast and exotic languages, and gaze with wonder upon machine-code hackers who wield their mysterious powers at 1.7 MHz.

Why this insatiable craving for faster programs? The answer is simple: GAMES. Every serious ATARI user has a secret desire to write the Ultimate Computer Game, a dazzling tour-de-force that would make Tempest look like Pong. Unfortunately, many would-be Chris Crawfords don't have time to master more than one programming language — and guess which excruciatingly slow language that one usually is.

If you've ever been frustrated by the speed of ATARI BASIC, then a BASIC compiler may be just what you need. The recent release of THREE new compilers for the ATARI offers programmers a long-overdue alternative to BASIC that TRS-80 and Apple users have been enjoying for years.

What Is A Compiler?

A compiler is a utility program that reads a program written in BASIC and translates it into a lower-level code that executes faster than the original. A compiled BASIC program is completely *self-contained*; it is treated exactly like a binary DOS file and does not need the BASIC cartridge or any other special software to run.

Monarch Data Systems' ABC, Datasoft's BASIC Compiler and BASM by Computer Alliance are significantly different in terms of features, performance and cost. Since ABC reached the market soonest, we'll examine it first.

Inside ABC

ABC is a single-pass integer compiler. "Single-pass" means that your BASIC program is scanned only once as it is being compiled. "Integer" means that numbers are stored in straight 3-byte binary instead of the 6-byte floating-point format used by the BASIC cartridge. The elimination of floating-point math is one of the main reasons for the speed of ABC.

The best way to understand ABC is to review what happens when you compose a BASIC program. Each time you hit RETURN over a line of BASIC, the instructions are "tokenized" into a special internal code that can be understood by the cartridge.

ABC takes this process a step further. It reads the tokens produced by the BASIC cartridge and translates them into an even more compact form called psuedo-code or "P-code." The P-code is then linked to a small machine-language program called a run-time interpreter, which reads and executes each P-coded instruction.

The big difference between tokenized BASIC and ABC P-code is **conciseness**. By using only whole-number integer arithmetic and a more efficient memory-management scheme, ABC simplifies the execution of each command in the ATARI BASIC repertoire. The result is a significant increase in the speed of the compiled program. According to Monarch, the speed improvement factor can range between four and twelve times, with seven times being a reasonable average.

It should be noted that the P-code produced by ABC is not 6502 machine language. It's essentially a series of pointers into the run-time interpreter, much like a FORTH program. You can't LIST, disassemble or make any sense at all out of the P-code without a detailed understanding of the ABC interpreter. This is an important feature if you're thinking about distributing your compiled software, because the code will be protected from all but the most determined pirates.

Easy To Use

Experienced BASIC programmers should have no trouble using ABC. First, you SAVE your completed BASIC program on a disk. Then you pull out the BASIC cartridge and boot the ABC disk. ABC asks for the name of your BASIC file and the name of a destination file, which will eventually become the compiled version of your program.

ABC next writes a copy of the run-time interpreter out to the destination file. It then scans your BASIC program and translates it into P-code, one line at a time. Finally, the P-code is appended to the interpreter, and you're left with a binary-format disk file that can be loaded and executed using DOS option "L." The original BASIC program is completely unaffected.

A couple of different run-time interpreters are included on the ABC disk. These provide a choice of loading addresses to match different memory configurations and DOS requirements. There is also a clever little program called MKRELO that makes your compiled software completely relocatable — a handy feature for commercial development because it assures that your software will run in virtually any machine with enough memory to fit it.

The Datasoft Compiler

Datasoft's BASIC Compiler is a 4-pass utility that converts BASIC programs directly into 6502 machine code. Because machine language doesn't need to be interpreted, the execution speed of the compiled code can be very impressive. Datasoft claims a speed improvement of 5 to 20 times over the original BASIC version.

Like ABC, a run-time support package must be linked to the code in order for it to run. Datasoft gives you a choice of two different run-time packages: a high-speed integer version or a slower version that will accept floating-point functions.

The compilation procedure for the Datasoft compiler is fairly involved. After specifying the source and destination filenames, the program asks you to select either integer or floating-point math; the appropriate run-time package is linked to your code. The compiler then studies your BASIC code and converts it into one or more mnemonic assembler files which are written out to the disk.

Next, the Datasoft system loads a 3-pass assembler which reads the intermediate files created by the compiler, and produces a machine-language output file which is the final, executable version of your BASIC program. All assembler files remain intact on the disk, and may be accessed by Datasoft's DATASM Editor/Assembler (sold separately) for later tweaking by hardcore hackers.

Datasoft's product is tricky to use if you have only one disk drive. Because the assembler and output files must be written onto the same disk as your BASIC code, you have to be sure to leave enough space for them. According to Datasoft, this limits the maximum size of your BASIC program to about 100 sectors (12.5K). Users with more than one drive can lessen the limitation by putting their BASIC source on a separate disk.

Good Diagnostics

An interesting feature of the Datasoft compiler is the Line Reference Map. This function displays each line number of your original BASIC program along with the exact address where its machine-language counterpart can be found. The map can be sent either to the screen, a printer or a disk file. Line references are very useful if you want to modify or debug the compiled version of a program.

The error handling of the Datasoft system is also helpful. Problems that occur during the execution of a compiled program produce a standard ATARI error number along with the address of the instruction that caused the foul-up. If you prepared a reference map of the program, you can determine which line of the original BASIC code produced the error. The Datasoft system also allows you to re-enter a crashed program at any point by specifying a new run address.

The Catch

It would be wonderful if you could take any old BASIC program, send it through one of these compilers and get a nice, speedy output file. Unfortunately, things aren't that simple. Both the Monarch and Datasoft products impose restrictions on the type of BASIC code that can be successfully compiled.

Listings 1 and 2 show the documented programming restrictions of the ABC and Datasoft BASIC compilers, respectively. Notice that the program access statements like LOAD, SAVE, ENTER and LIST are not supported by either system. This makes sense because of the self-standing nature of a compiled program. Also note that the floating-point math functions (SIN, COS, etc.) cannot be used by ABC, or by the integer version of the Datasoft compiler.

The documentation provided with ABC suggests a number of sneaky ways to get around its lack of floating-point arithmetic. It gives examples of how to simulate fractions, trigonometry and the RND() function without producing a compilation error. ABC's 24-bit integer math package allows a usable variable range of ±8 million, so it's possible to scale almost any value to a convenient whole number.

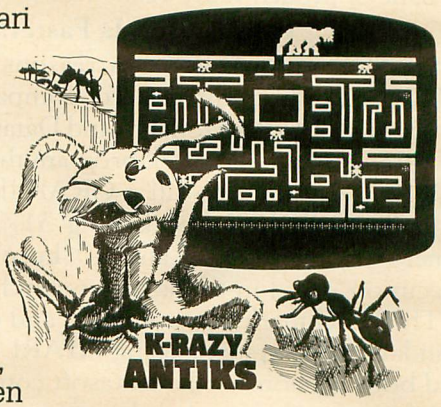
Both the integer and floating-point versions of the Datasoft compiler offer a nice implementation of the RND() function. Datasoft also allows you to use RUN statements as long as you don't include a filespec such as RUN "D1:PROGRAM."

Datasoft won't let you use variables as line references (GOTO X or GOSUB 100+Y, for example). Also, you can't imbed DATA statements in your BASIC code. You have to place them all at

Four new ways to get KRAZY!

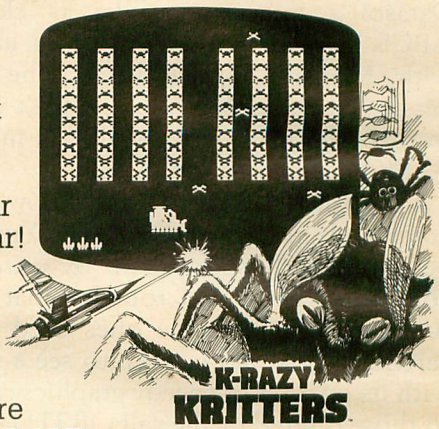
CBS Software introduces four new fast-action games for your Atari 400 and 800 or Commodore VIC-20 computers.* Each one is so challenging, you'll see why K-RAZY is the name of the game!

Take K-RAZY ANTIKS,™ for instance. If the carnivorous ants don't get you, their exploding eggs might. As you're defending your way through this multi-level maze game, watch out for the hungry anteater's sticky tongue and the flooding rainstorm, too!



If that doesn't make you crazy, try K-RAZY SHOOT-OUT!™ If you have terrific marksmanship, you can blast through the evil Alien Droids. But only if you're really sharp will you be able to avoid the radioactive walls and escape the deadly Control Sectors.

Or, try K-RAZY KRITTERS.™ You're in charge of the Command Ship, but it will take fast thinking to blast the invading Alien Attackers to save your Star Base from impending doom. Fail, and face the consequences—as your Command Ship is carted off to the Intergalactic Junkyard!



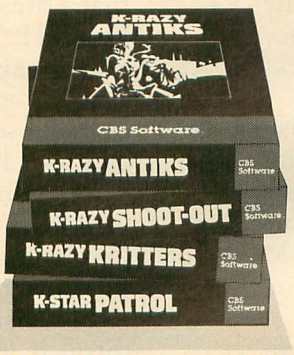
Just as crazy is K-STAR PATROL.™ Because you're the Star Ship Squadron's only hope. Maybe you can defend against the Alien Attack Forces, but can you escape the Intergalactic Leech, too? Oh, and one other thing: replenish your Star Ship's Force Field, or the end is near!

Also, this April, watch for our two new games that look, sound and play so different from anything else, you'll really have to see them to believe them. They're MOUNTAIN KING™ and BOULDERS AND BOMBS™ —both exploding with challenge!

After all, they're from CBS Inc. And everything that name stands for in quality entertainment is built into each and every game.

Remember. Get the most you can out of computer games...and get KRAZY!

*All programs are available as ROM cartridges for Atari® 400™ and 800™ computers. K-RAZY ANTIKS™ and K-STAR PATROL™ are also available for the Commodore VIC-20.™



CBS Software

A Unit of CBS Inc., 41 Madison Ave., New York, NY 10010

the very end of the program, preceded by an END, STOP or GOTO statement.

I like to keep DATA statements close to their corresponding READs because it makes programs easier to debug. I also like to use variables as line references because it makes my code self-documenting: statements like GOSUB NEXTLINE are inherently more meaningful than GOSUB 2011. Hopefully a later version of the Datasoft compiler will deal with this common stylistic approach more realistically.

Which One Is Faster?

Speed is one of the main reasons for using an ATARI BASIC compiler. To compare the speed performance of the Monarch and Datasoft products, I wrote a short benchmark program that uses nested FOR/NEXT loops to fill a GRAPHICS 24 screen with direct POKEs (see Listing 3). The hardware timers at locations 19 and 20 keep track of the execution speed in 60ths of a second or "jiffies."

The benchmark was compiled and executed on a standard 48K system, using ATARI BASIC, ABC and both versions of the Datasoft compiler. Just for the fun of it, I also tried the program on ATARI's disk-based Microsoft BASIC and Optimized Systems Software's BASIC A+ 3.05. The program was run 3 times under each system, and the results were averaged to produce the data in Listing 4.

The 5-to-20 times speed improvement claimed by Datasoft's integer compiler is clearly justified. ABC's increase is about 7.4 times, also right in line with Monarch's advertising. The floating-point version of Datasoft's compiler isn't very impressive in this example — it's not all that much faster than BASIC A+.

Prospective users should know that graphics commands like PLOT, DRAWTO and FILL will not be significantly speeded up by using one of these compilers. The ROM routines that perform these functions are the same ones used by the BASIC cartridge. It would be nice to see a super-compiler with its own set of speedy graphics routines, similar to those offered by the valFORTH language system.

Memory Requirements

The amount of memory required by a compiled BASIC program depends on three things: the size and type of program being compiled, the efficiency of the compilation, and the size of the run-time package required to support the code.

ABC's run-time interpreter takes up 36 disk sectors or about 4.5K of RAM. The floating-point package for Datasoft's compiler requires 32 sectors (4K), while the integer package needs 29 sectors (3.6K). These figures represent the minimum RAM overhead required by any compiled program, regardless of its size or function.

We looked far and wide for a large BASIC program that could be used as the basis for a size comparison between the Datasoft compiler and ABC. Most of the trouble was caused by the Datasoft product, which would not accept the imbedded DATA statements found in virtually every off-the-shelf BASIC program we tried. In desperation, I decided to write this issue's feature game *(Adventure in the 5th Dimension)* without using variable GOTOs or GOSUBs, "misplaced" DATA lines or anything else that would violate the restrictions documented by either product.

After thoroughly de-bugging the adventure, I SAVEd it onto a disk and checked its size. The BASIC version required 99 sectors, just below the maximum recommended by Datasoft for a single-drive system. So far, so good. Then I tried compiling the program with ABC using my single-drive 48K system. I experienced no problems until the very end of the compilation, when the program informed me of an Error #166 (Point data length). Puzzled, I called Monarch and spoke to the author of the program. He tracked down the problem (too many GOSUBs in line 66), suggested an easy fix and promised to eliminate the limitation in future releases. My second compilation was flawless; the P-code produced by ABC is 129 sectors in length, about 30% larger than the original. And the adventure plays perfectly.

Next I tried compiling *The 5th Dimension* with Datasoft, again using a single drive and 48K. I followed the instructions in the user's manual and transferred the system equate file SYSEQU.ABC onto the same disk as the BASIC program. Then I ran the compiler. Before the end of Pass 1, the compiler reported an Error #162 (Disk Full). I looked at the disk with DOS and found that the assembler files had completely filled the disk, leaving no room for the assembly itself!

I borrowed another drive and re-compiled, using a second disk containing copies of the assembler, system equate and run-time library files. Again I was greeted with an Error #162. Not to be deterred, I put the assembler file ASM.OBJ onto the same disk as the adventure and tried one more time. Success! The compiler just barely found enough room to write the assembler files, and I made it through Pass 1.

My disk space difficulty was caused by the fact that Datasoft always writes the assembler files onto Drive #1. The reference manual estimates that these files require about five times as much space as the BASIC source file. That places the maximum possible source file size at somewhere around 144 sectors (18K), regardless of the number of disk drives you can borrow.

Now the compiler started on Passes 2 and 3. In Pass 3, the compiler stopped to tell me I had some unresolved line numbers. It didn't say which lines were causing the problem, so I checked carefully through the BASIC program for GOSUBs or GOTOs that used variable line references. Nothing.

The RESTORE statements in lines 73 and 79 do use variable line references. But Datasoft's documentation doesn't say anything about RESTOREs. I wrote a little BASIC test program to see if the compiler would accept RESTOREs with a variable in the line reference. Sure enough, the test failed.

I consider this "undocumented restriction" (read BUG) to be very serious. Data line addressing is one of the most powerful features of ATARI BASIC. I used it extensively in the adventure program because it made object handling so much easier. Rewriting the adventure was out of the question; so I compiled the program one more time and ordered the assembler to ignore the "unresolved line numbers." The remainder of the compilation proceeded without error. Final program size was 214 sectors, more than twice the size of the original. Due to the presence of known errors, I did not try to run the compiled version.

Other bugs in the Datasoft BASIC Compiler have been discovered by users of the first release. I have personally verified difficulties with TRAP and VAL, along with some confusing problems with strings and numeric arrays. Datasoft is reportedly aware of these bugs and will hopefully offer updated disks to purchasers of the early release.

The Envelope, Please

The choice between Monarch's ABC and Datasoft's BASIC compiler is not an easy one. Each product has a unique personality that make it suitable for specific applications and programming styles.

If ultra-high speed is very important to you, then the machine code produced by the Datasoft integer compiler is tough to beat. Datasoft's product is also the better choice if you want to play around with the compiled versions of your software. And if you absolutely have to use transcendental math, the Datasoft floating-point package offers a slow but effective way to get it.

On the debit side, Datasoft's product is very greedy with disk space and RAM. You need at least two disk drives to compile anything except small programs; and you have to put up with an alarming range of BASIC programming restrictions. Before you buy the Datasoft compiler, I suggest that you check with your dealer to make sure you're getting a bug-free version.

ABC isn't as picky about your source code as the Datasoft compiler. It will compile just about anything that doesn't use fractions — and its wide usable number range gives it a decided advantage when it comes to simulating floating-point operations at high speed. The P-code produced by ABC offers a degree of software protection you can't get with straight 6502 machine code. Last but not least, Monarch's ABC costs $30 less than the Datasoft product.

And Then There's BASM

You may be wondering why I haven't yet mentioned BASM, the third "BASIC compiler" listed at the beginning of this article. The reason is simple: BASM isn't really a BASIC compiler at all. It's a BASIC assembler — an entirely new programming environment for the ATARI that looks like BASIC but acts like assembly language.

Take a look at Listing 5. This is the BASM equivalent of the speed benchmark used to test the ABC and Datasoft compilers. Notice that some of the lines look like ordinary BASIC, while others look like 6502 mnemonics. REM statements are included in those places where the BASM code differs significantly from the original BASIC.

BASM programs are composed using a text editor supplied with the software. Then the source file is saved onto a disk and assembled into machine language. A very small run-time library is linked to the code, and your application is ready to run.

The BASM system understands a very usable subset of ATARI BASIC, along with a number of statements and conditionals not found in the cartridge (see Listing 6). "Primitive" commands like PEEK and POKE must be replaced with their assembly-language equivalents, LDA (Load Accumulator) and STA (Store Accumulator). READ/DATA structures are implemented by using the 6502 X - and Y-registers as indexes.

BASM allows you to mix BASIC and assembly statements freely, even on the same logical line. This arrangement combines the simplicity of BASIC with the power of machine language in a most ingenious manner.

Because BASM programs have an assembly-like syntax, the efficiency of compilation is much greater than either ABC or Datasoft. Only the pure BASIC statements are actually "compiled" — the assembly-language sections are incorporated into the program as in-line machine code. This means that the speed of a BASM program can approach the limits of the hardware. I compiled and executed the BASM program in Listing 5 and obtained an execution time of 18 jiffies or less than 1/3 of a second. This is 231 times faster than the ATARI BASIC equivalent! Computer Alliance claims a more conservative speed improvement of up to 130 times.

Not For Beginners

BASM is not as straightforward to use as the ABC or the Datasoft compilers. You'll have a hard time following the 72-page reference manual unless you know something about 6502 architecture and assembly-language programming. It took me a while to grasp the syntax required for certain types of BASIC variables and addressing modes. More complete documentation is definitely called for — even if it means raising the price a bit.

I also ran across a bug in the disk interface. My

review copy of BASM bombed out whenever I tried to load and RUN a compiled program more than once. This made it impossible to repeat my benchmark demo without a complete system re-boot. When Computer Alliance fixes this problem, they will have a fascinating and very powerful "BASIC compiler" on their hands.

Implications

A stigma against BASIC programming has arisen in the ATARI software market over the past few years. The prejudice is based on the absurd idea that the quality of a program has something to do with the language it was written in.

The compilers reviewed in this article will help make BASIC programming respectable again. For this reason, I think they are the most important pieces of ATARI software to come down the pike since valFORTH. They may actually be more significant, because they offer much of the performance of FORTH without the need to learn a new programming language. That means BASIC hackers can spend less time puzzling over stacks, disk screens and other unfamiliar concepts, and more time improving the quality of their BASIC.

I'm happy to report that not one of the compilers mentioned in this article requires a licensing fee. You can sell your compiled software royalty-free as long as you include a credit in your documentation.

BASIC compilers are about to open the world of professional software development to a whole new range of talented authors. Let's hope the code they produce will be as sophisticated and valuable as these three products. □

Listing 1. ABC Programming Restrictions.

Unsupported Functions:
ATN CLOG COS EXP LOG RND SIN SQR

Unsupported Arithmetic Operators:
∧ (exponentiation)

Unsupported Statements:
BYE CLOAD CONT CSAVE DEG DOS ENTER LIST LOAD LPRINT NEW RAD RUN SAVE

Other Restrictions:
Cannot use fractional (non-integer) values.

Cannot use constants larger than 65,535 (variable range is ±8 million)

Listing 2. Datasoft Programming Restrictions.

Unsupported Functions (integer mode only):
ATN CLOG COS EXP LOG SIN SQR

Unsupported Arithmetic Operators:
None

Unsupported Statements:
BYE CLOAD CONT CSAVE DOS ENTER LIST LOAD NEW RUN - "FILESPEC" SAVE

Other Restrictions:
Integer mode values limited to ± 32,767 (except address constants).

DATA statements must be at end of program and cannot be "executed" (see text).

DIM statements cannot use variables for size allocation (e.g.; DIM X(A)).

GOTO and GOSUB cannot use variables for line references (e.g.; GOTO X).

Listing 3

```
10 REM ******************************
15 REM * BENCHMARK TEST FOR BASIC *
20 REM *         COMPILERS          *
25 REM ******************************
30 POKE 19,0:POKE 20,0
35 GRAPHICS 24
40 SETCOLOR 1,0,14:SETCOLOR 2,0,0
45 SCREEN=PEEK(88)+256*PEEK(89)
50 FOR I=0 TO 191:FOR J=0 TO 39
55 POKE SCREEN+J,255
60 NEXT J:SCREEN=SCREEN+40:NEXT I
65 GRAPHICS 0
70 PRINT PEEK(20);" jiffies"
75 PRINT PEEK(19);" jiffies x 256"
80 END
```

Listing 4. Speed Test Results.

RUN-TIME ENVIRONMENT	JIFFIES	SECONDS
ATARI BASIC Cartridge	4160	69.3
ATARI Microsoft BASIC (disk)	3348	55.8
OSS BASIC A+ 3.05 (disk)	2717	45.3
Monarch ABC Compiler	565	9.4
Datasoft Compiler (Integer Mode)	218	3.6
Datasoft Compiler (FP Mode)	2435	40.6

Listing 5

```
0100 REM * PROGRAM EQUATES
0110SCREEN=88
0120TIMER=20
0130TIMER256=19
0140 REM * POKE 19,0:POKE 20,0
0150 LET TIMER256 = 0 : LET TIMER = 0
0160 GRAPHICS 24
0170 SETCOLOR 5 , 0 , 14 : SETCOLOR 6 , 0 , 0
0180 FOR I = 0 TO 191 : FOR J = 0 TO 39
0190 REM * POKE SCREEN+J,255
0200 LDA #255 : LDY J : STA (SCREEN),Y : NEXT J
0210 REM * SCREEN=SCREEN+40
0220 REM * THIS IS A 16-BIT BINARY ADDITION
0230 CLC : LDA SCREEN : ADC #40 : STA SCREEN
0240 LDA SCREEN+1 : ADC #0 : STA SCREEN+1
0250 NEXT I
0300 REM * GRAPHICS 0
0310 FILE 0
0320 BPRINT TIMER : PRINT " jiffies"
0330 BPRINT TIMER256 : PRINT " jiffies x 256"
0340 RETURN : REM * BACK TO BASM
0350 REM * LINE 360 INITIALIZES THE VARIABLES I & J
0360 DIM I , J
```

Listing 6. BASM keywords.

BINPUT	BPRINT	CLOSE	COLOR
DATA	DEF/ENDDEF	DIM	DRAWTO
FILE	FILL	FOR/NEXT	GET
GOSUB	GOTO	GRAPHICS	IF
INPUT	LET	LOCATE	OPEN
PLOT	POSITION	PRINT	PUT
REM	RETURN	SETCOLOR	SOUND
STOP	TR	TRAP	WHILE

Copyright Basics

by Thomas M. Krischan

So, you've just written the successor to PAC-MAN and you want to protect your investment from the software pirates! Copyrighting it is a good initial step but there are some things you'll need to know. First of all, what exactly is a copyright? A copyright is a form of protection provided by the laws of the United States (title 17, U.S. Code) to authors. Specifically, it protects "original works of authorship" including computer programs. This protection is available to both published and unpublished works. The owner of the copyright has the exclusive right to do and to authorize others to do reproductions, prepare derivative works, distribute copies and display the copyrighted work publicly (section 106 of the Copyright Act). It is illegal for anyone to violate any of these rights. These rights, however, are limited to your particular literary or pictorial form of expression and nothing more. You do not own the exclusive right to any idea, method or system mentioned in your expression. My favorite example involves a mountain climber who successfully ascends to the summit of Mt. Everest. Upon reaching the top the climber unpacks his camera and snaps a picture of the sun as it sets behind an adjacent mountain. The climber owns exclusive rights to that picture, but not to all pictures of sunsets. In fact, if other climbers stood shoulder to shoulder with him on that mountain and took their own pictures, then each would own exclusive rights to their very own picture, even though all of the pictures are similar, perhaps identical, with each other. Copyright protection also excludes scientific or technical methods or discoveries, business operations or procedures, mathematical principles, formulas or any other sort of concept, process or method of operation (Circular 31). Inventions are subject matter for patents, not copyrights.

An expression is a description, explanation or illustration of an idea or system. Again, you cannot copyright an idea. But you can copyright a description of an idea, because it's your literary expression. It becomes obvious that proper wording in the copyright application is essential. Even the experts have difficulty in interpreting the fine points of copyright law and consequently its infringement. This is especially true of computer programs. For example, Apple Computer (May 1982) filed suit against Franklin Computer claiming that proprietary software designs were being infringed upon against copyright law. They also filed suit that proprietary components were being infringed upon, violating patent law. Franklin announced that they were filing a major antitrust counterclaim. Both parties decided to let the courts make the final judgement.

On July 30, 1982, U.S. District Judge Clarence C. Newcomer held that Apple was not entitled to a preliminary injunction against Franklin's sales. Score: Franklin 1, Apple nothing. Apple had applied for and obtained copyright registration for each of its programs involved in the suit. The judge, however, was not convinced that an operating-system program in object code can be protected under the copyright law. Apple's claim for copyright protection was based on these arguments:

1) That object code is a form of expression and a work of authorship.

2) That a computer's operating system is in the form of an expression, not an idea or system.

3) That ROMs and floppy diskettes are a tangible medium of expression and not mechanical devices.

Franklin argued that they wanted to be fully compatible with Apple, so that they could use independently produced software and hardware from third parties. Confused? You're not alone. Judge Newcomer wrote that "there is no clear consensus on how to describe the technology employed in microcomputers." The final word has yet to be said in this continuing legal battle. But, this is for certain, if ROM and floppy diskettes are found to be mechanical devices then they lose the protection reserved for writings and expressions under copyright.

In another case, Magnavox filed suit against Mattel Electronics claiming that proprietary patents had been infringed upon. On July 22, 1982, U.S. District Judge George Leighton held that Magnavox was entitled to an injunction against Mattel's sales. The decision forbids Mattel from manufacturing, selling or even using six popular video games. These games are Football, Tennis, Basketball, Hockey, Soccer and Baseball. My mind can't help but flash back to the TV commercial of George Plimpton standing in the snowy bleachers comparing video football games and hearing him say, "...kind of leaves you out standing in the cold." Score: Magnavox 1, Mattel nothing. As any good lawyer would do, Mattel filed and won a motion to stay the injunction with a second judge. The patent infringement allegedly refers to the rebound action of a ball-type object.

In yet another case, Astrocade filed suit against ATARI and Commodore claiming that licensed patents had been infringed upon. The patents are held by Bally, who refused to join the suit. Unfortunately, Astrocade did not request a preliminary injunction, so there's no score yet.

United States Copyright lawsuits are not confined to the United States. The U.S. holds copyright protection treaties with most countries, with the notable exception of Taiwan. Consequently, bootlegged materials are often shipped from Taiwan and seized in Hong Kong, New Zealand or some other country that does enforce the international law.

These lawsuits demonstrate that ownership of a copyright or patent is not necessarily total protection. Possession helps, but you can still be sued and you could even lose. Sometimes the suit may never appear in court, since the very threat of litigation and seizure of a product is often sufficient. Case in point-Visicorp hired a legal firm to confront certain advertisers who used their trademark prefix "visi" in their product name (Gee, I wonder what Citicorp thinks about the name Visicorp). As a result, most of the advertisers mended their ways and decided not to fight. The ultimate responsibility for enforcement against infrigement belongs with you. The Copyright Office does not compare deposit copies or check registration records to determine whether works submitted for registration are similar to any work already copyrighted. They just document the registration of the work, assign it a number and deposit a copy for reference purposes.

