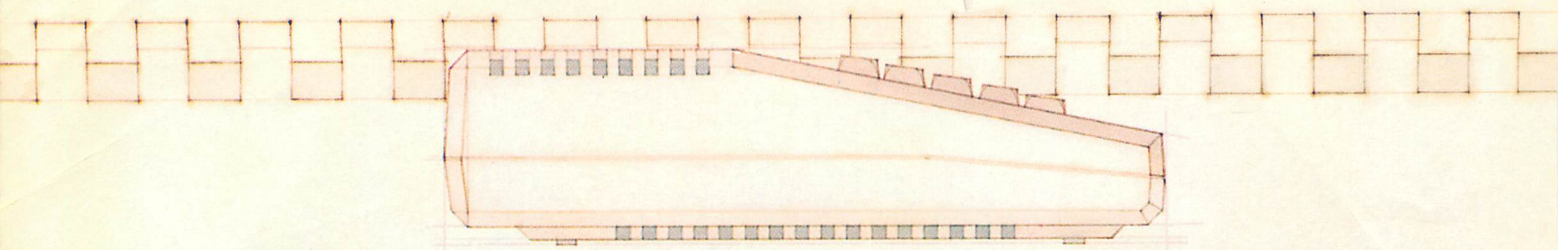
NO. 22 SEPTEMBER 1984 $3.00

THE MAGAZINE FOR ATARI® COMPUTER OWNERS

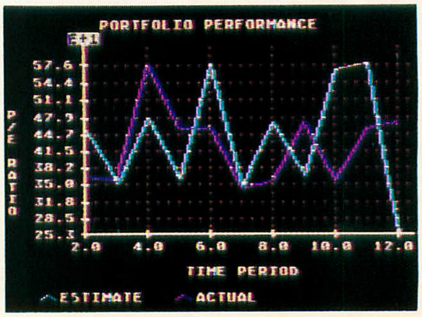
A.N.A.L.O.G. COMPUTING

Education and the Atari



Advanced spreadsheet
SynCalc™

Database management
SynFile+™



Graphing and statistics
SynTrend™

Now your Atari computer gets down to business.

If you're a serious home manager, a student, or run a small business at home, now you can get sophisticated, integrated software for your ATARI computer with the same features as the more expensive IBM and Apple packages.

SynCalc makes a spreadsheet more manageable.

First, there's SynCalc, the most advanced spreadsheet program ever created for ATARI Home Computers. Not only does SynCalc help you get all your numbers down easily, it also comes with a sorting feature, and the ability to label and name your formulas like "beginning inventory + goods purchased – goods sold = inventory on hand," as well as standard entries. And SynCalc also comes with "expert" commands, to use once you've become more familiar with its procedures. Plus many other features found in the more expensive programs.

SynFile+ keeps information more organized.

SynFile+ can function as your database, your filing system. With SynFile+, you can reorganize and sort parts or whole files instantly. Not only can you enter text, you can calculate and update data as well. And files from both SynCalc and SynFile+ can also be used by the ATARI word processor, AtariWriter,™ for uses such as mail merge.

SynTrend gives you a more graphic way to look at data.

Next, there's SynTrend, which can be the graphing and statistical arm of your operation. SynTrend allows you to visualize your data from SynCalc or SynFile+ with either bar graphs, pie charts, line graphs or scatter plots. To do statistical analysis, you can quickly calculate means and variances, standard deviations, or even linear and multiple regressions. It's pretty easy to understand, eh? And also pretty easy to operate because all three programs come replete with easy-to-understand "pop-up" menus, to take you through their paces step by step. And remember, all three programs can share data, which helps you get the job done even faster.

So get down to business with SynCalc, SynFile+, SynTrend, developed exclusively for ATARI by Synapse. And see for yourself why the cost of taking care of business doesn't have to put you out of it.

DISCOVER WHAT YOU AND ATARI® CAN DO.

CIRCLE #101 ON READER SERVICE CARD

NO. 22 SEPTEMBER 1984

THE MAGAZINE FOR ATARI® COMPUTER OWNERS

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

FEATURES

Spelling.SAM	Ed Rybczyk	13
Balldrop	Bill Boegelein	18
Spanish Study Guide	Larry Nocella	21
Math Attack	Manny Miller	23
Micro-Puzzler	Larry G. Hearin	34
Air Attack	Scott Sheck	39
The New Atari (CES Report)	Arthur Leyenberger	47
Word Scramble	Steven T. Murphy	52
Create-A-Font Datamaker	E.K. Garringer	57
Typing Evaluator	William Abell, Jr.	60
Money Hungry	Donald P. Murphy	63
Sound FX	John Carmody	72
Son of Solid States	Tom Hudson	79
Mathman	Francisco R. Moncada	85
The Reading Program	Ed Rybczyk	91

REVIEWS

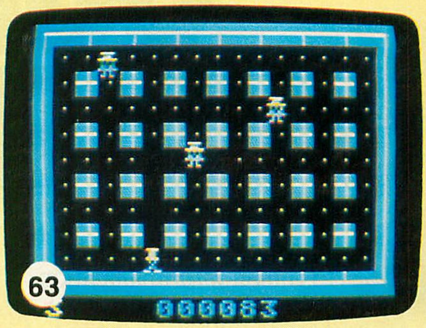
The New Letter Perfect (LJK, Inc.)	Arthur Leyenberger	10
Printer Interfaces	Charles Bachand	31
Gyruss (Parker Bros.)	Tom Hudson	89

COLUMNS

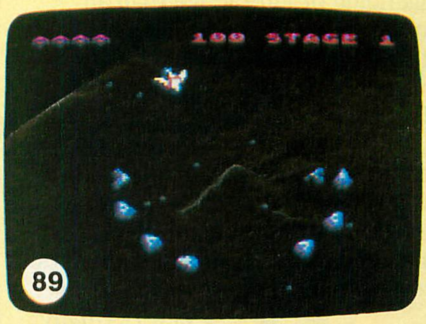
In This Issue	Jon A. Bell	4
Reader Comment		5
Griffin's Lair	Braden Griffin, M.D.	7
D:CHECK/C:CHECK		43
Boot Camp	Tom Hudson	74
Index to Advertisers		96

34

39



63



89

ANALOG COMPUTING (ISSN 0744-9917) is published monthly for $28 per year by ANALOG Magazine Corp., 565 Main Street, Cherry Valley, MA 01611. Second-class postage paid at Worcester, MA and additional mailing offices.

OUR STORIES LACK IMAGINATION.

Because Infocom's interactive fiction is designed to run on *your* imagination.

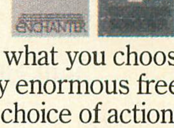
That's precisely why there's nothing more interesting, challenging or interactive than an Infocom disk—but only after you've put it in your computer.

Once it's in, you experience something akin to waking up inside a novel. You find yourself at the center of an exciting plot that continually challenges you with surprising twists,

unique characters (many of whom possess extraordinarily developed personalities) and original, logical, often hilarious puzzles.

Communication is carried on in the same way as it is in a novel—in prose. And interaction is easy—you type in full English sentences.

But there is this key difference between our tales and conventional novels: Infocom's interactive fiction is active, never passive.



The course of events is shaped by what you choose to do. And you enjoy enormous freedom in your choice of actions

—you have hundreds, even thousands of alternatives at every step. In fact, an Infocom

interactive story is roughly the length of a short novel in content, but because you're actively engaged in the plot, your adventure can last for weeks and months.

Find out what it's like to get inside a story. Get one from Infocom. Because with Infocom's interactive fiction, there's room for you on every disk.

INFOCOM™

For your: Apple II, Macintosh, Atari, Commodore 64, CP/M 8", DECmate, DEC Rainbow, DEC RT-11, HP 150 & 110, IBM PC* & PCjr,* KAYPRO II, MS-DOS 2.0,* NEC APC, NEC PC-8000, Osborne, TI Professional, TI 99/4A, Tandy 2000, TRS-80 Color Computer, TRS-80 Models I & III.

*Use the IBM PC version for your Compaq and the MS-DOS 2.0 version for your Wang, Mindset, Data General System 10, GRiD and many others.

CIRCLE #102 ON READER SERVICE CARD

For subscription information and service call toll-free:

1-800-345-8112

in Pennsylvania call 1-800-662-2444

ADVERTISING SALES

Gerald F. Sweeney & Associates
P.O. Box 662
New York, NY 10113
(212) 242-3540

ANALOG Computing
Home Office
Michael DesChenes
National Advertising
(617) 892-9230

Address all advertising materials to:
Michael DesChenes — Advertising Production
ANALOG Computing
565 Main Street, Cherry Valley, MA 01611

ANALOG COMPUTING (ISSN 0744-9917) is published monthly for $28 per year by ANALOG 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 ANALOG COMPUTING, P.O. Box 615, Holmes, PA 19043. No portion of this magazine may be reproduced in any form without written permission of the publisher. 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 ANALOG COMPUTING, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of ANALOG. 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.

IN THIS ISSUE

by Jon A. Bell

It's been three issues since we've printed commentary on what's graced our pages, and, since our readers have been clamoring for information on the changes that have taken place in **ANALOG**, we thought that we should address the queries.

*What happened to **Sally Forth**?*

Sally Forth has taken an extended vacation, resting her keyboard-weary fingers from the rigors of opening mail and punching Forth code into her computer. However, we do intend to resume our Forth coverage very soon, so expect to see Sally—or one of her relatives—return to our pages. Like Mr. Spock, you just can't keep a good Forth programmer down. . .

*Has Tom Hudson's **BASIC Training** been dropped? I noticed that it wasn't in the last issue.*

Never fear. **BASIC Training** will resume next issue. In issue 20, we wanted to fit in as many printer utilities as possible, so Tom decided to work on his third solid states article and hold **BASIC Training** until later. **Son of Solid States** appears on page 79 of this issue, in lieu of **BASIC Training**.

*What happened to **Our Game**?*

Our Game will not be returning to **ANALOG**, due to Joel Gluck's intensive work and school schedule. What we intend to do is use Tom Hudson's **BASIC Training** as the new springboard from which to develop a BASIC game. However, Joel is not permanently leaving our pages. He is working on a number of new programs (some in Action!), which we hope to publish in the not-too-distant future.

And now, center stage. . .

The topic of this issue is education and the Atari . . .and, to illustrate this, we're featuring a wealth of educational games and tutorial programs. In our educational review column, **Griffin's Lair**, Braden Griffin takes a look at **Atari Speed Reading**.

Kicking off this issue's educational programs is Ed Rybczyk's **Spelling.SAM**, a program that teaches spelling fundamentals by actually sounding out the word being spelled. This program is to be used with Don't Ask Software's **Software Automatic Mouth** (SAM) voice synthesizer. Another program by Ed Rybczyk, **The Reading Program**, appears on page 91.

Haven't brushed up on your foreign language study since your high school days? Then take a look at Larry Nocella's **Spanish Study Guide** on page 21. *Si usted no comprende esta frase, debe commencer con esta programma. . . rapidamente.*

Math Attack, by Manny Miller, shows that teaching math facts and having fun needn't be mutually exclusive. This fast-action game is for either one or two players. **Micro-Puzzler**, by Larry G. Hearin, enables you to take a Datasoft **Micro-Painter** screen and turn it into a computerized jigsaw puzzle.

Like to solve your daily newspaper's "jumbled word" puzzles? Then Steven T. Murphy's **Word Scramble** may be for you. Featuring three different levels of difficulty and a vocabulary of three hundred words, it should keep you and your friends amused for hours.

If your typing is in need of some improvement, try William Abell, Jr.'s **Typing Evaluator**, on page 60. It does the job of some commercially-available typing trainer programs for the cost of entering in less than one magazine page of code.

In **Boot Camp** this issue, Tom Hudson covers the remainder of the 6502 operation codes. Next issue, **Boot Camp** will begin exploring the world of useful 6502 programming. Tom also presents the third part in his series of three-dimensional object representation in **Son of Solid States**, page 79.

Should you desire to teach math to the younger set, check out Francisco R. Moncada's **Mathman** on page 85. Like **Math Attack**, it teaches addition, subtraction and multiplication, and the program itself is shorter, to boot (pun intended).

And finally, arcade game fans need not unsheathe their hari-kari swords, for we have Scott Sheck's **Air Attack** for BASIC fans and Donald Murphy's game **Money Hungry** for all you assembly freaks.

Time's a-wasting. Plug in that BASIC cartridge and start typing. □

Bacterion! Update.

Some of our readers have reported difficulty running **Bacterion!** (**ANALOG** issue 20) on their 600 and 800XL computers. Game authors Kyle Peacock and Tom Hudson are working on a fix for this problem, which we will present in a future issue.

READER COMMENT

Sound advice.

Since you only deal with Atari computers, maybe you could help me out with a little problem.

I also subscribe to another electronic magazine, which recently showed consumers how to hook their Commodore VIC-20 up to their stereo system.

Could you, in a future issue, show us how to hook up our stereo systems to an Atari?

Thank you very much.
Bill Fasser
El Paso, TX

If your Atari computer is an 800, 800XL or a 1200XL, then it is very easy to add sound that can be heard through your stereo to the system.

What you need is a 5-pin DIN connector (which may be purchased from any Radio Shack store) that plugs into the side or back of your computer. A shielded audio cable with an RCA-type phono connector on one end should be wired to the DIN connector by soldering the shield to pin 2 and the center conductor to pin 3. Then just plug everything together and enjoy.

Unfortunately, if you have a 400 or 600XL, this cannot be done. It would mean adding extra wiring and components to your computer — and voiding the warranty. While this might not bother some, we feel that giving out such information (which we don't have on hand, anyway!) would lead to the possible destruction of several home computers by those not familiar with electronics.

—CB

Boot cramp.

I have found a solution for all of those Atari owners who have a 600XL or 800XL and are struggling to load boot cassettes.

For XL owners who aren't used to holding down the OPTION key during power-up, BASIC will return at the end of the load. With your disk drive *turned off*, type DOS, and your program should come to life. This simple solution will prevent you from having to re-load the entire program all over again. Just make *sure* that your disk drive is off.

When you type DOS, the computer gives control to the disk operating system. If DOS does not exist in memory, then it gives control to the program that was just loaded in. I have had much success using this procedure; hope others do, too.

Mark Larson
Bakersfield, CA

If, after loading the boot tape, your computer has a black screen instead of the READY prompt, press RESET. The READY prompt will appear, and you can continue with the DOS operation.

—TH

400 Memory expansion.

I am new to computing and I'm interested in increasing the memory of my Atari 400.

I am also a recent subscriber and would like to know if there have been previous articles regarding memory expansion. If there are no such articles, would it be possible to run one in the future regarding the actual installment of the memory boards and use of the computer with these installed? Any information would be greatly appreciated.

Sincerely,
David Raudenbush
Pine Grove, PA

There are several companies that make expansion memory for the 400 computer, in either 32K or 48K sizes. They, however, require that the present 16K memory board in the computer be replaced with the new board. 32K boards required a simple board change, while 48Ks also need a small amount of soldering to be done.

These boards are made by such companies as Mosaic, Axlon, Intec, etc., and their ads can usually be found in our magazine, in either this issue or back issues.

As to an article possibly appearing in a future issue, I know that we are planning a 600XL memory expansion article, but — unfortunately — have no word for one on the model 400.

—CB

I would like to thank Raymond Berube for his fair review of our **MPP MicroRam** 64K memory board for the Atari 600 XL, which appeared in issue 19 of **ANALOG**. The review contained several criticisms of this product to which I would like to respond.

Apparently Mr. Berube had one of the very early pre-production models of the memory board. The circuit board in these early units (there were less than 100 released, mostly as samples to distributors, dealers, etc.) did not have the card edge connector on the back side for continuation of the expansion bus. Further, until the circuit board could be modified for the case, these early units had to be glued together. Neither of these less-than-satisfactory features are to be found on the regular production units.

Regarding price, the suggested retail price of this unit has been reduced to $119.95.

Jon North, Sales Manager
Microbits Peripheral Products
Albany, OR

Promotion or probity?

I feel I must comment on the editorial entitled *Darkness at Noon*, published in issue 19. This is the second time you have published an editorial in a similar vein, the first being *Whither Atari* in issue 9, and I feel that you are doing the Atari community a real disservice.

The function of **ANALOG**, as I see it, is to promote Atari computers. Any other goal would be counter-productive to the magazine's survival. This does not mean that you should not mention faults and/or criticisms, but that the criticism should be presented in a constructive manner. There is nothing in this recent editorial that would give one any reasonable hope that Atari will continue as a force in the microcomputer field.

Atari makes a fine machine, the equal or superior to many, more expensive machines currently on the market. I have owned mine for several years now, and, in spite of many modifications, it has never required service. Most of the people I know who own Ataris have had the same good fortune. Compare that to the 20-30% return rate on Commodores or the 50%+ rate on the Adam. The computing power is comparable to the Apple. It has a vast software base, including seven or eight languages. So why does it seem that so many Atarians downgrade, or are apologetic about their machines? Do we have some sort of a death wish?

This problem is compounded by magazines which seem to delight in finding fault with the very machines they claim to support. (Why is it that I never see editorials like this in Commodore, Apple or IBM magazines?) When I was shopping for a computer, I made a point of buying several machine-specific magazines to determine what the users thought of their machines. If I had read an editorial such as

(Reader Comment continues on page 71)

Griffin's Lair
Educational Programs Review

by Braden E. Griffin, M.D.

For those of you who immediately turn to **Griffin's Lair** as soon as you receive your issue of **ANALOG**, without so much as looking at the cover, I would like to say, "Hi, Mom." The rest of you are already aware of this issue's emphasis on education. The programs found in this month's **ANALOG** will prove to be worthwhile educational tools. Adapting these programs to individual needs provides the opportunity to create a unique educational environment. One does not have to spend megabucks on commercial software to enhance the learning process.

Unfortunately, the word "educational" has the connotation of being tedious, unexciting and, often, involuntary. In a Madison Avenue attempt to combat this attitude, one prominent software company has coined the term "edu-tainment!" William Safire could probably get a whole column out of that one. Everyone *should* appreciate the inherent pleasures of learning. We should also live in a world without war and hunger, but my plans for that are a little beyond the scope of this column. **ANALOG** readers, an erudite and select group (*really!*), realizing the value of knowledge, will find this issue very beneficial.

In an act of extreme mercy (divine intervention?), my soapbox has just been destroyed by a bolt of lightning. I guess it's high time to get on with the show.

ATARI SPEED READING
ATARI, INC., Home Computer Division
P.O. Box 50047
San Jose, CA 95150
16K Cassette $74.95

"Darlin', trace this call—'cause I don't know where I am." The glut of reading material inundating many people today *does* give one the feeling of being lost. Whether keeping pace with the news of the world, maintaining expertise in one's own business, cramming for a final (also, one's business), staying current with the plethora of computer magazines (whose numbers will soon outstrip porno mags) or simply trying to get through the book(s) we received for Christmas, the ability to read faster and comprehend more would be a great asset. In addition, the increase in leisure-time activities has made nearly everyone wish they had more time for reading. Unless we blow up our TV sets, disconnect our computers and move to Iceland, there is *never* going to be enough time. Since most of us aren't willing to sacrifice our pleasures, the only options are to stop reading (perish the thought) or to read faster.

To rescue us from this stagnation of priorities, methods to teach rapid reading skills have been developed. **Atari Speed Reading** incorporates many of these methods in a do-it-yourself program designed to improve both speed and comprehension.

The program consists of eight separate units on four cassettes and a superb reading workbook. A fifth cassette is included as an adjunct to outside reading and the maintenance of the newly-acquired skills. Each unit has a similar format and consists of six elements. Each session begins with a WARM-UP EXERCISE, which gets one moving fast and concentrating. A reading window highlights a word on the screen, then it moves over a separate series of words. When the appropriate matching word is highlighted, one presses the joystick fire button. The units differ slightly on the matching word required. The initial exercises are just seeking an exact word match. Others match synonyms, antonyms or phrases. The rate at which the entries are highlighted by the window can be varied, offering a greater challenge as progress is made.

One of the limiting factors in reading fast is that many people tend to subvocalize, or talk silently to themselves, as they read. Using the exercise above, one can practice seeing the words without pronouncing them. This is an important concept, emphasizing that one's reading speed, even for employees of overnight delivery services, relies on understanding, not on being able to speak fast.

The PHRASE-READING EXERCISE comes next. In this drill, the reading window will highlight phrases as it proceeds through a short portion of prepared text. A metronome-like tone accompanies the window movement. The rate can, again, be increased as skills improve. The purpose of this exercise is to force the reader to take in more information with each eye stop, or fixation. It promotes fast, efficient eye movements between these stops in a left to right procession, referred to as saccadic movements. By pushing one forward, it discourages regressions, or movements from right to left within the same line.

The next element is the PACED & TIMED READINGS. For these exercises, selected articles in the workbook are read while the computer supplies the pacing tone and timer. Pressing the fire button initiates the timer and then stops it when the reading is finished. The articles vary from 1500 to 2500 words, and most are quite interesting in themselves. Each selection is followed by a quiz, to help monitor how well one is retaining the material. A score of 70% is felt to be a minimum standard of achievement. The reading in this part of the program is fairly easy, enabling the user to concentrate on speed while still maintaining an acceptable degree of understanding.

Each segment focuses on the NEW TECHNIQUES used to improve reading efficiency. A variety of exercises in the workbook are employed to accomplish this. Scanning techniques are developed using an excerpt from a telephone directory. Several of the units stress the importance of skimming methods and include some excellent drills to refine them. One unit teaches how to incorporate the skimming techniques with note taking and outlining skills. This latter section is extremely well done and will unquestionably enhance study habits. The final unit deals with the use of the different techniques as they relate to the purpose of the reading.

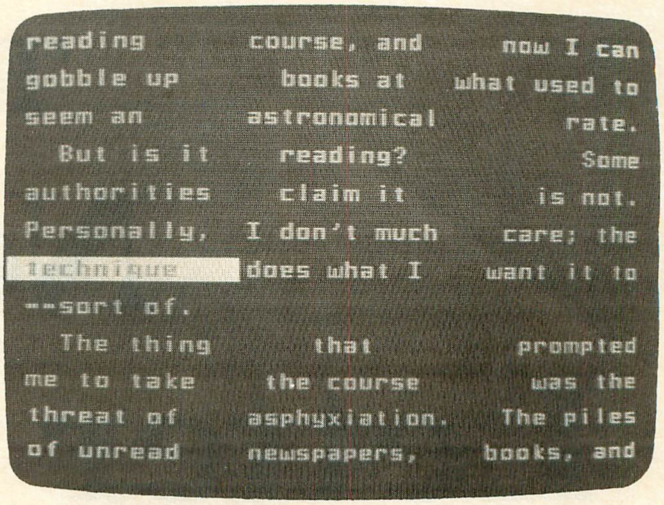
The fifth element is the FLEXIBLE READING portion. Using the computer-controlled timer without the paced tone, the selections are read as rapidly as possible, while maintaining a pace that does not interfere with one's understanding of the material. There is a wide range of reading difficulty in the excerpts used, serving to illustrate the flexible natures of these skills. Specialized topics with an unfamiliar vocabulary require a different degree of intensity and pace than an article about a subject with which the reader is quite conversant. A selection from *Gray's Anatomy* is used to demonstrate this. It is obvious that a great deal of thought went into the selection of the reading matter for each unit. Apart from specifically underscoring the respective techniques stressed in each unit, every one of the selected texts stimulates interest, helping maintain a high level of concentration. Again, a short quiz follows this section to ensure that an adequate measure of understanding is continued.

The final element is the READING PROGRESS GRAPH. At this point, the reading rates and comprehension scores for the paced, timed and flexible reading sections are entered, and a Reading Efficiency Index is calculated and plotted on a graph showing one's progress.

Before beginning the programmed course, the student is given a reading pretest. Using the computer-controlled timer, one reads a selection at a normal rate and answers the questions following the section. Current reading rate is thus determined and will serve as a baseline for the succeeding exercises. Each unit should take about two hours to complete. If two units are completed per week, the course can be finished in one month, a pace considered to promote the best progress.

RIF.

Does it really work? Definitely. The key is motivation. The success of any speed reading course depends heavily on the enthusiasm of the participant. I have always considered myself a slow reader. My excuse has been that, with so much to read from medical school on, I might only have the time to read something once, so I had better read slowly and remember it the first time. These habits influenced my reading in all areas. It took me days to read a novel most people finished in one night. Granted, I remembered every detail, but I'm not often quizzed on Asimov's *Foundation Trilogy*.



Atari Speed Reading.

Knowing that beginning a new book meant many hours of reading, I frequently opted not to even start. No more. Having completed most of the program, I now realize that I was never a slow reader, only a lazy one. My biggest surprise came during the very first unit. My reading speed nearly doubled after just a few exercises. Were the warm-up drills responsible? I don't think so. What happened was that I concentrated on reading fast and retaining the material. I mean, I really *worked* at it. How often have you sat down to read something important and found yourself rereading sections and drifting away? Most of us put very little effort into reading, and that makes us lazy readers. This program points out bad habits and helps eliminate them. I am not only reading faster, but with greater confidence.

Apparently, a number of studies have shown that the majority of people who develop speed reading skills revert back to their old habits after several months. These skills must be practiced. One may be able to ride a bike, but riding fast requires one to keep working at it. Mental skills are no different; they must be kept in shape. The final cassette helps maintain these skills by providing a pacing and timing program to use with reading material not found in the workbook.

Atari Speed Reading uncovers one's reading potential and develops it to the fullest. A good investment for adults, it is a *great* one for our children. And yet, I wonder. . . now that I can type *and* read incredibly fast, why does it still take me forever to write this column? □

Next issue, Dr. Griffin will take a look at Infocom's new "educational" adventure-type game, ***Seastalker.***

The New Letter Perfect

LETTER PERFECT 6.0
LJK, INC.
7852 Big Bend Blvd.
St. Louis, MO 63119
(314) 962-1855
48K Disk $99.95

by Arthur Leyenberger

Having been a user of LJK's **Letter Perfect** for almost two years, I've found it to be the premier word processor for the Atari. Now LJK has released a new version that adds significant capabilities to **Letter Perfect**'s already long list of features. The new program comes with an entirely rewritten manual and a spelling checking program (**Spell Perfect**). Rather than review the entire package, I'll describe the new features and how they differ from the older version.

A manageable manual.

The most obvious improvement is in the documentation. The previous manual was unreadable and confusing. With the new manual, the first-time user has it much easier. The 86-page typeset manual is now in a 5×8 inch spiral bound format containing many user-friendly features.

How can a *manual* be user-friendly, you ask? Well, it now includes a good index, a glossary of word processing terms and a well-written tutorial on how to use the program. There are also several printer worksheets for Centronics, Epson, NEC, Prowriter, Qume and Okidata printers. Information on various printer characteristics and default values used by the program are also included.

The printer material contained in the new manual corresponds to the many printer configuration files on the disk. Any of these can be edited and permanently saved, should you decide to reconfigure your system. Also, proportional printing is supported for the C.Itoh Prowriter, NEC8023 and Centronics-type (like the Epson FX-80) printers. Blank worksheets are provided for users whose printer is not one of the above.

Program clarity.

Another major change in **Letter Perfect** is the method of program delivery. Previously, two two-sided disks were supplied—one with a 40-column program, the other with an 80-column program (for the bit-3 80-column board). The back of each disk contained the printer driver editor program. Now, one disk is supplied, containing three variations. When the program is booted, you are asked to select a 40-column, bit-3 80-column or Austin-Franklin 80-column mode. (LJK has told me that they will support the new Atari 80-column card in the 1090 Expansion Box, when it becomes available.) Once this choice is made, the user selects the applicable printer and can then edit the parameters of the printer file. A separate printer driver editor is no longer necessary.

When the configuration process is complete, the information is saved to the program disk, so that it will be automatically used the next time the disk is booted. Should you decide to change any of this information, holding the ESC key down when booting allows you to reconfigure the program. Since **LP** is not copy-protected, you may have several program disks, each with a different system configuration.

As mentioned before, **Spell Perfect** is included in the main program. It counts the number of words in your document and allows you to check for misspellings. Unlike the separate **Spell Perfect** package, this version does not allow you to add to the dictionary disk or create your own dictionary. There just wasn't enough room on the program disk. It's still a useful feature and adds to the product's value.

By the way, if you're an owner of the earlier version of **Letter Perfect**, you may obtain the new program simply by sending a copy of your registration form and $30 to LJK. Since the new **Letter Perfect** is larger than the original by about 6K, some very large files created by the first will not completely fit into the new editor. LJK is aware of this, and their policy of allowing you to keep the older version when you upgrade is quite considerate.

Menus, modes and more.

Among the many improvements, menu selections can now be made by either positioning the cursor over the desired choice and pressing RETURN (as before) or by pressing the first letter of the menu option. Once a file has been loaded or saved, the program remembers its filename—and that becomes the default name when saving.

Double-density mode may be selected for your text disks, allowing you to save twice the amount of information on one disk. Trak, Rana, Indus, Percom and generic disk drives (with an ATR8000) are all supported. Although there is no direct single-to-double density conversion function, you can perform this operation if you have two drives. To do so, load a single-density file into **LP** and choose the configuration option on the main menu. Make your second drive the "file" drive and change the density to double. Then, all you have to do is save the file—it will be saved as a double-density file. Repeat this process for each single-density file, one at a time.

Other useful new features include: easier and more understandable delete functions for buffer, paragraph, tabs, entire file, before and after cursor and up to marker; a blinking cursor indicating that you are in insert mode, or overwrite mode is being maintained, even if you leave and re-enter the =I conditional page breaks; special characters like tilde and curly brackets; non-overwriting tab; immediate cursor movement to the end of the file without scrolling; and the ability to fix the width of your edit window, to be able to see how your text will look on paper.

Almost perfect.

Is there anything I don't like about the new version of **Letter Perfect**? Yes, a couple of things. I use an Epson FX-80 printer and have gotten used to the way it works with the original. The new version uses half-spacing for all Epson printers, so you will need to change your printer spacing, margins and header/footers. But, once you get it right, you can forget about it.

Another minor irritation is the character delete function. In the earlier **LP**, pressing CNTRL-DEL would continuously and rapidly delete characters from the right. Now, that same command results in a slower, jerky movement. Of course, you can always use the new word delete command to accomplish almost the same thing.

Finally, some of the commands have been changed. As in any major software rewrite, the addition of new commands forces new meanings on some of the old ones. For example, the key sequence CNTRL-, used to mean global replace. This same command now produces a left-curly bracket. CNTRL-CAPS has become the command for a global replace.

Fortunately, none of the reassigned meanings are destructive—you cannot inadvertently wipe out any text or files. In order to minimize confusion, I created a table of the old and new functions and commands (see Table 1 below). And, most importantly, the new **Letter Perfect** does not require the translator disk on Atari's XLs.

Aside from a few small gripes, version 6.0 of **Letter Perfect** is a major revision of the program that maintains its already impressive features while adding quite a few more. It remains the best currently available word processor for the Atari. □

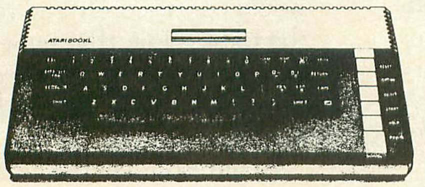
Table 1.

Letter Perfect Comparison Chart

Function	Old LP	New LP
Cursor movement		
Top of text	CNTRL-CLEAR	CNTRL-CLEAR
One word right		CNTRL-Y
One word left		CNTRL-W
Previous page	CNTRL-;	CNTRL-Q
Next page	CNTRL-Q	CNTRL-O
Next paragraph		CNTRL-6
Top of page	CNTRL-T	CNTRL-CLEAR
Bottom of page		CNTRL-0
Jump to marker		CNTRL-X
Deletion		
Word right		CNTRL-N
Word left		CNTRL-L
Current line	SHIFT-DEL	CNTRL-5
All after cursor	CNTRL-Y	CNTRL-KA
All before cursor	CNTRL-W	CNTRL-KB
Up to marker	CNTRL-N	CNTRL-KM
All text	CNTRL-X	CNTRL-KN
End of Paragraph		CNTRL-KP
Copy Buffer	CNTRL-K	CNTRL-KC
All tabs		CNTRL-KT
Text moves		
Copy to buffer	CNTRL-/	CNTRL-7
Copy from buffer	CNTRL-CAPS	CNTRL-9
Lift from buffer	CNTRL-L	CNTRL-8
Continuous scroll	CNTRL-O	CNTRL-4
Fix window width		CNTRL-3
Conditional pg. break		CNTRL-V%
Halt printer		CNTRL-V!
Global replace	CNTRL-,	CNTRL-CAPS
Show tabs		CNTRL-T

Always the Lowest Prices on the Finest Quality at. . . COMPUTER CREATIONS

ATARI®



ATARI PLATO SYSTEMS

The Electronic Highway To The Universe
Over 6000 Educational Programs

SYSTEM I	SYSTEM II
Atari 1030 Modem	Atari 850 Interface Module
Atari Plato Cartridge	Atari Plato Cartridge
	Anchor Mark XII Modem
$159	**$549**

ATARI HOME COMPUTER

ATARI 600™ Home Computer (16K RAM) .... CALL FOR
ATARI 800XL™ Home Computer (64K RAM) .... UPDATED PRICES

DISK DRIVES

RANA 1000
TRAK ATD 2 (Single/Double Density/Parallel Int./4k buffer:
TRAK ATD4 (Dual Drive)
TRAK TURBO DOS
INDUS GT (Free Software)
ASTRA 1620
ATR - 8000
HAPPY ENHANCEMENT for Atari 810 and 1050 Disk Drives

CALL FOR UPDATED PRICES

ATARI ADDITIONAL EQUIPMENT

ATARI 1010™ Program Recorder
ATARI 1020™ 40-Column Color Printer/Plotter
ATARI 1025™ 80-Column Printer
ATARI 1027™ Letter Quality Printer
ATARI 1030™ Direct Connect Modem
ATARI 1050™ Disk Drive
CX77 ATARI Touch Tablet™*

CALL FOR UPDATED PRICES

*Screen Dump Program **FREE!!** with purchase of CX77

HARDWARE COVERS

ATARI 1010	5.95
ATARI 1200	5.95
ATARI 1025	5.95
ATARI 600XL	5.95
ATARI 1050	5.95
ATARI 800XL	5.95
ATARI 1027	5.95
GEMINI 10X	5.95
GEMINI 15X	7.49
POWERTYPE	5.95
DELTA-15	7.49
RANA 1000	5.95
INDUS	5.95
TRAK	5.95

DISKETTE/CARTRIDGE/ CASSETTE FILES

Flip 'N' File	3.95
Flip 'N' File 15	6.95
Data Defender	16.99
Flip'N'File/The Original	19.95
Cassette'N'Game File	14.95
Disk Bank	3.99
Library Case	1.99
Color Coder	13.99

RAM (MEMORY) BOARDS

Microbits 64K (600XL) Expansion 129.00

AUTHORIZED SERVICE CENTER
ATARI ● TRAK
STAR MICRONICS (GEMINI)
Call for prices and services.

GENERIC DISKS AT FANTASTIC PRICES!!

Generic 100% Defect-Free/Guaranteed Mini-Floppy Disks

Diskettes (1 Box Minimum) - 10 Per Box

	SS/SD	SS/DD	DS/DD
1 or 2 Boxes	15.49	15.99	19.99
3 - 9 Boxes	14.49	14.99	18.99
10+ Boxes	13.99	14.49	17.99

Bulk Diskettes with Sleevės - Price per Disk

	SS/SD	SS/DD	DS/DD
10 - 29	1.45	1.49	1.89
30 - 99	1.35	1.39	1.79
100+	1.29	1.35	1.69

PRINTERS

GEMINI 10X (80 Column)	269.00
GEMINI 15X (136 Column)	399.00
DELTA 10 (80 Column)	419.00
RADIX 10 (80 Column)	639.00
RADIX 15 (136 Column)	749.00
POWERTYPE Daisywheel	379.00
EPSON RX-80 (80 Column)	339.00
EPSON RX-80 FT (80 Column)	469.00
EPSON FX-80 (80 Column)	555.00
EPSON FX-100 (136 Column)	749.00
SILVER-REED EXP 500 Daisywheel (80 Column)	399.00
OKIDATA 92P	449.00
EPSON RX100 (136 Column)	439.00

PRINTER INTERFACE CABLES

APE FACE	65.00
MPP-1150 Parallel Printer Interface	79.00
PARALLEL Printer Cable	30.00

PRINTER RIBBONS

GEMINI Printers (Black/Blue/Red/Purple)	3.00
EPSON Printers	6.95

MONITORS

Gorilla™ Hi Res 12" Non-Glare Green Screen	89.00
Gorilla™ Hi Res 12" Non-Glare Amber Screen	99.00
Sakata SC 100 Color Screen	239.00
Sakata Green Screen	99.00
Sakata Amber Screen	109.00
Monitor Cable	10.00

MODEMS

MPP-1000C Modem	129.00
Signalman Mark XII Modem	269.00

To order call TOLL FREE
1-800-824-7506
ORDER LINE ONLY

No extra charge for MasterCard or Visa

COMPUTER CREATIONS, Inc.

P.O. Bo 292467 - Dayton, Ohio 45429

For information call: (513) 294-2002 (Or to order in Ohio)

Order Lines Open - 8:30 a.m. to 8:00 p.m. Mon.-Fri.; 10:00 a.m. to 4:00 p.m. Sat. (Eastern Standard Time). FREE Catalog with each order. Minimum $10 per order. Prepaid Money Order - Free Shipping. MasterCard - Visa - C.O.D. (Add $2.50). All orders add $3.50 shipping and handling in Continental United States. Actual freight will be charged outside U.S. to include Canada, Alaska, Hawaii, Puerto Rico and A.P.O.'s. Price is subject to change without notice. Call for update prices and availability of product. Ohio residents add 6% sales tax.

CIRCLE #104 ON READER SERVICE CARD

Spelling.SAM

A spelling program using Don't Ask's S.A.M.

32K Disk

by Ed Rybczyk

When I first received **SAM** (Software Automatic Mouth by Don't Ask Software — $59.95) last Christmas, the first priority was to add speech into all the great "shoot 'em up, save the world" game program listings typed in over the past year. Hearing the score updated was great; and the verbal reassurance that the world had been saved, at least for that round, was comforting. It wasn't until later — and after complete satiation, for I am truly a games *junkie* — that another application for **SAM** smacked me in the face.

My older son came home from school with a list of words to be learned as part of his homework. As any computer parent knows, it's much easier to get the children to play with the computer than to do their homework. We had previously written programs for math facts, a joystick game to differentiate nouns from verbs, and experimented with cassette recorder spelling routines (POKE 54018,52 turns the prerecorded cassette on; POKE 54018,60 turns it off).

What a perfect application for **SAM**! How else can a spelling program work — without hearing the word to be spelled? Other elements of a good education program (immediate reinforcement, use of audio and visual reinforcers and correction of mistakes) were built in, and the result is **Spelling.SAM**. The program requires less than 8K and is loaded after **SAM** is booted.

Press the START key to hear the word. This can be done as many times as desired, and **SAM** will repeat the word. When you're ready to spell the word, press SELECT.

Spelling.SAM can be tinkered with, and any improvements are welcomed. Care must be taken when adding words in Lines 1000-1998. Incorrect phonetics will cause the word not to be spoken during program operation. Instead, the console will beep twice. Use the **SAM** documentation glossary for help. Spelling words are limited to ten spaces.

The program can be changed to use RECITER. This will alleviate phonetic problems but could cause poor enunciation. Change the following to use RECITER:

Line 20 — SAM=8199 and each of the phonetics to text in Lines 1000-1310. Also, change the phonetics to text in Lines 22, 331, 501, 701, 801, 851 and 1005.

The boot procedure is changed to: (1) Boot SAM; (2) USE a disk containing RECITER and MEM.SAV; (3) USE DOS option L to load RECITER; and (4) LOAD SPELLING.SAM.

(Program documentation and listing continued on next page.)

Program routines.

Line	Function
10 - 45	Opening graphics
55 - 97	Opening music
99 - 170	Speech
200 - 220	Evaluation graphics
300 - 410	Keyboard input
420	Evaluation
430	Random selection of correct reinforcements
440 - 499	Correct reinforcement #1
500 - 599	Correct reinforcement #2
600	Random selection of correcting mistakes
610 - 699	Correcting mistake #1
700 - 795	Correcting mistake #2
800 - 845	Scoreboard
850 - 852	Closing graphics
860 - 995	Closing music
1000 - 1999	Spelling words
10000 - 10020	Out of data graphics

If you don't own a **SAM** disk, what are you waiting for? Add the world of speech to your programs. It really *is* a lot of fun. □

```
4 REM ********************
5 REM *   SPELLING.SAM   *
6 REM *        BY        *
7 REM *    ED RYBCZYK    *
8 REM * ANALOG COMPUTING *
9 REM ********************
10 GRAPHICS 1+16:SETCOLOR 4,10,0
20 SAM=8192:DIM SAM$(255),D$(10),E$(10
)
22 SAM$="/HEH4LOW.  MAY4 NEY4M IHZ SAE
4M":A=USR(SAM)
25 POSITION 0,1:? #6;"##############
######"
30 POSITION 5,5:? #6;"WELCOME TO":POSI
TION 4,8:? #6;"SPELLING SAM"
35 POSITION 8,12:? #6;"□□□□"
40 POSITION 3,15:? #6;"WE'LL HAVE FUN"
:POSITION 3,18:? #6;"SPELLING WORDS"
45 POSITION 0,22:? #6;"##############
######"
55 READ L:IF L=-1 THEN 99
60 READ JA:JA=INT(250/JA)
65 SOUND 0,L,10,8
70 IF L=0 THEN 80
75 SOUND 1,L+1,10,8
80 FOR P=1 TO JA:NEXT P
85 SOUND 0,0,0,0
90 SOUND 1,0,0,0
92 FOR P=1 TO 5:NEXT P
94 GOTO 55
95 DATA 121,6,109,6,97,6,109,1,0,8,121
,8,109,8,97,8,109,1
96 DATA 0,8,121,4,96,6,91,8,96,4,91,2,
121,2,96,2,108,1
97 DATA 0,4,109,4,92,2,109,8,121,2,110
,6,109,2,97,4,109,1,121,1,-1
99 RESTORE 1000
100 READ SAM$,E$:D$=""
110 IF SAM$="000" THEN 10000
125 GRAPHICS 1+16:SETCOLOR 4,10,0:POSI
TION 2,2:? #6;"LISTEN CAREFULLY":POSIT
ION 3,4:? #6;"FOR THE WORD I"
130 POSITION 3,6:? #6;"WOULD LIKE YOU"
:POSITION 6,8:? #6;"TO SPELL.":POSITIO
N 0,11:? #6;"*□*□*□*□*□*□*□*□*□*□"
135 POSITION 1,14:? #6;"PRESS start TO
 HEAR THE WORD.":POSITION 1,19:? #6;"P
RESS select TO     CONTINUE."
140 POKE 53279,8
150 IF PEEK(53279)=6 THEN A=USR(SAM)
160 IF PEEK(53279)=5 THEN 200
170 GOTO 140
200 GRAPHICS 1+16:SETCOLOR 4,10,0:POSI
TION 4,2:? #6;"HOW IS THAT":POSITION 3
,4:? #6;"WORD SPELLED?"
210 POSITION 0,8:? #6;"***************
*****":POSITION 0,12:? #6;"***********
*********"
220 POSITION 0,17:? #6;"TYPE IN YOUR A
NSWER":POSITION 0,20:? #6;" THEN PRESS
 RETURN."
300 LENGTH=10:LINE=10:COL=5
310 IF KB=0 THEN OPEN #1,4,0,"K:":KB=1
312 POSITION COL,LINE
315 GET #1,X
320 IF X>=32 AND X<=95 AND LEN(D$)<LEN
GTH THEN D$(LEN(D$)+1)=CHR$(X):? #6;CH
R$(X);:GOTO 315
330 IF X<>126 THEN 400
340 IF LEN(D$)<1 THEN 400
350 IF LEN(D$)=1 THEN D$=""
360 IF LEN(D$)>1 THEN D$=D$(1,LEN(D$)-
1)
370 POSITION COL+LEN(D$),LINE
380 ? #6;" "
390 POSITION COL+LEN(D$),LINE
400 IF X=155 THEN 420
410 GOTO 315
420 IF D$<>E$ THEN 600
430 R=INT(2*RND(1))+1:ON R GOTO 440,50
0
440 GRAPHICS 1+16:SETCOLOR 4,6,0
441 SAM$="YUW3AAR RAY2IH7T.":A=USR(SAM
)
442 POSITION 3,3:? #6;"THAT'S GREAT!!"
:POSITION 2,6:? #6;"YOU GOT IT RIGHT"
445 POSITION 0,9:? #6;"□*□*□*□*□*□*□
*□*□*"
450 POSITION 3,12:? #6;"YOU SPELLED TH
E":POSITION 3,15:? #6;"WORD ";E$:POSIT
ION 5,18:? #6;"CORRECTLY."
460 ED=ED+1
470 FOR J=1 TO 3
475 FOR K=-160 TO 160 STEP 10
480 SOUND 0,ABS(K)+95,10,8
485 FOR L=1 TO 2:NEXT L:SETCOLOR 4,J+3
,0
490 NEXT K:NEXT J
495 SOUND 0,0,0,0
496 FOR P=1 TO 600:NEXT P
499 GOTO 800
500 GRAPHICS 1+16:SETCOLOR 4,6,0
501 SAM$="VEH4RIY GUH4D":A=USR(SAM)
502 POSITION 3,3:? #6;"YOU'RE RIGHT!!"
:POSITION 2,6:? #6;"I'M PROUD OF YOU"
505 POSITION 0,9:? #6;"□*□*□*□*□*□*□
*□*□*"
510 POSITION 3,12:? #6;"YOU SPELLED TH
E":POSITION 3,15:? #6;"WORD ";E$:POSIT
ION 5,18:? #6;"CORRECTLY."
530 JAN=JAN+1
550 FOR J=0 TO 48 STEP 4
560 SOUND 0,J,8,J/5:SETCOLOR 4,J,0
570 NEXT J
575 FOR J=48 TO 0 STEP -4
580 SOUND 0,J,8,J/6+3:SETCOLOR 4,J,0
590 FOR K=1 TO (70+J*3)/2:NEXT K
593 NEXT J
595 SOUND 0,0,0,0:SETCOLOR 4,8,0
596 FOR P=1 TO 600:NEXT P
599 GOTO 800
600 R=INT(2*RND(1))+1:ON R GOTO 610,70
0
610 GRAPHICS 1+16:SETCOLOR 4,3,0
611 SAM$="TRAY4 /HAA4RDER.":A=USR(SAM)
612 POSITION 7,3:? #6;"SORRY,":POSITIO
N 2,6:? #6;"THAT'S NOT IT !!"
615 POSITION 0,9:? #6;"□*□*□*□*□*□*□
*□*□*"
620 POSITION 3,12:? #6;"YOU SPELLED TH
E":POSITION 3,15:? #6;"WORD ";E$:POSIT
ION 5,18:? #6;"LIKE THIS:"
```

```
630 POSITION 5,21:? #6;D$:KURT=KURT+1
650 FOR J=10 TO 200 STEP 4
660 SOUND 0,J,10,J/25
670 FOR K=1 TO J/10:NEXT K
675 NEXT J
680 SOUND 0,20,0,14
690 SOUND 1,255,10,15
693 FOR K=1 TO 100:NEXT K
695 SOUND 0,0,0,0:SOUND 1,0,0,0
696 FOR P=1 TO 500:NEXT P
699 GOTO 800
700 GRAPHICS 1+16:SETCOLOR 4,0,0
701 SAM$="DHAE4TZ NAAT IH4T.":A=USR(SA
M)
702 POSITION 7,3:? #6;"WRONG,":POSITIO
N 2,6:? #6;"THAT'S INCORRECT"
705 POSITION 0,9:? #6;"▣*▣*▣*▣*▣*▣*▣
*▣*▣*"
710 POSITION 3,12:? #6;"YOU SPELLED TH
E":POSITION 3,15:? #6;"WORD ";E$:POSIT
ION 5,18:? #6;"LIKE THIS:"
720 POSITION 5,21:? #6;D$:ADAM=ADAM+1
730 FOR J=1 TO 8
740 SOUND 0,42,2,10
750 FOR K=1 TO 20:NEXT K
760 SOUND 0,0,0,0
770 FOR K=1 TO 20
780 NEXT J
790 FOR K=1 TO 1200:NEXT K
795 SOUND 0,0,0,0
800 GRAPHICS 1+16:SETCOLOR 4,6,0
801 SAM$="/HIY4RS YOH4R SKOH4R":A=USR(
SAM)
802 POSITION 2,4:? #6;"SO FAR YOU HAVE
:"
810 POSITION 5,8:? #6;ED+JAN;" CORRECT
"
812 POSITION 5,12:? #6;KURT+ADAM;" WRO
NG"
815 IF ED+JAN>KURT+ADAM THEN POSITION
4,16:? #6;"PRETTY GOOD!"
820 IF ED+JAN<=KURT+ADAM THEN POSITION
 5,16:? #6;"TRY HARDER"
822 POSITION 1,20:? #6;"ANOTHER WORD?
(Y/N)"
825 FOR J=1 TO 8
826 SOUND 0,121,10,8:SOUND 1,128,10,8:
SOUND 2,8,10,2:FOR X=1 TO 5:NEXT X
827 SOUND 0,128,10,8:SOUND 1,138,10,8:
SOUND 2,18,10,4
828 FOR X=1 TO 5:NEXT X
829 NEXT J
830 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND
2,0,0,0
831 GET #1,Z
840 IF Z=89 THEN 100
845 IF Z<>78 THEN 831
850 GRAPHICS 1+16:SETCOLOR 4,10,0
851 SAM$="THAE4NXK YUW4.":A=USR(SAM)
852 POSITION 5,10:? #6;"BYE BYE !!"
860 RESTORE 980
870 READ DURATION:IF DURATION=-1 THEN
END
880 DURATION=INT(DURATION*15)
890 READ PITCH:IF PITCH=0 THEN 910
900 PITCH=PITCH*3
910 SOUND 0,PITCH,10,8
920 SOUND 1,PITCH+1,10,8
930 FOR W=1 TO DURATION:NEXT W
940 SOUND 0,0,0,0
950 SOUND 1,0,0,0
960 FOR W=1 TO 3:NEXT W
970 GOTO 870
980 DATA 2,47,2,60,1,81,3,60,2,47,2,60
,1,53,3,53,2,47,2,60,1,45,2,45,1,45
990 DATA 1,47,1,47,1,53,1,53,3,60,1,0,
1.5,47,.5,53,1,60,1,53,1,47,1,47,2,47,
1,53,1,53,2,53
995 DATA 1,47,1,40,2,40,1.5,47,.5,53,1
,60,1,53,1,47,1,47,2,47,1,53,1,53,1,47
,1,53,3,60,1,0,-1
1000 DATA KUMPYUW3TER,COMPUTER
1010 DATA SER4CH,SEARCH
1020 DATA AHBAW4T,ABOUT
1030 DATA AE4FTER,AFTER
1040 DATA TRAY4AENXGUL,TRIANGLE
1050 DATA SER4KUL,CIRCLE
1060 DATA /HAW4S,HOUSE
1070 DATA IHNVEH4NT,INVENT
1080 DATA TEY4BUL,TABLE
1090 DATA BAA4TUM,BOTTOM
1100 DATA /HYUW4MUN,HUMAN
1110 DATA MAXSHIY4N,MACHINE
1120 DATA WAO4TER,WATER
1130 DATA PEH4NULTIY4,PENALTY
1140 DATA KWEH4SCHUN,QUESTION
1150 DATA BIH4ZIY,BUSY
1160 DATA TRAE4FIXK,TRAFFIC
1170 DATA SAY4IHNS,SCIENCE
1180 DATA BRAH4DHER,BROTHER
1190 DATA SOH4LJER,SOLDIER
1200 DATA JAH4MP,JUMP
1210 DATA TEH4RAXBUL,TERRIBLE
1220 DATA VEYKEY4SHUN,VACATION
1230 DATA AE4NSER,ANSWER
1240 DATA MAA4RKIXT,MARKET
1250 DATA KAE4NDIY,CANDY
1260 DATA CHIH4LDRIXN,CHILDREN
1270 DATA SIH4NXGUL,SINGLE
1280 DATA AE4PUL,APPLE
1290 DATA CHAO4KLIXT,CHOCOLATE
1300 DATA EH4ROHR,ERROR
1310 DATA AE4DREHS,ADDRESS
1999 DATA 000,000
10000 GRAPHICS 1+16:SETCOLOR 4,14,0
10005 SAM$="SAA4RIY.":A=USR(SAM)
10010 POSITION 7,5:? #6;"SORRY,":POSIT
ION 4,10:? #6;"I DON'T KNOW":POSITION
3,15:? #6;"ANYMORE WORDS."
10015 FOR X=1 TO 750:NEXT X
10020 GOTO 850
```

•

CHECKSUM DATA.

(see page 43)

```
4 DATA 560,216,352,946,509,570,136,464
,945,572,656,464,239,350,409,7388
60 DATA 473,177,260,627,45,296,285,209
,643,8,728,833,139,882,626,6231
125 DATA 396,395,422,285,690,531,716,8
22,777,71,690,489,75,575,183,7117
330 DATA 911,155,202,841,933,402,939,7
52,704,850,848,128,323,736,999,9723
450 DATA 983,438,309,549,197,434,479,1
07,564,739,118,732,533,989,973,8144
530 DATA 41,144,637,759,337,674,948,76
8,31,566,741,843,120,982,436,8027
615 DATA 994,928,79,81,672,363,766,574
,639,540,9,566,743,116,431,7501
702 DATA 712,993,927,874,313,570,145,9
9,148,766,329,113,124,764,291,7168
810 DATA 145,210,983,197,471,317,788,2
08,84,759,417,589,528,829,333,6858
851 DATA 935,186,241,962,22,759,392,14
5,160,243,97,103,82,755,720,5802
990 DATA 224,469,407,46,128,117,601,39
7,192,639,175,393,664,850,210,5512
1130 DATA 115,197,300,771,584,761,598,
888,93,357,403,565,291,248,580,6751
1280 DATA 196,328,237,534,350,428,202,
922,435,744,4376
```

•

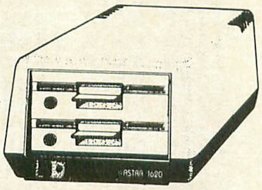
FOR ALL ATARI COMPUTERS

BEST SELLERS FROM THE PROGRAMMERS WORKSHOP

TWO DRIVES FOR THE PRICE OF ONE

$469.00

THE ASTRA 1620



MORE DISK DRIVE FOR YOUR MONEY ... In fact, with the ASTRA 1620, you get two superb Disk Drives for the price of one. The ASTRA 1620 is Single or Double Density (software selectable) and completely compatible with ATARI DOS or OSA+DOS. When used as Double Density, the ASTRA 1620 has the same capacity as Four ATARI 810® Disk Drives.

★ *Satisfaction Guaranteed* ★

INCLUDED: **at no extra charge**
One Copy of Home Writer and One Copy of Smart DOS.

THE HOME WRITER $39.00

The HOME WRITER is an easy to use word processor which includes a carefully selected group of functions that are at your disposal immediately. The functions are as follows: SAVE, LOAD, REVIEW, PRINT-OUT, or EDIT. All the popular editting features available on the ATARI Home Computer in direct programming mode are also available with HOME WRITER. You may also search for phrases or strings, load off of either disk or cassette, and number pages automatically. Unlike other small word processing programs, HOME WRITER does not wrap-around when at the end of a line. Right and left margin justification is available for any type parallel printer. 48K.

FILING SYSTEM $39.00

FILING SYSTEM allows the user to configure any type of data file imaginable. Examples are recipe cards, mail lists, reminders for birthdays, check-ups, etc..., complete inventories (home and business), personnel files, customer call-ups, price list, and much, much more. You may retrieve data using any field or combination of fields. Files also may be saved, sorted, and printed in a preset format that you configure. Uses either a single or a double density disk drive. Three preconfigured files are included – a Mailing List, a Price List, and a Household Inventory. 24K minimum. Now in double density.

THE PROGRAMMERS WORKSHOP
5230 Clark Ave., Suite 19
Lakewood, CA 90712
(213) 920-8809

DESK SET $39.00

DESK SET is a perpetual calendar, an appointment calendar and also a card file. The perpetual calendar is a calendar of every month, past, present or future. The appointment calendar allows up to 15 entries to be made each day. The card file is a mail list program which holds up to 200 addresses. The printing format of card file includes continuous lists, labels or envelopes. Files can be printed; all the files from one file number to another; by zip code; by state or by selected files. DESK SET is an easy way to organize your life. 40K

FINANCIAL CALCULATOR $29.00

The program answers virtually any questions concerning the cost of money, loans, and interest earned on savings, loans and investments. Plus, this program will give a complete interest earned table and amortization table. This program is a must for anyone serious about money. 32K.

FORECASTER $29.00

Forecast future events based on past information. Forecast profits, costs, sales trends, prices, test scores, virtually anything. Edit, save on disk and test various elements to determine the outcome. FORECASTER is a powerful "what if" program – a must for business. 24K

STAT PLUS $29.00

STAT PLUS is the most powerful statistic tool we have seen. Run sample space, mean, variance and standard deviation. Do probabilities using binomial, poisson or normal distribution. Also, do students' t-test, Mann-Whitney U test and Chi Square. Do linear regressions; may use 1-12 independent variables. Interact a sample from one module to another, edit, rank and print out. 24K min.

DISK FIX KIT $29.00

It takes more than a speed adjustment to properly set up an Atari® 810 Disk Drive. Do it Right – Test Disk, Cleaner, Tools, Special Oil, Swabs and complete instructions.

TO ORDER: **VISA/MasterCard, check or money order accepted. If charge, please include expiration date of card. Shipping and Handling software $1.50, disk drive, $10.00. California residents add sales tax. Phone or mail.**

® ATARI is a registered trademark of Warner Communications

CIRCLE #105 ON READER SERVICE CARD

MILES ACCOUNTING SYSTEM II™

The Finest Accounting Software Available for the Atari Today

It's good for business.

Who says the Atari is just a games machine? Not us. And we have the software to prove it.

There's no longer any reason to spend thousands for a "business computer" when your Atari and Miles Accounting System II can handle any accounting situation your business can throw at it.

Miles Computing has the hands-down, best accounting software available for the Atari today. We think you'll agree.

The Critics' Choice.

The critics agree. Here's what some of them said about Miles Payroll System:

"The package is flexible and capable of holding and manipulating a wide spectrum of information. In fact, it is hard to think of an area the program does not cover."
—*Desktop Computing*

"The performance of Miles Payroll System is excellent. It handles all its promised features quickly and easily. ...It is an easy-to-learn program that is very powerful."
—*InfoWorld*

"The documentation is excellent—well-written, clear and concise."
—*Desktop Computing*

"So much for the Atari's reputation of being a games-only machine."
—*Desktop Computing*

Hardware Requirements: Atari computer with 48k, any 80 column (minimum) printer, and 2 disk drives (single or double density). Double density drives will greatly increase the capacity of Miles Accounting System II.

Price: $145 per module. Buy any 3 modules and get the 4th one free.

InfoWorld
Report Card

Miles Payroll System

	Poor	Fair	Good	Excellent
Performance	☐	☐	☐	☑
Documentation	☐	☐	☐	☑
Ease of Use	☐	☐	☐	☑
Error Handling	☐	☐	☐	☑

"In all respects—features, performance, ease of use, documentation and support—this is a professionally conceived and executed program. We look forward to other business software from Miles Computing that will serve the needs of Atari owners."
—*InfoWorld*

The Complete Accounting System.

There are six separate modules in the system. Each functions individually as a complete stand-alone program. All six modules can also be easily linked resulting in an extremely powerful integrated accounting system.

G/L General Ledger System***

Allows interactive maintenance of the chart of accounts file, including add, change/inquiry and delete. Handles up to thirteen accounting periods, supports multiple profit centers, prints source cross-reference reports, financial statements (Income Statement, Balance Statement, Statement of Cash Flow).

A/R Accounts Receivable System**

This comprehensive system processes and maintains records from customer invoices and cash receipts, calculates service charges, maintains sales history and credit rating records. Allows aging. Handles both open items and balance forward customers.

A/P Accounts Payable System***

Keeps accurate status of all outstanding obligations: prints cash requirements report, allows flexable payment selection, including partial payments, prints A/P checks and check register, prints vendor analysis report.

PAYROLL Payroll System

Cumulative totals are maintained for each employee, as well as complete reporting, check writing, and W-2 reporting. Allows weekly, biweekly, semimonthly or monthly pay periods, handles Federal, State and City taxes, FICA, SDI, Group Insurance, Federal and State Unemployment Insurance, maintains cumulative totals and Worker's Compensation, prints paychecks and W-2's. Gives 941 information.

I/C Inventory Control System**

Provides complete control of your resale inventory: not in stock items, items on order, items at or below reorder point, complete Vendor Item Report, suggested Purchase Order to Vendor. Allows for inventory costing by either average cost, LIFO or FIFO, and handles multiple pricing per items.