The way in which copyright protection is secured is frequently misunderstood. The copyright is secured automatically when the work in created. A work is created when it is fixed to a visually perceived device such as books, manuscripts, videotape or microfilm. Registration in the Copyright Office and publication of the work are not required. There are, however, definite advantages to registration and several consequences to publication. Registration establishes a public record of the copyright claim, allows for infringement suits to be filed in court and will establish *prima facie* evidence in court of the validity of the copyright and of the facts stated in the certificate. When a work is published, all published copies should bear a notice of copyright, the year of publication and the name of the owner of the copyright. Failure to comply with the notice requirement can result in the loss of certain rights, such as the right to recover attorney's fees and statutory damages. The published works are subject to mandatory deposit with the Library of Congress. But here again, there is confusion on what publication means. The Copyright Act defines publication as the distribution of copies, but not their performance or display alone. Therefore, a performance on television does not constitute a publication, but the same performance in a theater or over cable television does. Does a turn on a PAC-MAN arcade machine constitute a publication or merely a performance? Publication requires the transfer of ownership and the unrestricted disclosure of its contents. What exactly does that quarter buy? No one is quite sure!

If you choose to register your program, send the Copyright Office a completed application (Form TX), $10.00 and two copies of your source code and manual. Here is an exclusive inside tip. Starting in September, 1982, you will be able to protect your visual displays by including two color photographs of every significant illustration. Significant illustration means detailed graphics like Mattel's baseball diamond or ATARI's PAC-MAN grid but not text modes or simple graphics patterns. You can request that forms be mailed to you by telephoning (202) 287-9100 and leaving a message on their recording machine or by sending a letter to the Register of Copyrights, Library of Congress, Washington, D.C. 20559. The Copyright Office is not permitted to give legal advice. If you need information on ownership disputes, infringement suits, publishing, royalty payments or the like, you will be told to consult an attorney.

Once you receive your application form, read the instructions, get a typewriter and reread this next part of the article. We will go through and fill out an application step by step. I am assuming the most simplified case of a lone author and their totally original program.

SPACE 1: TITLE. Your work must have a title. It does not have to be original or lengthy. Leave the rest blank.

SPACE 2: AUTHOR. Enter your name, your nationality and check NO for these three questions: "Was this author's contribution to the work a 'work made for hire'?", "Was this author's contribution to the work anonymous?", "Pseudonymous?". Enter under AUTHOR OF: "Text of computer program and description". Do not use the words: idea, method, system, develop, procedure, or principle. Words which imply expression are: description, explanation or illustration. Leave the rest blank.

SPACE 3: CREATION. Enter the appropriate year and leave the rest blank. The copyright is enforced for your lifetime plus fifty years.

SPACE 4: CLAIMANT. Enter your name, address and leave the rest blank.

SPACE 5: PREVIOUS. Check NO for the question: "Has registration for this work, or for an earlier version of this work, already been made in the Copyright Office?". Leave the rest blank.

SPACE 6: COMPILATION. Leave blank.

SPACE 7: MANUFACTURING. Leave blank unless you have signed a royalty agreement with someone. If you have, enter their name and address.

SPACE 8: REPRODUCTION. Check "a" for copies and phonorecords.

SPACE 9: CORRESPONDENCE. Enter your name and address under "CORRESPONDENCE" and leave the rest blank.

SPACE 10: CERTIFICATION. Check "author", enter your name, date and sign by the (X).

SPACE 11: ADDRESS. Enter your name and address.

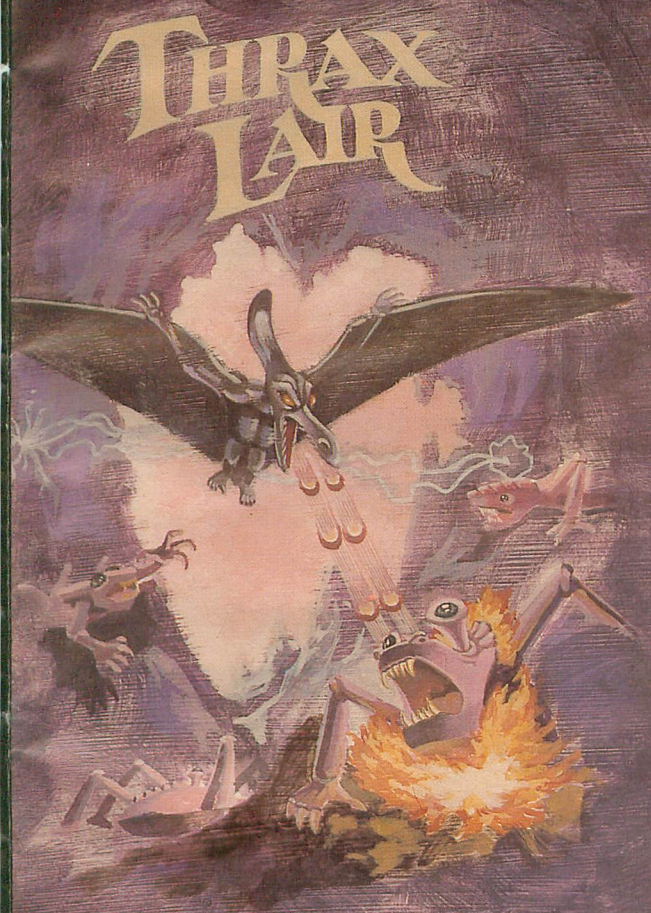
If you have followed my instructions, you should have entered your name 5 times; your address 3 times; the date (or at least year) 2 times; title, brief statement, nationality and signature 1 time each and 6 check marks. If you did this exactly as I told you to and you mail all of the items discussed earlier, the Copyright Office will send back a copy of the form. The copy should arrive within 8 to 12 weeks with a certification stamp, a registration number and a date. If you did something incorrectly, they will send you back your application with a new application form, a letter of explanation with a control number and a three month grace period to reapply or you forfeit your $10.00 and two copies. Don't feel bad if you get your application returned to you. I have obtained many copyrights but only once on the first try. I forgot to sign the application, or used a "bad" word. Often, they only wanted verification that I meant what I said, "Are you sure this is entirely your own work?". Many programs are written from scratch but some are compilations of other programs; data bases, operating systems, and the like. Be sure to give the proper credit to other works if you did incorporate them into your work. However, a program in BASIC or Assembler does not cite that language under "PREEXISTING MATERIAL". Language is considered an idea or system.

Copyright is not the only method of protection. There are also patents and trademarks, but their application forms are much more detailed and expensive. Beyond the legal aspects, there are also hardware and software techniques of protection. Diskettes and ROM's can be made uncopyable by their hardware dependence. Source and object codes can be made unreadable by encrypting them in any of various manners. But for ten bucks, copyrights are hard to beat. □

(Editor's note: a recent court decision ruled in favor of Franklin. The courts ruled that since the operating system of the Apple was in the public domain, computers emulating it were not an infringement of Apple's rights.)

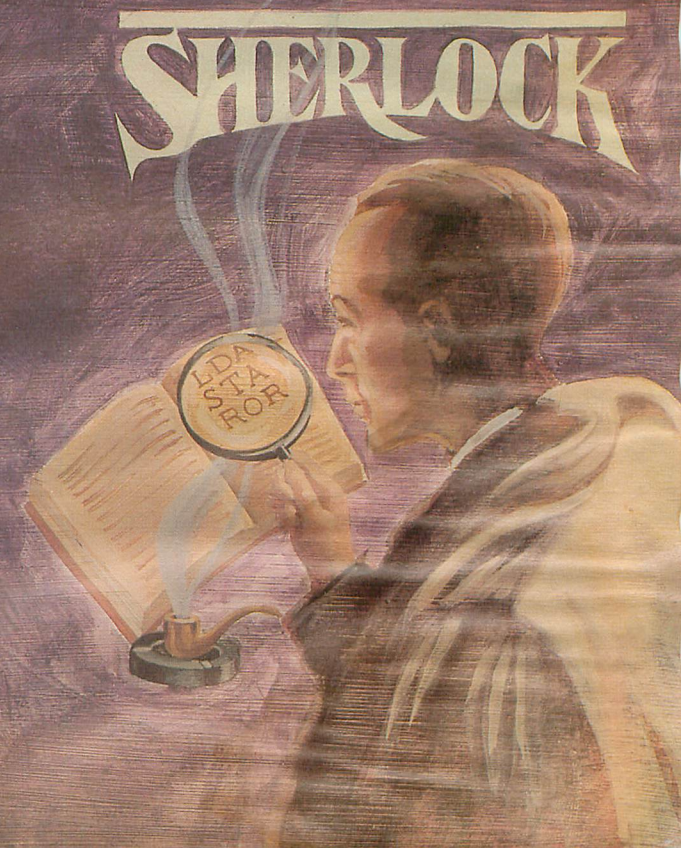
RANTOM
S·O·F·T·W·A·R·E

Offering a Whole New Spectrum in Programming: Family Fun, Arcade Action, or Programming Aids.



TRUST A delightful game for 1-4 players. Dazzling 3-D graphics, convincing animation, and infinite variety, with no violence. Perfect for the family. 48K on Atari* disk and cassette, soon on Commodore 64+ disk and cassette. **$34.95**

SHERLOCK Takes up where Atari* DOS II stops. Read or edit sectors, map sectors, disassemble programs, and more. The best buy in disk utilities. **$29.95**



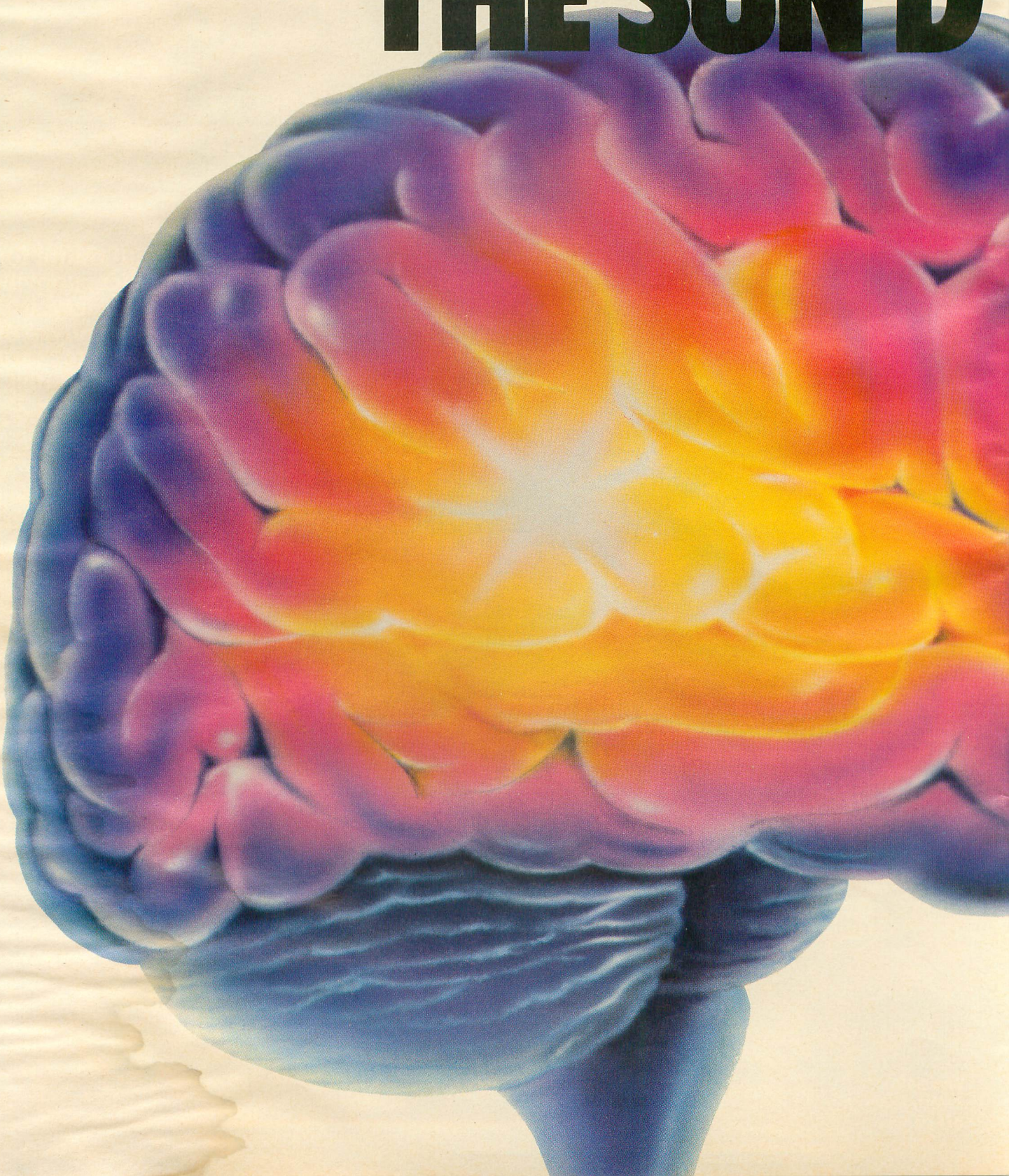
THRAX LAIR An exciting arcade game requiring lightning reflexes and a cool head. Great animation and fast action. 16K on Atari* disk and cassette, soon on Commodore 64+ disk and cassette. **$29.95**

*Atari is a registered trademark of Warner Communications Co.
+Commodore 64 is a registered trademark of Commodore Business Machines Co.

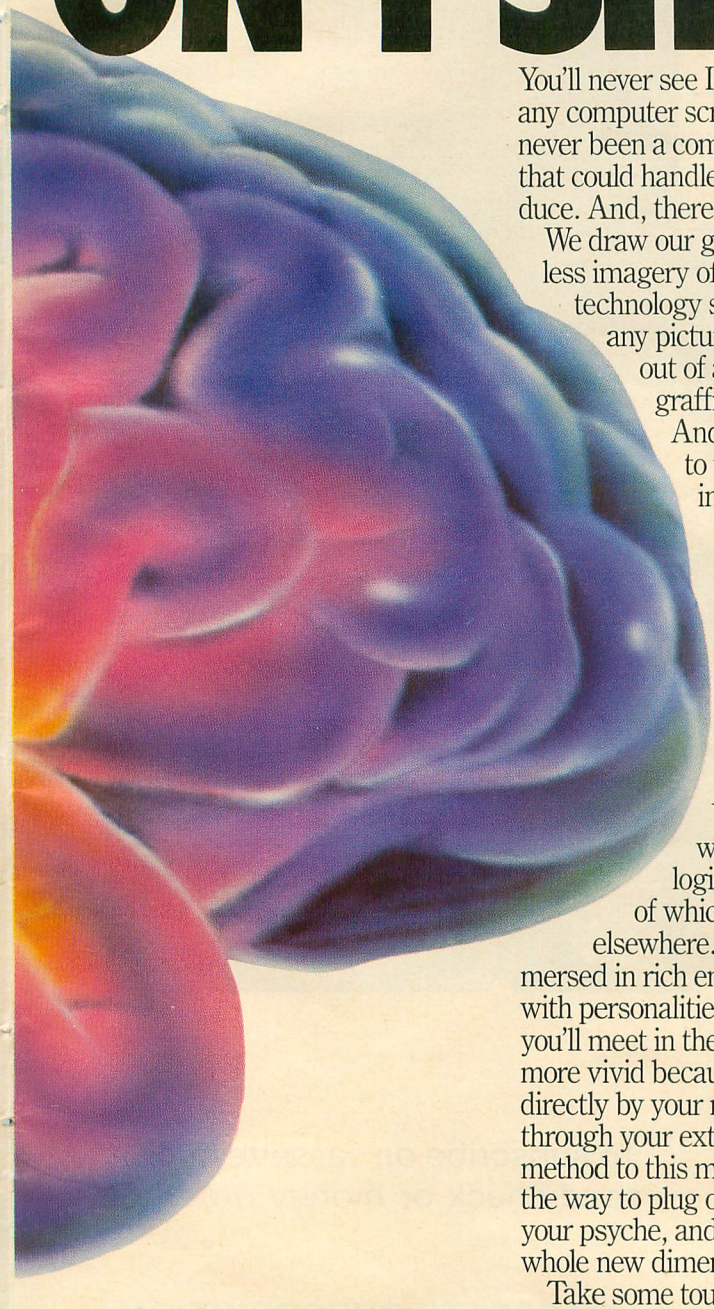
Attention Programmers: Rantom's unique marketing approach allows royalties of up to 60% for quality programs or 2 to 3 times greater royalties than industry standards. Call or write for details.

RANTOM
S·O·F·T·W·A·R·E
Box 5480
Avon, CO
81620
303-949-6646

WE STICK OUR G
THE SUN D



RAPHICS WHERE ON'T SHINE.



You'll never see Infocom's graphics on any computer screen. Because there's never been a computer built by man that could handle the images we produce. And, there never will be.

We draw our graphics from the limitless imagery of your imagination—a technology so powerful, it makes any picture that's ever come out of a screen look like graffiti by comparison.

And nobody knows how to unleash your imagination like Infocom.

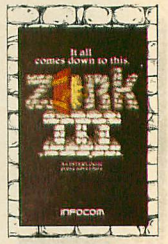
Through our prose, your imagination makes you part of our stories, in control of what you do and where you go—yet unable to predict or control the course of events.

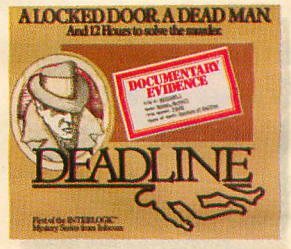
You're confronted with situations and logical puzzles the like of which you won't find elsewhere. And you're immersed in rich environments alive with personalities as real as any you'll meet in the flesh—yet all the more vivid because they're perceived directly by your mind's eye, not through your external senses. The method to this magic? We've found the way to plug our prose right into your psyche, and catapult you into a whole new dimension.

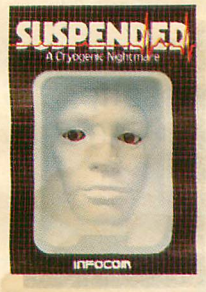
Take some tough critics' words about our words. SOFTALK, for example, called ZORK® III's prose "far more graphic than any depiction yet achieved by an adventure with graphics." And the NEW YORK TIMES saw fit to print that our DEADLINE™ is "an amazing feat of programming." Even a journal as video-oriented as ELECTRONIC GAMES found Infocom prose to be such an eye-opener they named one of our games their Best Adventure of 1983.

Better still, bring an Infocom game home with you. Discover firsthand why thousands upon thousands of discriminating game players keep turning everything we write into instantaneous bestsellers.

Step up to Infocom. All words. No graffiti. The secret reaches of your mind are beckoning. A whole new dimension is in there waiting for you.







INFOCOM™

The next dimension.

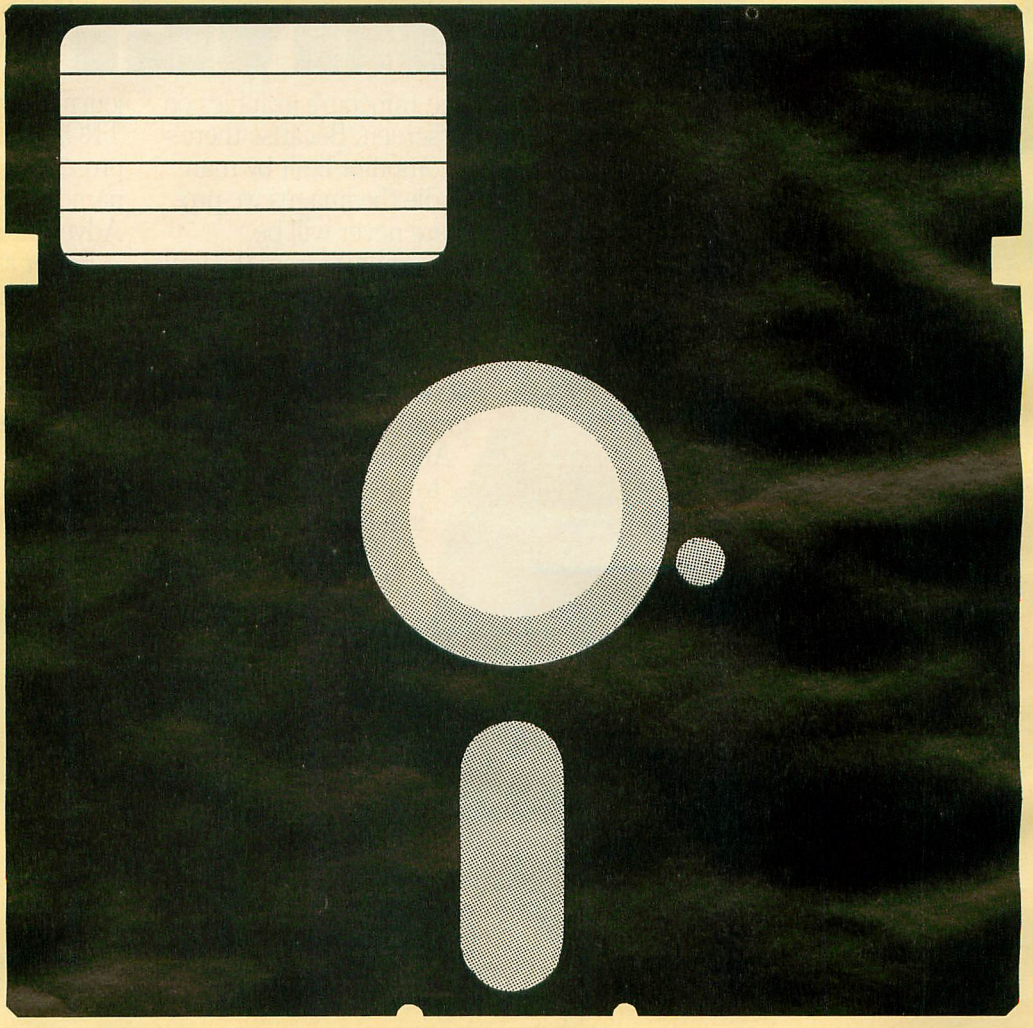
Infocom, Inc., 55 Wheeler St., Cambridge, MA 02138

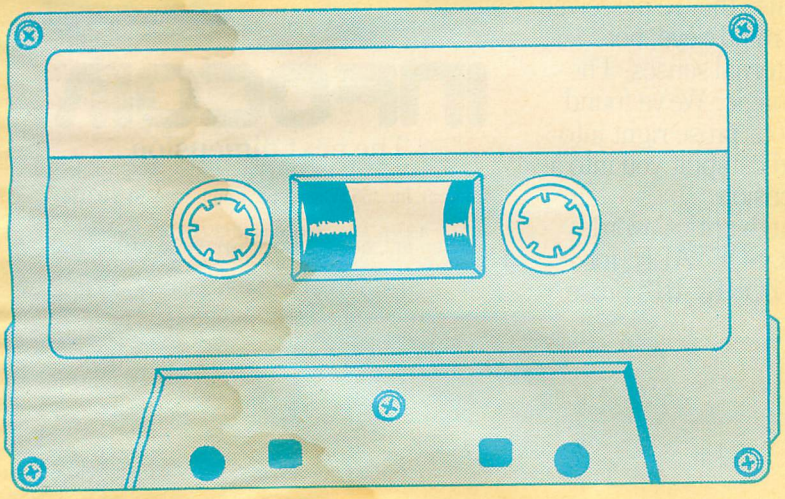
For your: Apple II, Atari, Commodore 64, CP/M 8," DEC Rainbow, DEC RT-11, IBM, NEC APC, NEC PC-8000, Osborne 1, *TI Professional, TRS-80 Model I, TRS-80 Model III.*

DO YOU SUBSCRIBE TO A.N.A.L.O.G. COMPUTING ON DISK OR CASSETTE?

. . . if not, then you should. Since issue no. 1, **A.N.A.L.O.G.**'s cassette and disk subscriptions have eliminated the need for you to spend hours typing in programs from the magazine. All of the programs in the magazine are on the cassette and disk versions. These programs are in the public domain, and are not copy-protected.

1 year (6 issues) on cassette, $36.00, disk, $60.00.





To subscribe on cassette or disk, send check or money order to:

CASSETTE SUBSCRIPTION
or
DISK SUBSCRIPTION
P.O. BOX 23
WORCESTER, MA 01603

DISK BACKUP PROGRAMS: DO THEY REALLY WORK?

by Brian Moriarty

This article is not another sermon on the evils of disk pirating. Anybody with two cents' worth of intelligence knows that pirating is wrong, and that unauthorized software duplication violates the author's right to a fair profit.

But what are the rights of a software BUYER? Is it legal for the original purchaser of a disk to make personal copies? Is it possible to duplicate a "copy-protected" disk? Is it moral to sell a disk copying program — or to advertise one?

This last question is of particular concern to the staff here at **A.N.A.L.O.G.** . Over a dozen ATARI disk copiers have appeared on the market over the past few months. These are supposed to be able to back up virtually any ATARI disk, even those protected by something called "bad sectoring." Many of the companies producing these products have bought ad space in the pages of this and other magazines.

A.N.A.L.O.G. has a responsibility to its other advertisers and to ATARI users in general. If the "super-dupers" being advertised in our magazine are contributing to the piracy of copyrighted software, it is not in the best interest of anyone to continue running them.

The reactionary way out of this dilemma would be to immediately reject all further ads for disk copiers. Instead, we decided it would be fairer (and more interesting) to test the copiers first to see if they really worked. We will also examine the technology of ATARI disk copying, and propose a set of protection standards for consideration by the ATARI community.

The Hardware.

Before discussing the operation of disk copiers, let's review the hardware that makes it all possible — the ATARI 810 Disk Drive.

The 810 is an **intelligent** drive. It contains its own 6507 microprocessor, memory buffers and a dedicated operating system that directly controls every reading, writing, and formatting function.

The 810 accepts commands from the main console over the serial bus. Only five commands are recognized by the 810 operating system: Read Sector, Write Sectors, Write Sector with Verify, Format Disk and Status Check. All ATARI disk I/O (including disk copying functions) works by using one or more of these fundamental operations.

It's important to understand that the computer can only tell the 810 what to do — NOT how to do it. That means that it's impossible to write a program that will make the 810 access a disk in a non-standard way. The ONLY way to accomplish this is by monkeying around with the drive, or with the disk itself.

The Disk.

When you format a disk, the 810 writes a predefined magnetic pattern consisting of 40 concentric rings or *tracks*. Each track is divided into 18 sectors which contain 128 data bytes apiece. Since there are 18 X 40 or 720 sectors, a standard ATARI disk can hold 720 X 128 or 92, 160 bytes of information.

The 810 also writes timing information and a unique indentification number onto each sector when it formats a disk. These sector headers cannot be altered once they are in place because the 810 doesn't know how to access them individually. You can play with the sector data all you like — but you can't touch those sector headers unless you completely reformat the disk.

The Copiers.

The principle of ATARI disk copying is very simple. All you do is read the data from each interesting sector into RAM, and write the data out to the same physical sectors on another disk. The copy disk should perform exactly the same as the original.

ATARI DOS II uses this method to duplicate disks. Whenever you create or modify a disk file, DOS updates a map of "in-use" sectors called the Volume Table of Contents (VTOC), which is maintained at sector 360. When you select option "J" (Duplicate Disk), DOS checks the VTOC to find

ATARI SINGS YOUR FAVORITE SONGS!!!

THE Original VOICE BOX Speech Synthesizer by the ALIEN GROUP has received rave reviews:

MICRO COMPUTING–"The VOICE BOX injects an endearing personality to your computer. The possibilities are enormous."

COMPUTE–"The VOICE BOX offers more human-like tones and does not blank out the screen."

CREATIVE COMPUTING–"English text and phonetic code may be freely intermixed rather than requiring separate modes as is the case without exception with every other speech system. A mode called talking face displays an animated face with impressive lip sync animation."

ANTIC–"There is a great potential for teaching children to spell and an added dimension to games overall. I believe the VOICE BOX is well worth the price tag."

ANALOG–"For ATARI owners who want to add speech to their programs, the Alien Group VOICE BOX is probably the best choice."

POPULAR SCIENCE–"The speech quality is excellent. Besides creating speech, the software has a bit of fun with graphics."

and on the new VOICE BOX II.....

TIME MAGAZINE–"Machine of the Year" "The VOICE BOX by the Alien Group enables an ATARI to say aloud anything typed on its keyboard in any language. It also sings "Amazing Grace" and "When I'm 64" or anything else that anyone wants to teach it.