OE/I Order Entry Invoicing System**

For entry of sales orders and shipping data, and printing customer orders, invoices and shipping papers. May also used to maintain address records, generating back orders for partially filled orders. Orders are automatically printed when shipping dates are entered into the system. Provides O/E and editing, handles credit memos, prints picking tickets, price lists and stocking status reports.

** Available end of 2nd quarter 1984
*** Available 3rd quarter 1984

See for yourself.

Ask your local computer dealer to show you one of our self-running demos, or call us directly.

MILES Computing

Miles ahead of the pack.

MILES COMPUTING
7136 Haskell Avenue Suite 300
Van Nuys, California 91406
(818) 994-7901

CIRCLE #106 ON READER SERVICE CARD

BALL DROP

16K Cassette or Disk

by Bill Boegelein

Interesting programs don't *always* have to be of eye-straining length; there are some which don't require typing until your fingerprints have vanished. **Balldrop** is one such program.

Program Listing 1 does manage to accomplish quite a bit in its less than forty BASIC lines. When run, it will simulate a random demo called **Balldrop**. Imagine a board mounted vertically, on which are arranged in a symmetrical triangular pattern 171 nails. Marbles of equal size and weight are dropped onto the top nail and can randomly bounce either to the left or right each time they come in contact with the nail. If a thousand marbles were dropped, how many would fill each of the nineteen compartments at the bottom?

Inner workings.

The program runs surprisingly fast for being written entirely in BASIC. Lines 0-50 set screen color, margins and DIMs, and clear needed variables. The variable MARBLES in Line 50 should be set to the desired number of marbles to be dropped. Lines 100-200 draw the screen. Lines 200-300 drop one marble at a time onto the top peg. Line 240 randomly sets RN to either +1 or −1. When this is added to the X in the POSITION statement, it moves the marble one position left or right, while Line 270 erases its old position. This is a fairly common way to create the pseudo-animation effect seen here. Lines 300-400 keep track of the totals seen at the screen's bottom in the array C(XPOS) and print this total vertically in the corresponding position.

Variations on a theme.

Adjust the variable MARBLES in Line 50 to the number of marbles to be dropped in the demonstration. Use CTRL-1 to halt and restart the program where desired. If the program is left running over approximately eight minutes, the Atari will automatically enter the attract mode and rotate the screen through various colors. Hit the space bar to bring things back to normal. The program can be renumbered in any increment, for it contains no internal line references, no GOTOs, GOSUBs or TRAPs. It can also be condensed with multiple statements per line to reduce its overall length and increase its speed slightly.

New and improved.

Add the following lines to Listing 1 to include a much faster machine language subroutine to speed up the **Balldrop** animation.

```
60 DIM ML$(150)
70 FOR I=1 TO 108:READ BYTE:ML$(I,I)=C
HR$(BYTE):TTL=TTL+BYTE:NEXT I
80 IF TTL<>14002 THEN ? "ERROR IN DATA
 LINES":STOP
90 SPEED=5:ML$(82,82)=CHR$(SPEED):REM
SPEED 1=FAST TO 255=SLOW
220 XXX=USR(ADR(ML$))
230 REM
240 REM
250 REM
260 REM
270 REM
280 REM
290 REM
1000 DATA 104,169,0,141,8,210,169,3,14
1,15,210,169,19,133,203,169
1010 DATA 1,133,204,169,5,141,0,210,16
9,168,141,1,210,165,203,133
1020 DATA 85,165,204,133,84,169,20,32,
164,246,165,203,133,85,165,204
1030 DATA 133,84,169,0,141,1,210,173,1
0,210,74,74,74,74,74,74
1040 DATA 74,201,0,208,2,169,3,24,101,
203,133,203,198,203,198,203
1050 DATA 162,5,202,160,0,136,192,0,20
8,251,224,0,208,244,169,32
1060 DATA 32,164,246,230,204,165,204,2
01,20,208,168,96
```

•

CHECKSUM DATA.

(see page 43)

```
60 DATA 929,60,805,97,959,85,88,91,94,
97,100,103,714,647,995,5864
1030 DATA 489,640,811,725,2665
```

Then delete Lines 220-290, replacing them with the single statement:

```
215 XXX=USR(ADR(ML$)):REM THIS LINE RE
PLACES LINES 220-290
```

The variable SPEED in Line 90 can be varied from 1 to 255 to increase the program's running speed. SPEED=1 will produce an effect barely visible, SPEED=255 slows it dramatically.

Normal distribution.

The program simulates what is known as "normal distribution," and the totals at the screen's bottom can be predicted fairly accurately by drawing what is called a "Pascal's triangle," like this:

```
                1
              1   1
            1   2   1
          1   3   3   1
        1   4   6   4   1
      1   5  10  10   5   1
    1   6  15  20  15   6   1
  1   7  21  35  35  21   7   1
1   8  28  56  70  56  28   8   1
```

Each number in the sequence is found by adding the two numbers immediately above it. If this triangle was carried out to the 19th level that is represented in the **Balldrop** demo, it would read:

```
1-18-153-816-3060-8568-18564-31824-437
58-48620-43758-31824-18564-8568-3060-8
16-153-18-1
```

Probability predicts that this should be the result after the demo drops 262,143 marbles. Using a more reasonable sample of 10,001, the result should come very close to:

```
0-1-6-31-117-327-708-1214-1669-1855-16
69-1214-708-327-117-31-6-1-0
```

An actual sample run of 10,001 resulted in totals of:

```
0-3-6-36-126-306-691-1155-1716-1843-16
44-1246-769-292-126-37-5-0-0
```

A predicted run of 46,663 should be:

```
0-3-27-145-545-1525-3305-5665-7789-865
5-7789-5665-3305-1525-545-145-27-3-0
```

The actual run showed:

```
0-4-27-150-511-1623-3358-5653-7709-850
5-7743-5815-3375-1482-546-126-33-1-0
```

. . . All fairly close approximations of the normal distribution curve.

In conclusion.

I'll leave it to our avid readers to design a similar 3-D version, in which the marbles are dropped onto a pyramid of suspended pegs, allowing each marble to fall randomly to the bottom. Good luck. □

```
0 REM ********************
1 REM *     BALLDROP     *
2 REM *  BILL BOEGELEIN  *
3 REM * ANALOG COMPUTING *
4 REM ********************
10 GRAPHICS 0:TRAP 40000
20 POKE 82,0:POKE 83,39:POKE 710,0:POK
E 752,1:? "K*************** BALLDROP *
**************";
30 DIM C(37),NUM$(7)
40 FOR CLEAR=1 TO 37 STEP 2:C(CLEAR)=0
:NEXT CLEAR
50 MARBLES=1000:REM SET MARBLES TO DES
IRED AMOUNT
100 REM ... DRAW SCREEN ...
110 SPACE=19:PEG=1:?
120 FOR J=1 TO 18
130 FOR I=1 TO SPACE:? " ";:NEXT I
140 FOR I=1 TO PEG:? CHR$(20);" ";:NEX
T I
150 SPACE=SPACE-1:PEG=PEG+1:? :NEXT J
160 ? :? :?
170 FOR Y=20 TO 23
180 FOR X=0 TO 38 STEP 2:POSITION X,Y:
? CHR$(124);:NEXT X:NEXT Y
190 POSITION 1,3:? "COUNT";
200 REM ... DROP MARBLE ...
210 FOR COUNT=1 TO MARBLES
220 X=19:FOR Y=1 TO 19
230 POSITION X,Y:? CHR$(20);:POSITION
X,Y
240 RN=INT(RND(0)+0.5):IF RN=0 THEN RN
=-1
250 SOUND 0,5,10,8
260 X=X+RN
270 ? " ";
280 SOUND 0,0,0,0
290 NEXT Y
```

```
300 REM ... COUNT MARBLES ...
310 XPOS=PEEK(91):C(XPOS)=C(XPOS)+1
320 NUM$="    ":NUM$(1,LEN(STR$(C(XPO
S))))=STR$(C(XPOS))
330 POSITION XPOS,20:? NUM$(1,1);
340 POSITION XPOS,21:? NUM$(2,2);
350 POSITION XPOS,22:? NUM$(3,3);
360 POSITION XPOS,23:? NUM$(4,4);
370 POSITION 7,3:? COUNT;
380 SOUND 0,50,10,8
390 NEXT COUNT
400 POSITION 0,0:POKE 752,0:? CHR$(253
):POKE 764,255:END
```

•

CHECKSUM DATA.

(see page 43)

```
0 DATA 552,889,260,499,560,250,197,420
,689,901,466,858,142,564,945,8192
150 DATA 262,577,303,588,587,442,601,2
5,503,107,304,707,379,95,789,6269
300 DATA 680,629,906,375,386,397,408,6
55,591,167,26,5220
```

•

Spanish Study Guide

16K Cassette or Disk

by Larry Nocella

This **Spanish Study Guide** program should help you increase your Spanish vocabulary. You'd better know your words well, because there is no special order the words are learned in.

Sometimes a word you have learned — or want to learn — may have a different gender than the one you are familiar with. Such a word is *purpula* or *purpulo* (purple). After CHECKing the program and making sure it works, you may change the DATA statements to suit yourself. You can even change them to German or French. Also, some words may have more than one meaning. To find out which meaning is in the program, just take a look at the DATA statements and change it if you like. *Miercoles* (Wednesday) does have an accent over the *e*, but not in the program. This is the only word which should have an accent.

When you RUN the program, you will be asked for Spanish to English or vice versa. In the first option, the computer prints the Spanish word, and you must type the English equivalent and press the RETURN key. The second option is printed the other way around.

After you choose one of these, the main menu will come up. Press the number of words you would like to learn. You will then be asked a question; type the answer and press RETURN. When you have been asked all the words in that section, you'll return to the main menu. The computer keeps score for you.

I hope this program works for you as well as it did for me. □

```
10 REM ****************************
11 REM *                          *
12 REM *    SPANISH STUDY GUIDE   *
13 REM *                          *
14 REM *     by:Larry  Nocella    *
15 REM *   Idea:Andrew Wodarczyk  *
16 REM ****************************
17 DIM N$(15),A$(15),Q$(15),RN(15)
20 POKE 710,0:POKE 709,14:? "K ↓↓↓↓ (1
) ENGLISH TO SPANISH"
25 ? " ↓↓ (2) SPANISH TO ENGLISH":? "↓
↓ TYPE THE NUMBER,THEN PRESS RETURN";
30 TRAP 20:INPUT ON:IF ON<1 OR ON>2 TH
EN 20
35 IF ON=1 THEN CON=315
38 IF ON=2 THEN CON=371
40 TRAP 40000
50 QR=0:THING=0
100 GRAPHICS 0:POKE 709,14:POKE 710,64
:POKE 752,1:QR=QR+RITE:THING=THING+THN
O
110 ? "   SPANISH STUDY GUIDE - MAIN M
ENU":?
115 ? "               by: Larry Nocella"
120 ? :? :? "▶(1) Days of the week"
125 ? :? "▶(2) Verbs"
130 ? :? "▶(3) Months of the Year"
135 ? :? "▶(4) Nouns"
140 ? :? "▶(5) Colors"
145 ? :? "▶(6) Adjectives"
150 ? :? "▶(7) Exit this program"
160 ? :? "▶ PRESS A NUMBER KEY":POKE 7
52,1
162 ? :? "            SCORE:";QR;" right out
 of ";THING
165 OPEN #1,4,0,"K:"
170 GET #1,Z:IF Z<49 OR Z>55 THEN GOSU
B 200:GOTO 170
175 CLOSE #1
180 RITE=0
185 ON Z-48 GOTO 300,500,700,900,1100,
1300,210
200 FOR I=30 TO 0:SOUND 0,I,10,15:FOR
XX=1 TO 20:NEXT XX:NEXT I:SOUND 0,0,0,
0:RETURN
210 ? "K↓↓↓BYE!-BYE!,ADIOS!"
215 POKE 752,0:END
```

```
300 POKE 710,113:? "K↓ LOS DIAS DE LA
SEMANA"
305 ? " ----------------------":LNO=199
9:THNO=7
310 ? " THE DAYS OF THE WEEK":? :GOTO
400
315 REM ****************************
316 POKE 752,0
317 FOR R=1 TO THNO
320 N$="                    ":A$=N$
325 RESTORE LNO+RN(R)
330 READ N$,A$
335 ? :? " ";R;".";N$;" in English is
";:INPUT Q$
340 IF Q$=A$ THEN ? " RIGHT!":GOSUB 20
0:RITE=RITE+1:GOTO 355
350 ? :? "Sorry,But the answer is ";A$
355 NEXT R
360 ? :? " You got ";RITE;" right out
of ";THNO;"!";
365 FOR XX=1 TO 800:NEXT XX
370 GOTO 100
371 POKE 752,0
372 FOR R=1 TO THNO
374 N$="                    ":A$=N$
376 RESTORE LNO+RN(R)
378 READ N$,A$
380 ? :? " ";R;".";A$;" in Spanish is
";:INPUT Q$
382 IF Q$=N$ THEN ? " RIGHT!":GOSUB 20
0:RITE=RITE+1:GOTO 355
384 ? :? "Sorry,But the answer is ";N$
386 NEXT R
388 ? :? " You got ";RITE;" right out
of ";THNO;"!";
389 FOR XX=1 TO 800:NEXT XX
390 GOTO 100
400 FOR I=1 TO THNO:RN(I)=I:NEXT I
410 FOR I=1 TO THNO
415 TMP=INT(RND(1)*THNO)+1
420 F=RN(I):RN(I)=RN(TMP):RN(TMP)=F
425 NEXT I
430 GOTO CON
500 POKE 710,208:? "K↓ VERBOS"
510 ? " ------"
520 ? " VERBS ":THNO=15:LNO=2027
530 GOTO 400
700 POKE 710,36:? "K↓ LOS MESES DEL AN
O"
705 ? " -----------------------"
710 ? " THE MONTHS OF THE YEAR"
715 THNO=12:LNO=2006:GOTO 400
900 POKE 710,42:POKE 709,2
910 ? "K↓ NOMBRES"
915 ? " -------":? " NOUNS"
920 LNO=2043
930 THNO=15:GOTO 400
1100 POKE 710,100:? "K↓ LOS COLORES"
1110 ? " -----------"
1120 ? " THE COLORS"
1130 THNO=9:LNO=2018:GOTO 400
1300 POKE 710,8:POKE 709,2
1310 ? "K↓ ADJECTIVOS"
1320 ? " -----------":? " ADJECTIVES"
1330 THNO=15:LNO=2058
1340 GOTO 400
1999 REM *** SPANISH THEN ENGLISH ***
2000 DATA DOMINGO,SUNDAY
2001 DATA SABADO,SATURDAY
2002 DATA MIERCOLES,WEDNESDAY
2003 DATA MARTES,TUESDAY
2004 DATA VIERNES,FRIDAY
2005 DATA JUEVES,THURSDAY
2006 DATA LUNES,MONDAY
2007 DATA MAYO,MAY
2008 DATA JUNIO,JUNE
2009 DATA SEPTIEMBRE,SEPTEMBER
2010 DATA ABRIL,APRIL
2011 DATA ENERO,JANUARY
2012 DATA DICIEMBRE,DECEMBER
2013 DATA FEBRERO,FEBRUARY
2014 DATA AGOSTO,AUGUST
2015 DATA MARZO,MARCH
2016 DATA OCTUBRE,OCTOBER
2017 DATA JULIO,JULY
2018 DATA NOVIEMBRE,NOVEMBER
2019 DATA AZUL,BLUE
2020 DATA ROJO,RED
2021 DATA BLANCO,WHITE
2022 DATA VERDE,GREEN
2023 DATA NARANJA,ORANGE
2024 DATA AMARILLO,YELLOW
2025 DATA PURPULA,PURPLE
2026 DATA NEGRO,BLACK
2027 DATA GRIS,GREY
2028 DATA GANAR,TO WIN
2029 DATA COMPRAR,TO BUY
2030 DATA NADAR,TO SWIM
2031 DATA LEER,TO READ
2032 DATA VER,TO SEE
2033 DATA VIVIR,TO LIVE
2034 DATA LLEGAR,TO ARRIVE
2035 DATA PASAR,TO PASS
2036 DATA TRABAJAR,TO WORK
2037 DATA ESCRIBIR,TO WRITE
2038 DATA HABLAR,TO SPEAK
2039 DATA TOCAR,TO TOUCH
2040 DATA COMENZAR,TO BEGIN
2041 DATA ESQUIAR,TO SKI
2042 DATA ESTAR,TO BE
2043 DATA PERDER,TO LOSE
2044 DATA PARQUE,PARK
2045 DATA PUEBLO,TOWN
2046 DATA CASA,HOUSE
2047 DATA CARNE,MEAT
2048 DATA TECHO,ROOF
2049 DATA MUCHACHO,BOY
2050 DATA MUCHACHA,GIRL
2051 DATA DINERO,MONEY
2052 DATA PADRE,FATHER
2053 DATA MADRE,MOTHER
2054 DATA DEPORTE,SPORT
2055 DATA PELOTA,BALL
2056 DATA PARTIDO,GAME
2057 DATA TORO,BULL
2058 DATA MESA,TABLE
2059 DATA FEO,UGLY
2060 DATA BAJO,SHORT
2061 DATA ALTO,TALL
2062 DATA MAL,BAD
2063 DATA FRESCO,FRESH
2064 DATA BUENO,GOOD
2065 DATA CIERTO,TRUE
2066 DATA FALSO,FALSE
2067 DATA ANCHO,WIDE
2068 DATA POBRE,POOR
2069 DATA FUERTE,STRONG
2070 DATA BONITA,PRETTY
2071 DATA GUAPO,HANDSOME
2072 DATA HUMILDE,HUMBLE
2073 DATA AMERICANO,AMERICAN
```

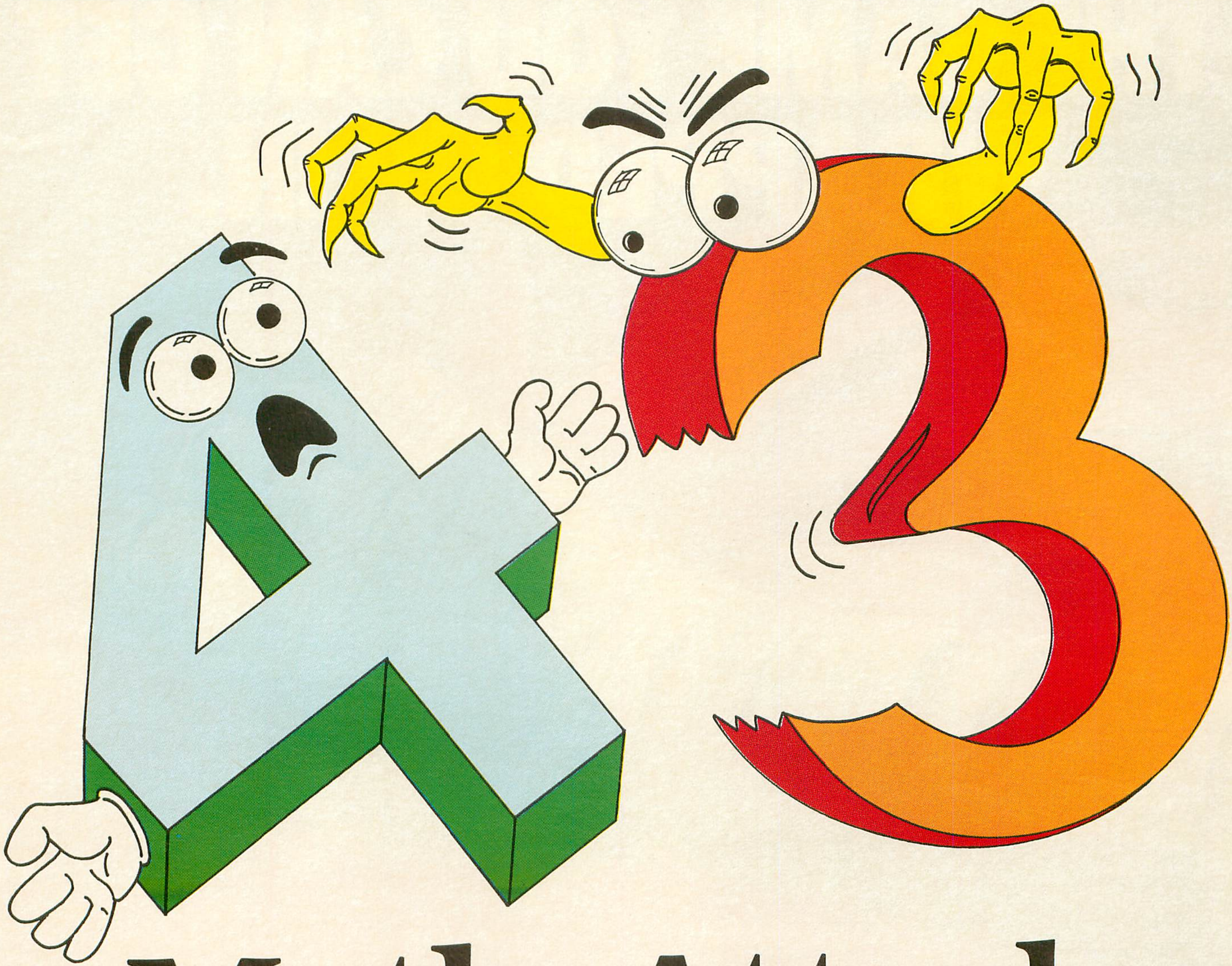
•

CHECKSUM DATA.

(see page 43)

```
10 DATA 519,32,475,38,243,720,537,981,
927,19,717,987,998,916,229,8338
100 DATA 970,947,722,54,141,465,189,46
1,984,390,613,88,281,714,661,7680
180 DATA 694,313,909,719,650,452,295,8
81,790,781,822,901,136,749,717,9809
340 DATA 267,846,770,598,661,708,794,8
35,920,152,769,720,320,885,780,10025
388 DATA 612,671,714,773,808,525,707,7
45,857,742,634,999,706,353,224,10070
710 DATA 187,723,205,608,78,686,636,51
3,126,439,653,113,976,270,791,7004
1340 DATA 884,982,449,690,89,422,484,7
86,300,580,753,370,916,251,997,8953
2013 DATA 724,225,14,781,804,170,857,5
68,199,949,411,775,544,925,913,8859
2028 DATA 230,482,129,267,614,194,510,
182,597,859,552,393,742,480,866,7097
2043 DATA 315,983,995,744,739,769,253,
135,263,273,301,303,932,308,938,8251
2058 DATA 732,509,793,847,208,244,708,
993,979,705,817,220,193,399,426,8773
2073 DATA 43,43
```

•



Math Attack

32K Cassette or 48K Disk

by Manny Miller

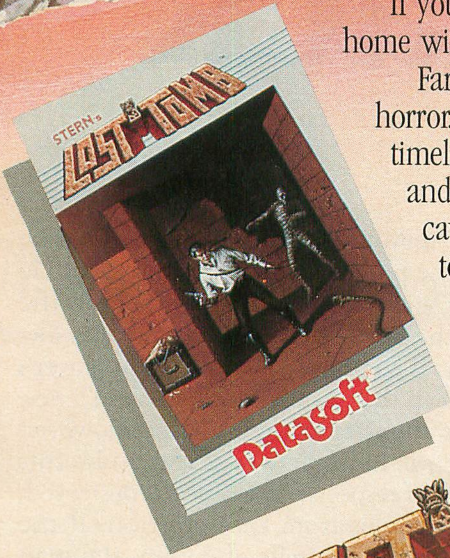
Math Attack is a fast-paced, joystick-operated mathematical game for one or two players. The game features randomly-generated problems in addition, subtraction and multiplication at four levels of difficulty. These various levels of play are also accompanied by several different playing speeds. All game conditions feature plentiful graphic and sound effects.

The game play consists of one or two graphic men standing on either side of the display screen. They are on a golden-colored platform and have three blocks above their heads. A randomly-generated problem is placed beside the men, and one of the digits of the answer is replaced by a randomly-generated digit, which can be changed by a joystick controller. In order to effect a change, the joystick must start in the neutral position, then may be moved to the up position for each digital increment.

While a player manipulates the joystick, a probe advances toward the player's man at a game-selectible speed. Whenever a player thinks he has chosen the correct digit for the answer, he should press his joystick's fire button. If the digit is correct, the player's man will shoot down the advancing probe, and a new probe will appear at the starting position. However, if the digit chosen is incorrect, or if the probe hits a block due to lack of response, the block is destroyed, and the probe continues from that position. If the probe should hit or fire a missile at a man, the man is disintegrated, and game participation is over for that player. The game continues until either the players are eliminated or ten problems have been generated for a player (ten problems for a one-player game, twenty for a two-player game).

(continued on page 25)

THE GAME YOU CAN GET WRAPPED UP IN.



If you thought Egypt was the home of the Pyramids, wait until you get home with Lost Tomb.™

Far more than mere chambers of wonder, these chambers are filled with horror. Poisonous scorpions, screeching bats and terrifying mummies. And in the timeless tradition of the most daring expeditions, you'll pack a pistol, plenty of ammo and a whip to crack the curse of the pyramids. Earthquakes rumble along cavernous passageways. Walls crumble and crackle with gunfire. Your mission is to make it through all 91 chambers and 13 levels. And then make off with the loot. The only things we can't give you are the things you'll need most. Cool reflexes, uncanny instincts and the courage to use them.

Lost Tomb.™ Can you unravel the mystery?

Available now for Atari, Commodore 64, Apple II series and IBM PC and PC/JR. Suggested retail price $29.95. Check with your local home computer software retailer for Lost Tomb,™ and to learn of other great programs from Datasoft® send for a free consumer catalog.

LOST TOMB

By
Datasoft®

19808 Nordhoff Place, Chatsworth, CA 91311 Phone (818) 701-5161

CIRCLE #109 ON READER SERVICE CARD

Game equipment.

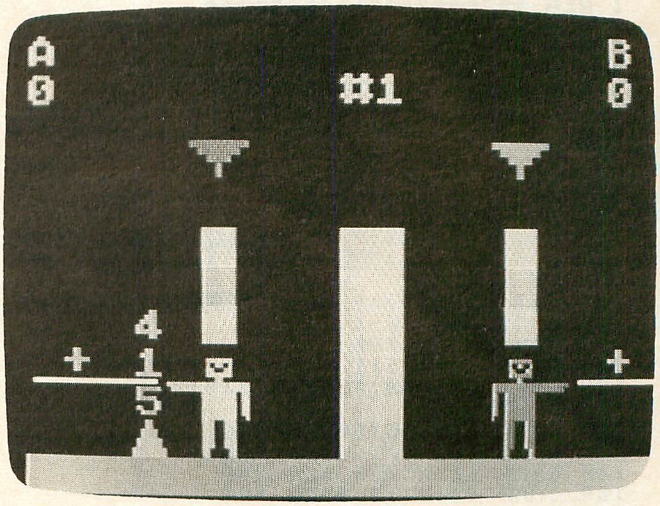
The following items are needed in order to play **Math Attack**: (1) an Atari 400 or 800 personal computer with at least 24K of RAM for cassette operation or 32K or RAM for a disk drive; (2) BASIC language cartridge; (3) black-and-white or color TV or monitor; (4) one or two joystick controllers (joystick 1 is placed in port 1 and joystick 2 in port 2).

Playing the game.

After typing or loading the program into memory, use the RUN command to begin program execution. The computer will display the title page for several seconds, then the game conditions screen is shown. The special function keys of the Atari are used to choose game conditions. There are eight speed options which can be chosen. Options 1 through 6 will give a fixed speed setting throughout the game, with option 1 representing the slowest speed and option 6 the fastest. Options 7 and 8 will cause the speed to slowly increase for each number generated, with speed 8 being faster-paced than 7.

The levels of difficulty range from 1 through 4 for each operation, with level 1 representing the easiest level and 4 the hardest.

Pressing the START key draws the game display and starts the game. During play, the game conditions display can be re-accessed while the probes are travelling down the screen (making a thumping sound), by first depressing the SELECT key and, while doing so, pressing the fire button of an active joystick. The conditions display can also be shown after a game by pressing only the SELECT key. □



Math Attack.

(Listing starts on next page.)

Program Description.

Lines 10 - 73 — Set up title page.
Lines 75 - 295 — Set up font.
Lines 305 - 410 — Read in m/l routine.
Lines 452 - 585 — Initialize variables.
Lines 500 - 780 — Game conditions display and subroutines to change game conditions.
Lines 900 - 950 — Subroutine to set up font.
Lines 1000 - 1990 — Set up P/M graphics, certain game parameters for speed and perform certain factors for a one- or two-player game.
Lines 2000 - 2085 — Main control for two players.
Lines 2100 - 2185 — Control when only left player remaining or for a one-player game.
Lines 2200 - 2285 — Control for only right player in game.
Lines 2500 - 2625 — Print message at end of successful game.
Lines 2800 - 2820 — Subroutine to choose factor for generating numbers.
Lines 2900 - 2930 — Message displayed if maximum problems are not obtained.
Lines 3000 - 3049 — Subroutines to generate the randomly generated problems and draw them to the screen.
Lines 3500 - 3650 — Subroutines to set up screen display.
Lines 3700 - 3750 — Subroutine to draw probe A (missile player 0).
Lines 3800 - 3850 — Subroutine to draw probe B (missile player 1).
Line 4000 — Set up players 2 and 3 (lines for problems).
Lines 5000 - 6160 — Subroutines and program sections for evaluating, answer, choosing wrong answer, time expiring or going back to game display.
Lines 6200 - 6495 — Subroutines for choosing right answer.
Lines 7000 - 7300 — Subroutines to zero parts of P/M area.
Lines 8000 - 8930 — Subroutines to draw various positions of graphics men.
Line 9000 — Subroutine to return to game conditions display.
Lines 9120 - 9500 — Zero P/M parameters.

```
1 REM ********************
2 REM *   MATH ATTACK     *
3 REM * BY MANNY MILLER   *
4 REM * ANALOG COMPUTING *
5 REM ********************
10 SETCOLOR 4,0,14:SETCOLOR 0,3,8:SETC
OLOR 1,3,8:SETCOLOR 2,8,8:SETCOLOR 3,0
,0
50 RAMTOP=PEEK(106)
60 POKE 106,RAMTOP-8
65 GRAPHICS 18
71 POSITION 4,3:? #6;"MaTh AtTaCk":POS
ITION 9,4:? #6;"BY":POSITION 4,5:? #6;
"MaNNY MILLeR"
73 POSITION 3,7:? #6;"CoPYRiGhT 1984"
75 CHBAS=RAMTOP-4
80 ADDR=CHBAS*256
90 FOR X=0 TO 511
100 POKE ADDR+X,PEEK(57344+X)
110 NEXT X
120 CHAR=4:N=55:GOSUB 900
130 CHAR=12:N=7:GOSUB 900
140 CHAR=14:N=7:GOSUB 900
160 CHAR=42:N=7:GOSUB 900
170 CHAR=49:N=7:GOSUB 900
180 CHAR=54:N=7:GOSUB 900
184 CHAR=31:N=7:GOSUB 900
185 CHAR=53:N=7:GOSUB 900
190 CHAR=58:N=47:GOSUB 900
215 DATA 231,231,231,231,231,231,195,1
95
220 DATA 7,7,6,6,6,6,6,4
225 DATA 224,224,96,96,96,96,96,32
230 DATA 3,7,14,28,56,32,0,0
235 DATA 192,224,112,56,28,4,0,0
240 DATA 255,127,0,0,0,0,0,0
245 DATA 255,254,0,0,0,0,0,0
250 DATA 7,7,0,0,0,0,0,0
253 DATA 224,224,0,0,0,0,0,0
263 DATA 0,4,6,6,6,6,6,6
265 DATA 0,32,96,96,96,96,96,96
268 DATA 0,1,3,7,15,28,56,0
270 DATA 255,231,195,129,0,0,0,0
273 DATA 0,128,192,224,240,56,28,0
275 DATA 0,0,0,24,24,0,0,0
278 DATA 0,0,60,60,60,60,0,0
280 DATA 0,126,126,126,126,126,126,0
285 DATA 24,24,60,60,126,126,255,255
290 DATA 0,0,0,126,90,102,126,24
295 DATA 255,255,255,255,255,255,255,2
55
305 FOR X=1536 TO 1548:READ N:POKE X,N
:NEXT X
310 DATA 104,160,0,200,177,205,136,145
,205,200,208,247,96
315 FOR X=1549 TO 1561:READ N:POKE X,N
:NEXT X
320 DATA 104,160,255,136,177,205,200,1
45,205,136,208,247,96
325 FOR X=1562 TO 1758:READ N:POKE X,N
:NEXT X
330 DATA 104,169,15,205,120,2,208,5,16
9,1,141,244,6,169,15,205,123,2,208,5,1
69,1,141,245,6,169
340 DATA 14,205,120,2,208,34,169,1,205
,244,6,208,27,169,0,141,244,6,160,0,16
9,89,209,203,208,7
350 DATA 169,80,145,203,76,92,6,177,20
3,170,232,138,145,203,169,14,205,123,2
,208,34,169,1,205,245,6
360 DATA 208,27,169,0,141,245,6,160,0,
169,89,209,207,208,7,169,80,145,207,76
,133,6,177,207,170,232
370 DATA 138,145,207,173,248,6,205,132
,2,208,6,169,1,141,246,6,96,173,249,6,
205,135,2,208,6,169
380 DATA 2,141,246,6,96,173,247,6,56,2
29,20,48,4,24,76,27,6,24,169,239,141,1
,210,173,252,6,133
390 DATA 206,32,14,6,173,253,6,133,206
,32,14,6,238,250,6,238,251,6,169,0,141
,1,210,133,20,205
400 DATA 6,208,240,1,96,205,7,208,240,
1,96,76,27,6
405 FOR X=1759 TO 1769:READ N:POKE X,N
:NEXT X
410 DATA 104,174,254,6,32,1,6,202,208,
250,96
452 DIM S1$(180),BLOCK(10),F1(4,5),F2(
4,5),F3(4,5),F4(4,5),OP$(1)
454 DIM ANS$(1),ANS1$(3)
460 S1$="QQQ<<///(((((((((((((((((((/
//////////////////////////<<<<</////////////////<
<<<<QQQQQQ           "
465 S1$(84,159)="QQQ<<///(((((((((((((
(((((((((((((((//////<<<<<QQQQQQQQQQQQQ
QQQQQQQ<<<<<<<<<<"
470 BLOCK(1)=63:BLOCK(2)=59:BLOCK(3)=5
8:BLOCK(4)=58:BLOCK(5)=32:BLOCK(6)=32:
BLOCK(7)=32:BLOCK(8)=32
475 F1(1,1)=6:F2(1,1)=0:F3(1,1)=5:F4(1
,1)=0:F1(1,2)=9:F2(1,2)=1:F3(1,2)=9:F4
(1,2)=1
476 F1(1,3)=41:F2(1,3)=10:F3(1,3)=40:F
4(1,3)=10:F1(1,4)=50:F2(1,4)=50:F3(1,4
)=50:F4(1,4)=50
477 F1(2,1)=6:F2(2,1)=0:F3(2,1)=5:F4(2
,1)=0:F1(2,2)=9:F2(2,2)=1:F3(2,2)=9:F4
(2,2)=1
478 F1(2,3)=90:F2(2,3)=10:F3(2,3)=9:F4
(2,3)=1:F1(2,4)=90:F2(2,4)=10:F3(2,4)=
90:F4(2,4)=10
479 F1(3,1)=6:F2(3,1)=0:F3(3,1)=6:F4(3
,1)=0:F1(3,2)=7:F2(3,2)=6:F3(3,2)=7:F4
(3,2)=6
480 F1(3,3)=13:F2(3,3)=13:F3(3,3)=13:F
4(3,3)=13:F1(3,4)=90:F2(3,4)=10:F3(3,4
)=9:F4(3,4)=2
485 PLAYERS=2:OP$="+":OPER=1:LEVEL=1:O
PERSUB=3000:SPEED=2:POKE 1791,80
500 GRAPHICS 18
505 POKE 756,224
510 POKE 16,64:POKE 53774,64
515 SETCOLOR 0,0,0
520 SETCOLOR 2,0,14
525 SETCOLOR 4,9,4
530 ? #6;"  GAME CONDITIONS:"
535 POSITION 0,2:? #6;"OPTION KEY-PLAY
ERS:";PLAYERS
540 POSITION 0,4:? #6;"SELECT KEY-SPEE
D:";SPEED
545 POSITION 0,6:? #6;"OP & SEL-OPERAT
ION:";OP$
550 POSITION 0,8:? #6;"OP & START-LEVE
L:";LEVEL
555 POSITION 0,10:? #6;"START KEY TO B
EGIN"
560 IF PEEK(53279)=3 THEN GOSUB 600
570 IF PEEK(53279)=5 THEN GOSUB 650
580 IF PEEK(53279)=1 THEN GOSUB 700
590 IF PEEK(53279)=2 THEN GOSUB 750
595 IF PEEK(53279)=6 THEN 1000
596 FOR N=1 TO 100:NEXT N
598 GOTO 560
600 IF PLAYERS=1 THEN PLAYERS=2:POSITI
ON 19,2:? #6;PLAYERS:RETURN
610 IF PLAYERS=2 THEN PLAYERS=1:POSITI
ON 19,2:? #6;PLAYERS:RETURN
650 SPEED=SPEED+1
660 IF SPEED=9 THEN SPEED=1
670 POSITION 17,4:? #6;SPEED
680 RETURN
700 IF OP$="+" THEN OP$="-":OPER=2:OPE
RSUB=3100:POSITION 19,6:? #6;OP$:RETUR
N
710 IF OP$="-" THEN OP$="X":OPER=3:OPE
RSUB=3200:POSITION 19,6:? #6;OP$:RETUR
N
720 IF OP$="X" THEN OP$="+":OPER=1:OPE
RSUB=3000:POSITION 19,6:? #6;OP$:RETUR
N
730 POSITION 19,6:? #6;OP$
740 RETURN
750 LEVEL=LEVEL+1
760 IF LEVEL=5 THEN LEVEL=1
770 POSITION 17,8:? #6;LEVEL
780 RETURN
900 FOR X=0 TO N
915 POS=ADDR+(CHAR*8)
920 READ NUM
930 POKE (POS+X),NUM
940 NEXT X
950 RETURN
1000 GRAPHICS 18:POKE 756,CHBAS:SETCOL
OR 0,10,8:SETCOLOR 1,1,8:SETCOLOR 2,6,
6:SETCOLOR 3,14,6
```

```
1010 POKE 16,64:POKE 53774,64
1020 A=PEEK(106)-16:POKE 54279,A:PMBAS
E=A*256
1030 POKE 559,62:POKE 53277,3
1040 GOSUB 9500
1060 FOR X=1770 TO 1780:READ N:POKE X,
N:NEXT X
1070 DATA 104,160,0,169,0,145,203,200,
208,251,96
1075 GOSUB 7000
1080 RESTORE 1070
1090 POKE 53256,3:POKE 53257,3:POKE 53
258,1:POKE 53259,1:POKE 623,4:POKE 704
,14:POKE 705,14:POKE 706,52
1095 PROBLEM=0:SCOREA=0:SCOREB=0:WRONG
A=0:WRONGB=0:POSA=63:POSB=63:NA=0:NB=0
1100 POKE 707,84:POKE 53278,0
1105 S1=0
1130 GOSUB 3500
1140 IF PLAYERS=2 THEN GOSUB 3600
1150 IF SPEED=1 THEN S=125
1155 IF SPEED=2 THEN S=50
1160 IF SPEED=3 THEN S=25
1165 IF SPEED=4 THEN S=10
1170 IF SPEED=5 THEN S=5
1171 IF SPEED=6 THEN S=3
1175 IF PLAYERS=2 AND SPEED=7 THEN S=6
3:S1=3
1176 IF PLAYERS=2 AND SPEED=8 THEN S=4
2:S1=2
1177 IF PLAYERS=1 AND SPEED=7 THEN S=6
6:S1=6
1178 IF PLAYERS=1 AND SPEED=8 THEN S=4
4:S1=4
1180 GOSUB 4000
1210 POKE 1788,A+6:POKE 1789,A+7
1220 FOR N=1 TO LEN(S1$)
1230 J=15
1240 IF S1$(N,N)=" " THEN J=0
1250 SOUND 0,ASC(S1$(N)),10,J
1255 FOR T=1 TO 1:NEXT T
1260 NEXT N
1265 SOUND 0,0,0,0
1270 IF PLAYERS=2 THEN 2000
1280 WRONGB=5:GOTO 2099
1990 GOSUB 3700
2000 GOSUB 3700:GOSUB 3800:POKE 1784,0
:POKE 1785,0
2001 GOSUB 2800:GOSUB OPERSUB:GOSUB 33
00:N4=0:GOSUB 3400:N4=16:GOSUB 3400
2004 PROBLEM=PROBLEM+1:POSITION 9,1:?
#6;"#";PROBLEM
2005 FOR N=1 TO 25:SOUND 0,255,14,10:N
EXT N:SOUND 0,0,0,0
2009 POKE 77,0
2010 POKE 1782,3:POKE 1783,S:S=S-S1
2014 N1=160:N2=160:N3=0:GOSUB 8600:N1=
128:N2=192:N3=9:GOSUB 8700
2015 POKE 53248,50:POKE 53249,178
2016 POKE 203,PLACEA2:POKE 204,PLACEA1
:POKE 207,PLACEB2:POKE 208,PLACEB1:NOA
=0:NOB=0:NA=0:NB=0
2025 GOSUB 7500
2030 R=USR(1562)
2035 IF PEEK(1782)<>3 THEN POSITION 6,
3:? #6;"ANS: ";F8
2040 ON PEEK(1782) GOSUB 5400,5800,500
0
2041 POKE 53278,0
2050 IF WRONGA>=4 AND WRONGB>=4 THEN 2
900
2060 IF WRONGB=4 THEN WRONGB=5:GOSUB 2
100
2070 IF WRONGA=4 THEN WRONGA=5:GOSUB 2
200
2071 IF PROBLEM=20 AND PLAYERS=2 THEN
2500
2073 IF PEEK(1782)=3 THEN GOTO 2016
2080 GOSUB 6300:POKE 1782,3
2085 GOTO 2001
2099 GOSUB 3700
2100 POKE 1785,255:POKE 1784,0:FOR N=7
 TO 10:POSITION 16,N:? #6;"    ":NEXT
N:POSITION 11,0:? #6;"     "
2101 POKE 53248,50:POKE 53249,0:GOSUB
7300:POSITION 6,3:? #6;"          ":POKE
 53278,0
2103 IF NB=1 THEN 2116
2104 GOSUB 6300:GOSUB 2800:GOSUB OPERS
UB:GOSUB 3300:N4=0:GOSUB 3400
2105 PROBLEM=PROBLEM+1:POSITION 9,1:?
#6;"#";PROBLEM
2106 FOR N=1 TO 25:SOUND 0,255,14,10:N
EXT N:SOUND 0,0,0,0
2108 POKE 1782,3
2109 POKE 77,0
2110 POKE 1783,S:S=S-S1
2114 N1=160:N2=160:N3=0:GOSUB 8600
2116 POKE 203,PLACEA2:POKE 204,PLACEA1
:POKE 207,255:POKE 208,6:NOA=0:NOB=0
2125 GOSUB 7500
2130 R=USR(1562)
2135 IF PEEK(1782)<>3 THEN POSITION 6,
3:? #6;"ANS: ";F8
2140 ON PEEK(1782) GOSUB 5400,5800,500
0
2144 IF WRONGA=4 THEN WRONGA=5:GOSUB 2
900
2145 IF PROBLEM=20 AND PLAYERS=2 THEN
2500
2148 IF PROBLEM=10 AND PLAYERS=1 THEN
2500
2173 IF PEEK(1782)=3 THEN POKE 53278,0
:GOTO 2116
2175 POKE 53278,0:POKE 1782,3
2185 GOTO 2104
2200 POKE 1784,255:POKE 1785,0:FOR N=7
 TO 10:POSITION 0,N:? #6;"    ":NEXT N
:POSITION 2,0:? #6;"     "
2201 POKE 53248,0:POKE 53249,178:GOSUB
 7200:POSITION 6,3:? #6;"          ":POK
E 53278,0
2203 IF NA=1 THEN 2216
2204 GOSUB 6300:GOSUB 2800:GOSUB OPERS
UB:GOSUB 3300:N4=16:GOSUB 3400
2205 PROBLEM=PROBLEM+1:POSITION 9,1:?
#6;"#";PROBLEM
2206 FOR N=1 TO 25:SOUND 0,255,14,10:N
EXT N:SOUND 0,0,0,0
2208 POKE 1782,3
2209 POKE 77,0
2210 POKE 1783,S:S=S-S1
2214 N1=128:N2=192:N3=9:GOSUB 8700
2215 POKE 53248,0:POKE 53249,178
2216 POKE 203,255:POKE 204,6:POKE 207,
PLACEB2:POKE 208,PLACEB1:NOA=0:NOB=0
2225 GOSUB 7500
2230 R=USR(1562)
2235 IF PEEK(1782)<>3 THEN POSITION 6,
3:? #6;"ANS: ";F8
2240 ON PEEK(1782) GOSUB 5400,5800,500
0
2245 IF WRONGB=4 THEN WRONGB=5:GOSUB 2
900
2250 IF PROBLEM=20 THEN 2500
2273 IF PEEK(1782)=3 THEN POKE 53278,0
:GOTO 2216
2275 POKE 53278,0:POKE 1782,3
2285 GOTO 2204
2500 GOSUB 7200:GOSUB 7300:POKE 53250,
0
2505 POSITION 6,3:? #6;"ANS: ";F8
2510 POKE 706,0
2520 FOR N=PMBASE+1648 TO PMBASE+1742:
POKE N,255:NEXT N
2525 POSITION 3,4:? #6;"GAME COMPLETED
"
2530 IF WRONGA<4 THEN N1=160:N3=0:GOSU
B 8700
2540 IF WRONGB<4 THEN N1=128:N3=9:GOSU
B 8600
2545 POKE 53250,120:O=1:POKE 706,24:PO
KE 623,1
2550 POSITION 9,5:? #6;"t "
2555 POSITION 9,6:? #6;"h "
2560 POSITION 9,7:? #6;"e "
2565 POSITION 9,8:? #6;" e"
2570 POSITION 9,9:? #6;" n"
2575 POSITION 9,10:? #6;" d"
2580 K=255:FOR N=14 TO 0 STEP -2:POKE
706,N:GOSUB 6000:NEXT N:SOUND 0,0,0,0
2590 FOR N=PMBASE+1649 TO PMBASE+1742:
POKE N,0
2600 FOR J=250 TO 0 STEP -O:SOUND 0,J,
14,12:NEXT J:O=O+1
```

```
2604 NEXT N
2605 FOR N=94 TO 130
2606 FOR J=250 TO 0 STEP -O:SOUND 0,J,
14,12:NEXT J:O=O+1:NEXT N:SOUND 0,0,0,
0
2620 IF PEEK(53279)=5 THEN 9000
2625 GOTO 2620
2800 F5=INT(F1(OPER,LEVEL)*RND(1))+F2(
OPER,LEVEL)
2810 F6=INT(F3(OPER,LEVEL)*RND(1))+F4(
OPER,LEVEL)
2820 RETURN
2900 POSITION 6,3:? #6;"ANS: ";F8
2910 POSITION 0,4:? #6;" GAME STOPPED
AT #";PROBLEM
2920 IF PEEK(53279)=5 THEN 9000
2930 GOTO 2920
3000 F8=F5+F6:RETURN
3100 IF F5<F6 THEN TEMP=F5:F5=F6:F6=TE
MP
3120 F8=F5-F6
3130 RETURN
3200 F8=F5*F6
3230 RETURN
3300 F0=INT(RND(1)*10)
3320 ANS1$=STR$(F8)
3325 PLACE=1
3330 IF (LEN(ANS1$)=2 OR LEN(ANS1$)=3)
 THEN PLACE=INT(RND(1)*2)+1
3331 IF LEN(ANS1$)=3 THEN PLACE=2
3334 ANS$=ANS1$(LEN(ANS1$)-PLACE+1,LEN
(ANS1$)-PLACE+1)
3340 SCREEN=PEEK(88)+256*PEEK(89):REM
CALCULATE SCREEN ADDRESS
3365 PLACEA1=INT((SCREEN+4-PLACE+180)/
256):PLACEA2=(SCREEN+4-PLACE+180)-PLAC
EA1*256
3370 PLACEB1=INT((SCREEN+20-PLACE+180)
/256):PLACEB2=(SCREEN+20-PLACE+180)-PL
ACEB1*256
3380 RETURN
3400 POSITION 4-LEN(STR$(F5))+N4,7:? #
6;STR$(F5)
3430 POSITION 2-LEN(STR$(F6))+N4,8:? #
6;OP$;" ";F6
3440 POSITION 4-LEN(ANS1$)+N4,9:? #6;A
NS1$
3450 COLOR F0+16:PLOT 4-PLACE+N4,9
3460 COLOR 93:PLOT 4-PLACE+N4,10
3479 RETURN
3500 POSITION 0,0:? #6;"A     "
3520 POSITION 0,1:? #6;SCOREA
3530 FOR N=0 TO 19:COLOR BLOCK(1)+64:P
LOT N,11:NEXT N
3540 FOR N=5 TO 10:COLOR BLOCK(1)+64:P
LOT 9,N:COLOR BLOCK(1)+64:PLOT 10,N:NE
XT N
3550 N1=160:N2=160:N3=0:GOSUB 8000
3560 COLOR BLOCK(1)+32:PLOT 5,5:COLOR
BLOCK(1)+64:PLOT 5,6:COLOR BLOCK(1)+32
:PLOT 5,7
3570 RETURN
3600 POSITION 17,0:? #6;"b"
3620 POSITION 17,1:? #6;SCOREB
3630 N1=128:N2=192:N3=9:GOSUB 8000
3640 COLOR BLOCK(1)+32:PLOT 14,5:COLOR
 BLOCK(1)+64:PLOT 14,6:COLOR BLOCK(1)+
32:PLOT 14,7
3650 RETURN
3700 IF WRONGA>=4 OR PROBLEM=20 OR (PR
OBLEM=10 AND PLAYERS=1) THEN RETURN
3705 FOR N=PMBASE+1576 TO PMBASE+1578:
POKE N,254:NEXT N
3710 FOR N=PMBASE+1579 TO 1581+PMBASE:
POKE N,124:NEXT N
3720 FOR N=PMBASE+1582 TO PMBASE+1584:
POKE N,56:NEXT N
3730 FOR N=PMBASE+1585 TO PMBASE+1590:
POKE N,16:NEXT N
3740 POKE 53250,85:POKE 1786,0
3750 RETURN
3800 IF WRONGB>=4 OR PROBLEM=20 THEN R
ETURN
3805 FOR N=PMBASE+1832 TO PMBASE+1834:
POKE N,254:NEXT N
3810 FOR N=PMBASE+1835 TO PMBASE+1837:
POKE N,124:NEXT N
3820 FOR N=PMBASE+1838 TO PMBASE+1840:
POKE N,56:NEXT N
3830 FOR N=PMBASE+1841 TO PMBASE+1846:
POKE N,16:NEXT N
3840 POKE 53251,157:POKE 1787,0
3850 RETURN
4000 POKE PMBASE+1199,254:POKE PMBASE+
1200,254:POKE PMBASE+1455,254:POKE PMB
ASE+1456,254:RETURN
5000 IF PEEK(53254)<>0 THEN WRONGA=WRO
NGA+1:NOA=1:NA=1
5020 IF PEEK(53255)<>0 THEN WRONGB=WRO
NGB+1:NOB=1:NB=1
5025 K=250:POKE PMBASE+768+58+PEEK(179
0),0:GOSUB 7100
5028 IF NOA=1 THEN POSA=POSA+17
5029 IF NOB=1 THEN POSB=POSB+17
5030 FOR N=1 TO 8
5035 IF N>4 THEN N5=0:N6=0
5040 IF NOA=1 AND WRONGA<4 THEN COLOR
BLOCK(N)+N5:PLOT 5,WRONGA+4
5050 IF NOB=1 AND WRONGB<4 THEN COLOR
BLOCK(N)+N6:PLOT 14,WRONGB+4
5060 IF WRONGA=4 THEN SETCOLOR 2,0,16-
2*N:N1=160:N2=160:N3=0:GOSUB 8200
5070 IF WRONGB=4 THEN SETCOLOR 3,0,16-
2*N:N1=128:N2=192:N3=9:GOSUB 8200
5080 GOSUB 6000:NEXT N
5090 SOUND 0,0,0,0:SOUND 1,0,0,0
5095 IF WRONGA=4 THEN FOR T=8 TO 10:PO
SITION 4,T:? #6;"    ":NEXT T
5096 IF WRONGB=4 THEN FOR T=8 TO 10:PO
SITION 13,T:? #6;"    ":NEXT T
5100 IF WRONGA=4 OR WRONGB=4 THEN GOSU
B 6100
5150 RETURN
5400 IF PEEK(53279)=5 THEN POP :GOTO 9
000
5410 IF VAL(ANS$)=PEEK(SCREEN+4-PLACE+
180)-80 THEN GOSUB 6200:RETURN
5415 GOSUB 5500
5420 RETURN
5500 POSITION 2,0:? #6;"WRONG"
5505 POKE 53252,92:WRONGA=WRONGA+1:POK
E 206,A+3
5508 FOR N=100 TO 110:SOUND 0,N,6,15:N
EXT N:SOUND 0,0,0,0
5510 NOA=1:POKE 1764,14:POKE 1790,POSA
-PEEK(1786):POKE PMBASE+768+58+PEEK(17
86),3:R=USR(1759)
5515 GOSUB 5025
5530 RETURN
5800 IF PEEK(53279)=5 THEN POP :GOTO 9
000
5810 IF VAL(ANS$)=PEEK(SCREEN+20-PLACE
+180)-80 THEN GOSUB 6400:RETURN
5815 GOSUB 5900
5820 RETURN
5900 POSITION 11,0:? #6;"WRONG"
5905 POKE 53253,163:WRONGB=WRONGB+1:PO
KE 206,A+3
5908 FOR N=100 TO 110:SOUND 0,N,6,15:N
EXT N:SOUND 0,0,0,0
5910 NOB=1:POKE 1764,14:POKE 1790,POSB
-PEEK(1787):POKE PMBASE+768+58+PEEK(17
87),12:R=USR(1759):GOSUB 5025
5920 RETURN
6000 FOR J=K TO 0 STEP -10
6010 SOUND 0,J,14,15
6020 IF WRONGA=4 OR WRONGB=4 THEN SOUN
D 1,J,12,10
6030 NEXT J:K=ABS(K-50)
6040 RETURN
6100 FOR N=1 TO 25:NEXT N
6110 FOR N=14 TO 0 STEP -2
6120 IF WRONGA=4 THEN POKE 706,N
6130 IF WRONGB=4 THEN POKE 707,N
6140 FOR J=1 TO 10:NEXT J
6150 NEXT N
6155 IF WRONGA=4 THEN GOSUB 7200
6158 IF WRONBB=4 THEN GOSUB 7300
6160 RETURN
6200 SCOREA=SCOREA+10:POSITION 0,1:? #
6;SCOREA:POSITION 2,0:? #6;"RIGHT"
6210 N3=0:N1=160:N2=160:GOSUB 8800
6213 FOR N=100 TO 110:SOUND 0,N,6,15:N
EXT N:SOUND 0,0,0,0
```

```
6215 POKE 1790,118-PEEK(1786):POKE PMB
A5E+768+150,3:POKE 53252,84:POKE 206,A
+3
6218 POKE 1764,1
6225 R=USR(1759):POKE PMBASE+768+150-(
118-PEEK(1786)),0:GOSUB 7100
6240 L=86:VPOS=PEEK(1786)
6245 POKE 206,A+6:POKE 1764,14:POKE 17
90,5:FOR T=250 TO VPOS STEP -5
6250 SOUND 0,T,14,8:SOUND 1,T,12,8
6255 R=USR(1759):POKE 53250,L:L=L+1:NE
XT T:SOUND 0,0,0,0:SOUND 1,0,0,0
6259 FOR I=100 TO 200:SOUND 0,1,0,15-I
NT((I-100)/6.66):POKE 712,I:NEXT I:POK
E 53250,0
6279 SETCOLOR 4,0,0
6280 SOUND 0,0,0,0
6285 GOSUB 7200:GOSUB 3700
6295 RETURN
6300 IF PROBLEM=20 OR (PROBLEM=10 AND
PLAYERS=1) THEN GOSUB 7200:RETURN
6310 POSITION 2,0:? #6;"
":POSITION 6,3:? #6;"          "
6320 FOR N=7 TO 10:POSITION 0,N:? #6;"
    ":POSITION 16,N:? #6;"     ":NEXT N
6325 POKE 53278,0
6330 RETURN
6400 SCOREB=SCOREB+10:POSITION 17,1:?
#6;SCOREB:POSITION 11,0:? #6;"RIGHT"
6410 N3=9:N1=128:N2=192:GOSUB 8900
6413 FOR N=100 TO 110:SOUND 0,N,6,15:N
EXT N:SOUND 0,0,0,0
6415 POKE 1790,118-PEEK(1787):POKE PMB
A5E+768+150,3:POKE 53252,170:POKE 206,
A+3
6418 POKE 1764,1
6425 R=USR(1759):POKE PMBASE+768+150-(
118-PEEK(1787)),0:GOSUB 7100
6440 L=159:VPOS=PEEK(1787)
6445 POKE 206,A+7:POKE 1764,14:POKE 17
90,5:FOR T=250 TO VPOS STEP -5
6450 SOUND 0,T,14,8:SOUND 1,T,12,8
6455 R=USR(1759):POKE 53251,L:L=L-1:NE
XT T:SOUND 0,0,0,0:SOUND 1,0,0,0
6459 FOR I=100 TO 200:SOUND 0,1,0,15-I
NT((I-100)/6.66):POKE 712,I:NEXT I:POK
E 53251,0
6479 SETCOLOR 4,0,0
6480 SOUND 0,0,0,0
6485 GOSUB 7300:GOSUB 3800
6495 RETURN
7000 POKE 1780,96:FOR N=2 TO 7:POKE 20
4,A+N:R=USR(1770):NEXT N:RETURN
7100 POKE 1780,96:FOR N=2 TO 3:POKE 20
4,A+N:R=USR(1770):NEXT N:RETURN
7200 POKE 1780,96:POKE 204,A+6:R=USR(1
770):RETURN
7300 POKE 1780,96:POKE 204,A+7:R=USR(1
770):RETURN
7500 IF WRONGA=0 OR WRONGA=2 THEN N5=3
2
7510 IF WRONGA=1 THEN N5=64
7520 IF WRONGB=1 THEN N6=64
7530 IF WRONGB=0 OR WRONGB=2 THEN N6=3
2
7540 POKE 20,0:RETURN
8000 COLOR 62+N2:PLOT 5+N3,8
8020 COLOR 63+N2:PLOT 5+N3,9
8030 COLOR 4+N1:PLOT 5+N3,10
8050 COLOR 5+N1:PLOT 4+N3,9
8060 COLOR 6+N1:PLOT 6+N3,9
8070 RETURN
8100 COLOR 5+N1:PLOT 4+N3,9
8110 COLOR 6+N1:PLOT 6+N3,9
8120 COLOR 32:PLOT 4+N3,10
8130 COLOR 4+N1:PLOT 5+N3,10
8140 COLOR 32:PLOT 6+N3,10
8160 RETURN
8200 COLOR 7+N1:PLOT 4+N3,9
8210 COLOR 8+N1:PLOT 6+N3,9
8220 COLOR 54+N2:PLOT 4+N3,10
8230 COLOR 31+N1:PLOT 5+N3,10
8240 COLOR 53+N2:PLOT 6+N3,10
8250 RETURN
8300 COLOR 32:PLOT 4+N3,8
8310 COLOR 9+N1:PLOT 4+N3,9
8320 COLOR 10+N1:PLOT 6+N3,9
8330 COLOR 32:PLOT 6+N3,8
8340 COLOR 32:PLOT 7+N3,10
8350 RETURN
8600 GOSUB 8300:GOSUB 8100
8610 COLOR 9+N1:PLOT 4+N3,9
8620 RETURN
8700 GOSUB 8300:GOSUB 8100
8710 COLOR 10+N1:PLOT 6+N3,9
8730 RETURN
8800 GOSUB 8300:GOSUB 8100
8810 COLOR 12+N1:PLOT 4+N3,9
8820 COLOR 42+N2:PLOT 4+N3,8
8830 RETURN
8900 GOSUB 8300:GOSUB 8100
8910 COLOR 14+N1:PLOT 6+N3,9
8920 COLOR 49+N2:PLOT 6+N3,8
8930 RETURN
9000 GOSUB 7000:GOSUB 9500:POKE 53277,
0:POKE 559,34:GOTO 500
9120 FOR N=2 TO 7:POKE 204,A+N:R=USR(1
173):NEXT N
9500 POKE 53248,0:POKE 53249,0:POKE 53
250,0:POKE 53251,0:RETURN
```

•

CHECKSUM DATA.