INCORPORATE THE SINGING HUMAN FACE
INTO YOUR PROGRAMS AND GAMES

VOICE BOX II
Speech & Singing Synthesizer

The New **VOICE BOX II** for **ATARI** plugs into the serial port of the **ATARI** 400/800 with sound coming out of the TV/monitor. 48K DISK is required. It has all of the features of the original **VOICE BOX** plus many exciting new hardware and software features:

- The ability to sing with voice and 3 part music.
- A library of 30 famous songs.
- A comprehensive music system that allows the user to easily enter or modify new songs.
- Software that can convert the bottom two rows of the **ATARI** keyboard into a piano with a range of 3½ octaves using the shift and control keys.
- Programmable musical sound effects such as tremolo, vibrato, glissando and click track.
- A singing *human* face with lip-sync animation designed by Jerry White.
- A talking or singing ***ALIEN*** face with software that allows the user to change the face and 8 mouth patterns as he sees fit.
- The ability to speak with inflection and feeling.
- Can speak in a foreign language with correct foreign spelling as input.
- A talk and spell program by Ron Kramer. Users can program any vocabulary for this spelling game. In fact, this program can even speak in a foreign language like French, where the user must spell the correct word in English, or vice versa.
- **GREEN GOBLINS**–A talking arcade game by John Wilson.
- Random Sentence Generator–An amusing grammar game that helps teach school children to identify parts of speech and recognize a variety of sentence structures.
- **NUMBER SPEAK**–A subroutine by Scott Matthews that converts up to a 9 digit number into normal English pronunciation. Ideal for building your own math games.
- **STUD POKER**–A talking poker game by Jerry White.
- The screen never blanks out while talking or singing.
- Singing or speaking subroutines can be incorporated into your programs, requiring as little as 100 bytes of RAM plus 5 bytes for each word.
- Entries into the $5000 talking or singing game contest can be written using the **VOICE BOX II**–send for contest information.
- Price $169.00 includes **VOICE BOX II** and all of the above software.
- Inquire about our discounts for educational institutions.

To order by mail send a check or money order to the ALIEN GROUP for $169. Then, try the VOICE BOX II for 10 days, and if it isn't the finest value you've ever seen in a computer peripheral, the most challenging and provocative addition you've ever made to your system, return it in its original condition for a full refund.

THE ALIEN GROUP
27 West 23rd Street (212) 741-1770
New York, NY 10010

ALSO AVAILABLE AT LEADING COMPUTER STORES THROUGHOUT THE WORLD.

out which sectors are marked as being active. It then reads the contents of those sectors and copies them.

DOS ignores sectors that haven't been marked in the VTOC. Some early disk protection schemes took advantage of this fact by "hiding" important data in unmarked sectors. If you tried to copy one of these disks with DOS, you wouldn't get a complete copy and the program would fail.

The first ATARI "disk backup" programs were merely dumb sector copiers. These products (with provocative names like Mirror Image, Superdup and Lockpik) bypassed DOS altogether by talking directly to the disk drive. Each and every one of the 720 sectors on a disk was read and duplicated, whether or not DOS thought those sectors were important. The resulting copies included all the "hidden" data and ran just like the original.

Software manufacturers were understandably concerned about these copy programs. Piracy was rampant, and for a while it seemed as if there was no way to circumvent a dumb sector copier.

The Breakthrough.

Towards the end of 1981, a new type of ATARI disk protection was introduced which put an end to "dumb" sector duping. Pirates were bewildered by these innocent-looking disks which copied perfectly but would not run.

The disk manufacturers found a way to modify selected sectors so that they could not be read by a standard 810 drive. The application program included a routine that checked for these "bad" sectors. If they were readable, the program assumed the disk was a copy and crashed itself.

Imagine what happens when you try to "Lockpik" a disk protected by bad sectoring. The copier will detect the bad sectors but will not be able to reproduce them because the 810 doesn't know how to write anything but good sectors. The resulting copy may contain every byte of program data, but it will stubbornly refuse to execute.

Software publishers were delighted by the effectiveness of this new protection scheme. The pirates shuffled away to sulk — and, inevitably, to tinker.

The New Copiers.

Hackers are known for their cleverness and dogged persistence. It didn't take long for one of them to figure out a way to beat "bad sectoring."

It seems that if you write to a disk sector with your drive speed adjusted far above or below its normal setting, the written data will be screwed up so badly that a normal-speed drive won't be able to read it. A similar result can be obtained by sticking a piece of tape on the disk jacket and pulling on it during a write operation. This throws the drive alignment off

just enough to produce a bad sector.

These two discoveries have led to a second generation of disk copiers that allows repeated read/write access to any selected sector, making it fairly easy to produce bad format. First you perform a sector-for-sector copy of the source disk, noting the sectors that are bad. Then you go back and destroy the integrity of the noted sectors on the dupe. The copied software will never know what happened to it — and neither will the software author, if the person doing the copying is lucky.

Most of the copy programs in our listing are the second-generation type with provisions for bad sectoring. Some include additional utilities that allow you to directly examine and edit sector data, and perform a variety of other useful (and totally legitimate) housekeeping function. Each listing is followed by a brief comment which notes the strong and weak points of that particular product. Contact the manufacturer if you need more detailed information.

The manufacturers of these copy programs do NOT endorse the use of their products for the purposes of duplicating copyrighted software. All of them include strict warnings to this effect in their documentation, along with a defensive note emphasizing the importance of being able to back up valuable software. It is amusing that most of these same publishers are distributing their copy programs on heavily protected disks! Did I hear someone use the work "hypocrisy?"

Another Breakthrough.

Software publishers aren't as worried about second-generation copy programs as you might think. Why? Because they have ALREADY adopted ingenious new disk protection methods that can foil even the most elaborate software-based copier.

Again it was the disk duplicators who came to the rescue, this time with sophisticated duplicating machines that can do all sorts of weird things to the sector headers on an ATARI disk. I won't divulge the details here, but I can tell you that **not one of the copy programs listed here can make an executable copy of a disk duplicated using these methods.**

What's more, there will NEVER be a copy program that will allow an unmodified 810 to dupe these disks, because there is no way for an 810 to access individual sector headers. The only way out is to scan the entire disk, sector by sector, locating and bypassing the machine-language instructions that look for the special format. This is a formidable task, especially if the code happens to be encrypted (and it nearly always is). So if your main reason for buying a "disk backup" program is to make dupes of Choplifter, Filemanager 800+, The Datasoft Compiler or any other current software hit for your friends, forget it!

What About Us Honest Guys?

The new disk protection technology has important implications for ALL software buyers, even the completely honest ones (yes, both of you). It is now more important than ever for software publishers to start providing separate backup disks with their products. Not just a mail-in coupon — I mean an extra physical disk, in the same package as the original. This is particularly true of systems and business programs, where an untimely disk failure can cause a lot of annoyance and expense.

Few things are more frustrating than an essential disk that crashes. The first prize for thoughtless software packaging goes to ATARI, for their disk-based version of Microsoft BASIC. Would you risk writing an important business program with this language, knowing that your only copy of the interpreter is a speck of dust away from total uselessness? Would you care to deal directly with ATARI in a time-critical emergency? I thought not.

It would cost publishers only a few dollars more to include a separate backup disk with their products. This simple precaution would go a long way towards protecting the interests of buyers and enhancing the professional image that ATARI software so desperately needs. It would also help reduce the temptation to produce unauthorized backups — a practice that usually leads to pirate copies at the next user's group meeting.

I don't think it's necessary to provide separate backups for games. But it would be an inexpensive comfort to have an extra copy on the flip side of the disk, just in case. This is definitely NOT recommended for professional programs. I know of at least one popular (and expensive) business-oriented program that comes only with a flip-side backup — a pointless courtesy if I misplace my disk.

What Do You Think?

A.N.A.L.O.G. is eager to hear your opinions on the issues presented in this article. I'd like to see feedback from everybody: consumers, authors, publishers and maybe even a pirate or two. We'll publish the most interesting replies in our next issue. So get out your favorite word processing program (hopefully a legitimate copy) and start typing! □

We invite all readers to share their thoughts with us on the subject of disk backups vs. pirating. Send letters to:

READER COMMENT
P.O. BOX 23
WORCESTER, MA 01603

Disk Copy Programs

DISKEY
ADVENTURE INTERNATIONAL
Box 3435
Longwood, Florida 32750
32K BASIC $49.95

One of the most comprehensive disk utilities available for the ATARI. Command-driven, with dozens of function keys and more options than you can shake a sector at. The 62-page reference manual is loaded with information, but a bit obscure in places. Strange that a major software company would release such an effective backup system — on a protected disk, of course.

SUPERCLONE
FRONTRUNNER COMPUTER INDUSTRIES
100 West Grove, Suite 115
Reno, Nevada 89509
32K BASIC $49.95

The first disk copier on the market with "bad sectoring" (the tape method). Designed for the total novice, Superclone uses a complicated disk-mapping routine to insure that only two copies can be made of any specific disk. This well-intentioned limitation makes the program extremely awkward to use. If you try to dupe a disk more than twice, you get a hi-res color picture of a pirate. Isn't that cute?

DISK WIZARD
C.A.P. SOFTWARE
69 New Boston Road
York, Maine 03909
32K BASIC $29.95

A friendly little package of four BASIC programs: sector copier and editor, speed checker and sector disassembler. Each menu-driven utility must be loaded separately. The Disk Wizard user's manual is one of the best; and unlike most disk copier publishers, C.A.P. actually encourages you to back up your master disk.

DISKED
AMULET ENTERPRISES
PO Box 25612
Garfield Heights, Ohio 44125
24K BASIC $89.95

Disked offers all kinds of exotic functions not found in any other disk utility. The BASIC programs are fully LISTable so that you can reach into the code and tinker to your heart's content. The human engineering is rough in places, and the price is rather steep, but the documentation is very complete and includes many hands-on examples that beginners should find very helpful.

DISKWIZ
ALLEN MACROWARE
1906 Carnegie Lane "E"
Redondo Beach, California 90278
16K BASIC $25.00

This is my personal favorite. Diskwiz combines all of the most-needed disk editing and copying features in a menu-driven package that runs very quickly and doesn't take a lot of RAM. Human engineering and error-trapping are superb; the documentation is a little sparse, but adequate. And the price is right!

MIGHTY BYTE
MIGHTY BYTE COMPUTER
828 Green Meadow Avenue, Dept. AG
Rockford, Illinois 61107
16K BASIC $29.95

A no-frills sector copier with provisions for "bad sectoring." The instructions for making bad sectors and adjusting your drive speed are unusually complete.

DISK PACK 1000
ALPHA SYSTEMS
4435 Maple Park Road
Stow, Ohio 44224
40K BASIC $14.95

Besides the usual copying features, Disk Pack includes a disk speed checker, directory menu, graphics dumper and even a little program that lets you adjust your screen colors. If a sector copier is all you want, Disk Pack is one of the cheapest ways to get it.

REPLICATOR ONE AND TWO
PLANETARY MACHINE INTELLIGENCE
2500 Lee Road, Suite 210
P.O. Box 2895
Winter Park, Florida 32790-2895
32K BASIC $34.95

The Replicator One and Two programs are intended for single- and dual-drive systems, respectively. Both programs are simple BASIC sector copiers, with no provision for bad sectoring and minimal error-trapping.

DISKSCAN
CDY CONSULTING
421 Hanbee
Richardson, Texas 75080
32K BASIC $40.00

Another menu-driven package of disk editing utilities. The "Assemble Into Sector" option is unique; it lets you type in 6502 mnemonics and assemble them directly into any part of a sector. The manual includes a brief but useful dissertation on ATARI disk structure.

A few other ATARI disk copiers have been advertised in various computer magazines and newsletters. These have the same features found in the ones listed here, and are priced in the $20-$50 range. Many user group software libraries offer sector copying programs that are free for the asking. And don't forget Tony Messina's **Disk Tool**, an excellent sector editor published in A.N.A.L.O.G. Issues 8 and 9; and **The Black Rabbit**, a 2-pass machine-language sector copier written by yours truly and published in Issue 9.

When choosing a disk copy program, beware of fly-by-night software publishers who promise the moon and vanish with the morning dew. Some of our readers have experienced problems with late deliveries, incomplete products and poor after-the-sale support. Please let us know about any difficulties you may experience with the companies advertised in this magazine. □

UTILITY # 5: ATARI MEMORY TEST

16K Cassette or Disk

by Tony Messina

Glancing at my calendar I see that it's time once again for another installment of (trumpet fanfare) **UTILITY WORLD!**...but first a brief message.

I would once again like to thank everyone who has written with questions, comments and additions to the utilities presented in this series. I am answering your questions as quickly as possible, so please be patient — I promise you will get a reply. Now, back to our program.

For the past three issues (8,9,10) I have been directing my utilities towards the disk drive crowd while leaving my cassette-based friends in the dark. For this I apologize and I hereby make it up to you by presenting this issue's utility (number 5), entitled ATARI MEMORY TEST. There are two program listings. One will create a disk file while the other will create a boot cassette. Prior to discussing the details of the test, let's talk about memory tests in general.

What Is A Memory Test?

A simple answer would be that a memory test tests your memory to see if it's OK. To be a bit more specific, a memory test should test the integrity of your memory chips to insure that any and all values stored into them remain there. Please keep in mind that I am speaking of RAM or Random Access Memory. And now...

Believe It Or Not

Believe it or not, when memory chips go flaky, it is possible to store a number in a location, load the number immediately back in and find that it is completely different! Or, how about this: storing a 0 into a location, immediately loading the contents of that location and finding out that it's not 0!! Weird, huh? Just think of the headaches in debugging programs where you store or load values from flaky memory chip locations. To bring this point home, the following story is true (at least that's what my informant told me). Only the names have been changed to protect the innocent victims.

A Memorable Story

Mr. X wrote a little math drill program on his ZAFARI at home. Mrs. X, a teacher at a local elementary school, thought this program was really neat and wanted to take it in to show her class. After all, Mr. Happy Face, who appeared and played music when a correct answer was given, and Mr. Frown Face, who appeared when an incorrect answer was given, were both really cute.

The next day Mrs. X, math program in hand, secured a ZAFARI from the school library, set it up for her class and even invited the principal down to see this unique program in action. Little Tommy Tucker was first. The program asked "How much is 2+2, Tommy?" to which little Tommy typed in 4 and hit return. At this point everything hit the fan. Mr. Frown Face appeared with half his head missing and the rest looking like he got hit by baseball bat (I'd frown too!). The music played but sounded like something you hear in a swamp after sunset. To top it all off, the answer line said "I'm sorry Tommy, but 2+2=12615"! Mrs. X was shocked, the principal was shocked, little Tommy Tucker was shocked. Mrs. Tucker, having heard about the incident after school, called the principal asking him what kind of a &$#%#" computer came up with 2+2=12615?! Needless to say, after all the smoke cleared it was found that a whole section of one memory board was flaky. Upon replacing it, everyone was pleased with the program and they all lived happily ever after. 'Nuff said.

OK? How Do We Test Memory?

The most comprehensive and reliable methods of testing memory (at least on 8 bit machines) write values from 1 to 255 in each byte of memory to be tested, verifying each value written and halting if an error is detected. Often this procedure will be repeated several times. Although this algorithm is thorough, one might guess that it does require a great deal of time (and it does.) I have such a test and it takes about 30 seconds to test 1K or 24 minutes to test 48K! I don't know about you but I don't feel like waiting that long. Fortunately there are other ways to achieve the same results with only a slight sacrifice in efficiency. This issue's utility uses one of the fastest methods available. Before discussing the algorithm I must make a confession.

Credits, Or "Someone Please Help"

The memory test I'm presenting this issue, or at least the algorithm that performs the actual testing, was not written by me. I found this code on a faded

handwritten paper stuffed inside an MOS programming manual which I purchased at a flea market for $2.00! The title written on this piece of paper was **6502 Memory Test** and was written for some other computer, since the original program's origin was set at 0. Anyone recognizing this code (lines 795-1195 of the source listing) please call and let me know who wrote it as I would like to give credit where credit is due. I will take credit for the bells and whistles, code commenting (the original had zip), conversion of the program into a multipass test, fitting the program in a reasonable address space of the ATARI, and the error address isolation printout routine. OK, with that out of the way, let's get down to the meat of this program.

The Meat Of This Program

Rather than give a blow-by-blow, line-by-line description of the source code, I suggest that you run the appropriate BASIC program to create a bootable image and then run the actual memory test to see the display created when it is run. Then read through the source code to see how it was done. I will, however, discuss the algorithm. The area tested runs from page 7 to the top of available user memory. Rather than have the test run from one specific memory location to another, it is designed to test memory as a series of pages. This not only simplifies things but also allows the test to run much quicker. The program origin is at $0480. This is normally the floating point storage area but since we run our test without the BASIC cartridge there isn't a conflict. I coded the program so it will run on any memory configuration up to 48K. So don't worry if you only have 8K or 16K... the program will figure out where it can test. The program goes through 10 iterations (0-9). Each iteration consists of the following steps.

1. Write the value $FF to all locations in the test area.
2. Write 00 in the one out of every three bytes in memory. Memory looks like FF FF 00 FF FF 00...etc...
3. Test the memory area and ensure the pattern above is there. Halt giving the memory address, Test value, and actual value if there is an error.
4. If no error write all FF again.
5. Write new pattern FF 00 FF FF 00 FF, etc....
6. Verify memory has FF 00 FF etc...halt if error.
7. If no error write FF.
8. Write new pattern 00 FF FF 00 FF FF etc.
9. Test and halt if necessary.
10. This ends the all FF phase.
11. Now write 00 to all locations.
12. Write pattern 00 00 FF.
13. Test and halt if error.
14. No error write 00 to all memory.
15. Write pattern 00 FF 00.
16. Test pattern and halt if error.
17. No error write 00 to all memory.
18. Write pattern FF 00 00.
19. Test and halt if error.
20. END OF 1 ITERATION.

The program does all of the above steps 10 times! As you can see we are turning all bits on and off in the test area by using FF or 00. What we are actually doing is checking to ensure that we can set and clear all bits within the memory area. Any change to our pattern indicates flaky memory in the form of stuck bits, or erratic retaining properties of the address flagged in error. Having the test do 10 iterations just makes things tougher for it to pass. You might think that this test would take a lot of time, right? Would you believe 11 seconds per iteration on a 48K ATARI?? Believe it! This test will waste about a minute and a half of your time if you have 48K and less if you don't have 48K. The secret is in the paging technique, zero page indirect addressing and keeping the major part of the program data on page 0. Check out the source. Whoever wrote the algorithm sure knew his 6502!!

Program Drawbacks

Because of the way I implemented this program, it will not test pages 0-6 nor will it test the area of memory containing the display list and screen data for graphics mode 0. It will test the area of memory used by your BASIC programs and the majority of memory where we all tend to store our ML things. This was the price I paid for bells and whistles! If you have heartburn about this, then a modification to have the program move itself to an already tested area could be made. This is just a program and as with all programs in this series, it is open for discussion, modification and just plain study. I hope you use it and enjoy.

Disk Directions

Listing 1 will create self-executing disk files. Type it in, save it and then run it. Be sure you have a disk that contains DOS 2 in drive 1 when you run it. The file created will be called MEMTST.OBJ. When the file has been created, pull out the BASIC cartridge and Reboot with only DOS 2. When the DOS Menu appears, do a binary load of MEMTST.OBJ. The program will load and execute. DOS will be creamed by this program but don't worry.

Cassette Directions

Listing 2 will create an auto-boot cassette for you. Type in the program and then run. When you hear the 2 beeps, insert a blank cassette into your tape drive, push record and play and hit return. The tape will be created. To load it, turn off your ATARI. Rewind the tape to the start of the program just created. Push play. Hold down the START key on your ATARI and at the same time turn on the power. You will hear a beep. Press return and the tape will load. When done, the program will start automatically.

FINAL ITEMS

The test will halt if it encounters a faulty memory location. The format of the message will be:

```
ERROR @->$A01A TVAL->$FF AVAL->$CD
```

For sample purposes the above message would mean address $A01A failed our test. The Test value (TVAL) was $FF but the actual value (AVAL) in address $A01A was $CD.

The program will print TEST DONE if no errors appear.

That's *it* for this article. I hope to hear from someone...anyone...if they know who wrote the original implementation. As always, any questions can be directed to me here at the A.N.A.L.O.G. offices. Keep hacking and may all your memories read true. □

```
10 REM ********************************
15 REM *  MEMTEST CASS FILE MAKER     *
20 REM *       BY TONY MESSINA       *
25 REM *     (C) A.N.A.L.O.G. 83     *
30 REM ********************************
31 GRAPHICS 2:POKE 712,96:? #6;"    ME
MORY  TEST":? #6;"   cassette maker":?
 #6;" ■■■■■■■■■■■■■■■■■■■■"
32 ? #6;" (C) analog 400/800":? #6;" *
■■■■■■■■■■■■■■■■■■■■"
33 ? #6;"    by tony messina"
34 POSITION 1,7:? #6;"CREATING boot fi
le":POSITION 0,9:? #6;"CKSUMS"
35 REM
40 DIM PROG$(510):PNTR=1:LINE=198
45 LINE=LINE+2:FOR COUNT=1 TO 10
50 READ BYTE:IF BYTE=999 THEN GOTO 75
55 PROG$(PNTR)=CHR$(BYTE)
60 PNTR=PNTR+1:TOTAL=TOTAL+BYTE:NEXT C
OUNT:? "LINE:";LINE
65 READ CHECKSUM:IF CHECKSUM=TOTAL THE
N TOTAL=0:GOTO 45
70 ? "BAD CHECKSUM: LINE ";LINE:STOP
75 TOTAL=TOTAL+BYTE:READ CHECKSUM:IF C
HECKSUM<>TOTAL THEN 70
80 OPEN #1,8,128,"C:"
85 PRINT #1;PROG$:CLOSE #1:END
100 REM
105 REM ********************************
110 REM * CAS PROGRAM DATA FOLLOWS *
115 REM ********************************
120 REM
200 DATA 0,4,128,4,136,4,24,96,169,60,
625
202 DATA 141,2,211,169,152,141,10,0,16
9,4,999
204 DATA 141,11,0,96,162,0,142,82,0,16
0,794
206 DATA 5,162,232,32,133,243,162,38,1
42,240,1389
208 DATA 2,169,61,142,225,5,32,164,246
,174,1220
210 DATA 225,5,202,16,242,160,6,162,40
,32,1090
212 DATA 133,243,173,88,0,24,105,180,1
33,207,1286
214 DATA 173,89,0,105,0,133,208,169,7,
141,1025
216 DATA 229,5,173,230,2,56,233,1,141,
230,1300
218 DATA 5,169,9,141,226,5,169,16,141,
224,1105
220 DATA 5,169,0,168,141,227,5,141,228
,5,1089
222 DATA 133,205,141,227,5,162,2,142,2
28,5,1250
224 DATA 173,229,5,133,206,174,230,5,1
73,227,1555
226 DATA 5,73,255,141,231,5,145,205,20
0,208,1468
228 DATA 251,230,206,228,206,176,245,1
74,228,5,1949
230 DATA 173,229,5,133,206,173,227,5,2
02,16,1369
232 DATA 4,162,2,145,205,200,208,246,2
30,206,1608
234 DATA 173,230,5,197,206,176,234,173
,229,5,1628
236 DATA 133,206,174,228,5,173,231,5,2
02,16,1373
238 DATA 5,162,2,173,227,5,209,205,208
,32,1228
240 DATA 200,208,238,230,206,173,230,5
,197,206,1893
242 DATA 176,229,206,228,5,16,159,173,
227,5,1424
244 DATA 73,255,48,144,206,226,5,16,78
,76,1127
246 DATA 169,5,132,205,32,192,5,141,88
,6,975
248 DATA 173,222,5,141,87,6,160,0,177,
205,1176
250 DATA 32,192,5,141,98,6,173,222,5,1
41,1015
252 DATA 97,6,165,205,32,192,5,141,78,
6,927
254 DATA 173,222,5,141,77,6,165,206,32
,192,1219
256 DATA 5,141,76,6,173,222,5,141,75,6
,850
258 DATA 160,6,162,65,32,133,243,160,6
,162,1129
260 DATA 100,32,133,243,76,176,5,160,0
,238,1163
262 DATA 224,5,173,224,5,145,207,76,22
9,4,1292
264 DATA 72,74,74,74,74,32,211,5,141,2
22,979
266 DATA 5,104,32,211,5,141,223,5,96,4
1,863
268 DATA 15,201,10,48,2,105,6,105,48,9
6,636
270 DATA 0,0,0,0,0,0,0,0,0,0,0
272 DATA 125,127,32,32,32,32,32,65,84,
65,626
274 DATA 82,73,32,77,69,77,79,82,89,32
,692
276 DATA 84,69,83,84,127,127,127,127,3
2,32,892
278 DATA 32,32,32,98,121,32,65,87,77,3
2,608
280 DATA 65,78,65,76,79,71,32,35,49,49
,599
282 DATA 127,127,155,127,127,127,127,1
27,127,127,1298
284 DATA 127,127,127,155,127,127,127,7
3,84,69,1143
286 DATA 82,65,84,73,79,78,83,127,127,
127,925
288 DATA 127,127,32,32,32,32,32,48,155
,69,686
290 DATA 82,82,79,82,32,64,45,62,36,88
,652
292 DATA 88,88,88,32,84,86,65,76,45,62
,714
294 DATA 36,88,88,32,65,86,65,76,45,62
,643
296 DATA 36,88,88,155,127,127,84,69,83
,84,941
298 DATA 32,32,68,79,78,69,155,0,0,999
,1512
```

CHECKSUM DATA

(See D:CHECK/C:CHECK,p.26)

```
10 DATA 532,746,142,858,536,112,943,67
2,410,270,420,310,184,634,166,6935
65 DATA 697,6,493,543,381,74,544,815,5
47,80,347,906,866,434,989,7722
210 DATA 106,459,154,98,163,916,965,45
2,170,529,186,439,486,189,164,5476
240 DATA 536,238,237,738,115,912,879,9
97,658,42,7,185,785,871,876,8076
270 DATA 283,633,808,254,939,813,635,3
49,57,914,762,802,793,89,8,8139
```

```
10 REM ******************************
15 REM *  MEMTEST DISK FILE MAKER   *
20 REM *       BY TONY MESSINA      *
25 REM *    (C) A.N.A.L.O.G. 83     *
30 REM ******************************
31 GRAPHICS 2:POKE 712,96:? #6;"    ME
MORY  TEST":? #6;"   DiskFile Maker":?
 #6;" ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒*"
32 ? #6;" (c) analog 400/800":? #6;" *
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
33 ? #6;"    by tony messina"
34 POSITION 1,7:? #6;"CREATING disk fi
le":POSITION 0,9:? #6;"cksums"
35 REM
40 DIM PROG$(510):PNTR=7:LINE=198
45 PROG$(1,1)=CHR$(255):PROG$(2,2)=CHR
$(255):PROG$(3,3)=CHR$(128)
50 PROG$(4,4)=CHR$(4):PROG$(5,5)=CHR$(
43):PROG$(6,6)=CHR$(6)
55 LINE=LINE+2:FOR COUNT=1 TO 10
60 READ BYTE:IF BYTE=999 THEN GOTO 85
65 PROG$(PNTR)=CHR$(BYTE)
70 PNTR=PNTR+1:TOTAL=TOTAL+BYTE:NEXT C
OUNT:? "LINE:";LINE
75 READ CHECKSUM:IF CHECKSUM=TOTAL THE
N TOTAL=0:GOTO 55
80 ? "BAD CHECKSUM: LINE ";LINE:STOP
85 TOTAL=TOTAL+BYTE:READ CHECKSUM:IF C
HECKSUM<>TOTAL THEN 80
90 OPEN #1,8,0,"D:MEMTST.OBJ"
95 PRINT #1;PROG$:CLOSE #1:END
100 REM
105 REM ******************************
110 REM *  DISK FILE DATA FOLLOWS   *
115 REM ******************************
120 REM
200 DATA 162,0,134,82,160,5,162,162,32
,133,1032
202 DATA 243,162,38,142,240,2,169,61,1
34,210,1401
204 DATA 32,164,246,166,210,202,16,244
,160,5,1445
206 DATA 162,226,32,133,243,165,88,24,
105,180,1358
208 DATA 133,207,165,89,105,0,133,208,
169,7,1216
210 DATA 133,214,173,230,2,56,233,1,13
3,215,1390
212 DATA 169,9,133,211,169,16,133,209,
96,169,1314
214 DATA 0,168,133,212,133,213,133,205
,133,212,1542
216 DATA 162,2,134,213,165,214,133,206
,166,215,1610
218 DATA 165,212,73,255,133,216,145,20
5,200,208,1812
220 DATA 251,230,206,228,206,176,245,1
66,213,165,2086
222 DATA 214,133,206,165,212,202,16,4,
162,2,1316
224 DATA 145,205,200,208,246,230,206,1
65,215,197,2017
226 DATA 206,176,236,165,214,133,206,1
66,213,165,1880
228 DATA 216,202,16,4,162,2,165,212,20
9,205,1393
230 DATA 208,28,200,208,240,230,206,16
5,215,197,1897
232 DATA 206,176,232,198,213,16,173,16
5,212,73,1664
234 DATA 255,48,161,198,211,16,74,76,1
13,5,1157
236 DATA 132,205,32,134,5,141,18,6,165
,217,1055
238 DATA 141,17,6,160,0,177,205,32,134
,5,877
240 DATA 141,28,6,165,217,141,27,6,165
,205,1101
242 DATA 32,134,5,141,8,6,165,217,141,
7,856
244 DATA 6,165,206,32,134,5,141,6,6,16
5,866
246 DATA 217,141,5,6,160,5,162,251,32,
133,1112
248 DATA 243,160,6,162,30,32,133,243,7
6,120,1205
250 DATA 5,160,0,230,209,165,209,145,2
07,76,1406
252 DATA 197,4,72,74,74,74,74,32,151,5
,757
254 DATA 133,217,104,32,151,5,133,218,
96,41,1130
256 DATA 15,201,10,48,2,105,6,105,48,9
6,636
258 DATA 125,127,32,32,32,32,32,65,84,
65,626
260 DATA 82,73,32,77,69,77,79,82,89,32
,692
262 DATA 84,69,83,84,127,127,127,127,3
2,32,892
264 DATA 32,32,32,98,121,32,65,87,77,3
2,608
266 DATA 65,78,65,76,79,71,32,35,49,49
,599
268 DATA 127,127,155,127,127,127,127,1
27,127,127,1298
270 DATA 127,127,127,155,127,127,127,7
3,84,69,1143
272 DATA 82,65,84,73,79,78,83,127,127,
127,925
274 DATA 127,127,32,32,32,32,32,48,155
,69,686
276 DATA 82,82,79,82,32,64,45,62,36,88
,652
278 DATA 88,88,88,32,84,86,65,76,45,62
,714
280 DATA 36,88,88,32,65,86,65,76,45,62
,643
282 DATA 36,88,88,155,127,127,84,69,83
,84,941
284 DATA 32,32,68,79,78,69,155,65,226,
2,806
286 DATA 227,2,128,4,224,2,225,2,197,4
,1015
288 DATA 999,999
```

•

CHECKSUM DATA

(See D:CHECK/C:CHECK,p.26)

```
10 DATA 532,756,142,858,536,986,943,67
2,383,270,426,547,177,312,187,7727
65 DATA 636,168,700,8,498,675,383,74,5
44,587,547,80,945,417,457,6719
206 DATA 292,222,189,512,452,477,745,6
11,156,569,589,142,776,498,17,6247
236 DATA 991,708,997,883,861,124,132,1
66,720,178,871,633,801,247,932,9244
266 DATA 813,635,342,50,907,762,802,78
6,82,998,872,289,7338
```

•

ATARI* ALERT !!!