(see page 43)

```
1 DATA 554,76,282,501,562,89,880,675,3
7,582,943,479,276,121,724,6781
110 DATA 761,499,512,521,527,551,541,5
32,543,341,541,807,958,328,513,8475
240 DATA 308,313,746,301,807,669,82,48
2,3,861,354,994,49,489,626,7084
305 DATA 776,48,778,179,783,836,875,24
3,63,796,897,574,304,803,668,8623
452 DATA 829,974,930,813,42,707,782,72
5,385,754,350,405,106,255,639,8696
515 DATA 445,709,482,669,580,218,986,5
69,802,328,347,329,345,758,550,8117
598 DATA 752,82,85,373,704,864,615,918
,57,5,244,605,395,742,916,7357
780 DATA 617,363,830,855,717,786,612,8
87,449,584,389,942,176,126,943,9276
1080 DATA 184,152,462,384,217,938,242,
290,452,451,451,196,197,622,617,5855
1177 DATA 635,626,934,784,56,197,193,3
64,182,511,220,444,5,972,942,7065
2001 DATA 530,3,25,705,279,985,989,70,
950,227,939,507,4,133,270,6616
2070 DATA 267,324,140,367,722,965,24,4
89,905,821,8,30,174,708,902,6846
2114 DATA 359,131,953,230,942,510,282,
332,332,699,400,735,938,835,909,8587
2204 DATA 653,11,33,177,711,905,419,70
0,81,956,233,945,513,292,767,7396
2273 DATA 704,403,739,760,103,907,305,
31,415,439,584,392,381,369,451,6983
2570 DATA 472,559,223,63,888,527,402,4
27,902,741,486,499,807,105,551,7652
2920 DATA 911,744,805,298,810,789,802,
792,925,697,681,791,178,807,200,10230
3365 DATA 287,679,800,954,945,232,511,
37,820,455,688,767,176,351,618,8320
3570 DATA 805,455,945,406,118,806,648,
325,297,921,913,674,809,830,322,9274
3810 DATA 314,937,929,880,812,868,844,
857,367,184,194,169,147,273,463,8238
5060 DATA 635,654,131,739,103,512,498,
795,986,831,965,801,911,885,140,9586
5510 DATA 395,969,805,998,979,985,813,
157,78,152,763,816,224,411,276,8821
6030 DATA 934,793,551,178,40,45,531,51
7,13,13,798,365,373,124,269,5544
6218 DATA 184,858,478,89,433,687,351,6
77,222,216,814,371,995,139,27,6541
6330 DATA 801,375,415,130,599,190,865,
301,98,439,698,360,683,228,225,6407
6495 DATA 820,961,960,141,145,972,487,
492,982,75,672,678,743,501,507,9136
8070 DATA 800,499,505,184,746,192,802,
504,510,500,494,506,804,368,510,7924
8320 DATA 677,377,201,807,226,519,813,
229,688,817,232,689,693,820,235,8023
8910 DATA 702,716,823,361,473,857,3932
```

ALOG COMPUTING **PRESENTS**

HIGH QUALITY PROGRAMS AT AFFORDABLE PRICES FOR
ALL ATARI COMPUTERS WITH 48K AND 1 DISK DRIVE.

SUMMER SPECIAL — BUY TWO & GET ONE FREE!*

*** WHEN ORDERING DIRECT FROM ALOG**

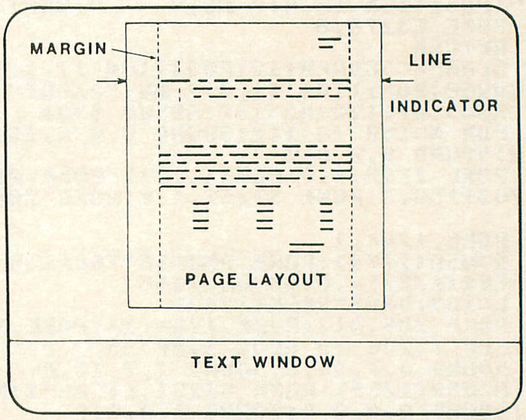
★★★★ NEW! ★★★★
THE ALOG CARDFILE

TURNS YOUR ATARI INTO AN ELECTRONIC CARD FILE. SEARCH & SORT ON ANY LINE. FORMATTED PRINT AND NUMERIC FIELDS MAKE THE CARDFILE IDEAL FOR YOUR EXPENSE RECORDS, INVENTORIES, AND RECEIPES.

```
ALOG COMPUTING PRODUCT CATALOG
***********************************************

PROGRAM     > THE_ALOG_CARDFILE__________
PROGRAMMER    DANIEL_F._HIGGINS__________
DATE          NOVEMBER_1983______________
PRICE         39.95______________________
COMPUTERS     ATARI_800,_800XL,_1200_____
REQUIRED      48K_&_1_DISK_______________
OPTIONS       2ND_DISK,_PRINTER__________
COMMENTS      TURNS_ATARI_COMPUTERS______
              INTO_AN_EASY-TO-USE________
              ELECTRONIC_CARDFILE________

    CARD # 4                4 CARDS TOTAL

              COMMAND MODE
 <E>DIT, <L>ABLE, <S>EARCH, <P>RINT
 <CTRL><S>ORT, <T>OTAL, <O>THER,<Q>UIT
```

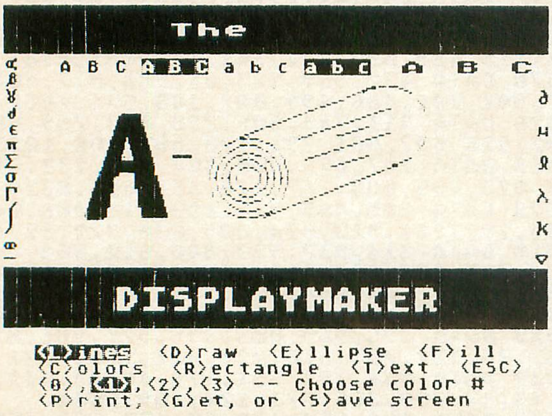
THE ALOG PAGEWRITER

TURNS YOUR ATARI AND 80 COLUMN PRINTER INTO AN EASY-TO-USE ELECTRONIC TYPEWRITER. THE ENTIRE PAGE LAYOUT IS DISPLAYED GRAPHICALLY WHILE YOU TYPE. IDEAL FOR SIMPLE WORD PROCESSING TASKS.



THE ALOG DISPLAYMAKER

MIXES COLOR GRAPHICS AND THREE TEXT SIZES ON THE SAME SCREEN. BUILT IN CHARACTER EDITOR AND SHAPE DRAWING ROUTINES. GRAPHICS DUMP TO MANY PRINTERS. GREAT FOR MAKING GRAPHS, CHARTS, AND HAVING FUN.



THE ALOG MAILLIST

IS AN EASY-TO-USE, MENU-DRIVEN DATA BASE PROGRAM DESIGNED FOR MAILING LISTS AND PRINTING LABELS. FAST 3 KEY SORT AND MULTI-LEVEL SEARCH ON ANY LINES. PRINT FORM LABELS 1 TO 2 INCHES HIGH.

```
           THE ALOG MAILLIST
  (C) Copyright 1983 by ALOG COMPUTING
----------------------------------------
   M A I N - M E N U

   <U>TILITIES

   <L>OAD FILE
   <S>AVE FILE
   <A>PPEND FILE
   <F>ILENAMES

   <C>REATE NEW RECORDS
   <E>DIT/VIEW RECORDS
   <D>ELETE MEMORY
   <P>RINT RECORDS

   <X>IT PROGRAM
----------------------------------------
 HIT <KEY> FOR OPTION:
```

All programs are only $39.95 each including tax. To order direct send check or money order to **Alog Computing, 1040 Veronica Springs Road, Santa Barbara, CA 93105.** We pay shipping. For information, credit card orders or C.O.D., call our distributor: **COMSTAR (805) 964-4660.**

Dealer and distributor inquiries invited.

CIRCLE #110 ON READER SERVICE CARD

Two Printer Interfaces for the Atari

MPP-1150
MICROBITS PERIPHERAL PRODUCTS
225 W. Third Street
Albany, OR 97321
(503) 967-9075
$99.95

APE-FACE
DIGITAL DEVICES CORPORATION
430 Tenth St., Suite N205
Atlanta, GA 30318
(404) 872-4430
$89.95

by Charles Bachand

You are at your local computer store and have just spent practically your last dollar buying that new whiz-bang parallel printer that everyone is talking about. You know, the one that will do everything but play "The Star Spangled Banner"? Anyway, you're just about ready to walk out the door when suddenly the salesman yells over to you, 'Oh, by the way, you *do* have an interface module for that printer, don't you?"

Interface module? Oh oh. . .an interface module! You had forgotten all about that, hadn't you? So you say — somewhat clumsily — "Oh. . .yeah, I need one of those, don't I? Um. . .how much are they?" His answer — over $200 retail for an Atari 850 — is way out of your means at the present time, and you start becoming visibly worried at the prospect of not being able to use your new toy. "Look," he says to you, "if all you need is a printer interface, and if about $100 retail won't break you, you can get one of those **MPP-1150**s or an **Ape-Face**!"

The **MPP-1150** is an interesting little device. It measures a mere 4⅜" × 3¼" × 1½" and contains only one 40-pin IC chip (yes, I'm a sucker when it comes to taking things apart!), but there are two sockets on the PC board. The extra socket can hold an optional 2K printer buffer chip that is available from MPP. A nice little option, if you ask me, one that will surely come in handy. It also has an Atari serial I/O connector to tie your disk drive, cassette recorder, etc. onto (it doesn't have to be the last device in the chain), and a three-foot ribbon cable with a Centronics-compatible 36-pin connector on the business end.

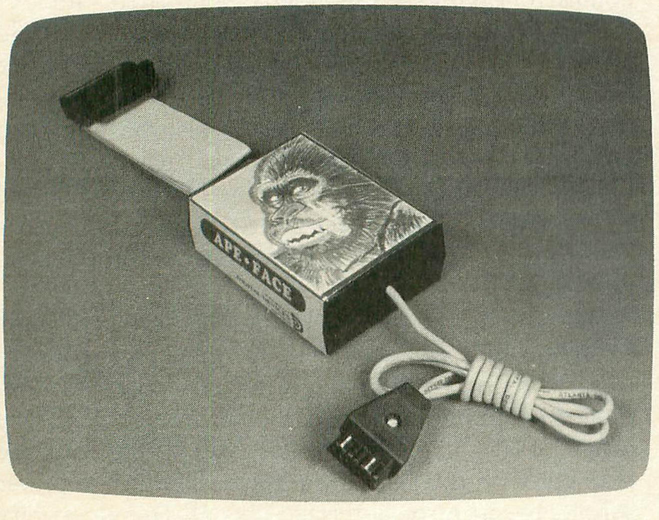
Hooking up the **MPP-1150** is very simple — practically nothing can go wrong! Just insert the connector at the end of the cable into your computer and plug the ribbon cable into your printer. If you have other Atari-compatible devices, they plug into the connector on the printer interface. There is no power supply, because the interface gets all the electricity it needs directly from the computer. That's all there is to it! Oh, by the way, MPP warranties the **1150** for two full years.

Splitting hairs.

The **Ape-Face** gives you the same song and dance as to size and operation, but there are some internal differences. Unfortunately, these two boxes perform the exact same operation and do it equally well, so well that I feel I am starting to split hairs in this review of the two models. Nevertheless, here goes. While the **Ape-Face** interface is ten dollars less than the one made by MPP and, internally, seems to be of better construction (solder masked PC board holding three IC chips and a voltage regulator), I prefer the **MPP** for several reasons.

1. Extra I/O Connector — The **Ape-Face** has only one Atari connector associated with it. If you also have a 410 cassette recorder in your system, you are in big trouble! Either you can use the recorder or you can use the interface, but not both. The reason is that connecting the **Ape-Face** leaves no place to plug in the recorder.

MPP-1150.



Ape-Face.

2. Cable Length — Both of the units have a multi-connector ribbon cable running from the box to the 36-pin connector that mates with your printer. The **Ape-Face**'s cable is only 18 inches long, while the **MPP**'s cable is 36 inches long. Some people might say that longer cables tend to pick up noise. However, I have used ten foot lengths in similar applications with no problems at all. Ribbon cable, in fact, tends to reduce noise pickup, since every other wire in the cable is by definition a ground wire.

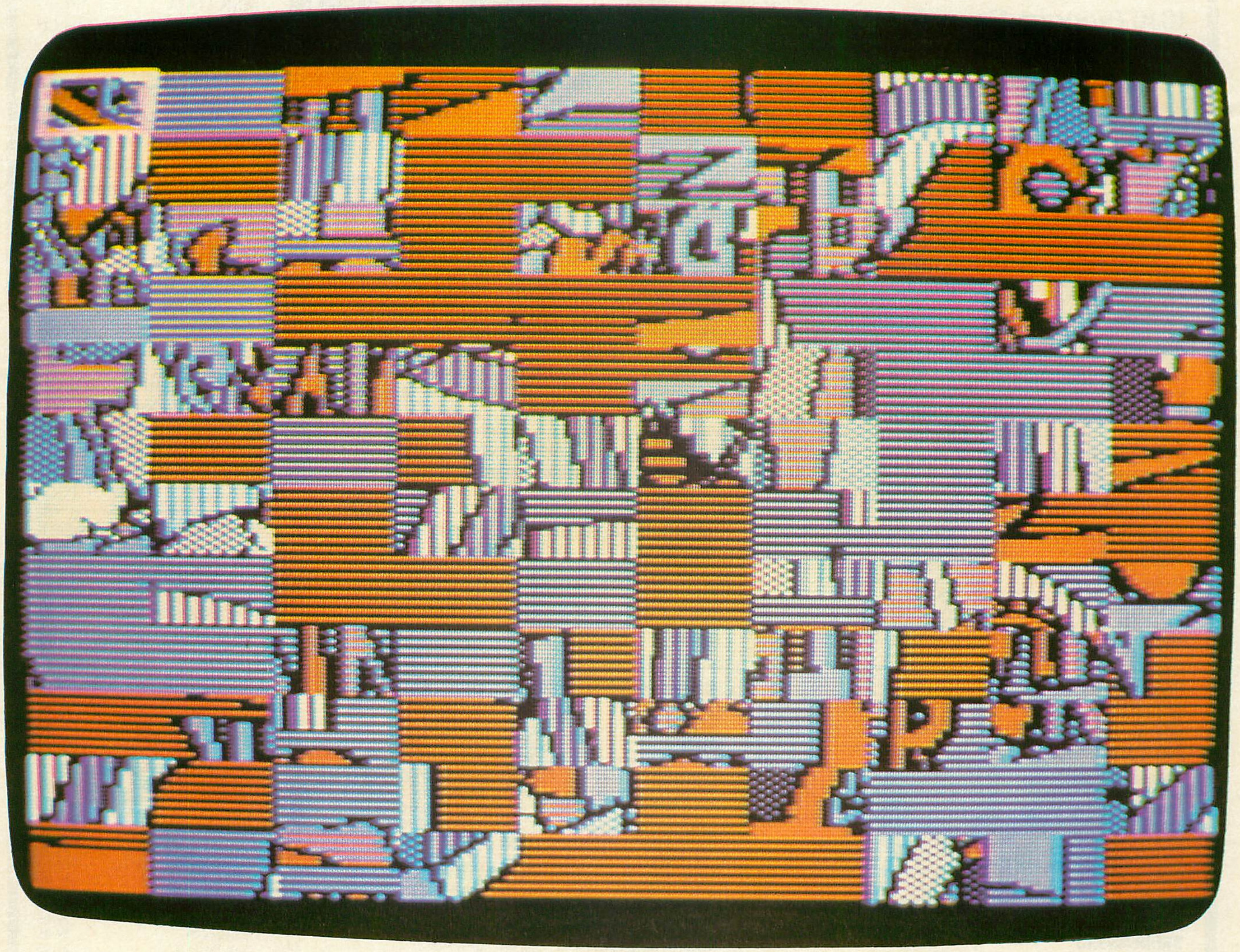
3. Power Requirements — Again, both of these units take their power from the Atari computer, but there seems to be a significant difference in the quantity of electrical current required. The **MPP-1150** needs to power only one large-scale integrated circuit chip, while the **Ape-Face** needs power for four. Not only that, but the **Ape-Face** comes in two models — one for the 1200XL and another for all the rest. With the **MPP**, one model fits all!

4. Printer Buffer — I think I mentioned this before . . .The MPP interface has the capability to accept a 2K print buffer. All that is needed is to insert a memory chip (available from MPP) into the extra socket on the interface. Again, no can do on the **Ape-Face**. Once you have used a printer buffer, it is hard to imagine having gone without one!

The big finish.

Both the **MPP-1150** and the **Ape-Face** seem to be well designed and worth the money. I prefer the **MPP**, even though the retail is ten dollars more. It is also more appealing aesthetically — a bright blue box wins over a black box with a monkey graphic on top every time!

Oh, I almost forgot! If for some reason, you have a serial printer instead of the more common parallel, the **MPP**'s interface will handle it with a plug-in chip. There is obviously not much call for it within the Atari community, but it can come in handy if your printer is somewhat of an orphan. □



Micro-Puzzler

48K Disk

by Larry G. Hearin

Hey, all you **Micro-Painter** owners! Have you ever wondered what you could do with all those beautiful screens you created, other than just *look* at them? Well, now there's **Micro-Puzzler**! This program will load a **Micro-Painter** screen file, divide it into 120 pieces, mix and rotate them, and then let you try to put it back together again — much like a jigsaw puzzle. As you may have guessed, the difficulty will be (mostly) determined by the complexity of the picture, so you can choose your own level by the screen you use.

Running the program.

When you run the program, there will be a few seconds of initialization, after which you will be prompted by the words ENTER SCREEN FILENAME. You may now enter the name of the **Micro-Painter** screen file that you want to use with **Micro-Puzzler**. If no device specification is given, disk drive 1 is assumed. If an error is encountered in trying to access this file, the program will return to the prompt for the screen filename. Instead of entering the screen filename, you may get a disk directory listing by hitting CTRL-D and then entering the drive number for which you want the directory. After listing the disk directory, the program will return to the prompt for the screen filename.

Once a valid screen filename is entered, the screen will be loaded, and the puzzle pieces will be shuffled and rotated. Then the new, mixed-up screen will be displayed, along with a rectangular cursor in the upper left of the screen. Puzzle pieces are moved by exchanging positions of two pieces at a time. To do this, move the cursor (using the arrow keys) to one of the pieces you want to exchange and hit RETURN. Then move the cursor to the other piece to be exchanged and hit RETURN. While you're moving from the first to the second piece, a secondary cursor will be left at the first position to mark the piece to be exchanged. After the exchange is made, the secondary cursor will disappear.

The only other type of puzzle piece manipulation that may be done is rotation. This may be done at any time by pressing the R key. This will always rotate the piece within the primary cursor. An interesting and sometimes helpful phenomenon to note is that some of the colors of a puzzle piece may change when the piece is rotated. So, if you see a color that isn't on the original picture, chances are that piece is upside-down.

For those of you who don't remember exactly what the original picture looked like, you may press the Atari key to toggle between the original and the mixed-up screen.

Once the picture is correct, you will be congratulated and may then press the ESC key to run again. To quit, you must hit SYSTEM RESET.

If you get tired of puzzling before you complete the picture, you may save your current status on disk, if desired. To do this, press the OPTION key, and then enter a disk filename to which the status will be saved. *Warning*: this file has to be saved to the same disk containing the original screen file. Otherwise, when you try to reload your status, it will not work. To reload, just enter this status filename instead of the original **Micro-Painter** filename when prompted with ENTER SCREEN FILENAME.

Summary.

Datasoft's **Micro-Painter** is an excellent graphics program for the Atari. And, by using **Micro-Puzzler**, you can get even more enjoyment out of your **Micro-Painter**. □

(continued on next page)

Micro-Puzzler.
Basic listing 1.

```
10 DIM FILE$(17),NAM$(17),T$(1),PCS$(1
20),PLAY$(8),RFLG$(120)
20 GRAPHICS 17:POSITION 3,2:PRINT #6;"
MICRO-PUZZLER":POSITION 1,5:PRINT #6;"
by larry g. hearin"
30 POKE 16,64:POKE 53774,64:IF PEEK(76
4)=28 THEN FOR I=1 TO 500:NEXT I:POKE
764,255:GOTO 50
40 POSITION 1,13:PRINT #6;"PLEASE STAN
D BY...":GOSUB 1030:REM Load ASM routi
nes
50 POKE 559,34:GRAPHICS 0:POKE 16,64:P
OKE 53774,64:TRAP 50:POSITION 3,2
60 ? "ENTER SCREEN FILENAME":? "(CTRL-
D FOR DISK DIRECTORY)":? :? "?";:CLOSE
 #2
70 IF PEEK(764)=255 THEN 70
80 IF PEEK(764)<>186 THEN ? CHR$(30);:
INPUT FILE$:GOTO 110
90 POKE 764,255:FILE$="D1:*.*":? "ENTE
R DRIVE # ";:INPUT T$:FILE$(2,2)=T$:TR
AP 31:OPEN #2,6,0,FILE$
100 TRAP 60:INPUT #2,FILE$:? FILE$:GOT
O 100
110 TRAP 60:IF FILE$(2,2)=":" THEN SN=
3:GOTO 130
115 IF FILE$(3,3)=":" THEN SN=4:GOTO 1
30
120 FOR I=LEN(FILE$) TO 1 STEP -1:FILE
$(I+2,I+2)=FILE$(I,I):NEXT I:FILE$(1,2
)="D:":SN=3
130 GRAPHICS 24:POKE 16,64:POKE 53774,
64:DL=PEEK(560)+PEEK(561)*256
140 SCR=PEEK(DL+4)+PEEK(DL+5)*256:SCR2
=SCR-8*1024:SCR2HI=PEEK(DL+5)-32
150 POKE DL+3,78:K=DL+6
160 SOUND 0,K,10,8:IF PEEK(K)=65 THEN
200
170 IF PEEK(K)=15 THEN POKE K,14
180 IF PEEK(K)=79 THEN POKE K,78:K=K+2
:KSAV=K
190 K=K+1:GOTO 160
200 SOUND 0,0,0,0:CLOSE #1:TRAP 50:OPE
N #1,6,0,FILE$
201 REP=0:INPUT #1,NAM$:IF NAM$(15,17)
="062" THEN 209
202 CLOSE #1:OPEN #1,4,0,FILE$:INPUT #
1;NAM$:INPUT #1;PCS$:INPUT #1;RFLG$:RE
P=1
203 FILE$(SN)=NAM$
209 CLOSE #1:TRAP 50:OPEN #1,4,0,FILE$
210 POKE 559,0:X=USR(ADR(LODE$)):IF X<
>1 THEN 50
220 GET #1,C12:POKE 712,C12:GET #1,C8:
POKE 708,C8:GET #1,C9:POKE 709,C9:GET
#1,C10:POKE 710,C10
230 CLOSE #1:POKE 559,34
235 IF REP=1 THEN 270
240 FOR I=1 TO 120:PCS$(I,I)=CHR$(I):N
EXT I
250 FOR I=1 TO 120:J=INT(RND(0)*120)+1
:T$=PCS$(J):PCS$(J,J)=PCS$(I,I):PCS$(I
,I)=T$
260 J=INT(RND(0)*100)+1:RFLG$(I,I)=STR
$((J>50)*1):NEXT I
270 FOR C=1 TO 10:FOR R=1 TO 12:X2=C:Y
2=R:K=(R-1)*10+C:Y1=INT((ASC(PCS$(K,K)
)-1)/10)+1
280 X1=ASC(PCS$(K,K))-(Y1-1)*10:SOUND
0,RND(0)*10+10,10,8:GOSUB 940
290 IF RFLG$(K,K)="1" THEN ADD1=ADD2+1
5*40+3:X=USR(ADR(ROT$),ADD2,ADD1)
300 NEXT R:NEXT C:SOUND 0,0,0,0
310 INC=32:POKE DL+5,PEEK(DL+5)-32:POK
E KSAV,PEEK(KSAV)-32
320 PMBASE=INT((SCR2HI-4)/4)*4:POKE 54
279,PMBASE:PMB=PMBASE*256:TRAP 40000
330 RESTORE 340:FOR I=1 TO 8:READ J:PL
AY$(I,I)=CHR$(J):NEXT I
340 DATA 255,129,129,129,129,129,129,2
55
350 FOR I=PMB+512 TO PMB+767:POKE I,0:
NEXT I:C1=0:POKE 623,1
360 POKE 559,46:POKE 53277,3:POKE 5324
8,48:POKE 53249,0:XP1=1:YP1=16:POKE 53
256,XP1:POKE 53257,XP1:YP2=YP1
370 FOR I=1 TO 8:POKE PMB+511+YP1+I,AS
C(PLAY$(I,I)):POKE PMB+639+YP1+I,ASC(P
LAY$(I,I)):NEXT I
380 SOUND 0,0,0,0:A=PEEK(764):C1=C1+1-
(C1=255)*256:POKE 704,C1:POKE 705,255-
C1:I=PEEK(623)
390 POKE 623,(I=1)*4+(I=4):IF PEEK(532
79)=3 THEN 1020
395 IF A=255 THEN 380
400 POKE 764,255:POKE 623,1
410 IF INC<0 THEN 540
420 SOUND 0,40,10,8
430 IF A<>6 THEN 450
440 XP1=XP1-1:GOSUB 970:POKE 53248,48+
16*(XP1-1)
450 IF A<>7 THEN 470
460 XP1=XP1+1:GOSUB 970:POKE 53248,48+
16*(XP1-1)
470 IF A<>15 THEN 500
480 YINC=8:GOSUB 1000
490 GOTO 520
500 IF A<>14 THEN 540
510 YINC=-8:GOSUB 1000
520 ADD2=PMB+512+YP1:ADD1=ADD2+YINC:X=
USR(ADR(MOV$),ADD1,ADD2,8,1,1):X=USR(A
DR(MOV$),ADR(PLAY$),ADD1,8,1,1)
530 YP1=YP1+YINC
540 IF A<>40 THEN 570
550 X1=XP1:Y1=INT((YP1-16)/8+1.5):ADD1
=SCR2+(X1-1)*4+(Y1-1)*640:ADD2=ADD1+15
*40+3
560 X=USR(ADR(ROT$),ADD1,ADD2):K=(Y1-1
)*10+X1:RFLG$(K,K)=STR$(1-(VAL(RFLG$(K
,K))=1)*1):GOTO 790
570 IF A<>39 THEN 630
580 SOUND 0,100,10,8
590 POKE DL+5,PEEK(DL+5)+INC:POKE KSAV
,PEEK(KSAV)+INC:INC=-INC
600 IF INC>0 THEN POKE 53248,48+16*(XP
1-1):POKE 53249,48+16*(XP2-1)-(XP2=0)*
32
610 IF INC<0 THEN POKE 53248,0:POKE 53
249,0
620 GOTO 380
630 IF A<>12 OR INC<0 THEN 380
640 SOUND 0,200,10,8
650 IF FLG=1 THEN 680
660 FOR I=0 TO 7:POKE PMB+640+YP2+I,0:
POKE PMB+640+YP1+I,ASC(PLAY$(I+1,I+1))
:NEXT I:YP2=YP1
670 XP2=XP1:POKE 53249,48+16*(XP2-1):F
LG=1:GOTO 380
680 X1=XP1:Y1=INT((YP1-16)/8+1.5)
690 X2=XP2:Y2=INT((YP2-16)/8+1.5):K=(Y
1-1)*10+X1:Y1=INT((ASC(PCS$(K,K))-1)/1
0)+1:X1=ASC(PCS$(K,K))-(Y1-1)*10
700 N=K:GOSUB 940
710 X1=XP2:Y1=INT((YP2-16)/8+1.5)
720 X2=XP1:Y2=INT((YP1-16)/8+1.5):K=(Y
1-1)*10+X1:Y1=INT((ASC(PCS$(K,K))-1)/1
0)+1:X1=ASC(PCS$(K,K))-(Y1-1)*10
730 AD2=ADD2
740 GOSUB 940:I=ASC(PCS$(K,K)):PCS$(K,
K)=PCS$(N,N):PCS$(N,N)=CHR$(I)
750 IF RFLG$(N,N)="1" THEN ADD1=AD2+15
*40+3:X=USR(ADR(ROT$),AD2,ADD1)
760 IF RFLG$(K,K)="1" THEN ADD1=ADD2+1
5*40+3:X=USR(ADR(ROT$),ADD2,ADD1)
770 T$=RFLG$(N,N):RFLG$(N,N)=RFLG$(K,K
):RFLG$(K,K)=T$
780 XP2=0:POKE 53249,0:FLG=0
790 X=USR(ADR(CMP$),SCR,SCR2,7680):IF
X=1 THEN 380
800 POKE 53248,0:POKE 53249,0
810 ADD1=10:FOR I=0 TO 12:FOR J=0 TO 8
:SOUND 0,200-ADD1*I,10,8:ADD1=-ADD1:FO
R K=1 TO 2:NEXT K:NEXT J:NEXT I
820 GOSUB 1012
860 POSITION 2,2:PRINT #6;"CONGRATULAT
IONS":POSITION 4,5:PRINT #6;"you did i
t!"
870 POSITION 4,8:PRINT #6;"PRESS ESC"
:POSITION 3,9:PRINT #6;"TO RUN AGAIN"
```

```
880 SOUND 0,100,10,8:POKE 559,0:POKE 5
60,0:POKE 561,PEEK(DL2+19):POKE 712,0:
POKE 708,40:POKE 709,202
890 POKE 710,148:POKE 559,34:FOR I=1 T
O 75:NEXT I:IF PEEK(764)=28 THEN SOUND
 0,0,0,0:GOTO 20
900 SOUND 0,200,10,8:POKE 559,0:POKE 5
61,INT(DL/256):POKE 560,DL-PEEK(561)*2
56:POKE 712,C12:POKE 708,C8
910 POKE 709,C9:POKE 710,C10:POKE 559,
34:FOR I=1 TO 75:NEXT I:IF PEEK(764)=2
8 THEN SOUND 0,0,0,0:GOTO 20
920 GOTO 880
930 REM Move card
940 ADD1=SCR+(X1-1)*4+(Y1-1)*640:ADD2=
SCR2+(X2-1)*4+(Y2-1)*640
950 X=USR(ADR(MOV$),ADD1,ADD2,4,16,40)
960 RETURN
970 IF XP1>10 THEN XP1=1:RETURN
980 IF XP1<1 THEN XP1=10
990 RETURN
1000 IF YINC+YP1>104 THEN YINC=16-YP1:
RETURN
1010 IF YINC+YP1<16 THEN YINC=104-YP1
1011 RETURN
1012 POKE 53248,0:POKE 53249,0
1013 DL2=(INT(SCR2/1024)+1)*1024:FOR I
=0 TO 2:POKE DL2+I,112:NEXT I:POKE DL2
+3,71
1014 POKE DL2+4,PEEK(DL+4):POKE DL2+5,
SCR2HI:POKE DL+5,SCR2HI+32:FOR I=6 TO
16:POKE DL2+I,7:NEXT I
1015 POKE DL2+17,65:POKE DL2+18,0:POKE
 DL2+19,INT(DL2/256)
1016 POKE 87,2:POKE 88,PEEK(DL2+4):POK
E 89,PEEK(DL2+5):POKE 559,0:POKE 560,0
:POKE 561,PEEK(DL2+19)
1017 FOR I=0 TO 11:POSITION 0,I:PRINT
#6;"                            ";:NEXT I:POKE
 559,34
1018 RETURN
1020 POKE 53248,0:POKE 53249,0:GRAPHIC
S 0:POKE 16,64:POKE 53774,64
1021 POSITION 2,2:? "must save to same
 disk as screen file":? "ENTER SAVE FI
LENAME"
1022 INPUT NAM$:IF LEN(NAM$)=0 THEN PO
KE 764,28:GOTO 20
1023 IF NAM$=FILE$ THEN ? :? "ERROR-TR
Y AGAIN":GOTO 1021
1024 CLOSE #2:TRAP 1020:OPEN #2,8,0,NA
M$:? #2;FILE$(SN):? #2;PCS$:? #2;RFLG$
:CLOSE #2:POKE 764,28:GOTO 20
1030 DIM LODE$(42):RESTORE 1090:FOR I=
1 TO 42:READ J:LODE$(I,I)=CHR$(J):NEXT
 I
1040 DIM MOV$(69):RESTORE 1120:FOR I=1
 TO 69:READ J:MOV$(I,I)=CHR$(J):NEXT I
1050 DIM CMP$(107):RESTORE 1160:FOR I=
1 TO 107:READ J:CMP$(I,I)=CHR$(J):NEXT
 I
1060 DIM ROT$(126):RESTORE 1220:FOR I=
1 TO 126:READ J:ROT$(I,I)=CHR$(J):NEXT
 I
1070 RETURN
1080 REM Load screen
1090 DATA 216,104,162,16,169,7,157,66,
3,169,0,157,72,3,169,30,157,73,3,165,8
8,157,68,3,165,89,157,69,3,32,86
1100 DATA 228,189,67,3,133,212,169,0,1
33,213,96
1110 REM Move card
1120 DATA 104,104,133,204,104,133,203,
104,133,206,104,133,205,104,104,133,20
7,104,104,133,208,104,104,133
1130 DATA 209,164,207,136,177,203,145,
205,136,16,249,198,208,240,29,165,203,
24,101,209,133,203,165,204
1140 DATA 105,0,133,204,165,205,24,101
,209,133,205,165,206,105,0,133,206,24,
144,213,96
1150 REM Compare memory
1160 DATA 104,104,133,204,104,133,203,
104,133,206,104,133,205,104,133,208,10
4,133,207,201,0,208,8,165,208
1170 DATA 201,0,240,4,198,208,198,208,
160,0,160,0,177,203,209,205,208,48,165
,207,201,0,240,31,198,207,165
1180 DATA 203,24,105,1,133,203,165,204
,105,0,133,204,165,205,24,105,1,133,20
5,165,206,105,0,133,206,24
1190 DATA 144,213,165,208,201,0,240,14
,198,208,24,144,214,169,1,133,212,169,
0,133,213,96,169,0,133,212
1200 DATA 133,213,96
1210 REM Rotate card
1220 DATA 104,104,133,204,104,133,203,
104,133,206,104,133,205,169,31,133,209
,169,3,133,210,160,0,162,8,177
1230 DATA 203,133,207,169,0,24,10,70,2
07,105,0,202,208,248,133,208,162,8,177
,205,133,207,169,0,24,10
1240 DATA 70,207,105,0,202,208,248,145
,203,165,208,145,205,165,209,201,0,240
,54,198,209,165,210,201,0,240
1250 DATA 35,169,1,133,207,198,210,165
,203,24,101,207,133,203,165,204,105,0,
133,204,165,205,56,229,207,133
1260 DATA 205,165,206,233,0,133,206,24
,144,165,169,37,133,207,169,3,133,210,
24,144,216,96
```