GET THEM WHILE THEY'RE HOT

IF YOU OWN AN ATARI* 400/800 AND WOULD LIKE TO BACK UP YOUR AUTO-BOOT (BINARY)PROGRAMS ON CASSETTE OR DISK, THESE ARE THE UTILITIES FOR YOU.

(1) **BOOT TAPE BACKUP:** ALLOWS YOU TO COPY AUTO-BOOTCASSETTESWITHOUT A DISK DRIVE. ALSO DISPLAYS A HEX LIST OF BOOT CASSETTE

(2) **TAPE TO DISK:** BRING YOUR AUTO-BOOT CASSETTESUP TO DISK (BINARY FILE).

(3) **DISK TO TAPE:** LETS YOU MAKE AN AUTO-BOOT CASSETTE FROM A BINARY DISK FILE.

NOTE:

- THESE UTILITIES REQUIRE 48K.
- THEY WILL NOT DO MULTISTAGE LOADS.
- PROGRAMS ARE NOT INTENDED FOR PIRATING BUT FOR BACKING UP PERSONAL ARCHIVES.
- BECAUSE IT IS POSSIBLE THAT THIS PROGRAM MAY BE REPRODUCED, THERE IS A GUARANTEE OF REPLACEMENT ONLY-NO REFUNDS.
- *REGISTERED TRADE MARK OF ATARI
- ALLOW 4-6 WEEKS FOR DELIVERY

MAIL TO:
PROGRAMS PLUS
P.O. BOX 369
DEER PARK, NEW YORK 11729
ORDER BY PHONE #516-242-1945

NAME _____
STREET _____
CITY _____
STATE/ZIP _____

✓ CHECK PROGRAM DESIRED:
( ) BOOT TAPE BACK-UP
( ) TAPE TO DISK
( ) DISK TO TAPE

CHECK ONE	CHECK ONE
( ) 1 FOR $29.95	( ) CHECK OR MONEY ORDER
( ) 2 FOR $39.95	( ) MASTERCARD
( ) 3 FOR $49.95	( ) VISA

CREDIT CARD # _____

ADD $2.50 FOR SHIPPING AND HANDLING

IF YOU HAVE WRITTEN ORIGINAL SOFTWARE FOR THE ATARI 400/800 AND WOULD LIKE TO SUBMIT A NON-RETURNABLE BACK-UP COPY, WE WILL EVALUTE YOUR PROGRAM FOR POSSIBLE NATIONAL MARKETING. SEND TO: PROGRAMS PLUS, P.O. BOX 369, DEER PARK, NEW YORK 11729.

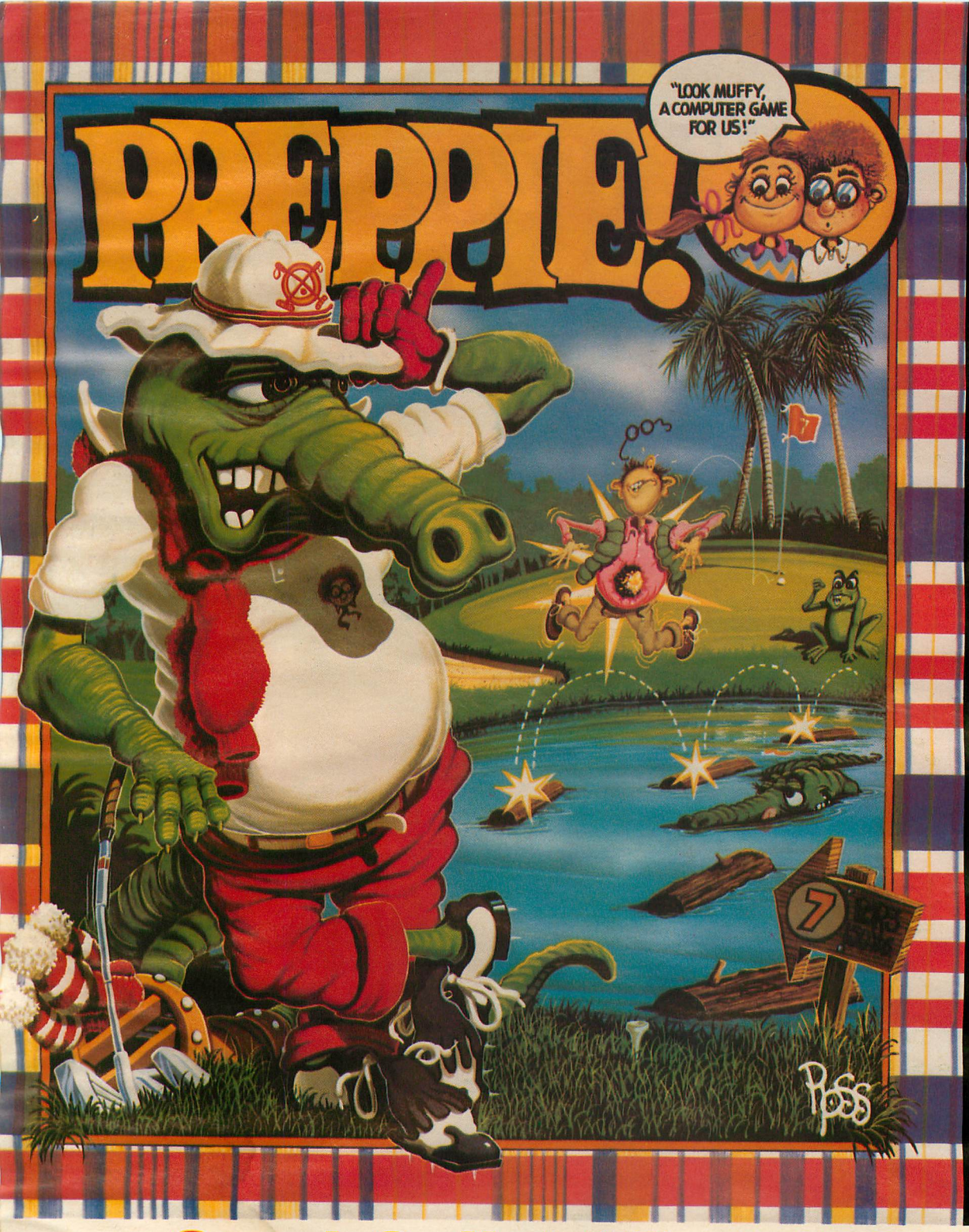
Hello, We're Adventure International

And we're publishers of some of the finest microcomputer software programs available.
If you can write a **top-quality** program, or can convert some of our best-sellers to other computers, we want to hear from you — Now.
We have the advertising, international distribution, manufacturing and marketing know-how to send top-quality programs to the top of the charts.
If your program **is top quality** — give us a call, or write for our Adventure International Author Information Kit.

We are publishers of the top-selling Scott Adams Adventure Series and other fine Entertainment and Applications Programs.

Adventure INTERNATIONAL

Box 3435
Longwood, Florida 32750
Telephone: (305) 862-6917
Ask for Author Assistance



PREPPIE II
IS HERE!

That's right you LaCoste-clad Atari owner. Wadsworth Overcash, our Weejuned young prepster continues his exploits — this time at college and just like daddy, he's decided to pledge a fraternity. Have we got surprises for him (and you)!

AVAILABLE NOW
AT FINE COMPUTER STORES EVERYWHERE

by Russ Wetmore

Good Golly! What A GAME!

The original arcade game **PREPPIE!** will give you hours of toe-tickling fun. **You** take the controls, moving your young prepster about on the golf course. And what a golf course! It's unlike any you have ever played on before. There are dangers **everywhere,** *and only you* can keep the little Ivy Leaguer from a fate worse then Penn State!

PREPPIE! is written *in state-of-the-art machine language,* is joystick compatible and fully utilizes the Atari's sound and graphics capabilities. Quite frankly, *it will give you the most fun you can have on an Atari microcomputer.*

PREPPIE! is available at fine computer, book, and hobby stores everywhere.

An exclusive game from

Adventure INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.

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

Published by ADVENTURE INTERNATIONAL
a division of Scott Adams, Inc.
BOX 3435 • LONGWOOD, FL 32750 • (305) 862-6917

"My Atari never did things like this before!"
—Holister Townsend Wolfe

"I had so much fun I almost blew my doughnuts."
—Theodore Boston III

"I haven't had this much fun since Buffy and I went to Princeton for the weekend."
—Martha Vineyard

A DISK CATALOGING UTILITY

16K disk

by Bert Williams and Tom Hamel

Anybody who owns and uses a disk drive quickly accumulates many diskettes loaded with various programs and utilities. It is not uncommon to wonder what files are on a particular diskette or on what diskette(s) a particular file can be found. CATALOG.BAS (see listing) creates a listed file (*FILENAME.LST*) that stores the above information for later retrieval, again by running CATALOG.BAS. The program is selfprompting and quite friendly. Simply type in listing #1 and save it with SAVE "D:CATALOG.BAS" or whatever file name you wish to use.

A. **Creating FILENAME.LST.**

Run CATALOG.BAS and select from the main menu option "1 FILE MAKER." Number the diskettes you wish to catalog using numbers from 1 to 726. You need not number the diskettes consecutively and you can catalog the diskettes in any order. When prompted, press "1 IF NEW *FILE*," place a numbered diskette in drive one, and press RETURN. CATALOG.BAS reads the disk directory and creates data statements with the information in the directory. Continue until all diskettes have been cataloged. Pressing 'O' will list FILENAME.LST to the disk in drive one. But first, be sure the diskette in drive one is the diskette you wish to write this file to.

B. **Updating FILENAME.LST.**

This is the same as A above except that when prompted you place the disk with FILENAME. LST in D1 then press "2 IF UP-DATE." You may update with a diskette number that has or has not been previously cataloged. Note: You can catalog diskettes that do not have a directory by interrupting the program with the BREAK key and adding the data statements as needed. For diskette n, data statements are created at lines 1000+40*n, 1000+40*n+2, etc., as needed. Study the data statements created by the program (lines 1040-30100) to see the format needed. If data statements are user created you still must go through the file-maker-update options to get this information written to FILENAME.LST on the disk.

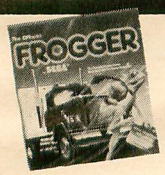
C. **Finding a File**

Run CATALOG.BAS and select from the main menu option "2 File Finder." From the file search menu select the "1 File Name" option and then indicate the name of the file you want to find. All occurrences, if there are any, of the indicated file will be listed on the screen. Wild cards will not work with this option.

D. **Listing the Contents of a Disk**

From the file search menu select the "2 Disk

Great Software • Great Service • ~~Great~~ incredible Prices



from Sierra On-Line

Frogger

16K Tape / 32K Disk **$23.88**

The official Atari computer version of the smash coin-op hit. Beautiful graphics. The disk version features seven different tunes.

All Sierra On-Line programs, including Ulysses & The Golden Fleece and The Wizard & The Princess in stock and available at similar savings.

from Lightning

Mastertype

32K Disk **$26.88**

A breakthrough in interactive educational software. Learn to type as you battle wave after wave of invading words riding in hi-res spaceships! 17 pre-programmed lessons, plus others that you can create, guarantee continuous challenge and help you develop mastery of the Atari Keyboard.

from SSI

The Cosmic Balance

48K Disk **$27.88**

Design and build starships, then battle it out in this combination strategic/arcade offering from SSI's Rapid Fire series.

All SSI programs including Tigers in the Snow and The Battle of Shiloh in stock and available at similar savings.

from Epyx

Crush, Crumble & Chomp

32K Tape or Disk **$21.88**

Take on the persona of any of six demonic horror-film beasties (or create your own on disk). Then, pick out a mouth-watering metropolis and head for those buildings and bridges.

All Epyx programs including Temple of Apshai, Dragon's Eye and Monster Maze in stock and available at similar savings.

from Datasoft

Text Wizard

48K Disk **$68.88**

A powerful, easy-to-use word processor. Perfect for writers, students and small business people. Recommended for use with Epson, Prowriter, NEC 8023 and Atari 825 line printers.

All Datasoft programs, including Zaxxon and Sands of Egypt available at similar savings.

from Infocom

Zork I, II, or III

32K Disk **$26.88**

your choice:

The feature-loaded Rolls Royce of text adventure games. These three games are so popular that there's now even a National Zork Users Group!

All Infocom programs including Deadline and Starcross in stock and available at similar savings.

from Sirius

Bandits

48K Disk **$23.88**

Bouncing nerve gas balloons, energy shields, space bandits attempting to steal your supplies — they're all on the screen in one of the best space arcade games yet.

All Sirius Software including Way Out and Sneakers in stock and available at similar savings.

from Thorn-EMI

Home Financial Management

16K Tape **$22.88**

Set up a budget and then see where you stand financially either in figures or through bar charts. This program should pay for itself!

All Thorn-EMI tapes including Figure Fun available at similar savings.

Please call or write for FREE CATALOG

Prices good thru May 31, 1983.

from MUSE

Castle Wolfenstein

48K Disk **$19.88**

One of the most popular games ever released for the Apple is now available to you. Try to find the war plans and escape from the Nazi stronghold in this exciting blend of strategy and arcade action.

from Thorn-EMI

Soccer

ROM Cart. **$34.88**

In our opinion, this is **the** best arcade sports game available today. Play vs. the computer or human opponents. Fantastic graphics.

All Thorn-EMI Cartridges including Submarine Commander and Jumbo Jet Pilot in stock and available at similar savings.

from Spinnaker

Facemaker

48K Disk **$23.88**

Let your kids learn to write simple programs while they have a ball creating and animating faces. It's more fun than Mr. Potato Head ever was! Ages 4 up.

All Spinnaker Software including Story Machine and Snooper Troops #1 and #2 in stock and available at similar savings.

from Broderbund

Choplifter

48K Disk **$23.88**

This was **the** Apple best-seller of 1982. Now it's your turn to battle tanks and fighter planes as you rescue hostages with your trusty chopper.

All Broderbund Software including Labyrinth, Stellar Shuttle and David's Midnite Magic in stock and available at similar savings.

MasterCard VISA

Mail order prices do not apply at our retail location.

HOW TO ORDER:

Use mail or phone. For immediate shipment pay by Money Order, Certified Check, Cashiers Check, VISA, or Master Charge (add 4% for credit card). Personal checks are accepted but require 1-3 weeks to clear. 10% required for C.O.D.

SHIPPING & HANDLING:

$2.50 for orders under $50; free shipping over $50; Mass. residents add 5% sales tax. Prices are mail order only and are subject to change without notice.

SOFTWARE PLUS

(617) 369-1992

for orders and information

phone hours: 10-6 EST Mon.-Sat.

45 Walden St., Box AA, Concord, Mass. 01742

Number" option and input the number of the disk you want searched. All files on this disk and sectors used/free information will be listed.

E. **List of Cataloged Disks**

Option "3 Cataloged Disks" of the file search menu lists all the disks that you have cataloged. This could be useful before using the "2 Disk Number" option.

Errors have not been trapped within the program so it is possible to cause the program to abort by inputting a wrong data type. Also, the break key can be used to stop execution since it has not been disabled. Should you unintentionally cause the program to abort simply run the program again and continue as before. The program and data statements created should still be in the computer. □

Following is a description of the major components of the program listing as referenced by *line numbers*.

10-25:	Initialize variables
30-65:	Heading
70:	Dimension strings
75-110:	Main menu options
115-125:	Enter FILENAME.LST if not already in machine
130-160:	File Search menu
165-305:	File and disk searching procedure
310-335:	File Maker menu options
340-495, 530-550:	Creates or updates and writes FILENAME.LST to disk
500-525:	Headings for file maker/file finder menus
555-635:	Various subroutines used in above procedures

```
10 READ Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,
Q10,Q11,Q12,Q13,Q14,Q17,Q20,Q22,Q30,Q3
1,Q34,Q43,Q75,Q82,Q125,Q255,Q500
15 READ Q535,Q555,Q559,Q709,Q710,Q712,
Q752,Q764,Q842,Q999,Q1000
20 DATA 0,1,2,3,4,5,6,7,8,9,10,11,12,1
3,14,17,20,22,30,31,34,43,75,82,125,25
5,500,535,555,559,709,710,712,752
25 DATA 764,842,999,1000
30 GRAPHICS Q1+Q17:? #Q6:? #Q6:? #Q6;"
   ▛▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▜"
35 ? #Q6;"   ▌ disk catalog  ▐"
40 ? #Q6;"   ▌      BY       ▐"
45 ? #Q6;"   ▌  bert williams ▐"
50 ? #Q6;"   ▌      AND      ▐"
55 ? #Q6;"   ▌   tom hamel    ▐"
60 ? #Q6;"   ▌CLARKSVILLE,TN. ▐"
65 ? #Q6;"   ▙▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▟":FOR WAI
T=Q1 TO Q1000*Q2:NEXT WAIT
70 DIM FILE$(Q17),C$(Q17),FA$(Q17)
75 GOSUB Q555:GRAPHICS Q1+Q17:Z=53279:
POSITION Q2,Q0:? #Q6;"PROGRAM  OPTIONS
":POSITION Q3,Q2:? #Q6;" 1 file maker"
76 POSITION Q1,Q1:? #Q6;"2":POSITION Q
1,Q2:? #Q6;"2":POSITION Q1,Q3:? #Q6;"3
":POSITION Q1,Q4:? #Q6;"4":POSITION Q1
,Q5:? #Q6;"5"
80 POSITION Q4,Q3:? #Q6;"2 file finder
":POSITION Q4,Q4:? #Q6;"3 exit program
"
85 POSITION Q9,Q6:? #Q6;"ON":POSITION
Q1,Q8:? #Q6;"PRESS START  FOR 1"
90 POSITION Q1,Q9:? #Q6;"PRESS SELECT
FOR 2":POSITION Q1,Q10:? #Q6;"PRESS OP
TION FOR 3"
95 B=PEEK(Q764):IF PEEK(Z)=Q6 OR B=Q31
 THEN GOTO Q31*Q10
100 IF PEEK(Z)=Q5 OR B=Q30 THEN GOTO Q
125-Q10
105 IF PEEK(Z)=Q3 OR B=Q20+Q6 THEN POK
E Q764,Q255:END
110 GOTO Q82+Q13
115 RESTORE Q1000:READ D,FILE$,S
120 IF D<>Q999 THEN GOTO Q125+Q5
125 GOSUB Q500+Q125+Q10:? "3"
130 GOSUB Q535-Q20
135 POSITION Q5,Q3:? "type number for
kind of listing":POSITION Q12,Q10:? "1
 file name":POSITION Q12,Q12
140 ? "2 disk number":POSITION Q12,Q14
:? "3 cataloged disks":POSITION Q12,Q1
4+Q2:? "4 program options"
145 B=PEEK(Q764):IF B=Q30 OR B=Q31 THE
N GOTO Q20*Q8
150 IF B=Q20+Q4 THEN POKE Q764,Q255:?
CHR$(Q125):GOTO Q75
152 IF B=Q20+Q6 THEN GOTO Q20*Q30
155 GOTO Q125+Q20
160 GOSUB Q555:IF B=Q30 THEN GOTO Q17*
Q10
165 ? "type in name of desired file":I
NPUT C$:POSITION Q10,Q2:? "I'm looking
...":GOTO Q82*Q2+Q11
170 ? "type in disk number desired";:I
NPUT X:IF X=Q999 THEN GOTO Q13*Q20
175 RESTORE Q1000+(Q30+Q10)*X
180 READ D,FILE$,S:IF FILE$="END OF DA
TA" THEN GOTO Q255
181 IF D<>X AND C$=" " THEN POSITION C
,L:? "SORRY I COULD NOT FIND DISK ";X
;"!":GOTO Q125*Q2+Q30
185 IF D=X THEN U=U+S:IF S=Q0 THEN ? "
 ";FILE$(Q1,Q8);"/";U;" USED":GOTO Q25
5+Q10
190 IF FILE$=C$ AND L<Q6 THEN ? CHR$(Q
125):POSITION Q10,Q2:? "I'm looking...
":FA$=C$
195 IF L=Q10+Q13 THEN POSITION Q4,Q2:?
 "I'M GOING TO CLEAR THE SCREEN":FOR W
AIT=Q1 TO Q1000:NEXT WAIT:L=Q5
200 IF FILE$=C$ THEN L=L+Q1:POSITION C
,L:? FILE$;" ";S;" IS ON DISK # ";D
205 IF F>Q9 THEN H=Q1:IF F>Q10+Q11 THE
N H=Q20+Q1:IF F>Q12+Q30 THEN H=Q2
210 POSITION H,T+Q1:IF S=Q0 THEN GOTO
Q125*Q2-Q20
215 IF D=X THEN ? F;" ";FILE$;
220 IF D=X AND H>=Q2 THEN GOSUB Q559+Q
11:? S:F=F+Q1:T=T+Q1
225 IF D=X AND H<Q2 THEN GOSUB Q559+Q1
1:? S:F=F+Q1:T=T+Q1:IF F=Q22 THEN POKE
 Q82,Q20:T=Q0:POSITION Q20,Q1
230 IF S=Q0 THEN IF D=X THEN ? F;" ";F
ILE$;" ":F=F+Q1:T=T+Q1:IF F=Q22 THEN P
OKE Q82,Q20:T=Q0:POSITION Q20,Q1
235 IF F=Q43 THEN POKE Q82,Q2:H=Q2:T=Q
0:POSITION Q2,Q22:? "THERE IS MORE--TO
 LIST PRESS RETURN"
240 IF F=Q43 THEN B=PEEK(Q764):IF B=Q1
2 THEN ? CHR$(Q125):GOTO Q30*Q6
245 IF F=Q43 THEN GOTO Q12*Q20
250 GOTO Q30*Q6
255 IF X=Q0 THEN GOTO Q125*Q2+Q20
260 IF D=Q999 AND F=Q1 THEN POSITION C
,L:? "SORRY I COULD NOT FIND DISK ";X
;"!":GOTO Q125*Q2+Q30
265 POSITION Q1,Q0:? "      Disk  #";
X;"                ":POKE Q82,Q2:POKE
Q752,Q0
270 IF X=Q0 AND C$<>FA$ THEN POSITION
C,L:? "I COULD NOT FIND ";C$;"!":? :?
 "      PLEASE RECHECK YOUR DATA"
280 POSITION Q10,Q22:IF X<>Q0 THEN POK
E Q752,Q1:? "Another disk? Y/N"
285 POSITION Q10,T+Q2:IF X=Q0 THEN ? "
another file? Y/N"
```

```
290 B=PEEK(Q764):IF B=Q43 AND X=Q0 THE
N GOSUB Q555:GOTO Q82*Q2+Q1
295 IF B=Q43 AND X<>Q0 THEN GOSUB Q555
:GOTO Q17*Q10
300 IF B=Q30+Q5 THEN GOTO Q75
305 GOTO Q30*Q10-Q10
310 POKE Z,Q7:POKE Q764,Q255
315 GRAPHICS Q1+Q17:POSITION Q0,Q1:? #
Q6;" press 1 IF new file":POSITION Q0,
Q3:? #Q6;" press 2 IF update"
320 POSITION Q0,Q5:? #Q6;"IF UPDATE PL
ACE DISK WITH 'filename.lst'          IN
 D1"
322 POSITION Q1,Q10:? #Q6;"press 3 FOR
 options "
325 B=PEEK(Q764):IF B=Q31 THEN GOTO Q1
25*Q3-Q20
330 IF B=Q30 THEN T=Q30:GOTO Q34*Q10
332 IF B=Q20+Q6 THEN POKE Q764,Q255:GO
TO Q75
335 GOTO Q255+Q75-Q5
340 RESTORE Q1000:READ D,FILE$,S
345 IF D<>Q999 THEN GOTO Q125*Q3-Q20
350 GOSUB Q500+Q125+Q10:? ""
355 POKE Z,Q7:GRAPHICS Q1+Q17:POSITION
 Q2,Q0:? #Q6;"number your disks

"
360 POSITION Q3,Q2:? #Q6;"put disk in
D1":POSITION Q3,Q5:? #Q6;"when prompte
d":POSITION Q4,Q4:? #Q6;"type disk "
365 POSITION Q3,Q7:? #Q6;"press return
":POSITION Q4,Q8:? #Q6;"WHEN READY"
370 B=PEEK(Q764):IF B=Q12 THEN GOSUB Q
500:GOTO Q255+Q125+Q10
375 GOTO Q255+Q125-Q10
380 GOSUB Q500:POSITION Q9,Q4:? "if do
ne type 0":POSITION Q9,Q5:? "and press
 RETURN":POSITION Q17,Q7:? "OR"
385 POSITION Q9,Q9:? "change disk "
390 POSITION Q9,Q11:? "type number of
disk":POSITION Q10,Q13:? "and press RE
TURN":POSITION Q17,Q17
395 INPUT X:IF X=Q0 THEN GOTO Q500-(Q3
0+Q5)
400 IF T=Q30 THEN ? CHR$(Q125):POKE Q5
59,Q0:POSITION Q2,Q4
405 IF T=Q30 THEN FOR I=Q1000+(Q30+Q10
)*X+Q2 TO Q1000+(Q30+Q10)*X+Q22+Q2 STE
P Q2:? I:NEXT I:GOSUB Q535
410 POKE Q559,Q0:LINE=Q1000+(Q30+Q10)*
X:C=X:OPEN #Q1,Q6,Q0,"D1:*.*"
415 ? CHR$(Q125):POSITION Q2,Q4:? LINE
;" DATA ";:FOR W=Q1 TO Q5
420 INPUT #Q1,FA$:IF FA$(Q5,Q8)="FREE"
 THEN ? C;",";FA$;",0":GOTO Q43*Q10+Q3
0
425 ? C;",";:FOR I=Q3 TO Q10:IF FA$(I,
I)<>" " THEN ? FA$(I,I);:NEXT I
430 IF FA$(Q11,Q11)<>" " THEN ? ".";
435 FOR I=Q11 TO Q13:IF FA$(I,I)<>" "
THEN ? FA$(I,I);:NEXT I
440 ? ",";FA$(Q2+Q13,Q17);
445 ? ",";:IF W=Q5 THEN ? "":REM ESC
DELETE
450 IF W=Q5 THEN GOSUB Q535
455 NEXT W:LINE=LINE+Q2:GOTO Q500-(Q75
+Q10)
460 CLOSE #Q1:GOSUB Q535:GOTO Q255+Q12
5
465 GOSUB Q500:POSITION Q10,Q6:? "a ne
w file will":POSITION Q10,Q7:? "be wri
tten to D1."
470 POSITION Q10,Q8:? "be sure the dis
k":POSITION Q10,Q9:? "has enough room.
":POSITION Q10,Q10:? "press RETURN whe
n ready"
475 B=PEEK(Q764):IF B<>Q12 THEN GOTO Q
500-(Q20+Q5)
480 POKE Q559,Q0:? CHR$(Q125):POSITION
 Q2,Q4:? "LIST";CHR$(Q34);"D1:FILENAME
.LST";CHR$(Q34);",";Q1000;",";Q1000*Q3
0
485 GOSUB Q535:GOSUB Q500
490 POKE Q559,Q34:POSITION Q7,Q10:? "a
 file 'FILENAME.LST' has              b
een  LISTED  on this disk."
495 ? "":FOR WAIT=Q1 TO Q1000:NEXT WA
IT:GOTO Q75
500 POKE Q764,Q255:GRAPHICS Q0:POKE Q7
52,Q1:POKE Q709,Q125+Q30:POKE Q710,Q0:
POKE Q712,146:POKE Q559,Q34:POKE Q82,Q
2
505 I=PEEK(560)+PEEK(561)*256+Q6:POKE
I-Q3,Q82-Q12:FOR J=Q0 TO Q2:POKE J+I,Q
6:NEXT J
510 POSITION Q3,Q0:? #Q6;"disk  direct
ory":POSITION Q5,Q1:? #Q6;"file  maker
":RETURN
515 POKE Q764,Q255:GRAPHICS Q0:POKE Q7
52,Q1:POKE Q709,Q125+Q30:POKE Q710,Q0:
POKE Q712,146:POKE Q559,Q34:POKE Q82,Q
2
520 I=PEEK(560)+PEEK(561)*256+Q6:POKE
I-Q3,Q82-Q12:FOR J=Q0 TO Q2:POKE J+I,Q
6:NEXT J
525 POSITION Q3,Q0:? #Q6;"disk  direct
ory":POSITION Q5,Q1:? #Q6;"file search
":RETURN
530 ? CHR$(Q125):POKE Q559,Q0:POKE Q76
4,Q255:POSITION Q2,Q4:? "ENTER";CHR$(Q
34);"D1:FILENAME.LST";CHR$(Q34)
535 ? :? :? :? "CONT":POSITION Q2,Q0
540 POKE Q842,Q13:STOP
545 POKE Q842,Q12
550 RETURN
555 ? CHR$(Q125):C=Q5:L=Q5:U=Q0:F=Q1:H
=Q2:T=Q0:C$=" ":FA$=" ":K=Q0
560 GRAPHICS Q0:POKE Q709,Q20+Q6:POKE
Q710,Q17:POKE Q712,Q16+Q16:POKE Q764,Q
255:POKE Q752,Q1:RETURN
570 C=H:IF H>=Q2 THEN C=C-Q1
575 IF S<Q10 THEN POSITION C+Q17+Q1,T+
Q1:GOTO Q559+Q31
580 IF S<Q10*Q10 THEN POSITION C+Q17,T
+Q1:GOTO Q559+Q31
585 POSITION C+Q14+Q2,T+Q1
590 RETURN
600 X=Q1:? CHR$(Q125):GOSUB Q535-Q20:P
OSITION Q4,Q1:? #Q6;"disk  numbers":?
605 RESTORE Q1000+X*Q2*Q20
610 READ X:IF X=Q999 THEN GOTO Q500+Q1
25-Q5
613 IF PEEK(Q82+Q3)>=Q30+Q5 THEN ? :?
615 ? X;" ";:X=X+Q1:GOTO Q30*Q20+Q5
620 POSITION Q8,Q20+Q1:? "Press RETURN
 for menu"
625 B=PEEK(Q764):IF B<>Q12 THEN GOTO Q
500+Q125
630 GOTO Q125+Q5
635 GOSUB Q555:POSITION Q3,Q10:? "Stan
d by while I load the file...":FOR WAI
T=Q1 TO Q752:NEXT WAIT:GOSUB Q535-Q5:R
ETURN
30100 DATA 999,END OF DATA,-1
```

•

CHECKSUM DATA

(See D:CHECK/C:CHECK,p.26)

```
10 DATA 266,444,239,822,52,897,941,994
,257,346,498,88,228,418,627,7117
80 DATA 1,870,40,142,838,933,411,226,1
17,781,478,879,708,354,251,7029
152 DATA 275,177,231,328,202,701,80,18
9,766,36,824,360,797,599,678,6243
220 DATA 568,576,645,142,816,916,154,3
39,267,169,66,553,986,48,414,6659
300 DATA 729,540,884,382,724,892,675,6
35,553,572,234,685,789,301,307,8902
365 DATA 329,461,38,458,48,179,830,604
,664,32,786,121,19,50,515,5134
440 DATA 876,463,602,818,974,520,64,27
,15,639,993,268,509,498,797,8063
515 DATA 517,499,429,445,809,994,185,6
04,47,948,680,141,284,535,616,7733
600 DATA 136,313,694,718,288,267,881,3
48,428,537,4610
```

•

WHEN YOU SIT DOWN TO A SCOTT ADAMS' GRAPHIC ADVENTURE . . .

. . . ANYTHING CAN HAPPEN!

That's right — anything can happen, and it usually does! Because with a Scott Adams Graphic Adventure, the fantastic is as close as your computer — and your own fertile imagination. Each S.A.G.A. features the finest in hi-resolution graphics — graphics which compliment and enhance the classic text — and, they can be toggled on or off with a single keystroke, too. Plus, optional Votrax Type 'N Talk™ and printer compatibility give you the flexibility to add new dimensions to your Adventuring — when you're ready.

The incomparable S.A.G.A. Series — experience the magic yourself. But be prepared for anything.

S.A.G.A. #1 — Adventureland (Skill Level: Moderate)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0201	$39.95
ATARI 48K Disk	052-0201	$39.95

S.A.G.A. #2 — Pirate Adventure (Skill Level: Beginner)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0202	$39.95
ATARI 48K Disk	052-0202	$39.95

S.A.G.A. #3 — Mission Impossible (Skill Level: Advanced)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0203	$39.95
ATARI 48K Disk	052-0203	$39.95

S.A.G.A. #6 — Strange Odyssey (Skill Level: Moderate)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0206	$39.95
ATARI 48K Disk	052-0206	$39.95

PRICES SUBJECT TO CHANGE

Also, look for the Scott Adams Text Adventure Series on these fine computers:
TRS-80 • COMMODORE VIC • NEC PC-8001
TEXAS INSTRUMENTS 99/4 • CP/M Z-80

Adventure INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC

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

ADVENTURE IN THE FIFTH DIMENSION!

16K Cassette/24K Disk

by Brian Moriarty

Adventure is an entertainment ideally suited to the home computer. No arcade game can offer the logical challenge and intellectual involvement of a good adventure program.

Unfortunately, most of the adventures available for the ATARI require more than 16K of memory to run. This prevents many owners of the ATARI 400 System from enjoying the excitement of adventure. And very few text adventures take advantage of the ATARI's unique hardware capabilities to produce a clear, easy-to-read display.

A.N.A.L.O.G.'s **Adventure in the 5th Dimension** addresses both of these problems. It's a beginner's-level text adventure with all the features you'd expect in a commercial product — a large vocabulary, blinking cursor, independent scrolling window and the ability to save and load games.

Don't be fooled by the fact that **The 5th Dimension** is written in ATARI BASIC. The program uses machine-language subroutines to give you speed performance that rivals even the most expensive commercial adventures. Best of all, the whole thing will fit (just barely) in a 16K cassette-based system!

THE GAME SCENARIO

Few of our national treasures are more secure than the original Declaration of Independence. It's kept at the Library of Congress in a closely guarded display case, sealed in an atmosphere of inert gas to preserve its integrity. The entire display can be retracted deep into the earth at the touch of a button. Stored in this underground vault, the Declaration is capable of surviving the most vicious enemy assault, including a thermonuclear attack.

One afternoon, as a gaggle of tourists stood admiring the priceless document, a shining door of light appeared over the display case. The crowd watched in helpless amazement as alien beings reached their 5-dimensional fingers through the

bulletproof glass, snatched the Declaration and vanished without a trace!

You are a top-notch private investigator, hired by the government to retrieve the Declaration. You must search the city of Washington for clues, find a way into the alien universe, locate the Declaration and return it to the police station.

In your search, you will encounter weird technologies and multi-dimensional terrors never before seen. It takes skill and insight to outwit the aliens — and plenty of patience to navigate the streets of Washington!

Typing It In

You will notice that our source listing for **The 5th Dimension** is un-encoded; that is, the code has not been scrambled to prevent you from seeing possible clues. We decided not to scramble our listing because encoded programs are very hard to type, and they make it nearly impossible to learn how the program works.

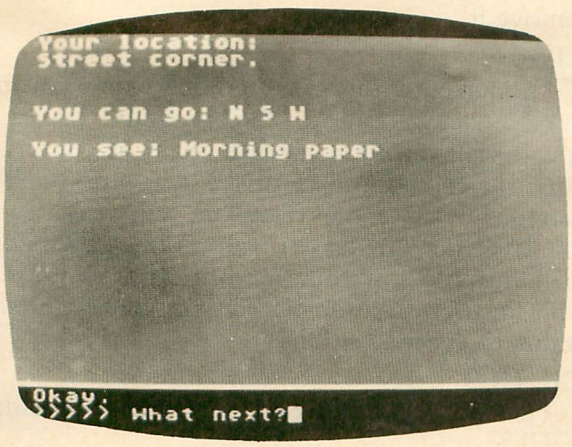
It is VERY IMPORTANT that you type each line of the program EXACTLY as you see it printed. Include all of the spaces and control characters; yes, even the REM statements! Save every few lines of new code in case your cat pulls out the power cord. Be especially careful with the DATA statements at the end of the program. And don't try to RUN anything until you have used D:CHECK or C:CHECK to guarantee the accuracy of your work.

Playing The Game

After your copy of **The 5th Dimension** has passed the CHECK routine without errors, SAVE a perfect copy into a disk or tape. Now you can type RUN. The title screen will appear along with the message "Initializing." After about five seconds, you'll see the following prompt:

Press START to being new game.
Press OPTION to restore old game.

Press the START key. Your screen should now look like this:



Adventure in the 5th Dimension

The screen is divided into five imaginary text areas or "windows." The black **response window** at the bottom accepts commands from the user, and displays descriptions of objects and the results of your actions. A blinking cursor in the response window indicates that the program is waiting for new commands.

The **location window** at the top of the screen gives you a brief description of your immediate surroundings. Underneath it is the **compass window**, which indicates all of the possible exits from that location.

The **objects window** shows a list of all objects visible at the current location. The bottom of the blue screen area is the **inventory window**, which lists whatever items you may be carrying.

Like most text adventures, **The 5th Dimension** understands two-word sentences in the form VERB-(space)-NOUN. Try typing the sentence TAKE PAPER on the starting screen. The "Morning paper" will vanish from the objects window and reappear in your inventory window.

You can interact with objects on the screen just like you can in real life. Watch the response window as you type the sentences EXAMINE PAPER and READ PAPER. When you're done, the sentence DROP PAPER will return it to the object window.

Part of the fun of adventuring is finding out which verbs and nouns the program will understand. If you type an illegal or misspelled word, you'll see the message "I don't understand — try again" in the response window.

Single-Character Commands

The 5th Dimension also understands a limited number of single-character commands. These are used to control your movement, and to perform other special functions.

MOVEMENT COMMANDS:
N — North S — South E — East W — West
U — Up D — Down

OTHER COMMANDS:
I — Inventory H — Help Q — Quit/Save Game

The movement commands let you go in any of the directions indicated in the compass window. The "I" command updates the inventory window so you can see what you're carrying. The inventory window is also updated whenever you TAKE, DROP or THROW anything.

The "H" command will give you a brief hint appropriate for that location. "Q" is used to exit the program, and to save games.

Saving And Loading Games

The 5th Dimension allows you to save your current game status on disk or tape. To use this feature, make sure your storage device is properly connected and loaded with a blank tape or formatted disk. Type the command Q (Quit) and answer Y to the "Save

Game?" prompt. Then indicate whether you are saving to disk or tape.

Don't be frightened when the screen goes black. The program turns off ANTIC during the game saving and loading functions to prevent the display from going crazy. When the save is complete, the screen *will* return in all its glory and you can continue the game. I/O errors will cause the console to squawk with irritation, and the "Disk or Tape?" prompt will reappear.

To load a previously saved game, type Q/RETURN and then RETURN again to exit. RUN the program again and press the OPTION key after the initialization is complete. When the screen appears, the game *will* be restored to exactly the way it was when you last saved it.

Don't try aborting the game with the BREAK key. The program disables it to prevent you from crashing the machine-language routine that blinks the cursor. Hit SYSTEM RESET and you will return safely to BASIC without erasing the program.

Hints For Successful Adventuring

1. Draw a map. You'll get hopelessly lost in the streets of Washington if you don't draw a map. The alien universe also contains areas that can be tricky to navigate without careful mapping.

The easiest way to map an adventure maze is use the "Hansel and Gretel" technique. Drop an item to mark your place and move one step in each direction, noting where it brings you. Then retrieve the item, drop it somewhere else and repeat the procedure.

The method works even better if you have more than one item to drop. Be careful - some directions loop around in circles, while others bring you back to the location you just left! With careful and methodical mapping, you should be able to figure out the entire street maze in less than 15 minutes.

2. Use the "H" (Help) command. Different locations have different hints. Some of these hints may prove very useful.

3. Examine everything. Objects may have important features that will not be evident unless you examine them closely. You should also keep track of the items you discover — most of them are essential to your success.

4. Save your game frequently. Use the "Q" command to save your current status after every important discovery, and before you try anything that might be dangerous. Otherwise you'll have to go back to the street corner and start all over again.

5. Try anything. Don't be afraid to find out what you can or can't do. The worst that can happen is that you *will* be blown into a million pieces.

6. Don't give up hope. It *is* possible to retrieve the Declaration and return it to the police station — I promise! If you're really stuck, ask for other peoples' suggestions. A fresh outlook might uncover a solution you didn't think of.

7. Don't call A.N.A.L.O.G. We are absolutely, positively and definitely not giving adventure hints over the telephone! If you're really stuck, drop us a desperate note and we'll print it in our next "Letters" column along with an appropriate reply. And remember to run D:CHECK or C:CHECK on the program before you try to play it. One byte in the wrong place can make **Adventure in the 5th Dimension** as hard to beat as an alien force field! □

Program Variables

C0-C710

Common variables. You can save lots of memory by defining commonly used constants as variables. I use the "C" prefix to indicate "common." Unfortunately, I couldn't use variables for common line references because I wanted to make the program compatible with the Datasoft BASIC Compiler. See review elsewhere in this issue.

Q,W,I,X,Y,Z

General-purpose working variables.

FLAG

Used to indicate whether the inventory window should be updated. See line 75.

M$

The master location matrix. It contains the data which determines the objects and legal exits available at each location. M$ is divided into 24 16-character sections or modules, each holding the data for one location. The first character position in each module is a unique letter from A-Y which identifies that location. The next six positions correspond to the six possible directions of movement. A letter in any position indicates which location you *will* enter if you go in that direction. A "?" character indicates no exit in that direction. Positions 8-16 contain "?" characters unless an object is present; if so, the code letter for that object is inserted in any location to a maximum of 8. Position 7 was a flag byte used in an early version of the program. I was too lazy to remove it.

CL$

The current location buffer. Every time you move into a new location, the program copies the appropriate location module from M$ into CL$. Any changes in the status of a location (objects dropped or taken, new exits opened, etc.) are made in CL$; when you leave that location, CL$ is copied back into M$ and a new module is copied into CL$. This insures that M$ will always reflect the latest status of every location.

CL, CL8

The address of CL$ and the address of position 8 in CL$.

ST$

Player status buffer. The first four positions will contain a "?" character unless you have picked up an

object; then the code letter for that object is stored into one of the positions. Dropping an object replaces the letter with a "?". Your inventory can contain no more than 4 objects simultaneously. Position 5 is a code letter indicating your current location.

ST

The address of ST$.

VERB$, NOUN$

These are the lookup tables for the legal verbs and nouns. The first four characters of each word are stored along with a unique alphabetic identifier. A machine-language routine rapidly compares the user input with the data in these tables. If a match is found, the routine returns the word's identifier code; if not, it returns a zero.

V,N

The addresses of VERB$ and NOUN$.

C$

The lookup table for single-character commands. It is scanned whenever you input a single character. A match returns the position of the matching character; no match yields a zero.

K$

The keyboard input buffer.

V$,N$

When you enter a sentence and press RETURN, lines 56 and 57 break K$ into verb and noun segments by determining the location of the space character. These segments are stored in V$ and N$ for later comparison with the lookup tables.

LOOK$,DLI$,F$,D$

Used to store the program's four machine-language subroutines. LOOK$ is a general-purpose character locator. DLI$ contains a display list interrupt handler that blinks the cursor and changes the color of the response window at the bottom of the screen. F$ is a high-speed screen clearing routine. D$ is the noun/verb decoder.

OK$,DH$

Used to store commonly used text phrases. See line 214.

```
10 CLR :GOTO 210
11 GRAPHICS C0:ST$=M$(401,405):Z=ASC(S
T$(C5,C5)):OPEN #C2,C8,C0,"S:":GOSUB 1
5:POKE 703,C4
12 POSITION C0,19:? #C2;"_____
_____";
13 Q=USR(ADR(DLI$),ADR(DLI$)+32):POKE
54286,192:POKE C16,112:POKE 53774,112:
GOTO 66
14 POKE C710,C0:POKE C709,C14:RETURN
15 POKE C709,C14:POKE C710,148:RETURN
:REM
16 FOR I=C0 TO C12 STEP C4:X1=USR(ADR(
F$),I):NEXT I
17 X1=USR(ADR(F$),C15):RETURN
18 SOUND C0,25,C10,C15:FOR I=C1 TO C4:
NEXT I:SOUND C0,C0,C0,C0:RETURN
19 ? "I don't understand.  Try again."
:GOTO 53
20 ? "That is impossible.":RETURN
21 ? "There isn't enough room here.":R
ETURN
22 ? OK$:? "You hear a powerful blast.
":RETURN
23 ? "It isn't here.":RETURN
24 X=USR(LOOK,CL8,N,C8):RETURN
25 Y=USR(LOOK,ST,N,C4):RETURN
26 GOSUB 25:IF Y THEN RETURN
27 POP :POP :? DH$:GOTO 53
28 Q=C16*(ASC(ST$(C5,C5))-65)+C1:RETUR
N :REM
29 POP :FLAG=C1:GOTO 72
30 ? #C2;"Street corner.":RETURN
31 ? #C2;"Lost in a maze of streets.":
RETURN
32 ? #C2;"West of police station.":RET
URN
33 ? #C2;"South of a store.":RETURN
34 ? #C2;"Lobby of police station.":?
#C2;"Sergeant eyes you suspiciously.":
RETURN
35 ? #C2;"Bathroom.":GOSUB 15:RETURN
36 ? #C2;"Inside store.  Sign reads:":
? #C2;"GREEN BATTERIES ONLY $1.00!":RE
TURN
37 ? #C2;"Dead-end alley.":RETURN
38 ? #C2;"Fire escape.":RETURN
39 ? #C2;"Roof of building.":RETURN
40 ? #C2;"Bedroom.":RETURN
41 ? #C2;"Kitchen.":RETURN
42 ? #C2;"White void.":POKE C709,C0:PO
KE C710,C14:RETURN
43 ? #C2;"Golden void.":POKE C709,C14:
POKE C710,24:RETURN
44 ? #C2;"Green void.":POKE C709,C14:P
OKE C710,212:RETURN
45 ? #C2;"Infinite void.":GOSUB 14:RET
URN
46 ? #C2;"Lost in a crimson void.":POK
E C709,C14:POKE C710,64:RETURN
47 ? #C2;"Dense forest.":GOSUB 15:RETU
RN
48 ? #C2;"River bank.":RETURN
49 POP :CLOSE #C2:GRAPHICS C0
50 GOSUB 14:POSITION C12,C10:? "Congra
tulations!":? "↓   You have saved the
Declaration!↓↓↓↓":END
51 POP :GRAPHICS C0:GOSUB 14:POKE 752,
C1:? "↓↓↓↓↓   An unearthly anti-matter
 blast"
52 ? "↓                OBLITERATES":? "↓
      this entire area!":GOTO 223
53 POP :? ">>>>>> What next";:GOSUB 18:
TRAP 53:INPUT K$:TRAP 40000:L=LEN(K$):
IF L=C0 THEN 19
54 IF L=C1 THEN V$=K$:GOTO 63
55 Q=USR(LOOK,ADR(K$),ADR(" "),L):IF Q
<C4 THEN 19
56 V$=K$(C1,Q-C1):IF Q=C4 THEN V$(C4,C
4)="!"
57 N$=K$(Q+C1,L):IF LEN(N$)<C4 THEN 19
58 Z=USR(ADR(D$),ADR(VERB$)-C5,V):IF Z
=C0 THEN 19
59 IF Z=76 THEN K$=K$(Q+C1,L)
60 Q=USR(ADR(D$),ADR(NOUN$)-C5,N):IF Q
=C0 AND Z<>76 THEN 19
61 N$=CHR$(Q):IF N$="X" THEN ? "Refer
to it by color.":GOTO 53
62 Z=Z-64:ON Z GOSUB 93,107,120,129,13
6,144,147,153,159,163,177,183,185,188,
203:GOTO 53
63 Q=USR(LOOK,ADR(C$),V,C9):IF Q=C0 TH
EN 19
64 IF Q>C6 THEN Q=Q-C6:ON Q GOTO 77,82
,88
65 Q=Q+C1:Z=ASC(CL$(Q,Q)):IF Z=63 THEN
 ? "You can't go that way.":GOTO 53
66 GOSUB 16:GOSUB 28:M$(Q,Q+C15)=CL$:S
T$(C5,C5)=CHR$(Z):GOSUB 28:CL$=M$(Q,Q+
C15)
67 ? OK$:POSITION C2,C0:? #C2;"Your lo
cation:";:POSITION C2,C1
68 Z=Z-64:ON Z GOSUB 30,31,31,32,31,31
,31,33,34,35,36,37,38,39,40,41,42,43,4
4,45,46,46,46,47,48
69 REM * SHOW LEGAL EXITS
```

```
70 POSITION C2,C4:? #C2;"You can go: "
;:FOR I=C1 TO C6:IF CL$(I+C1,I+C1)<>"?
" THEN ? #C2;C$(I,I);" ";
71 NEXT I
72 X1=USR(ADR(F$),C6):X1=USR(ADR(F$),C
10):POSITION C2,C6:? #C2;"You see: ";:
X=C6
73 FOR I=C1 TO C8:Q=ASC(CL$(I+C8,I+C8)
)-64:IF Q<>-C1 THEN RESTORE 245+Q:READ
 K$:POSITION 11,X:? #C2;K$:X=X+C1
74 NEXT I:IF X=C6 THEN POSITION 11,X:?
 #C2;"Nothing interesting"
75 IF FLAG=C1 THEN FLAG=C0:GOTO 78
76 GOTO 53
77 REM * I
78 ? OK$:GOSUB 17:POSITION C2,C15:? #C
2;"You have: ";
79 X=C15:FOR I=C1 TO C4:Q=ASC(ST$(I,I)
)-64:IF Q<>-C1 THEN RESTORE 245+Q:READ
 K$:POSITION C12,X:? #C2;K$:X=X+C1
80 NEXT I:IF X=C15 THEN POSITION 12,X:
? #C2;"Nothing"
81 GOTO 53
82 REM * Q
83 ? "Type Y to save, RETURN to quitN"
;:INPUT K$:IF K$<>"Y" THEN CLOSE #C2:G
RAPHICS C0:END
84 CLOSE #C1:POKE 559,34:? "Position S
ave: Disk or TapeN";:INPUT N$:IF N$<>"
D" AND N$<>"T" THEN 53
85 TRAP 84:K$="D1:SAVE.DAT":IF N$="T"
THEN K$="C:"
86 POKE 559,C0:POKE 54272,C0:OPEN #C1,
C8,C0,K$:TRAP 40000:M$(401,405)=ST$:GO
SUB 28:M$(Q,Q+C15)=CL$
87 FOR I=325 TO C1 STEP -81:? #C1;M$(I
,I+80):NEXT I:? #C1;NOUN$:CLOSE #C1:PO
KE 559,34:POKE 54286,192:GOTO 53
88 REM * H
89 IF USR(LOOK,ADR("ABCDEGHUVW"),CL,C1
0) THEN ? "A map is essential.":GOTO 5
3
90 IF USR(LOOK,ADR("JNORSY"),CL,C6) TH
EN ? "Examine everything.":GOTO 53
91 IF CL$(C1,C1)="T" THEN ? "No earthl
y power can help you.":GOTO 53
92 ? "How's your pitching arm lately?"
:GOTO 53
93 REM * VERB A
94 Q=Q-64:GOSUB 24:IF X=C0 THEN GOSUB
25:IF Y=C0 THEN 23
95 ON Q GOTO 96,96,96,96,96,97,96,96,9
8,96,98,96,99,100,101,99,102,103,96,96
,104,96,106
96 ? "Seems ordinary.":RETURN
97 ? "There's a battery inside!":RETUR
N
98 ? "Has a battery attachment.":RETUR
N
99 ? "Looks dangerous.":RETURN
100 ? "Looks like this: ^^^":RETURN
101 ? "They shimmer eerily.":RETURN
102 ? "Looks thirsty.":RETURN
103 ? "Stock is dated 1775!":RETURN
104 N$="V":GOSUB 24:IF X=C0 THEN GOTO
96
105 ? "It's protected by a powerful":?
 "force field.":RETURN
106 ? "Looks like a shining door.":RET
URN
107 REM * VERB B
108 GOSUB 25:IF Y THEN ? "You already
have that.":RETURN
109 Z=USR(LOOK,ST,ADR("?"),C4):IF Z=C0
 THEN ? "You can't carry any more.":RE
TURN
110 GOSUB 24:IF X=C0 THEN 23
111 IF USR(LOOK,ADR("NPQSTVW"),N,7) TH
EN 20
112 IF N$="U" AND USR(LOOK,CL8,ADR("V"
),C8) THEN ? "Force field won't let yo
u.":RETURN
113 IF N$="K" AND USR(LOOK,CL8,ADR("P"
),C8) THEN ? "Beast would rather you d
idn't.":RETURN
114 IF N$="R" AND USR(LOOK,CL8,ADR("Q"
),C8) THEN ? "Soldier won't cooperate.
":RETURN
115 IF N$="D" AND CL$(C1,C1)="K" THEN
? "Cashier says, '$1.00, please.'":RET
URN
116 IF N$="M" THEN 118
117 CL$(X+C8,X+C8)="?":ST$(Z,Z)=N$:GOT
O 29
118 IF USR(LOOK,ST,ADR("O"),C4) THEN 1
17
119 GOTO 51
120 REM * VERB C
121 GOSUB 26:X=USR(LOOK,CL8,ADR("?"),C
8):IF X=C0 THEN 21
122 IF N$="L" AND CL$(C1,C1)="Y" THEN
CL$(C9,C9)="?":N$="?":? "Soldier walks
 away with it."
123 IF USR(LOOK,ADR("NPQSTVW"),N,7) TH
EN 20
124 IF N$="M" AND CL$(C1,C1)="T" THEN
? "It's up against the force field."
125 IF N$="U" AND CL$(C1,C1)="I" THEN
49
126 IF N$="D" AND CL$(C1,C1)="K" THEN
? "Cashier returns it to shelf."
127 IF USR(LOOK,CL8,ADR("M"),C8) THEN
51
128 ST$(Y,Y)="?":CL$(X+C8,X+C8)=N$:GOT
O 29
129 REM * VERB D
130 GOSUB 25:IF Y=C0 THEN ? DH$:RETURN
:REM
131 IF USR(LOOK,ADR("NPQV"),N,C4) THEN
 POP :GOTO 19
132 IF N$="I" OR N$="K" THEN ? "Needs
a power source.":RETURN
133 IF N$="H" OR N$="J" THEN ? "It's a
lready activated.":RETURN
134 IF N$="S" OR N$="T" THEN ? "Indica
te a direction.":RETURN
135 ? "Be more specific.":RETURN
136 REM * VERB E
137 GOSUB 24:IF N$="S" AND X THEN 146
138 IF N$="T" THEN ? "It's already ope
n.":RETURN
139 GOSUB 26:IF N$="A" OR N$="U" THEN
? OK$:RETURN
140 IF N$="F" AND Y THEN 142
141 GOTO 20
142 N$="?":GOSUB 24:IF X=C0 THEN 21
143 ST$(Y,Y)="G":NOUN$(30,30)="G":CL$(
X+C8,X+C8)="E":GOSUB 97:? "It fell out
 on the floor.":GOTO 29
144 REM * VERB F
145 IF N$<>"S" THEN 20
146 ? "Lock is very secure.":RETURN
147 GOSUB 24:IF X=C0 THEN 23
148 Y=USR(LOOK,ST,ADR("C"),C4):IF N$="
S" AND Y THEN 151
149 IF N$="S" AND Y=C0 THEN ? "You hav
e nothing to break it with.":RETURN
150 GOTO 20
151 CL$(C5,C5)="O":CL$(X+C8,X+C8)="T":
M$(234,234)="C":ST$(Y,Y)="?":? "The st
one broke the window!"
152 NOUN$(80,80)="T":POP :GOTO 70
153 REM * VERB H
154 GOSUB 25:IF Y THEN ? "You already
have that.":RETURN
155 GOSUB 24:IF X=C0 THEN ? DH$:RETURN
:REM
156 Y=USR(LOOK,ST,ADR("B"),C4):IF Y=C0
 THEN ? "You don't have any money.":RE
TURN
157 IF N$<>"D" THEN 20
158 ST$(Y,Y)="D":CL$(X+C8,X+C8)="?":?
"Cashier takes your $1 bill.":GOTO 29
159 REM * VERB I
160 GOSUB 24:GOSUB 25:IF X=C0 AND Y=C0
 THEN 23
161 IF N$="M" AND USR(LOOK,ST,ADR("O")
,C4)=C0 THEN 51
162 GOTO 96
163 REM * VERB J
164 GOSUB 26:IF USR(LOOK,ADR("MNPQSTVW
"),N,C8) THEN 20
165 X=USR(LOOK,CL8,ADR("?"),C8):IF X=C
0 THEN 21
166 IF N$="C" THEN 168
167 ST$(Y,Y)="?":CL$(X+C8,X+C8)=N$:? "
```

```
Didn't go far.":GOTO 29
168 ? "Indicate a direction (N/S/E/W)"
;:GOSUB 18:INPUT K$:IF LEN(K$)<>1 THE
N POP :GOTO 19
169 Q=USR(LOOK,ADR(C$),ADR(K$),C4):IF
Q=C0 THEN POP :GOTO 19
170 IF CL$(Q+C1,Q+C1)="?" THEN 176
171 IF CL$(Q+C1,Q+C1)=CL$(C1,C1) THEN
167
172 Z=C16*(ASC(CL$(Q+C1,Q+C1))-65)+C9:
I=USR(LOOK,ADR(M$(Z)),ADR("?"),C8):X=U
SR(LOOK,ADR(M$(Z)),ADR("M"),C8)
173 IF X AND CL$(C1,C1)="Q" AND Q=C4 T
HEN M$(313,320)="???????U":ST$(Y,Y)="?
":GOTO 22
174 IF X THEN M$(Z,Z+7)="?????????":ST$
(Y,Y)="?":GOTO 22
175 IF I THEN M$(Z+I-C1,Z+I-C1)="C":ST
$(Y,Y)="?":? "Gone!":GOTO 29
176 ? "You can't throw it that way.":R
ETURN
177 REM * VERB K
178 GOSUB 24:IF X=C0 THEN 23
179 IF N$="P" OR N$="Q" THEN 181
180 GOTO 20
181 IF USR(LOOK,ST,ADR("R"),C4)=C0 THE
N ? "Not without a weapon.":RETURN
182 ? "Bayonet scared it away!":CL$(X+
C8,X+C8)="?":GOTO 29
183 REM * VERB L
184 ? K$:RETURN
185 REM * VERB M
186 GOSUB 26:IF N$<>"O" THEN 20
187 ? "You're already wearing them.":R
ETURN
188 REM * VERB N
189 GOSUB 26:X=USR(LOOK,ST,ADR("I"),C4
):Z=USR(LOOK,ST,ADR("K"),C4)
190 IF X=C0 AND Z=C0 THEN ? "You have
nothing to attach it to.":RETURN
191 IF N$="D" THEN 194
192 IF N$="E" THEN 196
193 GOTO 20
194 IF Z THEN ? "Cube absorbs the batt
ery and hums.":GOTO 199
195 IF X THEN ? "Green battery doesn't
 fit.":RETURN
196 IF X THEN ? "Spheroid absorbs the
battery and":? "displays a symbol: ^^^
":GOTO 198
197 IF Z THEN ? "Blue battery doesn't
fit.":RETURN
198 ST$(X,X)="H":M$(150,150)="Q":M$(15
3,153)="W":NOUN$(35,35)="H":GOTO 201
199 ST$(Z,Z)="J":M$(263,263)="J":M$(26
2,262)="?":NOUN$(40,40)="J"
200 IF CL$(C1,C1)="Q" THEN CL$(C6,C6)=
"?":CL$(7,7)="J"
201 IF CL$(C1,C1)="J" THEN CL$(C9,C9)=
"W":CL$(C6,C6)="Q"
202 ST$(Y,Y)="?":FLAG=C1:POP :GOTO 70:
REM
203 REM * VERB O
204 GOSUB 24:GOSUB 25:IF Y=C0 AND N$<>
"N" THEN ? DH$:RETURN
205 IF N$="N" AND X THEN 100
206 IF N$="A" AND Y THEN ? "HEADLINE:
Declaration Stolen!":? "Police Anxious
ly Await Recovery!":RETURN
207 IF N$="U" AND Y THEN ? "'We the pe
ople ... '":RETURN
208 IF N$="W" AND Y THEN 106
209 GOTO 20
210 READ FLAG,C0,C1,C2,C4,C5,C6,C7,C8,
C9,C10,C12,C14,C15,C16,C709,C710
211 GOSUB 14:POKE 752,C1:? "↑↓↓↓↓↓↓
      Brian Moriarty's":? "↓  ADVENTU
RE in the 5TH DIMENSION "
212 ? "↓       (C)1983 ANALOG Computing
":POSITION C12,C16:? "  INITIALIZING
":REM
213 DIM M$(406),CL$(16),K$(24),N$(4),V
$(4),LOOK$(43),VERB$(100),NOUN$(100),C
$(9),ST$(5),DLI$(64),F$(42),D$(76)
214 DIM OK$(5),DH$(20):OK$="Okay.":DH$
="You don't have that."
215 NOUN$="PAPEABILLBSTONCGREEDBLUEERA
DIFSPHEICUBEKTEABLMASSMSYMBNGLOVOBEASP
SOLDOBAYORWINDSDECLUFIELVDUCTWBATTX"
216 VERB$="EXAMALOOKATAKEBGET!BDROPCLE
AVCGIVECUSE!DOPENEUNLOFBREAGSMASGBUY!H
TOUCITHROJKILLKSAY!LWEARMATTANREADO"
217 C$="NSEWUDIQH":LOOK=ADR(LOOK$):CL=
ADR(CL$):CL8=CL+C8:V=ADR(V$):N=ADR(N$)
:ST=ADR(ST$)
218 M$(C1)="?":M$(406)="?":M$(C2)=M$:F
OR I=C1 TO 385 STEP C16:READ CL$:M$(I,
I+LEN(CL$))=CL$:NEXT I
219 M$(401,405)="????A":CL$=M$(C1,C16)
:FOR I=C1 TO 76:READ Q:D$(I)=CHR$(Q):N
EXT I
220 FOR I=C1 TO 43:READ Q:LOOK$(I)=CHR
$(Q):NEXT I
221 FOR I=C1 TO 64:READ Q:DLI$(I)=CHR$
(Q):NEXT I
222 FOR I=C1 TO 42:READ Q:F$(I)=CHR$(Q
):NEXT I
223 POSITION C4,C16:? "Press  START  t
o begin new game.":? "↓ Press  OPTION
 to restore old game.↑"
224 IF PEEK(53279)=C6 THEN 11
225 IF PEEK(53279)=3 THEN 227
226 GOTO 224
227 CLOSE #C1:POKE 559,34:POSITION C8,
20:? "Load from Disk or Tape↑";:INPUT
N$:IF N$<>"D" AND N$<>"T" THEN 227
228 POKE 559,C0:POKE 54272,C0:TRAP 227
:K$="D1:SAVE.DAT":IF N$="T" THEN K$="C
:"
229 OPEN #C1,C4,C0,K$:TRAP 40000
230 FOR I=325 TO C1 STEP -81:INPUT #C1
,M$:M$(I,I+80)=M$:NEXT I:M$(406,406)="
":INPUT #C1,NOUN$:CLOSE #C1
231 ST$=M$(401,405):GOSUB 28:CL$=M$(Q,
Q+C15):POKE 559,34:GOTO 11
232 DATA 0,0,1,2,4,5,6,7,8,9,10,12,14,
15,16,709,710
233 DATA AEB?G???A,BABCG,CDCEB,DFCID,E
EAFC,FFDGE???C,GHBAF,HKG?L,I?J?D
234 DATA JI??????NB,K?H?????D,L??H?M,M
????NL?S,N?????M?I,O??MP???F,P??O????L
,QURSTX??W,RQU?????M
235 DATA S???Q???PK,T??Q????VU,URQVW,V
WVWU???O,WWVUV,X??Y??Q?W,Y???X???QR
236 DATA 104,104,133,206,104,133,205,1
04,133,204,104,133,203,169,0,133,213,1
62,21,202,240,49,24,165,205
237 DATA 105,5,133,205,165,206,105,0,1
33,206,24,160,0,177,203,209,205,208,23
1,200,177,203,209,205,208
238 DATA 224,200,177,203,209,205,208,2
17,200,177,203,209,205,208,210,200,177
,205,133,212,96,169,0,133,212,96
239 DATA 104,104,133,206,104,133,205,1
04,133,204,104,133,203,169,0,168,133,2
13,177,203,133,207,104,104,168
240 DATA 136,48,10,165,207,209,205,208
,247,200,132,212,96,169,0,133,212,96
241 DATA 104,104,141,1,2,104,141,0,2,1
73,48,2,133,203,173,49,2,133,204,160,2
4,169,130,145,203
242 DATA 169,0,141,243,2,96,0,72,138,7
2,169,0,162,10,141,10,212,141,24,208,1
42,23,208,230,208
243 DATA 165,208,41,16,74,74,74,141,1,
212,104,170,104,64
244 DATA 104,104,104,170,165,88,133,20
3,165,89,133,204,216,24,202,48,15,165,
203,105,40,133,203,165,204
245 DATA 105,0,133,204,24,144,238,160,
159,169,0,145,203,136,208,251,96
246 DATA Morning paper
247 DATA $1 bill
248 DATA Stone
249 DATA Green battery
250 DATA Blue battery
251 DATA Broken portable radio
252 DATA Empty radio
253 DATA Spheroid with ^^^ symbol
254 DATA Alien spheroid
255 DATA Humming alien cube
256 DATA Alien cube
257 DATA Teabag
258 DATA Strange shimmering mass
```

```
259 DATA Alien symbol on wall
260 DATA Strange gloves
261 DATA 5-dimensional beast
262 DATA British soldier
263 DATA Bayonet
264 DATA Locked window
265 DATA Broken west window
266 DATA The Declaration
267 DATA Powerful force field
268 DATA Transdimensional duct
```