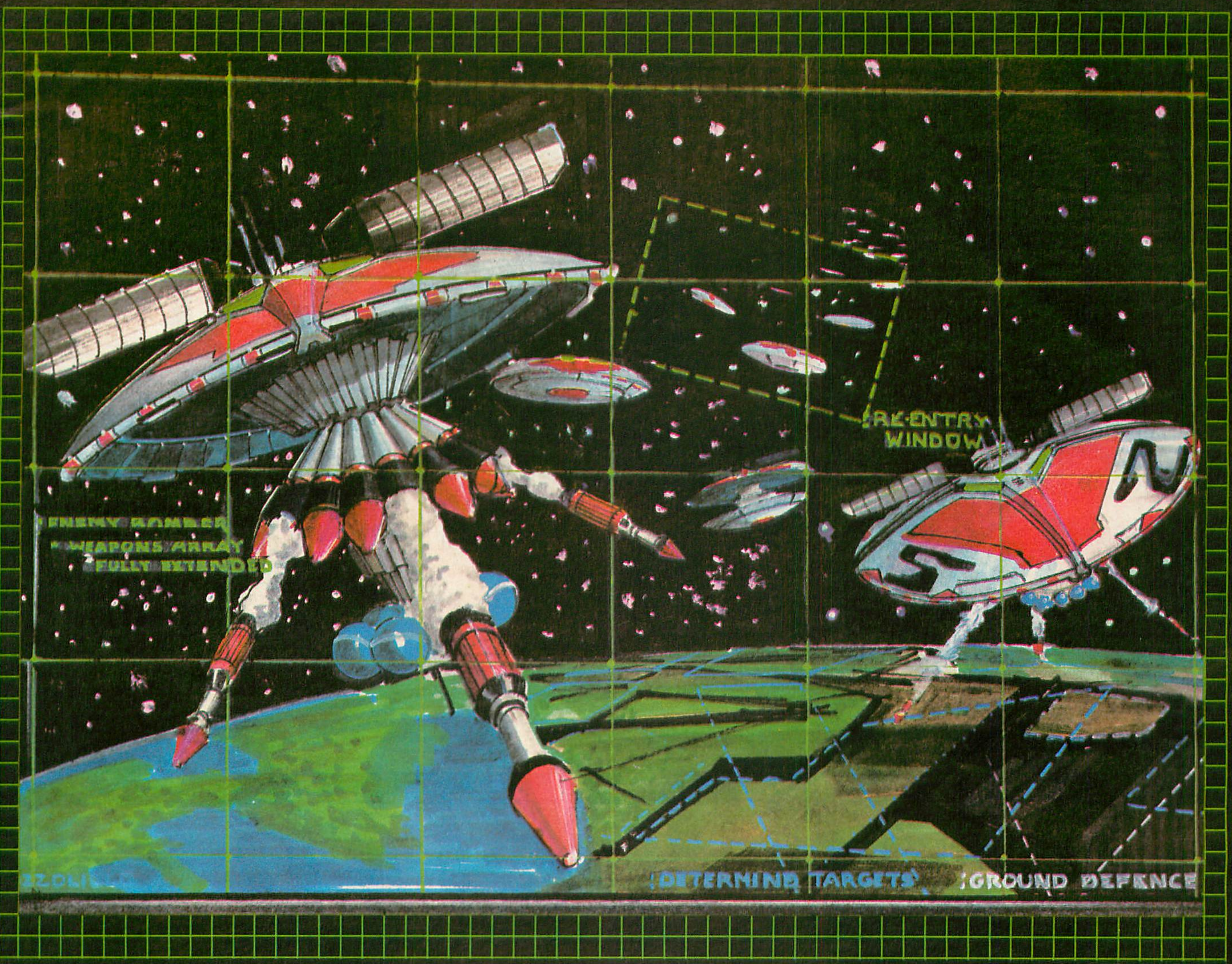
•

CHECKSUM DATA.

(see page 43)

```
10 DATA 134,525,257,210,914,214,367,35
7,897,678,975,146,514,268,965,7421
150 DATA 939,3,42,902,458,467,215,203,
494,644,114,929,85,845,700,7040
250 DATA 942,667,155,22,679,339,512,46
5,198,608,558,91,787,442,42,6507
395 DATA 731,364,787,573,457,845,470,8
49,713,866,730,705,152,96,512,8850
540 DATA 731,452,502,736,422,131,244,5
39,727,828,414,818,819,383,209,7955
690 DATA 839,236,194,816,686,364,545,6
80,669,844,245,477,193,815,382,7985
870 DATA 684,169,175,402,669,743,839,6
99,667,615,207,302,624,947,921,8663
1011 DATA 782,562,783,351,147,664,167,
796,870,541,126,40,332,822,744,7727
1050 DATA 693,774,786,294,93,959,917,6
28,456,694,971,670,715,326,408,9384
1200 DATA 275,369,935,316,683,950,103,
3631
```

•



Air Attack

24K Cassette or 32K Disk

by Scott Sheck

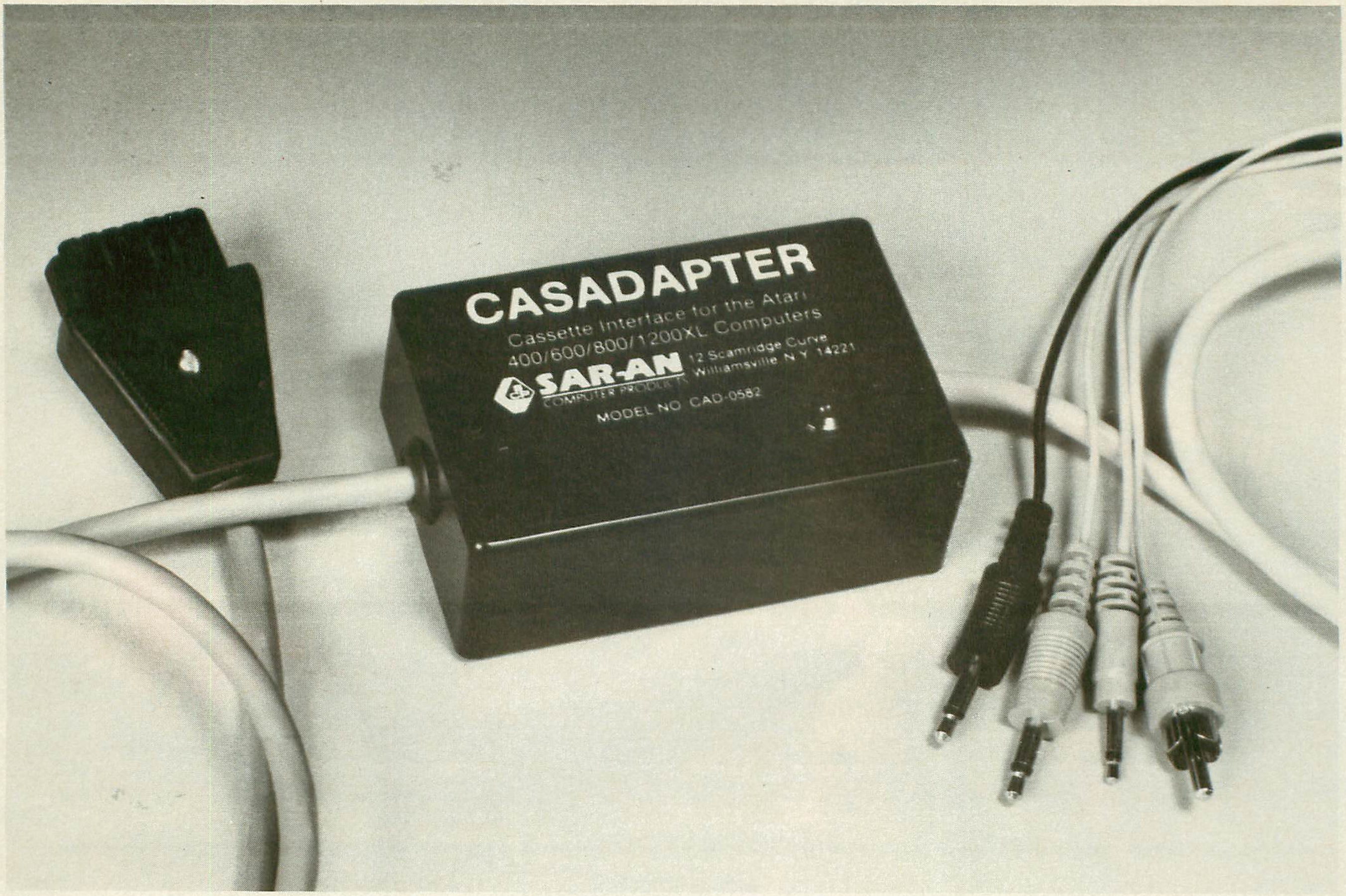
When the game begins the sky will be clear, but then a loud siren will sound, warning you of an **Air Attack**. You will begin to see missiles heading toward your central missile base and six missile factories. The only defense will be to fire your own high-speed missiles to intercept the oncoming ones. You are equipped with 30 missiles; however, if the enemy should bomb your missile base, you will be left defenseless.

Occasionally, an enemy craft will pass over your factories, dropping containers of explosive fuel. Should one of the containers hit a factory, it will explode. On the other hand, if the container hits the ground, the explosive fuel will spill out. It will be hazardous only if ignited by one of the enemy's missiles.

Periodically, enemy attacks will be suspended while you are replenished with missiles. The assault will start up again, but with much faster missiles than before, and enemy crafts which travel faster, with increased resistance to your interceptor missiles. The game ends when all of your missile factories have been destroyed.

Scoring.

Scoring will be as follows: enemy missiles — 5 points; enemy craft — 25 points; and fuel container — 50 points. Additional points: after each attack ceases, you will receive 100 points for each missile



CASADAPTER

THE ORIGINAL CASSETTE INTERFACE...

Allows you to use your own cassette recorder or stereo for loading and saving of programs. It makes your own home stereo or portable cassette player act just like an Atari 410 or 1010 recorder. CASADAPTER gives you motor control, audio, and data channels. The CASADAPTER plugs directly into the computer and requires no other interfaces. An L.E.D. shows the transfer of all data. CASADAPTER will work with all of the Atari brand computers.

$39.95

Add $2.50 shipping
Send check or money order
C.O.D. accepted
New York State Residents add 7% tax
®Atari is a trademark of Atari, Inc.

CIRCLE #114 ON READER SERVICE CARD

SAR-AN
COMPUTER PRODUCTS

12 Scamridge Curve
Williamsville, New York 14221
(716) 632-3441

factory that has remained standing and 5 points for each of your unused missiles. Bonus: every 2000 points, you will receive an extra missile factory.

Designing the game.

If you've never programmed a game, you might ask, "Where do you start in programming a game?" To answer this question, let's take a look at this game.

Step 1 — I first started by drawing the scenery (the non-moving objects). This included the missile base, the six missile factories, the ground, the interceptor missiles and the score. Printing the score (Line 25) involved modifying the display list, which description — for the sake of brevity — I have omitted here. The rest of the scenery was drawn by Lines 400-445 in graphics mode 7.

Step 2 — Next, I figured out what the moving objects were going to look like. Moving objects included the enemy missiles, the enemy craft, the fuel container and the aim. Before placing these objects on the screen, I had to find a way to make them all move at the same time. If you played the game and watched the objects move, it probably looked as if they were, indeed, all moving at the same time. Actually, they were not. Each object was taking turns moving.

To show how I did this, let's take as an example three objects labeled 1-3. To make it look as if all three objects are moving at the same time, I would do the following: a. move object 1; b. move object 2; c. move object 3; and d. go back to "a." Your computer could go through these steps so quickly that it would appear as if all three objects were moving simultaneously. To convince yourself of this RUN the short program below on your computer.

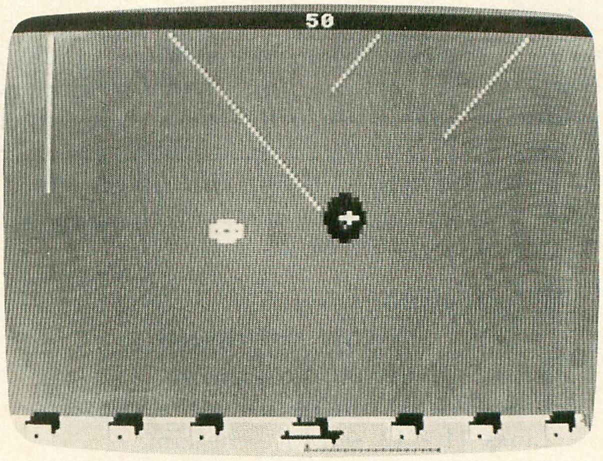
```
10 GRAPHICS 7:COLOR 1
20 A=1:B=1:C=1:REM starting point for
each object
30 PLOT 40,A:A=A+1:REM move object 1
40 PLOT 60,B:B=B+1:REM move object 2
50 PLOT 80,C:C=C+1:REM move object 3
60 GOTO 30
```

Now, suppose you wanted object 1 to move faster than the other lines. You would do this by adding the line below to the program:

```
25 PLOT 40,A:A=A+1
```

Step 3 — The next step I took was to detect collisions between the moving objects. I did this by using the two BASIC commands: COLOR and LOCATE. This is how I used the color registers: color 0 — sky (background); color 1 — ground; color 2 — enemy missiles; and color 3 — explosions, missile factories, missile base.

This is how I detected collisions (using the LOCATE command): enemy missile — if it touches color 3 then erase missile trail and place explosion there (Lines 120-130), if it touches color 1 then erase missile trail (Line 130); fuel container — if it touches color 1 then draw spilled fuel (Line 160), if it touches color 3 then erase container and place explosion there (Line 165); and enemy craft — if it touches color 3 then erase craft and place explosion there (Line 160).



Air Attack.

These are the steps I took when moving the objects in my game (Lines 50-96):

Lines 8 - 12 — Move the aim and check if the button is pressed.

Lines 55 - 60 — Move enemy craft if it should be moved and detect its collision.

Lines 80 and 150 - 200 — Move missile if it should be moved and detect its collision; move the aim.

Lines 100 - 136 — Move enemy missile corresponding to P in Line 50 and detect its collision.

Line 85 — Check if missile base is blown up. Go back to Step 1.

Setting up the game.

Type in the program and then SAVE it immediately before running the program. Only after you have saved the program, type RUN. The screen will then go blank for about fifteen seconds before the game begins.

This game requires an Atari with 24K of memory, however, you can play the game on a 16K Atari, if the title screen is removed. This is done by deleting the GOSUB 900 in Line 1011 and deleting Lines 899-995.

Final words.

What inspired me the most in writing this game was Tom Hudson's article, "Graphic Violence" (**ANALOG** issue 8). After seeing the demo that was included in the article, I was so impressed that I had to come up with a game using the explosions, so I used his routine in this game. I later wanted to include player/missile graphics in my game, and I found two very easy-to-use player (**ANALOG** issue 10) and missile (**ANALOG** issue 11) routines that Mr. Hudson had also written. Unfortunately, I couldn't use these routines due to the large size of my program. However, I would like to thank Mr. Hudson for showing me how to add the player routine to the G.V. routine.

In addition to player/missile graphics, I also have added two machine language routines which help speed up the game's action. These routines were written by D.K. Titchenell. One routine, which is stored in POK$ in my program, allows my program to make multiple POKEs. The other, stored in MOV$, allows fast movement of blocks of RAM to other areas of RAM. □

Program description.

Line 5 — Set up the scenery.
Lines 8 - 10 — Check joystick trigger.
Lines 11 - 12 — Move the aim.
Line 25 — Change the score.
Line 26 — Sound of spilled fuel.
Lines 50 - 96 — Main routine.
Lines 100 - 136 — Enemy missile movement and its collision detection.
Lines 150 - 200 — Fuel container movement and its collision detection.
Lines 300 - 319 — Data for the different enemy missile paths (used in Line 600).
Line 400 — Draw the ground.
Line 421 — Data for position of each missile factory.
Lines 425 - 430 — Draw the missile factories.
Line 434 — Draw the thirty interceptor missiles.
Lines 435 - 445 — Draw the central missile base.
Lines 450 - 460 — Joystick movement data.
Line 600 — Get a different enemy missile path.
Lines 700 - 842 — Set up players, missiles, POK$ and MOV$.
Lines 850 - 860 — Sound of the siren.
Lines 1000 - 1020 — Dimension variables and initialize values.
Lines 2850 - 2880 — Check for a free missile factory.
Lines 2900 - 3509 — Change difficulty level of each new attack.
Lines 4000 - 4100 — Count remaining mis-

Some program listings reproduced in **ANALOG** 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
```

sile factories and interceptor missiles.

Lines 4200 - 4205 — Is the game over?

Lines 10010 - 15230 — Graphic Violence initialization.

Program variables.

AMMO — Holds an even number from 80 to 144.

XAIM — P/M X-coordinate of the aim.

YAIM — P/M Y-coordinate of the aim.

XX() — Array holding joystick's X-direction.

YY() — Array holding joystick's Y-direction.

SCORE — Current score.

PNTS — Points to be added to current score.

P — Current enemy missile being referred to.

STREAMS — Number of enemy missiles for a particular attack.

YP1 — P/M Y-coordinate of the enemy craft.

PLYR — P/M X-coordinate of the enemy craft (220 - off the screen).

RAND — Value from 0 to 255.

PROB — Probability used to determine if craft or container should appear.

SPEED — Used to speed up craft and enemy missiles.

M1 — If M1=1 then container has been dropped, else it's not dropped.

MX — Gr. 7 X-coordinate of container.

MY — Gr. 7 Y-coordinate of container.

X(P) — X-position of enemy missile P.

Y(P) — Y-position of enemy missile P.

SLOPE(P) — Slope of enemy missile P's path.

START(P) — X-coordinate of where enemy missile P began.

STCNT — Number of enemy missiles destroyed.

STLIMIT — Limit on the number of enemy missiles for that attack.

MISPD — Used to speed up container speed.

ATTACK — Difficulty level.

B$ — Determines if a missile factory is still standing (e.g., if B$="111111" then all six factories are still standing).

S — Stick direction (5 - 15).

PL1$ — Shape of enemy craft.

AIM$ — Shape of the aim.

MOV$ — Machine language routine which moves blocks of RAM.

POK$ — Machine language routine which allows multiple POKEs.

FREECITY — Value which signals when to give a free missile factory.

WHAT IS D:CHECK/C:CHECK?

Most program listings in **ANALOG** 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 and C:CHECK, which appeared in the **ANALOG Compendium** and Issue No. 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 either article, send for a copy of back issue 16 ($4.00) or **The ANALOG Compendium** ($14.95 plus $2.00 shipping and handling) from:

ANALOG Computing
P.O. BOX 615
HOLMES, PA 19043

```
1 GRAPHICS 2+16:POSITION 5,3:? #6;"air
 attack":? #6:? #6;"    by: scott sheck
":FOR D=1 TO 1000:NEXT D
5 GOSUB 1000:GOTO 50
8 IF AMMO<79 OR PEEK(644)=C1 THEN 11
9 I=XAIM-45:T=YAIM-14:A=USR(ADR(EXPL$)
,I,T):COLOR C1:PLOT AMMO,94:AMMO=AMMO-
C2
10 PLOT 81,87:DRAWTO I,T:COLOR CZ:DRAW
TO 81,87:DRAWTO I,T
11 XAIM=XAIM+XX(S)*6:XAIM=XAIM+((XAIM<
50)-(XAIM>197))*6:YAIM=YAIM+YY(S)*4:YA
IM=YAIM+((YAIM<24)-(YAIM>92))*4
15 S=PEEK(632):POKE 53249,XAIM:A=USR(A
DR(MOV$),ADR(AIM$),PMBASE+640+YAIM,11)
:RETURN
25 SCORE=SCORE+PNTS:POKE 87,CZ:POSITIO
N 20,CZ:? #C6;SCORE:POKE 87,7:RETURN
26 FOR W=95 TO 12 STEP -10:SOUND C3,W,
10,W:NEXT W:GOSUB 860:RETURN
50 S=15:FOR P=CZ TO STREAMS:GOSUB 8
55 IF PLYR<220 THEN 58
56 IF RAND>PROB THEN 80
57 PLYR=40
58 PLYR=PLYR+SPEED:A=USR(ADR(POK$),532
50,PLYR,53278,C1)
60 IF PEEK(53254)=4 THEN PLYR=225:POKE
 53250,225:PNTS=25:GOSUB 25:A=USR(ADR(
EXPL$),PLYR-45,YP1-15)
80 IF M1 THEN GOSUB 150
85 GOSUB 11:COLOR C2:GOSUB 100:LOCATE
80,89,Z:IF Z=CZ THEN COLOR C1:PLOT 80,
94:DRAWTO AMMO,94:AMMO=78
86 NEXT P:RAND=PEEK(53770)
88 IF M1=CZ THEN IF PLYR<205 AND RAND<
PROB THEN M1=C1:MX=PLYR-42:POKE 53252,
PLYR+4:MY=YP1+7:POKE 53767,(16*10)+8
92 IF PLYR<220 THEN YP1=YP1+INT(RND(0)
*5)-C2:A=USR(ADR(MOV$),ADR(PL1$),PMBAS
E+768+YP1,9)
96 GOTO 50
100 X=X(P):Y=Y(P):PLOT X,Y:X=X+SLOPE(P
):Y=Y+SPEED:LOCATE X,Y,Z
115 IF Z<>C1 AND Z<>C3 AND Y<89 THEN D
RAWTO X,Y:X(P)=X:Y(P)=Y:RETURN
120 IF Z=C3 THEN A=USR(ADR(EXPL$),X,Y)
130 COLOR CZ:DRAWTO START(P),C2:GOSUB
600:PNTS=5:GOSUB 25
135 STCNT=STCNT+C1:IF STCNT=STLIMIT TH
EN 2900
136 RETURN
150 T=PMBASE+384+MY:A=USR(ADR(POK$),53
766,MY,53278,1,T+MISPD,C3)
155 IF MY>108 THEN POKE T+MISPD,CZ:M1=
CZ:GOSUB 860:GOTO 165
160 IF PEEK(53248)=C1 THEN POKE T+MISP
D,CZ:M1=CZ:COLOR C3:PLOT MX,88:DRAWTO
MX-C1,90:GOSUB 26:PNTS=50:GOSUB 25
165 POKE T,CZ:MY=MY+MISPD:IF PEEK(5324
8)=4 THEN A=USR(ADR(EXPL$),MX,MY-20):P
OKE T+MISPD,CZ:M1=CZ:GOSUB 860
200 RETURN
300 DATA 45,2
301 DATA 150,-2
302 DATA 3,2
303 DATA 36,0
304 DATA 54,2
305 DATA 99,-2
306 DATA 117,-2
307 DATA 99,-2
308 DATA 48,2
309 DATA 148,0
310 DATA 23,2
311 DATA 138,-2
312 DATA 16,0
313 DATA 108,0
314 DATA 56,0
315 DATA 16,0
316 DATA 32,2
317 DATA 128,0
318 DATA 84,0
319 DATA 20,2
400 COLOR C1:FOR X=CZ TO 7:PLOT X,95-X
:DRAWTO 159-X,95-X:NEXT X:COLOR C2:PLO
T 7,87:DRAWTO 152,87
421 DATA 11,31,51,103,123,143
425 Y=91:RESTORE 421:FOR M=C1 TO C6:RE
AD X
426 IF B$(M,M)="0" THEN 430
427 COLOR C2:FOR B=CZ TO 5:PLOT X+B,Y:
DRAWTO X+B,Y-C2:NEXT B
428 COLOR C3:PLOT X,Y-C3:DRAWTO X+5,Y-
C3
429 PLOT X+C1,Y-4:DRAWTO X+C6,Y-4:DRAW
TO X+C6,Y-C1:PLOT X+7,Y-C2:DRAWTO X+7,
Y-5:DRAWTO X+C2,Y-5:PLOT X+C2,Y
430 NEXT M
434 COLOR CZ:FOR B=80 TO 144 STEP C2:P
LOT B,94:NEXT B:AMMO=144
435 RESTORE 439:F=CZ:COLOR C3
436 FOR T=C1 TO 4:READ X,Y,Z:PLOT X,Y:
DRAWTO Z,Y:NEXT T:IF F=CZ THEN F=1:COL
OR C2:GOTO 436
437 COLOR C3:PLOT 84,89:PLOT 87,92:PLO
T 85,87:DRAWTO 85,89:COLOR CZ:PLOT 81,
87:PLOT 82,87:PLOT 80,93
439 DATA 74,91,88,75,90,89,77,88,84,78
,87,84,73,93,86,73,92,86,76,90,83,76,8
9,83
445 RETURN
450 RESTORE 460:FOR X=5 TO 15:READ N:X
X(X)=N:READ N:YY(X)=N:NEXT X:RETURN
460 DATA 1,1,1,-1,1,0,0,0,-1,1,-1,-1,-
1,0,0,0,0,1,0,-1,0,0
600 STRMTYPE=INT(RND(CZ)*4)+(4*P):REST
ORE STRMTYPE+300:READ B,C:X(P)=B:Y(P)=
C2:START(P)=B:SLOPE(P)=C:RETURN
700 A=PEEK(106)-20:PMBASE=256*A:A=USR(
ADR(POK$),54279,A,559,46,53277,3,53248
,P1,623,2,704,206,706,62,705,15)
730 PLYR=40:Y=384
840 Y=384
842 POKE PMBASE+Y,CZ:A=USR(ADR(MOV$),P
MBASE+Y,PMBASE+Y+1,255):IF Y=384 THEN
Y=640:GOTO 842
843 RETURN
850 Y=ATTACK*-C2:FOR X=C1 TO C2:SOUND
C3,174+Y,10,8:SOUND C1,176+Y,10,7:FOR
Y=50 TO 300:NEXT Y:NEXT X
860 SOUND C3,CZ,CZ,CZ:SOUND C1,CZ,CZ,C
Z:RETURN
899 REM ---DRAW VERTICAL LINES---
900 RESTORE 990:FOR D=1 TO 350:NEXT D
910 READ X,Y:IF X=-1 THEN 929
911 COLOR 3:PLOT 81,87:DRAWTO X,Y:SOUN
D 0,0,0,0:COLOR 0:DRAWTO 81,87:DRAWTO
X,Y
915 FOR I=Y TO Y+7
925 COLOR I/2:PLOT X,I:SOUND 0,I,0,8
927 NEXT I:GOTO 910
929 COLOR 2:PLOT 119,35:DRAWTO 112,42:
DRAWTO 119,50:PLOT 97,24:DRAWTO 104,31
930 REM ---DRAW HORIZONTAL LINES---
932 READ X,Y:IF X=-1 THEN 950
935 FOR I=X TO X+7
945 COLOR I:PLOT I,Y:SOUND 0,Y,10,8
947 NEXT I:GOTO 930
950 RESTORE 990:FOR D=1 TO 300:NEXT D:
SOUND 0,0,0,0
952 READ X,Y:IF X=-1 THEN 960
955 A=USR(ADR(EXPL$),X+1,Y+3):FOR D=1
TO 30:NEXT D:GOTO 952
960 FOR Y=0 TO 6:A=USR(ADR(EXPL$),104,
16+Y*7):NEXT Y:FOR Y=0 TO 3:A=USR(ADR(
EXPL$),119,35+Y*8):NEXT Y
962 FOR D=1 TO 500:NEXT D:RETURN
990 DATA 70,16,70,24,77,16,77,24,87,16
,87,24,97,16,97,24,104,16
991 DATA 50,35,50,43,57,35,57,43,65,35
,65,43,76,35,76,43,83,35,83,43,90,35,9
0,43,100,35,100,43,112,35,112,43,-1,-1
993 DATA 70,16,70,24,84,16,84,31,97,16
,97,24
995 DATA 50,35,50,43,62,35,73,35,83,35
,83,43,100,35,100,50,-1,-1
1000 DIM SLOPE(10),X(10),Y(10),START(1
0),B$(6),XX(15),YY(15),PL1$(9),AIM$(11
),MOV$(39),POK$(25)
1005 CZ=0:C1=1:C2=2:C3=3:C6=6:POKE 559
,CZ:GOSUB 10010:GOSUB 450:POKE 559,34
1009 GRAPHICS 7+16:DL=PEEK(560)+256*PE
EK(561)
1010 SCORE=CZ:FREECITY=2000:ATTACK=CZ:
B$="111111":GOSUB 700:GOSUB 3000
1011 GOSUB 900:GOSUB 850:A=USR(ADR(POK
$),710,2,712,148,DL+3,66)
```