•

CHECKSUM DATA
(See D:CHECK/C:CHECK,p.26)

```
10 DATA 340,840,723,954,975,660,574,3,
94,386,669,532,340,532,491,8113
25 DATA 294,939,506,558,138,173,348,66
,692,825,485,698,276,855,837,7690
40 DATA 34,31,795,851,959,322,53,973,5
57,922,874,649,843,129,793,8785
55 DATA 375,74,537,559,661,561,226,812
,847,95,346,556,962,398,631,7640
70 DATA 246,387,242,860,808,660,641,58
8,144,789,324,628,591,970,702,8580
85 DATA 997,233,148,591,792,546,969,90
0,33,92,450,206,870,868,402,8097
100 DATA 870,128,5,334,206,443,993,350
,763,982,828,793,147,137,359,7338
115 DATA 684,801,62,144,503,351,83,391
,798,463,899,554,186,51,362,6332
130 DATA 693,688,557,227,593,395,364,7
03,163,308,519,495,248,949,367,7269
145 DATA 950,976,844,991,353,497,669,2
13,373,774,700,984,925,183,381,9813
160 DATA 139,509,534,380,59,936,811,74
,590,736,113,701,759,28,820,7189
175 DATA 204,918,389,854,645,506,593,6
32,390,397,394,118,288,399,662,7389
190 DATA 682,811,817,512,152,364,915,2
76,235,118,237,450,341,374,721,7005
205 DATA 512,76,518,538,493,11,48,638,
830,683,691,479,843,307,672,7339
220 DATA 578,126,103,992,631,545,709,1
60,472,982,283,659,402,337,95,7074
235 DATA 574,677,261,81,958,641,502,54
9,814,479,495,22,590,343,149,7135
250 DATA 637,492,558,59,152,672,478,51
0,839,226,63,52,283,935,103,6059
265 DATA 12,211,368,644,1235
```

•

Assembly Language Listing

```
0100 ; NOUN/VERB DECODER
0105 ;
0110 ; Syntax: N=USR(ML,TL-5,NL)
0115 ; ML=addr of this routine
0120 ; TL=addr of lookup table
0125 ; NL=addr of current noun/verb
0130 ;
0135 ; Program equates
0140 ;
0145 NOUN=$CB  ; noun addr pointer
0150 TABLE=$CD ; table addr pointer
0155 NRET=$D4  ; BASIC return addr
0160 ;
0165      PLA           ; # arguments
0170      PLA           ; msb of table addr
0175      STA TABLE+1
0180      PLA           ; lsb
0185      STA TABLE
0190      PLA           ; msb of noun addr
0195      STA NOUN+1
0200      PLA           ; lsb
0205      STA NOUN
0210      LDA #$00
0215      STA NRET+1    ; zero msb
0220      LDX #$15      ; noun/verb count
0225 NEXT DEX
0230      BEQ NOPE      ; illegal entry
0235      CLC
0240      LDA TABLE     ; +5 to pointer
0245      ADC #$05
0250      STA TABLE
0255      LDA TABLE+1
0260      ADC #$00
0265      STA TABLE+1
0270      CLC
0275      LDY #$00      ; init index
0280      LDA (NOUN),Y  ; get 1st char
0285      CMP (TABLE),Y ; equal?
0290      BNE NEXT      ; no; next noun
0295      INY           ; in-line for speed
0300      LDA (NOUN),Y  ; try 2nd char
0305      CMP (TABLE),Y
0310      BNE NEXT
0315      INY
0320      LDA (NOUN),Y  ; try 3rd char
0325      CMP (TABLE),Y
0330      BNE NEXT
0335      INY
0340      LDA (NOUN),Y  ; last char!
0345      CMP (TABLE),Y
0350      BNE NEXT
0355      INY           ; must be legal
0360      LDA (TABLE),Y ; get iden #
0365      STA NRET      ; give to BASIC
0370      RTS           ; and return
0375 NOPE LDA #$00      ; 0=illegal entry
0380      STA NRET      ; give to BASIC
0385      RTS           ; and return
```

•

```
0100 ; CHARCTER SEARCH ROUTINE
0105 ;
0110 ; Syntax: X=USR(ML,SVT,V,R)
0115 ; ML=addr of this routine
0120 ; SVT=addr of $ to be searched
0125 ; V=addr of search character
0130 ; R=# bytes to search
0135 ;
0140 ; Program equates
0145 ;
0150 CADR=$CB  ; char addr pointer
0155 TABLE=$CD ; verb table pointer
0160 CHAR=$CF  ; character buffer
0165 BRET=$D4  ; BASIC return addr
0170 ;
0175      PLA           ; # arguments
0180      PLA           ; msb of table addr
0185      STA TABLE+1
0190      PLA           ; lsb
0195      STA TABLE
0200      PLA           ; msb of verb addr
0205      STA CADR+1
0210      PLA           ; lsb
0215      STA CADR
0220      LDA #$00
0225      TAY
0230      STA BRET+1    ; zero msb
0235      LDA (CADR),Y  ; get the char
0240      STA CHAR      ; save for later
0245      PLA           ; msb of range (ignore)
0250      PLA           ; lsb
0255      TAY           ; use as the index
0260 NEXT DEY
0265      BMI NOPE      ; must be illegal
0270      LDA CHAR      ; get char
```

```
0275      CMP (TABLE),Y   ; match?
0280      BNE NEXT        ; no; try another
0285      INY             ; yes; give position
0290      STY BRET        ; to BASIC
0295      RTS             ; and return
0300 NOPE LDA #$00        ; 0=char not found
0305      STA BRET        ; give to BASIC
0310      RTS             ; and return
```

•

```
0100 ; DLI/BLINK ROUTINE
0105 ;
0110 ; Syntax: USR(DLI,DLI+X)
0115 ; DLI=addr of this routine
0120 ; X=offset to DLI handler
0125 ;
0130 ; Program equates
0135 ;
0140 COLPF1=$D017
0145 COLPF2=$D018
0150 WSYNC=$D40A
0155 SDLSTL=$0230
0160 VDSLST=$0200
0165 CHACTL=$D401
0170 CHACT=$02F3
0175 BUFFER=$CB
0180 BLINCT=$D0
0185 ;
0190 ; First set up the DLI
0195 ;
0200      PLA             ; # arguments
0205      PLA             ; msb of DLI addr
0210      STA VDSLST+1
0215      PLA             ; lsb
0220      STA VDSLST
0225      LDA SDLSTL      ; find start of
0230      STA BUFFER      ; display list
0235      LDA SDLSTL+1
0240      STA BUFFER+1
0245      LDY #$18        ; mode line 20
0250      LDA #$82        ; DL instruction
0255      STA (BUFFER),Y
0260      LDA #$00        ; turn off
0265      STA CHACT       ; inverse video
0270      RTS
0275      BRK             ; mark end of init
0280 ;
0285 ; This is the actual DLI handler
0290 ;
0295      PHA             ; save accumulator
0300      TXA
0305      PHA             ; save X
0310      LDA #$00        ; black bkgrnd
0315      LDX #$0A        ; white chars
0320      STA WSYNC
0325      STA COLPF2
0330      STX COLPF1
0335      INC BLINCT      ; Blink cursor
0340      LDA BLINCT
0345      AND #$10
0350      LSR A
0355      LSR A
0360      LSR A
0365      STA CHACTL
0370      PLA             ; restore A and X
0375      TAX
0380      PLA
0385      RTI             ; back to BASIC
```

•

```
0100 ; SCREEN ERASE SUBROUTINE
0105 ; CLEARS IN 4-LINE BLOCKS
0110 ;
0115 ; Syntax: X=USR(ML,ST)
0120 ; ML=addr of this routine
0125 ; ST=starting line number
0130 ;
0135 ; PROGRAM EQUATES
0140 ;
0145 BUFFER=$CB ; scr address buffer
0150 SAVMSC=$58 ; screentop pointer
0155 ;
0160      PLA             ; # arguments
0165      PLA             ; msb of line#; ignore
0170      PLA             ; lsb
0175      TAX             ; save in x-register
0180      LDA SAVMSC      ; get screen
0185      STA BUFFER      ; address
0190      LDA SAVMSC+1
0195      STA BUFFER+1
0200      CLD             ; clear decimal mode
0205      CLC
0210 ADD40 DEX            ; find window addr
0215      BMI CLEAR
0220      LDA BUFFER      ; add 40
0225      ADC #$28
0230      STA BUFFER
0235      LDA BUFFER+1
0240      ADC #$00
0245      STA BUFFER+1
0250      CLC
0255      BCC ADD40
0260 CLEAR LDY #$9F       ; clear 4 lines
0265      LDA #$00        ; space char
0270 SPACE STA (BUFFER),Y
0275      DEY
0280      BNE SPACE
0285      RTS
```

•

GRAPHICS 7+ HANDLER

16K Cassette/24K Disk

by Tom Hudson

Hidden deep inside the ATARI 400/800 computer systems are several capabilities that ATARI apparently chose to keep a deep, dark secret. Player-missile graphics, an incredibly powerful graphics tool, are barely hinted at in most ATARI documentation. Optimized Systems Software's **BASIC A+** was a step in the right direction as far as P/M graphics are concerned, allowing easy manipulation in BASIC.

Unknown to many ATARI users, the 400/800 computers actually have 14 graphics modes (17 if you count GTIA), not just the nine BASIC modes 0-8. These additional modes are available in the hardware, but the Operating System (OS) doesn't support them directly. Shown below is a table of the ANTIC (hardware) modes and their corresponding BASIC modes.

ANTIC MODE	BASIC MODE	DESCRIPTION
02	0	40 X 24, 2 COLOR, TEXT
03	-	40 X 19, 2 COLOR, TEXT
04	-	40 X 24, 4 COLOR, TEXT
05	-	40 X 12, 4 COLOR, TEXT
06	1	20 X 24, 5 COLOR, TEXT
07	2	20 X 12, 5 COLOR, TEXT
08	3	40 X 24, 4 COLOR, GRAPHIC
09	4	80 X 48, 2 COLOR, GRAPHIC
10	5	80 X 48, 4 COLOR, GRAPHIC
11	6	160 X 96, 2 COLOR, GRAPHIC
12	-	160 X192, 2 COLOR, GRAPHIC
13	7	160 X 96, 4 COLOR, GRAPHIC
14	-	160 X192, 4 COLOR, GRAPHIC
15	8	320 X192, 2 COLOR, GRAPHIC

ANTIC mode 3 is a nifty text mode similar to GRAPHICS 0 which will allow true descenders on lower-case letters. ANTIC 4 and 5 are very powerful text modes allowing 5 colors WITHIN EACH CHARACTER! Mode 4 was used for ATARI's adaptation of PAC-MAN, for the maze and bonus nuggets. ANTIC 12, which we call GRAPHICS 6+, is identical to BASIC GRAPHICS 6, but each plotted block (or pixel) is only one scan line tall, giving a higher resolution display of 160 by 192.

The mode this article is concerned with is ANTIC 14, or GRAPHICS 7+. It is identical to GRAPHICS 7, but has a resolution of 160 by 192. Using this mode will allow the generation of high-resolution displays in four colors. The best example of this mode is Datasoft's graphics package, "Micropainter." This article will present a machine-language subroutine which will allow you to use GRAPHICS 7+ from BASIC. It also has some nice enhancements which make plotting and drawing much faster.

The "Plot" Thickens

Listing 1 is the BASIC code necessary to use GRAPHICS 7+. As written, it will run a continuous demonstration, plotting rectangles in 3 colors at random points on the screen. Type this listing into your computer and SAVE it before running it.

After SAVEing the program, RUN it. If the listing was entered correctly, you will see rectangles plotted continuously on your screen. The program has some error-checking that will catch some errors in the DATA. A "CALC DATA ERROR" indicates an error in the plot calculator data in lines 460-500. A "MAIN DATA ERROR" indicates an error in the main routine data in lines 540-680.

Listings 2 and 3 are the machine-language source listings of the handler, for those interested in the assembler side of the routine.

Once you have an operating copy of the GRAPHICS 7+ handler, you are ready to use it in your own programs.

Inside The Program

As noted earlier, the program presented here will plot random rectangles on your screen. The code that performs this function is in lines 230-420. Lines 50-220 and 440-680 MUST be left as is (of course, you can delete the REMarks if you wish).

Line 50 - READs the DATA in lines 460-500 and places it in the user memory. This DATA, the machine-language form of listing #2, actually performs the calculations needed to PLOT in GRAPHICS 7+.

Lines 110-115 - READs the DATA in lines 540-680 and places it in the string variable G7P$. This DATA is the machine-language form of listing #3, and handles GRAPHICS 7+ initialization, PLOTting and DRAWing. Note that a simple checksum routine is used to check for DATA errors.

Line 170 - This line sets entry points into the

ATARI®
MEANS BUSINESS

BUSINESS PACKAGES FOR $195.00 EACH

For the first time, Financial Software Plus introduces businesss packages for the Atari 400 or 800 computer that simulates business packages found on much larger and expensive computer systems. All our packages are written by an accountant-programmer and program designer so that you can be assured that our packages will meet your individual requirements.

INVENTORY

- Capacity of 500 part numbers, 250 customers, 250 vendors per disk.
- Maintains customer files, vendor files, and sales history by month and year-to-year date.
- Generates purchase orders, invoices, quotations, regular mailing labels, and shipping labels.
- Reports Include: Quantity in stock, quantity on order, backorder, reorder, and sales activity.
- Ability to limit reports by part numbers and vendor or both.

PAYROLL

- Capacity of 100 Employees per disk.
- User changeable tax codes-five different tax tables may be set up.
- Miscellaneous earnings and deductions categories are user specified and changeable at any time.
- Prints checks, W2's, month-end, quarterly, annual and tax reports.
- Fixed or variable deduction capability

COMMON FEATURES

- Will run on a one or two disk drive system at any time.
- All data disks are created automatically-unlimited storage capacity.
- All reports can be printed to the screen or printer-printer is optional.
- Can act as a stand alone package or a fully integrated system with our general ledger package.

Stop on down to your nearest dealer and ask for a demonstration.

Atari is a registered trademark of Atari Incorporated.

General Ledger, Accounts Payable, and Accounts Receivable Packages Available Soon.

FINANCIAL SOFTWARE PLUS

121 WEST CEDAR
KALAMAZOO, MICH. 49007
(616) 345-8546

DISTRIBUTOR AND DEALER INQUIRIES WELCOME

machine-language program located in G7P$. INIT is the address of the initialization routine, PL is the address of the PLOT routine, and DR is the address of the DRAWTO routine.

Line 220 - This line initializes the GRAPHICS 7+ screen. First, the user must set up a GRAPHICS 8 screen, either a full-screen or split-screen mode. In this case, we want a full-screen GRAPHICS 7+ screen so we use GRAPHICS 8+16. Using only GRAPHICS 8 will allow the use of a text window at the bottom of the screen. We use a GRAPHICS 8 call because it reserves the same amount of memory that GRAPHICS 7+ needs. Next, we do a USR call to the INIT routine to actually set up the GRAPHICS 7+ screen. It's that simple!

Line 230 - This line sets COLOR 0 to red.

Line 280 - This line randomizes the X and Y coordinates of the rectangle's upper-left corner. It also sets a random COLOR of 1, 2, or 3. We don't allow the color value to be zero, as this would be the same color as the background, and wouldn't show up.

Line 330 - This line PLOTs the first point of a rectangle. It does the same thing as the BASIC statement:

PLOT 10+X,10+Y

One interesting function incorporated into the GRAPHICS 7+ handler is the ability to plot multiple points with one PLOT statement. For example, examine the following BASIC statements:

```
PLOT X,Y
PLOT X+2,Y+2
PLOT X,Y+2
PLOT X+2,Y
```

These four commands could be done in GRAPHICS 7+ with ONE command, as shown below:

A=USR(PL,X,Y,X+2,Y+2,X,Y+2,X+2,Y)

Using one command for such multiple PLOTs can speed up program execution and makes life a little easier when keying in programs. Just remember to always give the routine an EVEN number of arguments (X and Y coordinates). If the GRAPHICS 7+ handler receives an odd number of arguments, it will not plot, but will simply return to BASIC. If the X or Y values exceed the screen limits (X = 0-159, Y = 0-191), the PLOT will be ignored.

To summarize, whenever a PLOT is desired in GRAPHICS 7+, use the command:

A=USR(PL,X,Y)

where X and Y are the coordinates of the pixel to be PLOTed.

Line 380 - This line DRAWs the four sides of the rectangle. As with the GRAPHICS 7+ PLOT handler, the DRAW handler will accept multiple DRAWTOs! As you can see, we can draw a rectangle in GRAPHICS 7+ with only TWO commands, where normally FIVE BASIC commands would be necessary. This line is the same as the four BASIC commands:

```
DRAWTO 10+X,Y
DRAWTO X,Y
DRAWTO X,10+Y
DRAWTO 10+X,10+Y
```

This multiple-argument DRAWTO capability can be very powerful, allowing many lines to be drawn with one statement.

Whenever a DRAWTO is desired with GRAPHICS 7+, use the command:

A=USR(DR,X,Y)

where X and Y are the corrdinates of the pixel a line is to be drawn to. This line will originate from the last point plotted by the GRAPHICS 7+ routine.

Line 420 - This line simply transfers control back to line 280, where the plotting of another rectangle begins.

Using Graphics 7+ In Your Own Programs

You can easily create programs that use GRAPHICS 7+. Simply remove lines 230-420 and place your program code after line 220. You may change the "GRAPHICS 8+16" command in line 220 if a split-screen graphics mode is desired.

I think you will find that GRAPHICS 7+ is a happy medium between the somewhat "chunky" GRAPHICS 7 and the one-color hi-res GRAPHICS 8. Its added resolution in the Y-axis brings it close to mode 8, and the four-color capability gives spectacular displays.

Interestingly enough, the new ATARI 1200XL computer supports ANTIC modes 4, 5, 12 (or 6+) and 14 (or 7+)! Therefore, this program is unnecessary for those future 1200XL owners. Of course, any program written with this GRAPHICS 7+ handler will work on a 1200XL, without modification.

Whichever ATARI computer you own, the GRAPHICS 7+ handler will let those hidden graphics capabilities shine through. □

```
1 REM ********************************
2 REM *                              *
3 REM *     GRAPHICS 7+ HANDLER      *
4 REM *                              *
5 REM *        BY TOM HUDSON         *
6 REM *                              *
7 REM *  A.N.A.L.O.G. COMPUTING #11  *
8 REM *                              *
9 REM ********************************
10 REM
20 REM *** PLACE PLOT CALCULATOR ***
30 REM *** ON PAGE 6              ***
40 REM
50 FOR X=0 TO 116:READ N:POKE 1536+X,N
:CK=CK+N:NEXT X:IF CK<>15155 THEN ? "C
ALC DATA ERROR!":END
60 REM
70 REM *** PLACE GRAPHICS 7+ ***
80 REM *** MACHINE-LANGUAGE  ***
90 REM *** IN STRING G7P$    ***
100 REM
```

```
110 DIM G7P$(371):FOR X=1 TO 371:READ
N:G7P$(X,X)=CHR$(N):CK=CK+N:NEXT X
115 IF CK<>63809 THEN ? "MAIN DATA ERR
OR!":END
120 REM
130 REM *** SET INITIALIZATION, ***
140 REM *** PLOT AND DRAWTO      ***
150 REM *** VARIABLES            ***
160 REM
170 INIT=ADR(G7P$):PL=INIT+77:DR=INIT+
129
180 REM
190 REM *** SET UP GRAPHICS 7+ ***
200 REM *** GRAPHICS MODE      ***
210 REM
220 GRAPHICS 8+16:A=USR(INIT)
230 SETCOLOR 0,3,2
240 REM
250 REM *** GET RANDOM X,Y, AND ***
260 REM *** COLOR VALUES        ***
270 REM
280 X=RND(0)*140:Y=RND(0)*180:COLOR IN
T(RND(0)*3+1)
290 REM
300 REM *** PLOT THE FIRST POINT ***
310 REM *** IN THE RECTANGLE     ***
320 REM
330 A=USR(PL,10+X,10+Y)
340 REM
350 REM *** DRAW THE 4 SIDES OF ***
360 REM *** THE RECTANGLE       ***
370 REM
380 A=USR(DR,10+X,Y,X,Y,X,10+Y,10+X,10
+Y)
390 REM
400 REM *** DO ANOTHER RECTANGLE ***
410 REM
420 GOTO 280
430 REM
440 REM *** PLOT CALCULATOR DATA ***
450 REM
460 DATA 173,241,6,10,133,203,169,0,42
,133,204,6,203,38,204,6,203,165,203,13
3,207,38,204,165,204
470 DATA 133,208,6,203,38,204,6,203,38
,204,165,203,24,101,207,133,203,165,20
4,101,208,133,204,165,88
480 DATA 24,101,203,133,203,165,89,101
,204,133,204,173,240,6,41,3,170,173,24
0,6,74,74,24,101,203
490 DATA 133,203,165,204,105,0,133,204
,164,200,189,113,6,57,105,6,133,206,18
9,109,6,160,0,49,203
500 DATA 5,206,145,203,96,0,85,170,255
,63,207,243,252,192,48,12,3
510 REM
520 REM *** GR. 7+ MACHINE CODE ***
530 REM
540 DATA 104,173,48,2,24,105,3,133,203
,173,49,2,105,0,133,204,160,0,177,203,
201,79,208,21,169
550 DATA 78,145,203,165,203,24,105,2,1
33,203,165,204,105,0,133,204,169,0,240
,15,201,15,208,6,169
560 DATA 14,145,203,208,5,201,65,208,1
,96,165,203,24,105,1,133,203,165,204,1
05,0,133,204,169,0
570 DATA 240,197,216,104,240,13,133,20
5,41,1,240,8,166,205,104,104,202,208,2
51,96,104,104,201,160,176
580 DATA 22,141,240,6,104,104,201,192,
176,6,141,241,6,32,0,6,198,205,198,205
,208,229,96,104,104
590 DATA 169,0,240,243,216,104,240,217
,133,205,41,1,240,6,166,205,169,0,240,
200,104,104,201,160,176
600 DATA 14,141,242,6,104,104,201,192,
176,7,141,243,6,144,9,104,104,198,205,
198,205,208,228,96,205
610 DATA 241,6,144,14,56,237,241,6,141
,247,6,169,1,141,249,6,208,15,173,241,
6,56,237,243,6
620 DATA 141,247,6,169,255,141,249,6,1
73,242,6,205,240,6,144,14,56,237,240,6
,141,246,6,169,1
630 DATA 141,248,6,208,15,173,240,6,56
,237,242,6,141,246,6,169,255,141,248,6
,169,0,141,245,6
640 DATA 141,244,6,173,246,6,205,247,6
,144,15,141,250,6,133,209,74,141,245,6
,169,0,240,14,240
650 DATA 147,173,247,6,141,250,6,133,2
09,74,141,244,6,173,250,6,240,237,173,
245,6,24,109,247,6
660 DATA 141,245,6,197,209,144,19,173,
245,6,56,229,209,141,245,6,173,241,6,2
4,109,249,6,141,241
670 DATA 6,173,244,6,24,109,246,6,141,
244,6,197,209,144,19,173,244,6,56,229,
209,141,244,6,173
680 DATA 240,6,24,109,248,6,141,240,6,
32,0,6,206,250,6,208,182,169,0,240,159
```

•

CHECKSUM DATA
(See D:CHECK/C:CHECK,p.26)

```
1 DATA 746,168,556,172,240,176,687,180
,762,251,614,549,257,382,261,6001
70 DATA 720,893,600,74,882,737,80,443,
114,659,92,497,98,370,220,6479
210 DATA 79,871,448,88,180,960,97,297,
103,768,370,84,938,90,92,5465
360 DATA 978,99,68,105,663,83,721,89,6
84,95,854,415,819,965,752,7390
510 DATA 85,181,91,548,871,853,279,713
,142,6,358,431,477,618,912,6565
660 DATA 735,711,81,1527
```

•

Assembly Language Listing

```
;       ==========================
;          GRAPHICS 7+ HANDLER
;       ==========================
;
;       ==========================
;          WRITTEN BY: TOM HUDSON
;          A.N.A.L.O.G. COMPUTING #11
;       ==========================
;
;       ------------------------
;       OPERATING SYSTEM EQUATES
;       ------------------------
;
SAVMSC  =   $58         ;SCREEN ADDRESS
COLOR   =   $C8         ;COLOR REGISTER
;
;       --------------------
;       MY WORKING VARIABLES
;       --------------------
;
LO      =   $CB
HI      =   $CC
HOLD    =   $CE
LOHLD   =   $CF
HIHLD   =   $D0
;
;       --------------
;       PLOT WORK DATA
;       --------------
;
PLOTX   =   $6F0
PLOTY   =   $6F1
;
        ORG $0600       ;PAGE 6
;
;       ---------------------
;       GR. 7+ PLOTTER ROUTINE
;       ---------------------
;
PLOTCL  LDA PLOTY       ;MULT. Y BY 40:
```

```
        ASL A
        STA LO
        LDA #0
        ROL A
        STA HI          ;*2
        ASL LO
        ROL HI          ;*4
        ASL LO
        LDA LO
        STA LOHLD
        ROL HI
        LDA HI
        STA HIHLD       ;*8
        ASL LO
        ROL HI          ;*16
        ASL LO
        ROL HI          ;*32
        LDA LO
        CLC
        ADC LOHLD
        STA LO
        LDA HI
        ADC HIHLD
        STA HI          ;+*8=*40
        LDA SAVMSC      ;ADD THE DISPLAY
        CLC             ;ADDRESS TO GET
        ADC LO          ;THE ACTUAL
        STA LO          ;ADDRESS OF THE
        LDA SAVMSC+1    ;BYTE THAT WILL
        ADC HI          ;BE ALTERED FOR
        STA HI          ;THE PLOT.
        LDA PLOTX       ;MASK PLOTX FOR
        AND #3          ;PLOT INDEX,
        TAX             ;PLACE IN X.
        LDA PLOTX       ;GET PLOTX AND
        LSR A           ;DIVIDE
        LSR A           ;BY 4,
        CLC             ;ADD TO
        ADC LO          ;PLOT ADDRESS
        STA LO          ;FOR FINAL PLOT
        LDA HI          ;ADDRESS.
        ADC #0
        STA HI
        LDY COLOR       ;GET COLOR
        LDA BMASK2,X    ;AND MASK OFF
        AND COLORS,Y    ;PIXEL POSITION
        STA HOLD        ;SAVE IT,
        LDA BMASK1,X    ;MASK OFF PIXEL
        LDY #0          ;OF THE ADDRESS
        AND (LO),Y      ;TO BE ALTERED
        ORA HOLD        ;SET THE PLOT
        STA (LO),Y      ;BITS AND STORE!
        RTS             ;FINIS!

;       ---------------
;       PLOT MASK TABLES
;       ---------------

COLORS  DB  $00,$55,$AA,$FF
BMASK1  DB  $3F,$CF,$F3,$FC
BMASK2  DB  $C0,$30,$0C,$03

        END
```

•

```
;       ==========================
;          GRAPHICS 7+ HANDLER
;       ==========================

;       ==========================
;         WRITTEN BY: TOM HUDSON
;       A.N.A.L.O.G. COMPUTING #11
;       ==========================

;       ------------------------
;       OPERATING SYSTEM EQUATES
;       ------------------------

DLISTL  =   $230        ;DISPLAY LIST LOW
DLISTH  =   $231        ;DISPLAY LIST HIGH
COLOR   =   $C8         ;BASIC COLOR

;       --------------------
;       MY WORKING VARIABLES
;       --------------------

LO      =   $CB
HI      =   $CC
ARGNUM  =   $CD
LOHLD   =   $CF
HIHLD   =   $D0
ENDPT   =   $D1

;       ----------------------
;       PLOT ADDRESS CALCULATOR
;       ----------------------

PLOTCL  =   $0600

;       --------------
;       PLOT WORK DATA
;       --------------

PLOTX   =   $6F0        ;PLOT X LOC.
PLOTY   =   $6F1        ;PLOT Y LOC.
DRAWX   =   $6F2        ;DRAWTO X LOC.
DRAWY   =   $6F3        ;DRAWTO Y LOC.
ACCX    =   $6F4        ;X ACCUMULATOR
ACCY    =   $6F5        ;Y ACCUMULATOR
DELTAX  =   $6F6        ;DRAWTO WORK AREA
DELTAY  =   $6F7        ;DRAWTO WORK AREA
INCX    =   $6F8        ;DRAW X INCREMENT
INCY    =   $6F9        ;DRAW Y INCREMENT
COUNTR  =   $6FA        ;DRAWTO COUNTER

;       --------------------
;       RELOCATABLE ROUTINES
;       --------------------

        ORG $6000       ;ANYWHERE

;       --------------------------
;       DISPLAY LIST INITIALIZATION
;       --------------------------

SETUP   PLA             ;(DISCARD)
        LDA DLISTL      ;FIND THE
        CLC             ;ADDRESS OF
        ADC #3          ;THE BASIC
        STA LO          ;DISPLAY LIST
        LDA DLISTH      ;AND PLACE
        ADC #0          ;IN A PAGE 0
        STA HI          ;WORKING AREA
        LDY #0          ;NO Y OFFSET
SCANDL  LDA (LO),Y      ;SCAN THE DISPLAY LIST:
        CMP #$4F        ;CHANGE
        BNE NOMLD       ;$4F (GR. 8)
        LDA #$4E        ;TO
        STA (LO),Y      ;$4E (GR. 7+)
        LDA LO          ;SINCE THIS WAS
        CLC             ;A 'LOAD MEMORY'
        ADC #2          ;INSTRUCTION (3 BYTES)
        STA LO          ;WE WILL SKIP
        LDA HI          ;THE NEXT 2 BYTES
        ADC #0          ;TO GET THE NEXT
        STA HI          ;DISPLAY LIST
        LDA #0          ;INSTRUCTION
        BEQ NXTDLB      ;ADDRESS.
NOMLD   CMP #$0F        ;CHANGE $0F (GR. 8)
        BNE NOREGL      ;TO
        LDA #$0E        ;$0E (GR. 7+)
        STA (LO),Y      ;AND GO TO
```

```
        BNE NXTDLB      ;NEXT D.L. INSTRUCTION.
NOREGL  CMP #$41        ;END OF DISP. LIST?
        BNE NXTDLB      ;NO: GET NEXT BYTE
        RTS             ;YES, EXIT!
NXTDLB  LDA LO          ;INCREMENT THE
        CLC             ;MEMORY POINTER
        ADC #1          ;TO GET THE
        STA LO          ;NEXT BYTE
        LDA HI          ;OF THE
        ADC #0          ;DISPLAY LIST
        STA HI          ;AND
        LDA #0          ;FORCE BRANCH
        BEQ SCANDL      ;BACK TO LOOP!

;       ------------------------
;       GRAPHICS 7+ PLOT HANDLER
;       ------------------------

PLOT    CLD             ;CLEAR DECIMAL MODE
        PLA             ;PULL # OF ARGUMENTS
        BEQ PULLED      ;OOPS-NONE!!
        STA ARGNUM
        AND #1          ;NEED EVEN#
        BEQ GOODPL      ;OK!
        LDX ARGNUM      ;NOT EVEN,
PULLEM  PLA             ;CLEAR STACK
        PLA
        DEX
        BNE PULLEM
PULLED  RTS             ;EXIT TO BASIC

GOODPL  PLA             ;(DISCARD)
        PLA             ;GET PLOT X
        CMP #160        ;ONSCREEN?
        BCS NOPLOT      ;NO!
        STA PLOTX       ;YES, SAVE
        PLA             ;(DISCARD)
        PLA             ;GET PLOT Y
        CMP #192        ;ONSCREEN?
        BCS DECARG      ;NO!
        STA PLOTY       ;YES, SAVE
        JSR PLOTCL      ;PLOT IT!!!
DECARG  DEC ARGNUM      ;HEY, WE HAVE
        DEC ARGNUM      ;2 LESS ARGS!
        BNE GOODPL      ;ANOTHER PLOT
        RTS             ;FINIS!
NOPLOT  PLA             ;PULL Y-COORD
        PLA             ;OFF STACK
        LDA #0          ;FORCE BRANCH
        BEQ DECARG      ;TO NEXT PLOT

;       ------------------------
;       GRAPHICS 7+ DRAW HANDLER
;       ------------------------

DRAW    CLD             ;CLEAR DECIMAL MODE
        PLA             ;PULL # OF ARGUMENTS
        BEQ PULLED      ;NONE!!
        STA ARGNUM
        AND #1          ;NEED EVEN #
        BEQ GOODDR      ;OK!
        LDX ARGNUM      ;NOT EVEN,
        LDA #0          ;FORCE BRANCH
        BEQ PULLEM      ;TO ABORT.

GOODDR  PLA             ;(DISCARD)
        PLA             ;GET DRAWTO X
        CMP #160        ;ONSCREEN?
        BCS NODRAW      ;NO!
        STA DRAWX       ;YES, SAVE IT
        PLA             ;(DISCARD)
        PLA             ;GET DRAWTO Y
        CMP #192        ;ONSCREEN?
        BCS DECPLA      ;NO!
```

```
        STA DRAWY       ;YES, SAVE IT
        BCC DRWCAL      ;DRAW LINE!
NODRAW  PLA             ;DISCARD Y
        PLA             ;COORDINATES
DECPLA  DEC ARGNUM      ;2 LESS
        DEC ARGNUM      ;ARGUMENTS
        BNE GOODDR      ;NOT DONE YET!
        RTS             ;FINISHED!

;       ---------------------
;       CALCULATE DRAW VECTOR
;       ---------------------

DRWCAL  CMP PLOTY       ;IS DRAWY)PLOTY?
        BCC YMINUS      ;NO!
        SEC             ;SUBTRACT
        SBC PLOTY       ;PLOTY FROM DRAWY
        STA DELTAY      ;AND SAVE DIFFERENCE.
        LDA #1          ;Y INCREMENT
        STA INCY        ;= 1 (DOWN)
        BNE XVEC        ;BRANCH!
YMINUS  LDA PLOTY       ;SUBTRACT
        SEC             ;DRAWY
        SBC DRAWY       ;FROM PLOTY
        STA DELTAY      ;AND SAVE DIFFERENCE.
        LDA #255        ;Y INCREMENT
        STA INCY        ;= -1 (UP)
XVEC    LDA DRAWX       ;IS DRAWX
        CMP PLOTX       ;) PLOTX?
        BCC XMINUS      ;NO!
        SEC             ;SUBTRACT
        SBC PLOTX       ;PLOTX FROM DRAWX
        STA DELTAX      ;AND SAVE DIFFERENCE.
        LDA #1          ;X INCREMENT
        STA INCX        ;IS 1 (RIGHT)
        BNE VECSET      ;BRANCH!
XMINUS  LDA PLOTX       ;SUBTRACT
        SEC             ;DRAWX FROM
        SBC DRAWX       ;PLOTX
        STA DELTAX      ;AND SAVE DIFFERENCE.
        LDA #255        ;X INCREMENT
        STA INCX        ;IS -1 (LEFT)
VECSET  LDA #0          ;ZERO OUT:
        STA ACCY        ;Y ACCUMULATOR
        STA ACCX        ;X ACCUMULATOR
        LDA DELTAX      ;IS DELTAX)
        CMP DELTAY      ;DELTAY?
        BCC YMAX        ;NO!
        STA COUNTR      ;SAVE DELTAX
        STA ENDPT       ;IN COUNTR, ENDPT.
        LSR A           ;DIVIDE BY 2 AND
        STA ACCY        ;STORE IN Y ACCUM.
        LDA #0          ;FORCE BRANCH
        BEQ DRAWGO      ;TO DRAWGO.

JDPLA   BEQ DECPLA      ;LEAPFROG JUMP

YMAX    LDA DELTAY      ;DELTAY LARGER,
        STA COUNTR      ;STORE IT IN
        STA ENDPT       ;COUNTR, ENDPT.
        LSR A           ;DIVIDE BY 2 AND
        STA ACCX        ;STORE IN X ACCUM.

;       ------------------------------
;       NOW WE START THE ACTUAL DRAWTO
;       FUNCTION!
;       ------------------------------

DRAWGO  LDA COUNTR      ;IF COUNTR=0...
        BEQ JDPLA       ;NO DRAW!
BEGIN   LDA ACCY        ;ADD DELTAY
        CLC             ;TO Y ACCUMULATOR
        ADC DELTAY
        STA ACCY
        CMP ENDPT       ;AT ENDPOINT YET?
        BCC BEGIN2      ;NO, GO DO X.
        LDA ACCY        ;SUBTRACT ENDPT
        SEC             ;FROM Y ACCUMULATOR
        SBC ENDPT
        STA ACCY
        LDA PLOTY       ;AND INCREMENT
        CLC             ;THE Y POSITION!
        ADC INCY
        STA PLOTY
BEGIN2  LDA ACCX        ;ADD DELTAX TO
        CLC             ;X ACCUMULATOR
        ADC DELTAX
        STA ACCX
        CMP ENDPT       ;AT ENDPOINT YET?
        BCC PLOTIT      ;NO, GO PLOT.
        LDA ACCX        ;SUBTRACT ENDPT
        SEC             ;FROM X ACCUMULATOR
        SBC ENDPT
        STA ACCX
        LDA PLOTX       ;AND INCREMENT
        CLC             ;PLOT X
        ADC INCX
        STA PLOTX
PLOTIT  JSR PLOTCL      ;PLOT THE POINT!
        DEC COUNTR      ;MORE TO DRAW?
        BNE BEGIN       ;YES!
        LDA #0          ;NO MORE,
        BEQ JDPLA       ;FORCE BRANCH.

        END
```

•

ADD A FLOPPY DISK TO YOUR ATARI* FOR LESS THAN $450

Now Atari* users can upgrade to 5″ or 8″ floppy disks at an affordable price with **MICRO MAINFRAME.**

Micro Mainframe, the premier manufacturer of low cost add on peripherals for the TRS-80 computer, is proud to announce our lineup of products for the Atari* computer. Micro Mainframe is not a newcomer to the computer marketplace, as we have been supplying quality disk drives for the TRS-80* for three years. We now bring our expertise to the Atari* market and pledge to provide the highest quality peripherals at the lowest price.

Our floppy disk drive is designed to sell for under $450, and check out the features:

- **Low initial cost includes one disk drive and controller**
- **Single and Double Density operation STANDARD**
- **Additional drives available for under $200**
- **Controller features a proprietary digital phase lock loop data separator for unparalleled reliability**
- **Operates 8″ disk drives with our MaxiDos A operating system (AVAILABLE MID-1983)**
- **Intermix 5″ and 8″ disk drives with MaxiDos A**
- **Includes a parallel printer port to allow the use of low cost printers**
- **Expandable to include hard disk operation (requires expansion chassis and host adapter)**
- **Includes a SWITCHING POWER SUPPLY that is capable of running two disk drives**
- **Can be used in conjunction with other Micro Mainframe peripherals**

The expensive Atari* 810 disk drive includes a costly disk controller and power supply for each drive. This is wasteful as one controller can easily control up to four disk drives. With a **MICRO MAINFRAME** drive, you pay only once for the disk controller. To add a second disk drive, you merely purchase a disk drive and case. The power supply in your first drive will power an additional disk drive. Other manufacturers use cheap linear power supplies which generate considerable heat that can shorten the life of your electronic equipment.

ADD A DISK DRIVE TO YOUR ATARI* FOR LESS THAN $90

If you already own a TRS-80 Model-III*, you can use your Model-III* as an intelligent disk controller with the addition of our **CONNECTION A™.** This peripheral connects between the 50 pin expansion bus on the Model-III* computer and the Atari* 400 or 800 computer. After loading the supplied software package, your Model-III* computer thinks that it is an Atari* disk drive and you can read and write Atari* diskettes on your Model-III* for less than $90.

Dealer Inquiries are Solicited

MICRO MAINFRAME

11325 Sunrise Gold Circle
Building E
Rancho Cordova, CA 95670
(916) 635-3997

*Atari is a trademark of Atari, Inc. TRS80 is a trademark of the Tandy Corp.

FIRST BORN IN 1978!

the original & continuously updated

CCA Data Management System

Now Available For Atari Computers	$ 99.50
For Apple Computers	150.00
For CPM Based Computers	225.00

CCA Data Management System

Uses	Features And Capabilities
• Business Accounts Receivable Accounts Payable Inventories Billing Lists and Rosters • Home Phone Lists Budgets, Hobbies	• Long record lengths • Up to 24 fields per record • **Not Copy Guarded** • Alpha numeric items • Numeric only items • Add, update, scan, etc. files • 10-Level sort ascending, descending, allows alphabetizing data file.

• Contact your local dealer for details or write us for our catalog

DIVISION OF CUSTOM ELECTRONICS, INC.
SOFTWARE

238 Exchange St., Chicopee, Massachusetts 01013
(413) 592-4761
Mastercard & VISA Accepted
• Dealer And Distributor Inquiries Invited
• Closed Mondays — Open Daily 'Til 5:30 — Fridays 'Til 8

NEW FOR YOUR ATARI

MAGIC MAIL*

An exciting New Data Base Mailing and Personal Information System for your Atari 400/800** Personal Computer. Never before have all these features been designed into a single Mailing and Information System, so easy to use and so powerful.

COMPARE THESE FEATURES:

- Data Fields: Name, Address, City, State, Zip Code, Area Code, Phone Number, Birthday (m/d/y), and Entry Type.
- Up to 1027 records per Diskette!
- Less than 1 second Search Time over 95% of your Data, and 2-4 seconds over the other 5%.***
- Up to 62 SEPARATE and UNIQUE files permitted per Diskette.
- Single or Double Disk Drive Capability.
- Built in SORT, capable of Sorting ANY Data File by ANY Field with up to 10 Sub-fields.
- Command-Driven with simple English-like commands: FIND, MODIFY, PRINT, ADD, SORT, etc.
- Position Independent & Position Dependent Fields.
- Files compatible with Atari DOS II.**
- Recorded on High-Quality Verbatim Diskettes.
- Every Diskette Pre-tested for Bad Sectors and recording integrity.
- Versions for: Epson MX-80, Prowriter, Centronics
- Print Single-width Mailing Labels or Directory Listing.
- 100% Machine Language.
- 16 K Minimum Memory.
- 60-Page User Manual.
- 90 Day Buyer Protection Guarantee. Call or write for FREE additional information.

*MAGIC MAIL is a Trademark of A-BIT-BETTER Software
**ATARI is a Registered Trademark of Atari Inc.
***Search time is for an Indexed File.

ABBS
A-Bit-Better Software
Post Office Box 28
Laurel Md 20707
(301) 953-7256

Dealer Inquiries Invited.

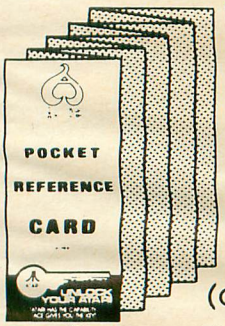
ONLY $59.95
*Plus tax, shipping and handling.
VISA AND MASTERCARD ACCEPTED.

GETTING SERIOUS ABOUT PROGRAMMING YET?!?

ATARI*

UNLOCK YOUR ATARI

WITH THE WORLD'S MOST COMPLETE POCKET PROGRAMMING AID.



16 comprehensive pages

- error codes
- basic commands with abbreviations
- peek and poke locations
- internal codes
- machine language aides
- much, much more!!

Get Yours Now only $9.95

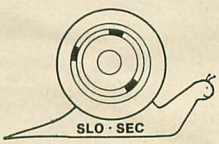
(dealer ad space available)

ORDER NOW-CASH, CHARGE OR C.O.D.

ADVANCED COMPUTING ENTERPRISES
5516 ROSEHILL A-1
SHAWNEE MSN., KS 66216
(913) 262-2875 • (913) 631-4180

ATARI IS A REGISTERED TRADEMARK OF ATARI INC

ACE



SLO·SEC

(SLOw speed SECtor write kit)

CREATE YOUR OWN PROTECTED DISKS WITH YOUR ATARI DISK DRIVE

* SLO-SEC enables your ATARI 810 disk drive to write unreadable (bad) sectors used in bad-sector disk protection routines.
* Easy disk drive modification requires only two solder connections.
* SLO-SEC will not interfere with normal disk drive operation.

Complete SLO-SEC kit:
Hardware and Step-by-Step Instructions: $9.95

Note: SLO-SEC is designed for second generation drives (manufactured after December, 1981)

For fast service use certified check, money order or charge card.
Please allow 2 to 3 weeks for personal checks to clear.

Please send _____ SLO-SEC kits at $ 9.95 each.

Name _____
Street _____
City _____ State _____ Zip _____

MASTERCARD # _____ Exp. date _____
VISA # _____ Exp. date _____

Signature _____

COMPUWARE
1704 Princess St.
Wilmington, NC 28405

ATARI is a registered trademark of ATARI, Inc.

The New AtariWriter Word Processor

by Richard Kushner

This article is being written using the new **AtariWriter** word processor. This immediately tells you that it can't be too hard to use, since I've only had it in my hot little hands for a few hours and have only skimmed the manual. The ease of my learning to use this software is partly due to its simplicity and partly due to its obvious debt to the Text Wizard word processor. (Everyone has probably heard the rumor (fact?) that the ***AtariWriter*** was written by the author of Text Wizard.) In any event, we are not here to speculate but to demonstrate!

The **AtariWriter** is a 16K ROM cartridge that will work with either disk or tape. With 48K of memory, you have about 20K of memory free for your text file. This amounts to about 14 double spaced pages. Longer documents are handled by chaining files together. This means that you put a code at the end of "part one" of a long file that calls the next piece into memory to be printed and continue this process until the entire file is printed.

The menu that confronts the user initially (and the only menu) consists of eight commands. You merely type the first letter of the command and you are then prompted what to do next. The function of each command is:

—CREATE File: Use this to start from scratch to create a text file. If there is already a file in memory, you are prompted whether or not you want it erased.

—DELETE File: You use this to erase a file on disk. There is failsafe prompt before it will actually delete the file. Tape users should just write over a file to delete it.

—EDIT File: This will bring you back to the text in memory so that you can insert, delete, add or perform whatever your editing need happens to be.

—FORMAT Disk: This is handy if you find yourself ready to save, to disk, but don't have any formatted disks. This will not write DOS files on the disk. (Note that DOS files must be on your data disk when you turn on your computer to use **AtariWriter.**)

—INDEX of Disk Files: Like it says, this will give you an on-screen listing of all the files on your date disk without destroying any file currently in memory. You can also, at your option, print out this listing (a very nice feature!).

—LOAD File: This will move a text file from disk or tape into memory where it can then be edited or printed.

—PRINT File: This is used to print your finished document on your printer. Read on for more information on this command.

—SAVE File: The obvious opposite to LOAD. If your filename for saving is the same as one already on the disk, you will be asked if you wish to erase the file currently on disk.

Once you go into the CREATE file mode, you will see an array of items surrounding the blank text window. Along the bottom are a series of arrows that indicate where the TAB positions are set. Default values are every five columns and they can be altered by the user. However, the altered TAB positions are not saved when the file is saved, so any later editing that needs other than standard TAB settings will have to have them changed again. This seems to be only a minor nuisance and falls into the "you can't have everything in 16K" category. Also on the bottom of the screen are the name of the file currently in memory (if it was retrieved from disk) and two numbers that tell you the current line and column location of the cursor (of dubious value). By the way, the cursor is a flashing underline rather than the usual ATARI flashing block and is much better for word processor work.

The very first line at the top of the screen shows the default settings for file formatting. These are all easily alterable by the user. The default values are:

Bottom margin: 12 half-lines (1 inch)
Paragraph spacing: 4 half-lines
Print style: 10 characters per inch
Paragraph indentation: 5 spaces from left margin
Justified right margin: Off
Left Margin: 10 spaces from left edge of page
Right margin: 70 spaces from left edge of page
Line spacing: 2 half-lines (single spacing)
Top margin: 12 half-lines (1 inch)
Page length: 132 half-lines (11 inches)

Like **Text Wizard,** the **AtariWriter** works in half-lines, but differs in using "real" spaces to measure across the page (much more convenient than **Text Wizard**). Any modifications that are made in these formatting values are saved when the file is saved.

To create a text file, you merely type on the blank

screen "page". You do not use the return key unless you want to start a new paragraph. If a word at the end of a line will not fit on that line, the word processor moves it to the next line. All typing is done on a screen that has 36 characters per line. This was presumably intended to take care of television sets that overscan and, therefore, will not show the full 40 characters/line that are available. To start a paragraph with indentation you can just type CTRL-P as the first character and you will see a "P" symbol, which is easily recognizable as a paragraph sign. Upon printing, this will be interpreted as a command to indent five spaces.

Having typed your document on the screen you can then edit it to remove mistakes and alter its content. It is the editing capabilities that really make a word processor plus computer more than just an expensive typewriter. Gone is whiteout. Gone is retyping an entire document because of missing one word in the middle. With the screen editing capabilities of **AtariWriter** you can easily correct those "typos" before they get on paper. You can use the usual CTRL+arrow keys to move around the text file one character at a time or you can use other commands to move around in bigger steps. SELECT+T moves you instantly to the start of the file (T for top) and SELECT+B moves you to the end of the file (B for bottom). OPTION+up or OPTION+down arrow moves you up or down one screen at a time. CTRL+A moves to the beginning of the line that the cursor is on and CTRL+Z moves to the end of that line (A and Z being the first and last letters of the alphabet as well as being conveniently close to the CTRL key). All of these commands allow you to quickly get to where the editing needs to be done. You then can type in any additions and the program will move text out of the way to make room. You can also delete the character to the left of the cursor, the character above the cursor, to the end of the line or to the file. You can also delete blocks of text that you delineate with CTRL-X characters. If you accidentally delete text you wanted to keep, you can also take back you last command (with some limitation on the amount of deleted text that can be restored). This is possible because there is a buffer that temporarily holds any deleted text.

Additional powerful features are the ability to duplicate or move blocks of text. This can be useful to improve the meaning of a text file by shifting around its parts without having to retype or when a document has several similarly worded parts and, again, retyping can be minimized. Typing changes can also be minimized by using the Search and Replace functions. For example, you may have

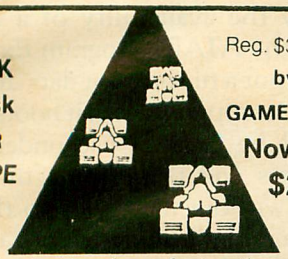
Royal Software

Make Royal Your ATARI® Source!

ATARI®

Over 1000 items for your Atari!

BAJA BUGGY



16K Disk OR TAPE

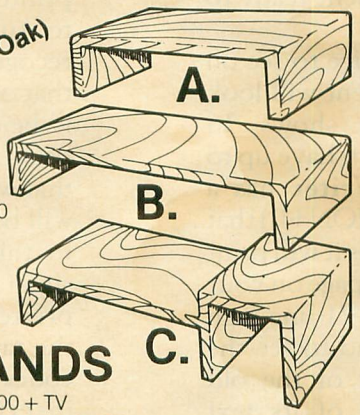
Reg. $32.95 by GAMESTAR

Now Only $29.60

If you're looking for a fast action driving game then we recommend this one. Good graphics and sound.

Sits over computer to hold your TV/Monitor.

(Solid Oak)



Add $7.00 each for shipping.

T.V. STANDS

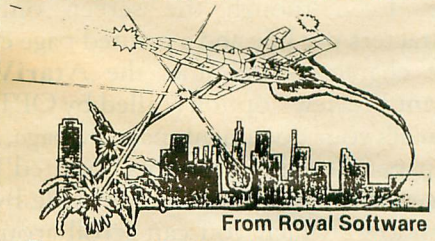
A. 800/400 + TV
B. 800/400 + 810 + TV
C. 800/400 + 2-Percoms + TV

A. = $29.95 B. = $39.95 C. = $44.95

METEOR STORM

16K Tape/Disk

NOW ONLY $29.95



From Royal Software

The city is about to be destroyed and ONLY you can save its people from destruction. FAST, arcade type action game.

FOOT BALL

★ Pass
★ Run
★ Kick

- One or Two Players
- 24K RAM Disk or Tape

Great Graphics Scrolling Field

Reg. $31.95

OUR PRICE $28.90

By Gamestar

INTRODUCING. . .

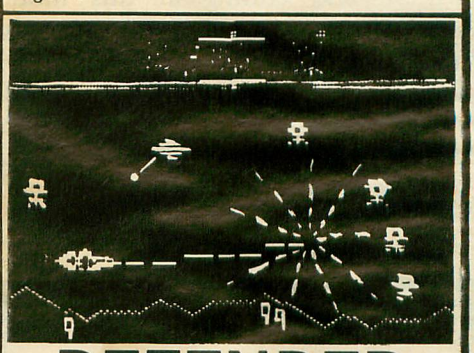
From Royal Software

48K DISK

$49.95

The most user-friendly mail maintenance program available! Here are a few of the outstanding features. . .

- New OS that offers fewer 'crashed-disks'-True random access-Store 500 to 2000 names/addresses per disk.
- Works on single or double density.
- Automatic delete of duplicates.
- Print a disk directory.
- Official state abbreviations are built-in.
- Make back-up copies. ● Merge files
- Search files ● Sort files
- Create sub-files ● Print lables/file copy



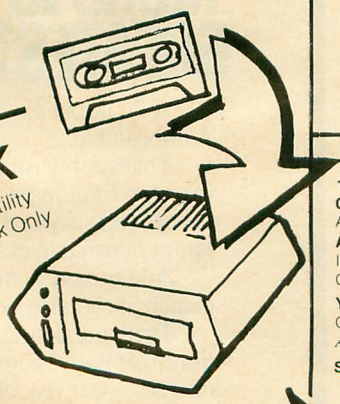
DEFENDER

New! ● 16K Cartridge From Atari®
● Super Excitement

Reg. $44.95 **NOW ONLY $39.50**

TAPE —TO— DISK

- A Disk Utility
- 32K Disk Only



BONUS!!!

★ Disk Speed Check
★ Screen Dump To Printer

From Royal Software

$29.95

BOOKS-BOOKS-BOOKS

1st BOOK OF ATARI, by Compute **$12.00 each**
Computes's 2nd BOOK OF ATARI
A collection of relevant articles from Compute Magazine.

ATARI TECHNICAL USERS NOTES, by Atari
Includes: Hardware manual, Operating system manual, Operating system source listing... Very Useful! **$26.90**

YOUR ATARI COMPUTER, by Lon Poole
One of the most comprehensive books on the Atari!
A must for the beginner. . . **$16.95**

SHIPPING NOTE: Add $1.00 per book for shipping.

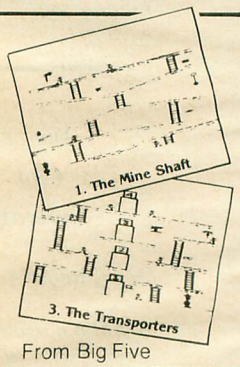
New!

MINER 2049er

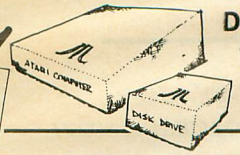
- 10 Different Screens
- 16K Cartridge
- Spectacular Sound & Graphics

Reg. $49.95

OUR PRICE ONLY $44.90



From Big Five



DUST COVERS

- 400/800 Computer
- 410 Recorder
- 810 Disk Drive

Only $8.95 Each

Free Catalog

Full of program descriptions
►WITH ANY ORDER◄ OR
SEND $1.00 Refundable with first order
Send us your name/address for Free Flyers!

Royal Software

2160 W. 11th Avenue
Eugene, Oregon 97402

48 Hour Shipping Available

For Credit Card Orders Call
1-800-452-8013

Open 8am - 6pm, Mon-Sat.

We specialize in *ATARI Compatible Products*
So we know what works best!!!
Call for the latest info. . . (503) 683-5361

Shipping: UPS, Parcel Post = $2.00—UPS Air = $3.90
Shipping on hardware & paper extra-Call for costs

consistently misspelled a word. You can easily correct this by using these functions to locate and then replace the offending word. You have the option whether to change the word at each occurrence or whether to change all of them at once. Combining all of these commands and features will get you to the point of having a document ready to see the light of the printed page.

But wait a minute! Wouldn't it be nice to see on your TV what the final printed document will look like? Even though the screen only shows 36 characters per line and a printed page can have up to 132 characters per line, the **AtariWriter** has a "Print Preview" option (called by OPTION+P) that formats your document, page by page, as it will print on paper. You then view this "printed" page on your screen through a 36 character wide by 21 line high "window" which you can scroll around to see the entire page. This is very reminiscent of the old ATARI Word Processor that did all of its text creation and editing in this mode. With the **AtariWriter**, however, you can only look at the final form of the document. To do more editing you must go back into the normal 36 character per line edit mode. Nonetheless, the ability to preview the final printed version is a very nice feature to have.

We are now ready to transfer that document in memory onto the printed page. The first time you do this in any **AtariWriter** session you will be asked to choose a printer from among the ATARI 1025, the ATARI 825, the ATARI 820 and the ATARI 822. What, you say, you don't have any of those printers! You then choose the ATARI 822 option to get your printout. This word processor was originally written to support only the ATARI printers (failing to recognize that the Epson was the most popular and that others were also in use by ATARI owners). This is being remedied by the availability of a disk of printer drivers from the ATARI Program Exchange that will allow you to put a file on your data disk that will be loaded into the computer when it boots DOS. In this way you will be able to use underlining, superscripts, subscripts and print fonts that your printer may support without having to go through the task of inserting a complicated series of control codes and numbers. The **AtariWriter** does permit inserting other printer codes using CTRL+0 followed by the decimal value of the printer code (for example, 27 for the ESCape code). This is a nice capability to have, since it is not possible to include all possible features of all possible printers in the word processor itself. It is, however, quite cumbersome to use and requires that you have ready access to the control code values. Thus, it is nice to

have the most useful control codes built into the **AtariWriter**. The optional printer driver disk will give you this capability.

You will notice that I have pretty much avoided comparing the **AtariWriter** with other word processors available for the ATARI computer. I have done this because I believe that this product addresses itself best to those who are new at finding applications for their computer and want to get involved in some "simple" word processing (i.e., a note to Aunt Em, a letter to their Congressman, a brief article for their local ATARI group newsletter). The **AtariWriter** has good documentation, is reasonably easy to learn and to use, doesn't seem to leave you hung up anywhere and has sufficient commands and flexibility to meet the needs of those who are most likely to use it. You really can't ask for much more from a word processor that sells for less than $100. Yet, as your uses for word processing grow (and they will), you will still be able to use **AtariWriter** by employing some of its more advanced features. After a series of strikeouts, ATARI has a hit on their hands.□

EPSET

6K PRINTER UTILITY (DISK/CASS.)

by Dick Tedeschi

Those of you with new Epson printers, if you are like me, have been frustrated by all those wonderful CONTROL CODES available to do anything from change the number of characters per line, change line spacing, select full width/double-width/half-width, etc.

There are an impressive number of these "print options" available, especially when you "mix and match" them to create unusual combinations for a specific application.

This program allows you to choose from among the more commonly used commands by way of a menu which issues the codes to the printer as well as updating the menu as the option is chosen.

I have found this valuable in setting up VISICALC (R) sheets for printing and have used it with a multitude of programs in addition to printing out program listings (e.g., fit a 1 ½ page listing on one page).

I leave the PAPER OUT switch (#1-3) ON (Left) which turns the paper out sensor OFF (does that make sense?). Then if I want to print a single sheet I can do so without buzzers ringing and the printer halting at the halfway point. If I am using pin-fed forms I can select (P) on the menu and the sensor is enabled which enables me to walk away and not worry about the print roller getting messed up.

The great part of all this is that the EPSET program can be run and then the 400/800 and/or the disk drive can be turned off and any program (except a few like LETTER PERFECT which internally reset & control the printer) can be loaded without affecting the print status previously selected.

(H)elp and (R)eset options are included. Reset will simply return the printer to power up state while Help gives a brief (one screen) description of the program.

More options which can be added include italics (I use inverse char.) and 480/960 dot graphics select.□

Program Variables

A$-W$: print option strings
K : input Character variable
CH : number of characters per line
DL : number of dots per line (1-85)
FL : number of lines per 11″ form
PF : number of lines to skip before perf

```
10 REM  EPSET.BAS
15 REM
20 REM  Epson MX-80 III  F/T Printer
25 REM  Control Code Set-up Program
30 REM  for use with VISICALC, Etc.
35 REM
40 REM  DICK TEDESCHI  Jan.16, 1983
45 REM  1014 Main Street
50 REM  Norwell, MA 02061
55 REM
60 REM  Try: A C 5 Q (really small)
65 REM  Try: A 5 Q (AQ-double width)
70 REM
110 CLOSE #1:CLOSE #7
115 TRAP 115:STATUS #7,PE:IF PE=138 TH
EN GRAPHICS 2:POSITION 3,6:? #6;"TURN
ON printer":END
120 OPEN #1,4,0,"K:"
130 OPEN #7,8,0,"P:"
135 ? #7;CHR$(27);CHR$(64)
140 GRAPHICS 0
150 REM 155 POKE 710,254:POKE 712,138:
POKE 709,202
155 POKE 710,130:POKE 712,130:POKE 709
,10
160 CLR
170 DIM A$(40),B$(40),C$(40),D$(40),E$
(40)
180 DIM F$(40),L$(40),P$(40),Q$(40),S$
(40),U$(40),W$(40)
190 A$="(A) subscript FONT (toggles B5
:C:S)"
200 B$="(B) N/72 INCH LINE"
210 C$="(C) COMPRESSED"
220 D$="(D) DOUBLE STRIKE"
230 E$="(E) EMPHASIZED NOT with (C) or
 (S)"
240 F$="(F) SKIP OVER PERF"
250 L$="(L) CHARACTERS PER LINE"
255 P$="(P) PAPER-OUT ENABLE (SW.1-3 =
L)"
260 Q$="(Q) QUIT"
270 S$="(S) SUBSCRIPT"
280 U$="(U) UNIDIRECTIONAL PRINT"
290 W$="(W) DOUBLE WIDTH"
295 TRAP 297:GOTO 300
297 RUN
300 POSITION 8,0:? "**********************
********"
310 POSITION 8,1:? "* EPSON PRINTER OP
TIONS *"
315 POSITION 8,2:? "**********************
********"
320 ? "        (H) HELP       (R) RESET"
:?
330 ? A$:? B$:? C$:? D$:? E$:? F$
340 ? L$:? P$:? Q$:? S$:? U$:? W$
430 POSITION 7,22:? "  Type 'Q' when f
inished  "
440 POSITION 10,18:? "YOUR CHOICE(S):
";
450 GET #1,K
455 IF CHR$(K)="H" THEN GOSUB 800:? "K
"
457 IF CHR$(K)="R" THEN RUN
460 IF CHR$(K)="A" THEN ? #7;CHR$(27);
CHR$(65);CHR$(5);CHR$(15);CHR$(27);CHR
$(83);CHR$(1)
465 IF CHR$(K)="A" THEN B$(4,5)="*5":C
$(4,4)="*":S$(4,4)="*":DL=5
470 IF CHR$(K)="B" THEN POSITION 2,19:
```

```
? ")DOTS/ LINE";:INPUT DL:? #7;CHR$(27
);CHR$(65);CHR$(DL):B$(4,5)="*N"
480 IF CHR$(K)="C" THEN ? #7;CHR$(15):
C$(4,4)="*"
490 IF CHR$(K)="D" THEN ? #7;CHR$(27);
CHR$(71):D$(4,4)="*"
500 IF CHR$(K)="E" AND C$(4,4)()"*" AN
D S$(4,4)()"*" THEN ? #7;CHR$(27);CHR$
(69):E$(4,4)="*"
510 IF CHR$(K)="F" THEN POSITION 2,20:
? ")PERF LINES";:INPUT PF:? #7;CHR$(27
);CHR$(78);CHR$(PF):F$(4,4)="*"
520 IF CHR$(K)="L" THEN POSITION 2,21:
? ")CHAR/ LINE";:INPUT CH:? #7;CHR$(27
);CHR$(81);CHR$(CH):L$(4,4)="*"
525 IF CHR$(K)="P" THEN ? #7;CHR$(27);
CHR$(57):P$(4,4)="*"
530 IF CHR$(K)="Q" THEN ? #7:Q$(4,4)="
*":GOTO 580
540 IF CHR$(K)="S" THEN ? #7;CHR$(27);
CHR$(83);CHR$(1):S$(4,4)="*"
550 IF CHR$(K)="U" THEN ? #7;CHR$(27);
CHR$(85);CHR$(1):U$(4,4)="*"
560 IF CHR$(K)="W" THEN ? #7;CHR$(27);
CHR$(87);CHR$(1):W$(4,4)="*"
570 GOTO 300
580 ? "K"
590 POSITION 7,5:? "** EPSON PRINT MOD
E **"
600 POSITION 7,6:? "   ---------------
-"
610 ?
620 IF C$(4,4)()"*" THEN ? " REGULAR F
ONT"
630 IF A$(4,4)="*" THEN ? A$(4)
640 IF B$(4,4)="*" THEN ? B$(4);"  (";
DL;"/72)"
650 IF C$(4,4)="*" THEN ? C$(4)
660 IF D$(4,4)="*" THEN ? D$(4)
670 IF E$(4,4)="*" THEN ? E$(4,15)
680 IF F$(4,4)="*" THEN ? F$(4);"  (";
PF;" LINES)"
690 IF L$(4,4)="*" THEN ? L$(4);"  (";
CH;"/LINE)"
695 IF P$(4,4)="*" THEN ? P$(4,20);"D"
700 IF S$(4,4)="*" THEN ? S$(4)
710 IF U$(4,4)="*" THEN ? U$(4)
720 IF W$(4,4)="*" THEN ? W$(4)
721 IF DL=0 THEN DL=12
722 FL=INT(66*(12/DL))
725 ? :? "   This mode gives ";FL;" li
nes/ 11 in."
730 IF Q$(4,4)="*" THEN ? :? " YOU MA
Y NOW ENTER OR RUN A PROGRAM"
735 IF Q$(4,4)="*" THEN ? " EVEN TURN
 OFF THE 800 AND/OR DRIVE"
737 IF Q$(4,4)="*" THEN ? " THE EPSON
 WILL REMAIN IN THIS MODE"
740 CLOSE #1:CLOSE #7
750 END
800 ? "K          EPSET  INSTRUCTIONS"
810 ? :? "    This is a program to sen
d all"
820 ? "those CONTROL CODE combinations
 to"
830 ? "the Epson MX-80 printer for suc
h"
840 ? "things as compressed print or '
sub-"
850 ? "script font' (try this I use it
 to"
860 ? "get 150+ lines on an 11 inch sh
eet"
870 ? "of paper!!!) or Double-Width or
 any"
880 ? "other things you may need (mix
and"
890 ? "match).(subscript font = A C S
Q )"
900 ? :? "All you need do is:"
910 ? "1. TURN ON PRINTER & INTERFACE
MOD."
920 ? "2. RUN THIS PROGRAM (RESETS PRI
NTER)"
930 ? "3. ENTER THE LETTER OF THE OPTI
ON(S)"
940 ? "   YOU DESIRE (A '*' WILL APPEA
R"
950 ? "   NEXT TO THE OPTION AS IT IS
SENT)"
960 ? "4. WHEN DONE TYPE 'Q' (AS IN QU
IT)."
970 ? "5  YOU CAN NOW TURN OFF THE COM
PUTER"
980 ? "   AND/OR THE DISK DRIVE. THE E
PSON"
990 ? "   WILL REMEMBER!!!!!"
1000 POSITION 10,23:? "HIT START TO BE
GIN";
1010 IF PEEK(53279)()6 THEN 1010
1020 RETURN
```

•

CHECKSUM DATA
(See D:CHECK/C:CHECK,p.26)

```
10 DATA 573,866,231,177,613,870,494,43
7,129,874,367,589,863,731,160,7982
120 DATA 264,298,909,896,847,807,64,32
6,706,756,263,949,294,810,524,8713
250 DATA 106,38,173,186,679,551,899,12
8,973,307,985,66,641,819,578,7129
440 DATA 414,571,52,820,238,67,458,719
,777,691,153,92,811,839,689,7391
550 DATA 706,723,716,394,529,517,898,6
37,623,900,637,644,939,430,630,9923
695 DATA 131,688,699,710,26,942,204,28
0,922,147,752,53,880,641,255,7330
830 DATA 619,635,304,686,304,434,518,5
28,836,721,659,164,495,595,829,8327
980 DATA 578,445,277,601,781,2682
```

•

```
10 REM *** SNOWFLAKE GENERATOR ***
20 REM
30 REM BY TOM HUDSON
40 REM
50 REM SET UP GRAPHICS MODE, COLORS
60 REM
70 GRAPHICS 8+16:SETCOLOR 2,0,0:COLOR
1
80 REM
90 REM SET UP DEGREES, X AND Y TABLES
100 REM
110 DEG :DIM D(10),X(10),Y(10)
120 REM
130 REM RANDOMIZE SHAPE
140 REM
150 FOR I=1 TO 10:D(I)=0:X(I)=RND(0)*8
0:Y(I)=RND(0)*I*4:NEXT I:POKE 77,0
160 REM
170 REM ECHO AND ROTATE SHAPE
180 REM
190 PLOT 160,96:FOR I=1 TO 10:DRAWTO 1
60+(X(I)*COS(D(I))+Y(I)*SIN(D(I))),96+
(-X(I)*SIN(D(I))+Y(I)*COS(D(I)))
200 D(I)=D(I)+60:NEXT I:IF D(1)<360 TH
EN 190
210 FOR I=1 TO 10:D(I)=0:NEXT I
220 PLOT 160,96:FOR I=1 TO 10:DRAWTO 1
60+(X(I)*COS(D(I))-Y(I)*SIN(D(I))),96+
(-X(I)*SIN(D(I))-Y(I)*COS(D(I)))
230 D(I)=D(I)+60:NEXT I:IF D(1)<360 TH
EN 220
240 REM
250 REM LEAVE IT ON SCREEN A WHILE
260 REM
270 FOR DELAY=1 TO 5000:NEXT DELAY:RUN
```

•

CHECKSUM DATA
(See D:CHECK/C:CHECK,p.26)

```
10 DATA 663,253,798,257,711,261,187,26
5,25,74,612,80,292,86,825,5389
160 DATA 92,940,98,542,156,151,527,152
,88,624,94,193,3657
```

HERE ARE SOME EXAMPLES OF WHAT EPSET CAN DO.

This is SUBSCRIPTED font (B5,C,S,D)

This is SUBSCRIPTED italic

THIS IS DOUBLE STRIKE,EMPHASIZED
WITH 9/72 INCH LINE SPACING

THIS IS COMPRESSED,DOUBLE STRIKE, UNIDIRECTIONAL,DOUBLE WIDTH WITH NORMAL (12/72 INCH) LINE SPACING

THIS IS A DEMO OF SUBSCRIPT FONT
TO SHOW THE EFFECT OF 5/72 INCH LINE SPACING
YOU CAN GET A LOT OF LINES (OVER 150) TO A PAGE THIS WAY

THIS IS D,E,W (DOUBLE STRIKE, EMPHASIZED,DOUBLE WIDTH)

THIS IS COMPRESSED, DOUBLE STRIKE
WITH ONLY 8/72 INCH SPACING (NOT ENOUGH SHOULD BE 9/72)
BUT IT DOES WORK AND GIVES 99 LINES TO AN 11 INCH FORM.....

HOW ABOUT COMPRESSED,SUBSCRIPT,DOUBLE WIDTH?

THIS IS DOUBLE STRIKE, SUBSCRIPT, DOUBLE WIDTH

THIS IS SUBSCRIPTED, DOUBLE WIDTH

THIS IS SUBSCRIPT ONLY
IT IS STILL A REDUCED SIZE BUT LARGER THAN COMPRESS SUBSCRIPT
THIS IS "NORMAL" (12/72 INCH) SPACING)

Finally, a computer game you can BEAT your kid at

It's the same old story. Your kid boots up his favorite space swashbuckler and pleads, "Ah, come on, Dad, just one game of Starstruck."
Naturally, you resign yourself to humiliating defeat. Another day, another loss.
Well, take revenge with Golf Challenge™!
Practice by yourself, rain or shine, then challenge your kid or the whole family to nine or 18 holes of championship golf - up to four at a time.
Soon you'll be saying, "C'mon, son, just one more game of Golf Challenge™."

Order Golf Challenge™ cassette for Atari 400/800 from your local computer dealer for $24.95 or directly from Sierra On-Line, Inc., Sierra On-Line Building, Coarsegold, CA 93614 (209) 683-6858. Add one dollar for shipping and handling. VISA • MASTERCARD • CHECK • COD ACCEPTED

Sierra ON-LINE INC

SIERRAVISION™

WHEN YOU SIT DOWN TO A SCOTT ADAMS' GRAPHIC ADVENTURE . . .

. . . ANYTHING CAN HAPPEN!

That's right — anything can happen, and it usually does! Because with a Scott Adams Graphic Adventure, the fantastic is as close as your computer — and your own fertile imagination. Each S.A.G.A. features the finest in hi-resolution graphics — graphics which compliment and enhance the classic text — and, they can be toggled on or off with a single keystroke, too. Plus, optional Votrax Type 'N Talk™ and printer compatibility give you the flexibility to add new dimensions to your Adventuring — when you're ready.

The incomparable S.A.G.A. Series — experience the magic yourself. But be prepared for anything.

S.A.G.A. #1 — Adventureland (Skill Level: Moderate)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0201	$39.95
ATARI 48K Disk	052-0201	$39.95

S.A.G.A. #2 — Pirate Adventure (Skill Level: Beginner)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0202	$39.95
ATARI 48K Disk	052-0202	$39.95

S.A.G.A. #3 — Mission Impossible (Skill Level: Advanced)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0203	$39.95
ATARI 48K Disk	052-0203	$39.95

S.A.G.A. #6 — Strange Odyssey (Skill Level: Moderate)

APPLE 2 PLUS 48K Disk (DOS 3.3 req.) . .	042-0206	$39.95
ATARI 48K Disk	052-0206	$39.95

PRICES SUBJECT TO CHANGE

Also, look for the Scott Adams Text Adventure Series on these fine computers:
TRS-80 • COMMODORE VIC • NEC PC-8001
TEXAS INSTRUMENTS 99/4 • CP/M Z-80

Adventure INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC

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