```
1020 RETURN
2850 IF SCORE<FREECITY OR B$="111111"
THEN 2880
2855 FOR M=C1 TO C2:FOR T=C1 TO C3:FOR
 X=15 TO CZ STEP -4:SOUND C1,40-T*5,10
,X:NEXT X:NEXT T:NEXT M
2856 SOUND C1,CZ,CZ,CZ
2860 X=INT(RND(CZ)*C6)+C1:IF B$(X,X)="
1" THEN 2860
2865 B$(X,X)="1":FREECITY=FREECITY+200
0
2880 RETURN
2900 M1=CZ:PLYR=220:COLOR CZ:FOR P=CZ
TO STREAMS:PLOT X(P),Y(P):DRAWTO START
(P),C2:NEXT P
3000 GOSUB 840:SOUND C3,CZ,CZ,CZ:ATTAC
K=ATTACK+C1:IF ATTACK>9 THEN ATTACK=9
3015 RESTORE 3500+ATTACK:READ A,B,C,D,
E:STLIMIT=A:STREAMS=B:SPEED=C:PROB=D:M
ISPD=E
3018 FOR P=CZ TO STREAMS:GOSUB 600:NEX
T P
3020 IF ATTACK>C1 THEN GOSUB 4000:FOR
D=C1 TO 200:NEXT D:GOSUB 2850:GOSUB 42
00
3200 STCNT=CZ:YP1=60:YAIM=60:XAIM=99
3220 POKE 752,C1:? #C6;"K":PNTS=CZ:GOS
UB 25:GOSUB 400:IF ATTACK>C1 THEN GOSU
B 850:GOTO 50
3400 RETURN
3501 DATA 15,3,2,5,1
3502 DATA 20,3,3,10,1
3503 DATA 20,4,3,15,2
3504 DATA 20,4,5,20,1
3505 DATA 20,4,6,40,1
3506 DATA 20,4,6,95,2
3507 DATA 20,4,7,99,2
3508 DATA 20,4,7,140,3
3509 DATA 20,4,9,200,3
4000 FOR D=C1 TO 200:NEXT D:RESTORE 42
1:COLOR CZ:Y=91:B$="000000"
4005 FOR X=C1 TO C6:READ A:LOCATE A+C3
,Y-C2,Z
4010 IF Z=C2 THEN PNTS=100:GOSUB 25:FO
R B=CZ TO 7:PLOT A+B,Y:DRAWTO A+B,Y-5:
NEXT B:GOSUB 4100:B$(X,X)="1"
4040 NEXT X
4055 COLOR C1:FOR B=AMMO TO 80 STEP -C
2:PLOT B,94:GOSUB 4100:PNTS=5:GOSUB 25
:NEXT B
4090 RETURN
4100 FOR S=5 TO 45 STEP 10:SOUND CZ,S,
8,S/C3:NEXT S:SOUND CZ,CZ,CZ,CZ:FOR D=
C1 TO 20:NEXT D:RETURN
4200 IF B$<>"000000" THEN RETURN
4201 POKE 53277,0:GRAPHICS C2:? #C6;"
PRESS FIRE BUTTON","  TO PLAY AGAIN":P
OSITION 4,5:? #C6;"score:";SCORE
4202 IF STRIG(0)=0 THEN POP :GOSUB 100
9:GOTO 50
4205 GOTO 4202
10010 DIM INIT$(41),EXPL$(29),MAIN$(35
5),COORD1$(89),COORD2$(89):RESTORE 110
00
10020 FOR X=1 TO 89:READ A:COORD1$(X,X
)=CHR$(A):NEXT X:FOR X=1 TO 89:READ A:
COORD2$(X,X)=CHR$(A):NEXT X
10040 FOR X=1 TO 41:READ A:INIT$(X,X)=
CHR$(A):NEXT X:FOR X=1 TO 355:READ A:M
AIN$(X,X)=CHR$(A):NEXT X
10050 FOR X=1 TO 29:READ A:EXPL$(X,X)=
CHR$(A):NEXT X
10060 FOR X=1 TO 25:READ A:POK$(X,X)=C
HR$(A):NEXT X:FOR X=1 TO 39:READ A:MOV
$(X,X)=CHR$(A):NEXT X
10070 FOR X=1 TO 9:READ A:PL1$(X,X)=CH
R$(A):NEXT X:FOR X=1 TO 11:READ A:AIM$
(X,X)=CHR$(A):NEXT X
10080 POKE 1568,192:POKE 1569,48:POKE
1570,12:POKE 1571,3
10100 A=USR(ADR(INIT$),ADR(MAIN$),ADR(
COORD1$),ADR(COORD2$),0,1)
10110 RETURN
11010 DATA 0,1,255,0,255,0,255,2,1,1,0
,254,255,1,0,1,254,254,2,0,1,255,2,2,2
,255,254,1,253,3,3,4,252,253,254
11020 DATA 255,254,2,3,3,253,0,0,0,4,4
,252,255,2,0,3,2,1,253,254,254,252,253
,3,253,252,251,251,252,4,3,4,255
11030 DATA 5,5,5,253,1,254,0,255,252,2
53,251,253,252,3,4,3,1,255,1,2,4
12000 DATA 0,255,1,2,254,255,0,1,254,0
,1,0,255,1,253,253,2,255,255,254,2,3,2
,0,254,2,1,3,254,1,254,255,0,1,253
12010 DATA 253,254,3,2,0,3,252,4,3,0,2
,2,4,4,5,3,253,252,0,3,4,254,252,252,2
,1,1,0,255,254,255,1,251
12020 DATA 0,255,1,4,4,252,251,252,253
,253,255,255,3,253,253,4,251,5,5,252,3
13000 DATA 104,169,0,141,0,6,141,1,6,1
04,170,104,168,169,7,32,92,228,104,133
,204,104,133,203,104,133,206
13010 DATA 104,133,205,104,104,141,11,
6,104,104,141,12,6,96
15000 DATA 216,165,16,41,127,133,16,14
1,14,210,173,11,6,240,20
15010 DATA 173,14,6,24,105,16,141,14,6
,173,198,2,41,15,13
15020 DATA 14,6,141,198,2,173,12,6,240
,22,173,13,6,240,17
15030 DATA 56,233,1,141,13,6,74,74,74,
141,1,210,169,40,141
15040 DATA 0,210,173,0,6,240,31,238,1,
6,174,1,6,173,2
15050 DATA 6,157,64,6,173,3,6,157,85,6
,169,127,141,13,6
15060 DATA 169,0,157,106,6,141,0,6,141
,5,6,238,5,6,173
15070 DATA 1,6,205,5,6,16,3,76,98,228,
174,5,6,169,0
15080 DATA 141,4,6,189,106,6,201,89,48
,51,238,4,6,56,233
15090 DATA 89,201,89,48,41,138,168,232
,236,1,6,240,2,16,21
15100 DATA 189,64,6,153,64,6,189,85,6,
153,85,6,189,106,6
15110 DATA 153,106,6,200,208,227,206,1
,6,206,5,6,169,0,240
15120 DATA 176,254,106,6,168,189,64,6,
24,113,203,141,6,6,201
15130 DATA 160,176,159,189,85,6,24,113
,205,141,7,6,201,96,176
15140 DATA 146,10,133,207,169,0,240,2,
240,137,133,208,165,207,10
15150 DATA 133,207,165,208,42,133,208,
165,207,10,133,207,141,9,6
15160 DATA 165,208,42,133,208,141,8,6,
165,207,10,133,207,165,208
15170 DATA 42,133,208,165,207,10,133,2
07,165,208,42,133,208,165,207
15180 DATA 24,109,9,6,133,207,165,208,
109,8,6,133,208,165,88
15190 DATA 24,101,207,133,207,165,89,1
01,208,133,208,173,6,6,41
15200 DATA 3,168,190,32,6,142,10,6,173
,6,6,74,74,24,101
15210 DATA 207,133,207,165,208,105,0,1
33,208,160,0,173,4,6,208
15220 DATA 11,173,10,6,81,207,145,207,
169,0,240,132,173,10,6
15230 DATA 73,255,49,207,145,207,169,0
,240,241
16000 DATA 104,173,1,6,201,20,48,5,104
,104,104,104,96,104,104,141,2,6,104,10
4,141,3,6,169,1,141,0,6,96
17000 DATA 104,74,170,160,0,104,133,25
5,104,133,254,104,240,4,200,145,254,13
6,104,145,254,202,208,237,96
18000 DATA 104,104,133,215,104,133,214
,104,133,217,104,133,216,104,133,218,1
04,170,160,0,177,214,145,216
18010 DATA 200,208,4,230,215,230,217,2
02,208,242,198,218,16,238,96
19000 DATA 0,0,60,255,165,255,60,0,0
20000 DATA 0,0,0,0,8,62,8,0,0,0,0
```

•

(*CHECKSUM DATA* *continued on page 46*)

CHECKSUM DATA.

(see page 43)

```
1 DATA 141,580,472,292,546,689,712,346
,992,782,167,197,740,423,303,7382
80 DATA 959,51,404,381,239,639,485,968
,999,546,354,596,340,521,563,8045
165 DATA 975,583,894,911,618,894,897,7
51,911,753,911,728,887,916,892,12521
313 DATA 713,902,895,892,723,906,887,4
13,183,535,305,913,509,831,751,10358
434 DATA 859,77,16,760,692,604,898,18,
729,176,317,340,637,610,638,7371
860 DATA 349,429,178,702,548,530,780,8
71,710,687,689,531,587,879,603,9073
952 DATA 697,325,764,529,892,863,846,9
62,822,314,526,32,527,781,537,9417
2855 DATA 469,98,800,988,813,735,5,392
,505,737,842,274,795,242,437,8132
3503 DATA 453,449,455,473,485,732,731,
864,663,418,539,707,794,39,963,8765
4201 DATA 802,894,735,321,122,917,862,
341,903,758,198,43,901,920,385,9102
12000 DATA 964,263,34,432,51,377,13,7,
217,450,996,743,441,863,301,6152
15100 DATA 958,239,326,614,853,887,899
,169,370,615,960,409,269,122,387,8077
17000 DATA 525,454,933,930,561,3403
```

•

The New Atari

The 1984 Summer Consumer Electronics Show

by Arthur Leyenberger

The Consumer Electronics Show is a twice-yearly event at which manufacturers of electronic products display new products and announce what will be forthcoming in the following months. The show is held in Las Vegas in January and in Chicago in June, and is attended by close to 100,000 people each time. The June show is especially important, because new products are announced that will become available for the Christmas buying season.

Currently Atari is down to under 1000 employees, from a high of over 7000 in 1982. Alay Kay, Chris Crawford and long-range research and development are no longer part of our favorite company. There is continued talk of a buy-out by Phillips Corp. But just when many people thought that Atari was down for the final count, along comes the 1984 Summer CES, in which Atari rolls out their new products and their new corporate identity.

There is no doubt that *Atari* is an emotional word for most of us. Our love/hate relationship goes back many years. Being loyal Atari enthusiasts, we continue to hope that Atari will eventually come out with a series of computers and peripherals that will again lead the industry. Only this is not to be, at least not in the way that we have wanted. Based upon what was seen in Chicago, Atari has become primarily a publisher of software and a marketeer of hardware. This is not necessarily bad; it just means that Atari will no longer be a full line company with long-term hardware and software research and lengthy product development cycles. If you look at the products announced at CES, the company's new direction is readily apparent. Atari even billed their opening press conference, held on the first day of the show, as "June 3, 1984, the day the future began." Let's take a look at the new products and see what the *new* Atari has to offer.

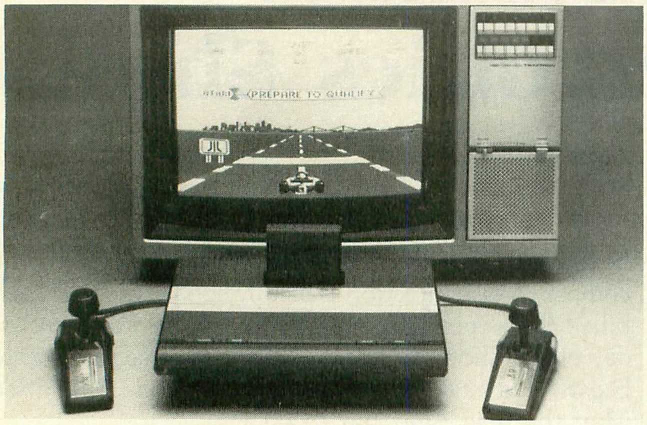
Hardware.

Although Atari wasn't openly showing the 1450XLD computer and the 1090XL expansion box, they did have them available for viewing by third-party software developers. The 1450XLD is no longer being called that but is simply referred to as the new high-end computer. Currently scheduled for a late fall introduction, the new machine will have 64K of RAM and will be compatible with existing Atari software and peripherals, at a cost of under $1000. It will contain a built-in double-sided, double-density disk drive capable of storing 352K bytes on a disk. The disk drive is connected directly to the processor bus, which means it will operate five times faster than other drives using the serial I/O interface.

The new computer contains a built-in 300 baud modem and a speech synthesizer rumored to be better than the one first shown a year ago. Also, telecommunications software and a mini-database called The **Grapevine** are built in. One of the uses of the **Grapevine** is allowing customers to receive customer service information via their computers. The new computer was said to be "70 to 80 percent compatible" with the IBM PC.

The 1090XL Expansion Box was also shown to software developers. It will have five expansion slots and contain a 64K RAM card. This will increase the memory of a 600XL to 64K (80K bank select) and an 800XL or high-end computer to 128K. Atari is supposedly working on an 80-column video card, a clock/calendar card and a CP/M card.

The most exciting piece of hardware introduced by Atari was the **MindLink** system. This device is composed of a headband connected to an infrared transmitter and a receiver that connects to a video game or computer up to 20 feet away. Using electromyogram transducers, the headband can detect minute electrical energy generated from the muscles in the forehead. By tensing and relaxing the muscles in your face and forehead, you can control a computer or video game screen without using a joystick. The **MindLink** will sell for approximately $100 and, initially, be available for the BCS and 7800 video game this fall. The computer version will be available in early 1985.



7800 ProSystem.

Atari was showing two **MindLink** VCS games. One, **Bionic Breakthrough**, was the familiar **Breakout** game with the paddle at the bottom of the screen controlled by your forehead. The screen changes color and the player is rewarded with higher point totals as he or she relaxes during the game. Interestingly, if the infrared beam is broken when a person walks between you and the receiver, the game instantly pauses. Likewise, if the phone rings, just get up and answer it, and the game will wait for you.

I was able to try the game for about fifteen minutes. By the time I finished, I was playing fairly well. With continued practice, I think I could get used to this unique input device. Anyway, I was impressed with **MindLink**. With a hands-free input like this, who needs a mouse?

There are many possible applications for **MindLink**. Software planned by Atari covers a range or areas, including relaxation, education, ESP and thought games that rely on memory and intuition, biofeedback and relaxation. The product manager of **MindLink** told me that it would eventually gain serious computer application software. Its use by physically impaired persons would be a major breakthrough. In a word processing program, for example, with some clever programming, **MindLink** could be used for two-dimensional cursor positioning by a quadriplegic.

Atari was also showing the 7800 ProSystem and the computer keyboard upgrade for it. The keyboard will operate with 4K of RAM and is expandable to 20K. It is compatible with Atari home computer peripherals but not with existing computer software. The 7800 will list for $150 and should be available in July. The computer keyboard will probably cost less than $100 and be out by the end of the year.

Software.

Five software titles were announced for the 7800 computer keyboard. They include a terminal program, word processor and BASIC. **AtariLab** and **Typing Tutor** will also be available. Prices were not disclosed.

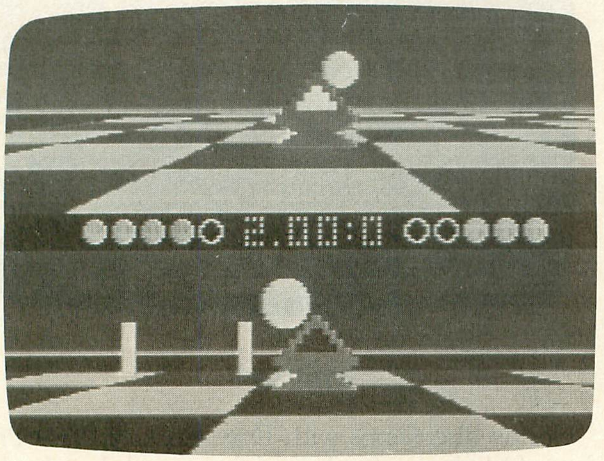
A dozen new Atarisoft titles were announced for other computers, including their first educational program, **Typo Attack**. Atari plans to add educational software to its already extensive list of game titles under the Atarisoft label.

New life was pumped into the aging VCS video game system with the introduction of the "superchip" technology series of games. VCS games in this series have upgraded graphics due to the increase in ROM (read only memory). Instead of the previous 8K maximum, superchip VCS games have 16K of ROM and a special interface chip. New VCS titles include **Track and Field**, **The Last Starfighter** (based on the recent film), **Jr. Pac-Man**, **Millipede**, **Stargate**, **Crystal Castles** and (David's) **Midnight Magic**. These same titles have been announced for the 5200 game and computer.

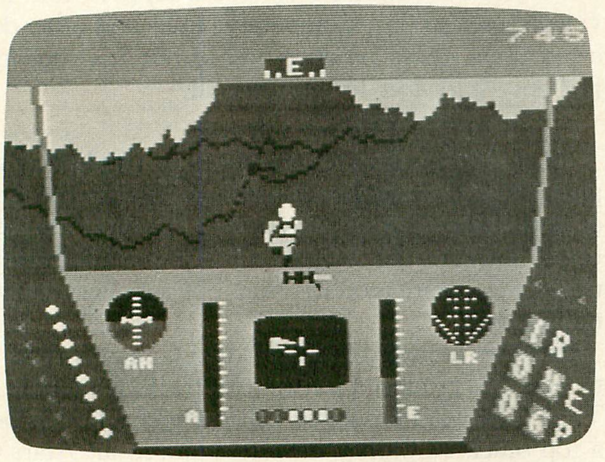
The two new Lucasfilm games were being displayed on the 5200 and 7800 video games and the computer. The graphics in both games are excellent—outstanding when seen on the new 7800 system. Lucasfilm's computer division used sophisticated animation techniques and graphics technology to develop these games, and it really shows.

Ballblazer is a futuristic, high-speed soccer type of game that uses a split screen to convey the action. Each of the players gets their own unique first-person view with a three-dimensional perspective. The music that accompanies the game is an improvisational jazz score that is as innovative as the game play. The other Lucas title is called **Rescue on Fractalus**. Here, you navigate your Valkyrie Fighter through the treacherous canyons of Fractalus in search of downed pilots. A first-person viewpoint is used for the flight simulation, and fractal geometry effects the three-dimensional random graphic sequences.

The theme of this particular Consumer Electronics Show seemed to be educational software. Atari fell in line with a series of new educational titles, some of them very well done. Under the Atari Learning Systems umbrella, several series of programs were announced for a wide range of ages. The **Milestone** series represents top of the line educational software from Atari. The previously announced **AtariLab Starter Set** (temperature module) and **Light Module** lead off this educational line. With these two products, the home computer science student can learn about temperature and light by conducting experiments and completing workbook exercises. They are geared for elementary and junior high school students.



Ball Blazer.



Escape from Fractalus.

Find It! is a group of computer activities designed for the development of visual perception skills. Ranging from simple to moderately complex, these allow young children to participate in such tasks as finding the animated figure in a crowd, matching geometric shapes or solving visual puzzles. **The ABC of CPR** is the first entrant in the home health software library, focusing on the basics of health/medicine. The first of a two-part tutorial is titled *First Aid.* This program is intended to build awareness and background information as a prelude to formal training in medical assistance. Graphics, sound and animation are used to teach first aid techniques to both children and adults.

Wheeler-Dealer is a simulation of an automobile assembly plant, aimed at children twelve and older. Supply and demand economics are taught by setting up and maintaining a profitable business. The player actually designs and assembles vehicles, acquires raw materials and selects staffing and pricing, based upon options given for maximum profitability. Up to four players can compete in this simulation, which even includes price freezes and strikes.

The Simulated Computer uses a computer to show what goes on inside a computer. While not a new title —Atari bought this one, as they did several others— **Simulated Computer** lets the user program a see-through mock-up of a computer system, then see the results of each action as the computer carries out the program. Turtle designs and sound effects can also be achieved with this program designed for children ages 10 and up.

Telly Turtle is the next in the series. It is a pictorial pre-Logo version of the turtle graphics concept. There are four levels available, with the top level being a true programming language. Once this level is mastered, the user would continue on to Atari Logo.

All of these educational titles will be available for other computers in addition to the Atari. Apple, Commodore and IBM computers will be supported. In addition, **AtariLab** will become available for other machines as well.

The most exciting Atari educational programs announced were the "Futuremakers" series. There are two initial titles, aimed at ages 10 to adult, that deal with the space program. **This Is Ground Control** is a simulated voyage through our solar system. The journey involves spacecraft design, course planning and flight operations—as you deal with the principles and technical constraints of real space travel. Excellent graphics, using a three-dimensional view perspective, create a feeling of actually being out in space. Planet fly-bys are unreal.

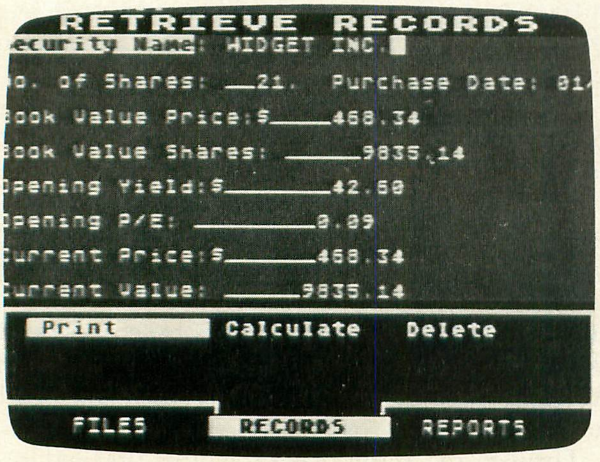
Through the Starbridge is the other title and incorporates fact and science fiction, as you travel the universe and explore everything from black holes to quasars to aliens. Theories and facts about physics, logic, chemistry, mathematics and astronomy all blend together to make this program educational and entertaining.

Both of these "Futuremakers" titles share several elements. A heads-up display is used to present the view out of the craft's window, with 3-D animated graphics of planetary approaches and swing-bys. A joystick, lightpen or touch tablet may be used as the control. The game can be saved to disk for future continuation. Random start-up scenarios present the user with a different identity and a different set of parameters each time the program is used.

These programs appear to be excellent. Only demos could be seen at the show but, as mentioned earlier, they were very impressive. **This Is Ground Control** and **Through the Starbridge** will be available by September and will sell for $39.95.

The SYN-series—**Syncalc, Synfile+** and **Syntrend**—were present but not prominently displayed at CES, since they are already on retailers' shelves. Synapse created the three programs exclusively for Atari and announced them a year ago. **Proofreader** is a revised spelling checking program for the Atari, similar to APX's **Atspeller**. Expect to see it soon.

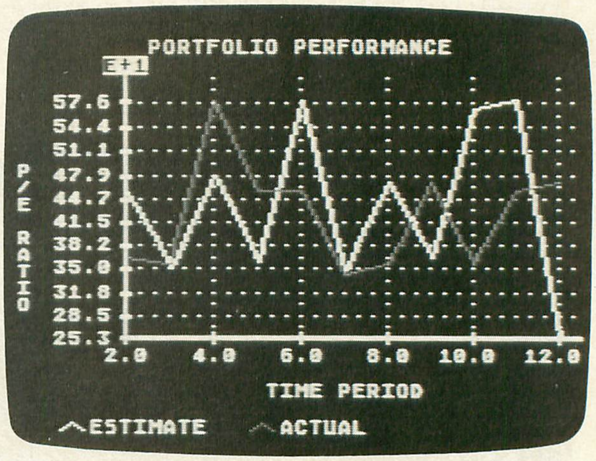
Syntrend, Synfile+ and Syncalc.



Synfile+.

Additional game titles for the computer include: **Gremlins** (based on the hit movie), **The Last Starfighter, Hobgoblin** (Atari's first text-adventure game, tentatively titled), **The Final Legacy, Track and Field, Crystal Castles, Pole Position II, Elevator Action** and **Jr. Pac-Man**. Most of these games will be available in the third quarter of 1984.

As you have seen, Atari announced quite a few new products at the Summer Consumer Electronics Show in Chicago. Many of the products were not developed in-house, but, rather, were purchased or licensed from the outside. Almost all of the software that was announced will become available for other computers. Even hardware products like **MindLink** and **AtariLab** will eventually find their way onto such computers as Commodore and Apple. This is in keeping with Atari's new role as a publisher and marketeer.



Syntrend.

During discussions with Dave Ruckert, Vice President of Marketing for Atari, I discovered that the decision to actually go ahead with the high-end computer (1450XLD type) and the expansion box was made with the advanced user in mind. According to Rupert, if it wasn't for the continued input and support from individuals, user groups, Compuserve users and other dedicated users, these products would not have made it. Perhaps Atari *does* listen, after all.

Other software.

Much of the software introduced by third-party suppliers at CES was either educational, not for the Atari, or both. There is no space here to list all of the additional software that was seen for the Atari, however, two new software products from one supplier were very impressive.

Batteries Included is a Canadian company that, until now, has produced software for the Commodore 64. Their C64 word processor, called **Paper Clip** has been a top seller. Now they have announced an Atari version of **Paper Clip** that may become the ultimate word processor for the Atari. It is impossible to describe all of the features of this product, so I will just mention a few.

Paper Clip is compatible with standard Atari DOS files and is the first word processor to interface with Atari's new 80-column card (presumably Atari's new **Super AtariWriter** will, too). The program disk comes with over fifty printer configuration files, and each one may be further customized. A macro command allows a single keystroke to enter and display a set of repeatedly used strings of text—or even entire sentences—at any point in the body of the text. There is on-screen display of up to 132 columns, which can be formatted to 80 columns with the print preview command. Finally, there are dual text windows which allow the simultaneous editing of two files, plus cut and paste transfers from one file to another. **Paper Clip** for the Atari will list for $89.95 and be available during the second half of 1984.

The other product announced by Batteries Included is called **Homepak**. This $49.95 program is really a combination of three programs in one: a smart telecommunications program, an information management system and an easy-to-use word processor. **Hometerm**, the telecommunications program, features X-Modem protocol for exchange of data between computers, bulletin boards and data banks such as CompuServ. Features such as an on-screen clock and unlimited capture buffer make this a very promising program.

Homefind is the information manager which lets the user employ English language commands. For example, I may store an item like this: "Atari's chief executive officer is James Morgan." Later, I can simply ask, "Who is James Morgan?" and I will see displayed on the screen, "Atari's chief executive officer."

The third program in **Homepak** is called **HomeText**. While not as sophisticated as **Paper Clip**, this straightforward word processor offers many features—such as cut and paste, mail merger (with **Homefind**), headers, footers and page numbers.

Batteries Included looks like a company that is headed for success in the Atari market, based upon the first two products they were showing. Best of luck to them and other software companies that continue to support the Atari computer and the Atari computer user.

Postscript.

No description of a Consumer Electronics Show is complete without mention of the one product that was the undeniable hit of the show. In this case, it was the Amiga personal computer. While not specifically an Atari product, the fate of Atari users and future Amiga PC owners seems likely to be intertwined. Here's why.

With a Motorola 68000 CPU, 128 bytes of RAM, very high resolution graphics, built-in disk drive, modem, NTSC (television), composite and RGB outputs, this machine makes Apple's much-touted Macintosh look primitive. Its IBM compatibility, 16-bit operating system and phenomenal sound and graphics features could make this the graphics computer of the 1980s that we've all been waiting for.

The Amiga PC wasn't being shown to the public, but I was able to sit in on a brief demo. For owners anxious for Atari's next generation graphics machine, the Amiga isn't "next generation" but the one *after* that. Priced under $2000, it should be available by the end of the year. Personally, I can't wait. □

Word Scramble

16K Cassette or 24K Disk

by Steven T. Murphy

Word Scramble is an amusing game for verbalists, in which one or more persons try to unscramble words scrambled by the computer.

There are three hundred words in the computer's memory, each coinciding with a particular skill level. After the title screen, you will be prompted to enter a level of difficulty. Beginner level has three to four letters in each word; intermediate has five to six per word; and expert has anywhere from seven to twelve letters per word. Once you have selected the level, the computer will start scrambling a word at random so that, if a word appears twice, the letters may not appear in the same order. To get back to the level entry menu, just hold OPTION while the computer is scrambling a word.

If you make a mistake typing in a word, simply hit the backspace key, and the word will disappear.

I hope you will have hours of fun with **Word Scramble**; I know I have. I've noticed that, even though I put in the words to be scrambled, I still have a very hard time unscrambling words from the expert level.

Program outline.

Line(s)	Function
19 - 31	Title page
34 - 95	Level entry
97 - 120	Draw screen
125 - 205	Scramble section
210 - 265	Word entry
270 - 285	Sound routine
290 - 435	Score and advancement
997 —	Data words (must be typed in according to line number)
999 - 1009	Data for beginner
1999 - 2009	Data for intermediate
2999 - 3009	Data for expert

Adding words.

To add more words to your **Word Scramble**, first select ten words for each skill level. It is necessary to have thirty words because of the random selection.

Next, locate the corresponding data statements for the skill level of the chosen words, and enter the words in a data statement, incrementing by one line number. There can be only ten words per data line. Then, in Line 135:

```
135 RESTORE ((J*1000)+INT(RND(0)*10)):
 ...
```

For every thirty words you enter (ten for each skill level), add one to the ten in the first random statement only. For example, you add thirty more words to the list, ten for each skill level. Line 135 would look like this:

```
135 RESTORE ((J*1000)+INT(RND(0)*11)):
 ...
```

It's that simple. □

```
1 REM ********************************
2 REM ***        WORD SCRAMBLE      ***
3 REM ***                           ***
4 REM ***             BY            ***
5 REM ***                           ***
6 REM ***        STEVE  MURPHY      ***
7 REM ********************************
8 REM
9 REM
10 REM
15 CLR :DIM WORD$(50),A$(50),B$(50),B(
50),LEVEL$(20):SCORE=290:S=270
18 GRAPHICS 18
19 REM     TITLE PAGE
20 POSITION 8,0:? #6;"ROWd":POSITION 6
,3:? #6;"MsaBEICr":POSITION 9,5:? #6;"
Yb":POSITION 4,8:? #6;"vSETe YMUhRp"
24 FOR W=1 TO 1400:NEXT W
25 FOR W=1 TO 60:POKE 755,2:FOR E=120
TO W STEP -W:SOUND 0,E,10,6:SOUND 1,E+
10,10,6:POKE 755,4:NEXT E:NEXT W
29 SOUND 0,0,0,0:SOUND 1,0,0,0:POKE 75
5,2
30 POSITION 8,0:? #6;"WORd":POSITION 6
,3:? #6;"ScrAMbLe":POSITION 9,5:? #6;"
BY":POSITION 4,8:? #6;"sTEVe MURpHy"
32 FOR W=1 TO 2000:NEXT W
34 REM    LEVEL ENTRY SECTION
35 ? #6;"K"
40 POSITION 3,0:? #6;"SELECT level"
45 POSITION 4,11:? #6;"push start"
47 POSITION 0,3:? #6;"BEGINNER"
50 POSITION 0,5:? #6;"intermediate"
55 POSITION 0,7:? #6;"expert"
60 POSITION 16,3:? #6;"<<<":J=1
65 POKE 53279,8:X=PEEK(53279)
70 T=T+1:SETCOLOR 2,8,T*2:SOUND 0,40,1
4,T:IF T>2 THEN T=0
75 IF X=5 AND J=1 THEN POSITION 16,3:?
 #6;"   ":POSITION 16,5:? #6;"<<<":J=2
:FOR W=1 TO 50:NEXT W:GOTO 65
80 IF X=5 AND J=2 THEN POSITION 16,5:?
 #6;"   ":POSITION 16,7:? #6;"<<<":J=3
:FOR W=1 TO 50:NEXT W:GOTO 65
85 IF X=5 AND J=3 THEN POSITION 16,7:?
 #6;"   ":POSITION 16,3:? #6;"<<<":J=1
:FOR W=1 TO 50:NEXT W:GOTO 65
90 IF X=6 THEN FOR W=1 TO 3:FOR E=15 T
O 0 STEP -1:SOUND 0,20,14,E:NEXT E:NEX
T W:GOTO 99
95 GOTO 65
97 REM    DRAW SCEEN ROUTINE
99 ? #6;"K"
100 POSITION 4,0:? #6;"unscramble"
105 POSITION 0,2:? #6;"____________________
_____"
110 POSITION 0,4:? #6;"____________________
_____"
115 POSITION 3,6:? #6;"________________"
120 SETCOLOR 2,8,4:POSITION 0,9:? #6;"
YOU              atari":GOSUB 500
125 REM    SCRAMBLE SECTION
130 FOR W=1 TO 15:B(W)=0:NEXT W:B$="":
WORD$="":A$=""
135 RESTORE ((J*1000)+INT(RND(0)*10)):
FOR Q=1 TO INT(RND(0)*10)+1:READ WORD$
:NEXT Q
140 X=LEN(WORD$)
145 FOR W=1 TO X
150 IF K=10 THEN K=0:GOTO 130
155 G=PEEK(53279):A=INT(RND(0)*X)+1:SO
UND 0,A,14,4:FOR V=1 TO 4:NEXT V:SOUND
 0,0,0,0:IF G=3 THEN 34
160 FOR E=1 TO X
165 IF A=B(E) THEN K=K+1:GOTO 150
170 NEXT E
175 K=0
180 B(W)=A
185 NEXT W
190 FOR W=1 TO X:B$(W,W)=WORD$(B(W),B(
W)):NEXT W
195 IF B$=WORD$ THEN B$="":GOTO 135
200 POSITION 4,7:? #6;"
";:POSITION 4,5:? #6;"                 "
:POSITION 4,3:? #6;"                 "
205 POSITION 4,3:FOR W=1 TO X:SOUND 0,
50,14,15:FOR T=1 TO 14:NEXT T:SOUND 0,
0,0,0:? #6;B$(W,W);:NEXT W
210 REM    WORD ENTRY SECTION
215 CLOSE #1:OPEN #1,4,0,"K:":T=0
220 POKE 702,64:POKE 694,0
225 GET #1,K:PITCH=INT(RND(0)*40)+10:S
OUND 0,PITCH,14,4:SOUND 1,PITCH,14,3
226 FOR W=1 TO 10:SETCOLOR 1,13,8:NEXT
 W
230 SOUND 0,0,0,0:SOUND 1,0,0,0:SETCOL
OR 1,12,5
240 IF K=155 THEN GOTO 260
243 IF K=126 THEN T=0:A$="":POSITION 4
,5:? #6;"                ":GOTO 220
245 IF K<65 OR K>90 THEN SOUND 0,0,0,0
:SETCOLOR 1,12,5:GOTO 220
247 POSITION 4+T,5:? #6;CHR$(K);:IF (T
+4)>15 THEN GOTO 265
250 T=T+1:A$(T,T)=CHR$(K)
255 GOTO 220
260 IF A$=WORD$ THEN Y=Y+1:P=0:POSITIO
N 4,7:? #6;"CORRECT":GOSUB S:GOTO SCOR
E
265 IF A$<>WORD$ THEN POSITION 4,7:? #
6;WORD$:P=1:GOSUB S:M=M+1:GOTO SCORE
270 REM    SOUND ROUTINE
275 IF P=0 THEN FOR L=1 TO 30 STEP 3:F
OR Q=60 TO L STEP -L+2:SOUND 0,Q,10,10
:SETCOLOR 2,8,Q:NEXT Q:NEXT L
280 IF P=1 THEN FOR W=30 TO 4 STEP -4:
FOR E=W TO 1 STEP -1.5:SOUND 0,W-E,12,
E:NEXT E:SETCOLOR 3,3,W:NEXT W
285 SOUND 0,0,0,0:SETCOLOR 2,8,5:RETUR
N
290 REM    SCORE AND ADVANCEMENT
300 POSITION 1,11:? #6;Y:POSITION 16,1
1:? #6;M;
310 IF (Y-M)=40 AND J=1 THEN LEVEL$="M
EDIUM":J=2:GOTO 400
315 IF (Y-M)=30 AND J=2 THEN LEVEL$="E
XPERT":J=3:GOTO 400
320 GOTO 125
400 POSITION 4,1:? #6;"ADVANCEMENT
":POSITION 4,3:? #6;LEVEL$:POSITION 4,
5:? #6;"LEVEL"
405 FOR E=1 TO 5 STEP 0.3
410 FOR W=6 TO 2 STEP -0.4
415 SOUND 0,50,12,W:SETCOLOR 2,8,W
420 NEXT W:NEXT E
425 SOUND 0,0,0,0:SETCOLOR 2,8,5
430 FOR W=1 TO 800:NEXT W
435 POSITION 4,1:? #6;"              ":PO
SITION 4,3:? #6;"           ":POSITION 4
,5:? #6;"     "
500 REM    RESET OF SCORE
505 M=0:Y=0:POSITION 1,11:? #6;Y;"  ":
POSITION 16,11:? #6;M;"  ";
510 GOTO 135
997 REM *** WORD SCRAMBLE DATA ***
998 REM
999 REM    DATA FOR BEGINNER
1000 DATA THE,WAIT,ROAD,TOAD,BIRD,SKY,
AIR,DOG,NOSE,EAR
```

The Latest Innovations From CDY For Your Atari System

OMNIMON! Resident Monitor

ANTIC July '83 review by David Duberman:
"OMNIMON! by David Young is a machine-language monitor that should have come with the ATARI. In fact, every microcomputer should have this sort of hardware based monitor installed. Most, however, do not. Now, for a relatively low cost, you can equip your ATARI 400/800 with a truly sophisticated programming tool. Whether you're an experienced programmer or a wondering beginner, OMNIMON can, if wisely used, help you to fully understand the working of your computer."

ANALOG July '83 review by Brian Moriarty:
"OMNIMON! can be a great addition to your ATARI computer if you know what to do with it. The ability to "freeze" a running program on-the-fly and examine the hardware registers is invaluable for testing and debugging; the sector-level disk functions are alone worth the price of the board . . . OMNIMON! might be one of the smartest investments you can make."

September '83: "Those of you who read my review of OMNIMON! in issue #12 know what a godsend it is for serious programmers. This ROM-resident monitor has saved me many hours of program development and debugging time, and recently made it possible for me to recover several otherwise unsalvageable text files that were lost when my word processor accidentally destroyed a disk directory. Ironically, the review you are reading is one of those salvaged files! Three of the ATARIs in our offices are now equipped with OMNIMON! boards, and more are on the way. Staff programmers Tom Hudson and Charlie Bachand both swear by OMNIMON!"

What is OMNIMON!?

OMNIMON! is a PC board which plugs into your 400/800 (soon to be available for the XLs also) and gives you complete control of your computer. Even though it is always available (by pressing SELECT and SYSTEM RESET) it takes up no user memory because it resides in the unused 4K block at $C000. Use it to interrupt, examine, and manipulate any program in memory whether it be disk, cassette, or cartridge based. It is especially good for program development or customization of existing programs. The flexible disk I/O allows you to write to or read from disk in either single or double density. You can edit raw sector data or even load a file without DOS. Many debugging tools are at your disposal: Display/Alter memory or 6502 registers, Disassemble memory, Search memory, Hex/Char modes, Single Step execution, JSR or GOTO address, Push/Pull stack, Printer dump, etc. After interrupting a program with OMNIMON!, many times it is possible to return to the program as if you had never left it (e.g., BASIC, DOS, etc.). Instructions are provided for the addition of a simple toggle switch to make OMNIMON! invisible, thus making it compatible with all software. An external cable is now provided to eliminate the need to solder directly on the board.

New 8K OMNIMON! Upgrade

This enhancement, which is available to all OMNIMON! users, includes a substantial number of features not available in the standard version. The 8K OMNI resides in an 8K ROM which has been modified by the addition of a switch for selecting either of two 4K banks. The additional features include Hex Conversion and Hex Arithmetic, Block Move, a Relocater, and a Line Assembler. A Binary Load command allows you to load any binary load file without DOS and doubles as a disk directory command which prints out the start sector of each file. Lockup recovery allows you to recover from system lockup, meaning that when your computer freezes, you can usually salvage the program or text file in memory by popping into 8K OMNI and dumping memory to disk. Advanced users will like the user extendibility feature which allows them to make use of the interface routines of 8K OMNI in their own software. One of the most exciting features of the 8K OMNI is the resident Ramdisk handlers. They allow AXLON Ramdisk owners to use this powerful device with any DOS which uses standard SIO calls and even with boot programs like word processors and games which access the disk a lot. Several additional features make this version very valuable for advanced programmers, but if you have a Ramdisk, 8K OMNI is a MUST!

New OMNIVIEW 80 Column Upgrade

Did you know that for most applications you do not need an expensive, slot consuming 80 column board to enjoy the power of 80 columns? Would you 400 owners like the convenience of 80 columns? OMNIVIEW takes advantage of the high resolution graphics mode built into the ATARI to generate an 80 column screen editor essentially identical to the ATARI screen editor (E:, S:). Thus, you can use OMNIVIEW in any environment where you would normally use the 40 column "E:" (e.g., BASIC, Assembler/Editor, etc.). The 80 column "E:" of OMNIVIEW has been optimized for speed so that it is not significantly slower than 40 column "E:". In addition, the character font was specially designed to be legible on an ordinary TV set! A monitor is recommended, but not really necessary for casual 80 column operation. The Bit-3 version of LJK's 80 column Letter Perfect has been modified to support OMNIVIEW and other programs are sure to follow. Lastly, the Ramdisk handlers described under 8K OMNI are also incorporated in OMNIVIEW.

New RAMROD-XL

800XL owners will soon be able to equip their computers with OMNIMON and OMNIVIEW. In addition, the Newell enhanced operating system and Fastchip floating point package will be included at no extra charge. This will essentially turn your 800XL back into a 400 / 800 compatible machine and allow it to run most of the software which the XL-OS will not. A switch will allow you to select the XL-OS when needed.

Pricing

Hardware:	Standard OMNIMON! Piggyback Board (400/800)	$99.95
	RAMROD-XL with OMNIMON-XL (800XL)	$119.95
	OMNIVIEW-XL Addon (RAMROD-XL)	$45.00
Enhancements:	(subtract $5.00 if ordered with board)	
	8K OMNIMON Enhancement	$45.00
	8K OMNIVIEW Enhancement - (4K OMNIMON with 4K OMNIVIEW)	$45.00
	4K OMNIVIEW Enhancement	$30.00

Newell RAMROD OS Board

This is a new operating system board which replaces the existing OS board. It allows you to use EPROMs in place of the ATARI OS ROMs and comes with an enhanced OS which includes additional graphics modes and a fast cursor. It also has a socket which will accept any version of OMNIMON and thus is an alternative to the OMNIMON! piggyback board. For the 800 only.

RAMROD OS Board with Standard OMNIMON	$149.95
RAMROD OS Board with 8K OMNIMON or 8K OMNIVIEW	$189.95
Same as above with Fastchip Floating Point Package	$209.95
RAMROD OS Board with 8K OMNIMON and 4K OMNIVIEW	$209.95
Fastchip Floating Point Package by itself	$29.95

How To Order

We accept checks, money orders, or credit cards (Visa or MC). COD must be payed with cash or MO. Our toll free order desk is (800) 227-3800 ext. 561. Call this number only if you know **exactly** what you want and be explicit! They can answer no questions. For faster service or if you have questions please call or write:

CDY Consulting
421 Hanbee
Richardson, TX 75080
(214) 235-2146

DEALER INQUIRIES SOLICITED

```
1001 DATA EYE,BYE,HAND,FOOT,TREE,PUSH,
BACK,BALL,BIKE,ANT
1002 DATA BEE,BELL,TALK,POKE,NEAR,FAR,
BET,ABLE,WORD,KEY
1003 DATA GIVE,HAVE,PUT,TAKE,AGE,ALL,A
IM,CRY,DATE,BAIT
1004 DATA BOOK,THAT,HOT,KNOT,DATA,LIMP
,JOY,HORN,PEN,HELP
1005 DATA HIT,ART,CITY,SHED,SILK,DOWN,
OFF,SEA,HERE,THEN
1006 DATA BULB,BOX,TAPE,READ,OUT,FISH,
MEAN,WORD,NAIL,PIN
1007 DATA TEST,STAR,GAME,MENU,OIL,WELL
,CUBE,HAT,SHOE,PLUS
1008 DATA TRY,SHOT,BEST,FILL,PLAY,LATE
,HOME,WORK,WHAT,FLAG
1009 DATA MAN,HANG,RAIN,BOW,MOVE,MAIL,
LIST,BUG,BOOT,BASE
1999 REM  DATA FOR INTERMEDIATE
2000 DATA POSTAL,GUESS,MUSIC,SCHOOL,FR
IEND,BORDER,DRILL,ANGRY,PALACE,RADIO
2001 DATA TRACK,BOXES,SEARCH,PRICE,UNI
TED,CHINA,RECORD,PRINT,SCROLL,MARGIN
2002 DATA WIDTH,FIRST,PACMAN,MONTH,SPE
ED,STRING,COAST,TRUCK,ROBOT,STREET
2003 DATA SORRY,FORCE,RETURN,STAFF,PLE
ASE,BLACK,SHEEP,DONKEY,RANDOM,SEVEN
2004 DATA ATTACK,GATHER,BRAVE,THROW,AL
ONE,BEWARE,ORDER,TODAY,EFFECT,SCOUT
2005 DATA TABLE,HOUSE,CHECK,SKETCH,SPI
RAL,SOUND,BANNER,STOCK,RANGE,BLINK
2006 DATA HAPPY,COLOR,LATEST,SHOOT,MOT
OR,NUMBER,CRASH,SUPER,CLOCK,LASER
2007 DATA PLANE,ERROR,DRILL,BORDER,HEA
VEN,EASTER,WATER,PAPER,STICK,BOARD
2008 DATA STATE,MENTAL,SHADE,BALLAD,HA
NGER,WIZARD,SPELL,EIGHT,EARTH,HOLDER
2009 DATA APART,SPEAK,SUMMER,WINTER,SP
RING,BASIC,MANUAL,LIGHT,BRIEF,ATARI
2999 REM  DATA FOR EXPERT
3000 DATA CHARACTER,CHEMISTRY,REPLACE,
OPERATOR,TOMORROW,WASHINGTON,TELEPHONE
,CAPITAL,SPECIAL,RELEASE
3001 DATA UNDERLINE,IMPROVE,DISPLAY,MA
GAZINE,PROGRAM,PRODUCE,EXAMPLE,CASSETT
E,COMPUTER,DEVELOP
3002 DATA DIRECTION,DOMINOES,POPULAR,C
ONTROL,PERFECT,THOUSAND,COMPLETE,BEGIN
NING,BEAUTIFUL,ALLIGATOR
3003 DATA ENGLISH,LECTURE,WATERBED,TON
IGHT,PRINTER,EXCHANGE,INVADER,AMERICAN
,POLITICS,AMBASSADOR
3004 DATA CARRIAGE,UNDERLINE,DEFAULT,S
TANDARD,REVIEW,FUNCTION,BALANCE,CONDEN
SER,MICROPHONE,PORTABLE
3005 DATA DIGITAL,ASSEMBLY,CARTRIDGE,E
XPERIENCE,PERSONAL,DISCOVER,LANGUAGE,B
ETWEEN,ADDRESS,RESERVED
3006 DATA CONNECTED,LOCATION,REGISTER,
MACHINE,DETERMINE,DECIMAL,EQUIVALENT,P
ERFORM,TRANSFER,RESULTS
3007 DATA STATEMENT,DUNGEON,GOLDRUSH,O
UTPOST,COMMUNIST,AROUND,HOMEWORK,LATES
T,POLYGON,AIRPLANE
3008 DATA HANGMAN,MOVEMENT,RAINBOW,LAS
ERS,CHECKBOOK,BASEBALL,FOOTBALL,COPYCA
T,CHAPTER,ALGEBRA
3009 DATA INVENTORY,KEYBOARD,HORIZONTA
L,PICTURE,GEOMETRY,TECHNIQUES,ADVANCED
,GRAPHICS,GENERATOR,ROUTINE
```

•

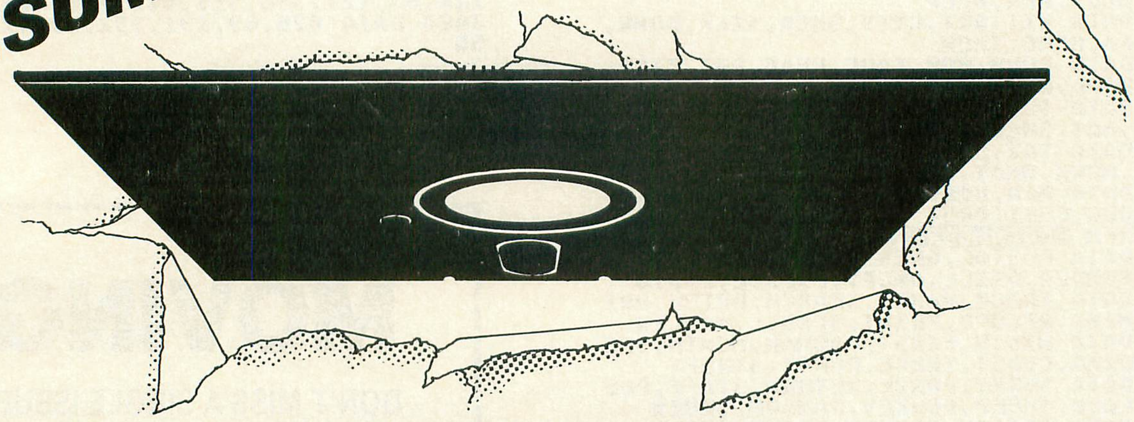
CHECKSUM DATA.

(see page 43)

```
1 DATA 507,16,31,181,35,137,519,1,3,25
1,890,36,389,517,528,4041
25 DATA 152,951,464,517,80,376,951,238
,335,405,740,869,566,420,670,7734
80 DATA 668,676,15,647,516,400,206,602
,604,284,313,993,505,130,499,7058
145 DATA 386,553,106,369,136,741,227,5
69,785,79,342,56,767,683,967,6766
220 DATA 218,124,469,688,283,970,883,3
29,351,713,740,638,870,195,527,7998
285 DATA 414,669,824,612,636,704,451,2
82,136,608,467,19,565,804,942,8133
505 DATA 614,708,992,125,480,440,514,5
88,295,817,786,641,110,302,600,8012
1999 DATA 283,242,317,399,503,447,117,
114,63,121,396,915,441,768,404,5530
3003 DATA 626,69,191,992,894,292,91,31
55
```

•

BREAKS THROUGH WITH AWESOME POWER!



TOP-DOS

POWER YOU COMMAND.

FRIENDLY POWER

TOP–DOS unleashes the latent power of your ATARI computer–an amazing machine. TOP–DOS puts this power under your control. Here are just a few of its friendly features:

COMMAND MENU & HELP FILES speed your mastery of the system.
SUPERIOR STATUS DISPLAY keeps you informed. Shows free memory, disk-drive configuration, and state of TOP-DOS options.
ERROR–CODE TRANSLATOR deciphers numeric codes into English.
BREAK–KEY ABORT lets you change your mind in mid-command.
FULL SCREEN USE shows you what you've done. Keeps 23 ines of past operations on display.
UNDELETE COMMAND rescues an accidentally-deleted file.

ECLIPSE

POWER TO PLEASE

Upgrade to TOP–DOS. Owners are delighted. You will be too. Only $49.95. No risk, 30-day MONEYBACK GUARANTEE.

SPECIFICATIONS

Memory required: 32K
Computers: ALL ATARI
Disk drives: 1 to 8
5¼": Single-density, Double-density, Double-sided Double-density
8": Double-density.
Ramdisk: AXLON or MOSAIC.
Files: Single density: Up to 128
Double density: Up to 256
Sectors: Single density: Up to 944
Double density: Up to 1968
Memory–residency: 0700-1A80 (hex)
(Same as ATARI DOS-2)
Commands: 58
(All ATARI DOS-2 + 43 more)
Command options: 42

See TOP–DOS at your dealer. If not available, you may order direct from ECLIPSE SOFTWARE, 1058-A Marigold Court, Sunnyvale, CA 94086,(408) 246-8325.

DEALER INQUIRIES WELCOME

SOPHISTICATED POWER

TOP–DOS offers professional features found only in systems on much larger machines. Whatever your experience level, you will appreciate the flexibility and power of this advanced system. Here are some examples of TOP–DOS's powerful features:

MACHINE LANGUAGE MONITOR & EDITOR allows you to access and change bytes in memory.
COMMAND FILE CAPABILITY permits you to simply and rapidly execute a complex sequence of commands.
"HELLO" FILE executes automatically on boot-up.
SET COMMAND enables you to customize your system: Configure disk drives and select TOP–DOS options.
FILE DIRECTORY COMMAND lets you choose: Alphabetization, the number of columns in the listing, and the inclusion of deleted & open files.
MEMORY MAP shows you the memory areas used by the Binary Load command.
ONE-LINE COMMANDS saves you time and conserves screen space, once you are familiar with the command syntax.
DOS–RESIDENT OPTION speeds your transfer between TOP–DOS & BASIC, or other programs.

TOP-DOS includes all the features of its predecessor, DOS-MOD. TOP-DOS and DOS-MOD are trademarks of ECLIPSE. ATARI is a registered trademark of Atari, Inc. Prices are subject to change without notice. Shipping charges are prepaid. California residents add 6.5% sales tax ($49.95 + 3.25 = $53.20).

CIRCLE #121 ON READER SERVICE CARD

Create-A-Font Datamaker

16K Cassette or Disk

by E.K. Garringer

Create-A-Font, published in February's **ANALOG**, issue 16, makes redefining a character set easy enough for a child — and as entertaining as a game. But, after you have enjoyed creating your own special character set, you may have asked, "What now?"

Font Datamaker makes using your character set as easy as creating it. It does all the work for you. Here's how to use it.

1. Type in **Font Datamaker** and SAVE it to tape or diskette.
2. Design a character set with **Create-A-Font** and SAVE it to disk or tape.
3. RUN **Font Datamaker** and follow the prompts: (a) enter the *complete* name of your data file saved by **Create-A-Font**; (b) enter the *complete* name of the subroutine you will be creating; and (c) enter the line number with which you would like the subroutine to begin.

At this point, **Datamaker** will read your font file into memory and then ask you to insert your output diskette or tape. Your font will be compared to the ROM internal character set. Only those characters which have been changed will generate data statements for the subroutine. This effectively cuts down the length of the subroutine and the amount of memory required to store your program.

As **Datamaker** works, it will generate your subroutine and write it to diskette or tape. This process takes about one minute. When complete, **Datamaker** returns control to BASIC, and the READY prompt appears on the screen.

Note: The disk write process occurs only when the buffer is full. Some drives shut down between writes. **Datamaker** is not finished until the READY prompt appears!

Font Datamaker generates a subroutine which does all the work required to store and access your redefined character set in memory. To use your subroutine, follow these steps:

the XL boss

for ATARI* 800XL* and 1200XL* computers

providing the the greatest compatibility with all atari* software and featuring:

- no disk or translator to load
- full compatability with Text Wizard, Letter Perf., games and many programs that a translator just can't handle
- proper RESET operation
- easy user access to extra RAM for word processing, P.M. ,etc. ,etc.
- power up push button commands for all RAM, extra RAM, BASIC [one touch]
- warmstart with coldstart option and more!
- easy no-solder 10 min installation

Price: 800XL US79.95 1200XL US89.95
Terms: check, M.O., cashiers chk
No charge cards--cod add $5
foreign add $3-No cod outside US
Calif. res. add 6/6.5%
subject to price change and availability

POB 2205 REDONDO BEACH CA 90278 (213)3764105

* trademark of atari, inc.

Allen Macroware

EPSON*, NEC*, PROWRITER*, GEMINI*, OKIDATA 92*
OKIDATA 82A/OKIGRAPH, M-T SPIRIT, DMP-80, PANASONIC KXP-1070

NEW! ATARI*

Allen Macroware

PRINTWIZ

The only self-booting grafix handler for dumps in **horizontal** format — all mach. lang. — **Lister** incl. — all modes — mixed modes — change aspect ratios, etc. while running other programs — assem ed — basic or no cartridge — demos, utilities, fonts, included — dump, create forms, stationery, calendars, requires interface. **$29.95**

diskwiz-II

Now for single/double density. Repair, explore, alter, duplicate, map, speedcheck, bad sector (810), block move, trace, special print capabilities, disassembler, new speed, new ease, new functions, special printing functions, excellent repair tool w/instr. — even better than before! The best repair/editor/duplicator at any price — still at the lowest price. (Updates avail. for a small fee.) **$29.95**

1st Class Postage Paid
California Residents add 6%, Foreign Orders add $2.50
C.O.D. add $3.00 — No credit cards
Prices subject to change
(213) 376-4105

ALLEN MACROWARE
P.O. Box 2205
Redondo Beach, CA 90278

** Indicates Trademark of non-related company*

CIRCLE #122 ON READER SERVICE CARD

INTRODUCING

the NEW ASTRA 2001

Double Density Double Disc Drive

- ☐ More Reliable
- ☐ Quieter Operation
- ☐ Faster Read Write
- ☐ More Capacity (360 K Bytes)
- ☐ Easier Data Read

ALL THIS WRAPPED UP IN ONE ATTRACTIVE UNIT

ASTRA SYSTEMS INC.
2500 S. FAIRVIEW, UNIT L SANTA ANA, CA 92704
Call **(213) 804-1475** for your nearest dealer.

CIRCLE #123 ON READER SERVICE CARD

1. Write and debug your program using the ROM character set for trial runs.

2. When satisfied, ENTER the routine created by **Font Datamaker.**

3. Start your program with GOSUB X, where X is the beginning line number of your subroutine.

4. To access your character set, [POKE 756, CHSET/256] after each GRAPHICS call.

5. SAVE your complete program, with subroutine intact, to cassette or diskette.

6. Press SYSTEM RESET before RUNning your program and between RUNs.

Additional information.

As stated in **Create-A-Font**, a scrolling text window can destroy your character set. If you wish to use a scrolling window, protect your character set by changing Line 340 to:

```
340 ? #2;LINO;" POKE 106,PEEK(106)-8:G
RAPHICS 0:CHSET=(PEEK(106)+4)*256:? ";
CHR$(34);"ONE MOMENT";CHR$(34)
```

If your player will be using Player/Missile Graphics, location 106 must be adjusted to suit your specific application.

You may wish to save the original contents of location 106 to a numeric variable before the GOSUB X [i.e., RTOP = PEEK(106)]. □

BASIC listing.

```
10 DIM INFILE$(14),OUTFILE$(14)
20 REM CREATE-A-FONT DATAMAKER
25 REM by E. K. GARRINGER
30 REM CREATES A SUBROUTINE FROM
35 REM YOUR CREATE-A-FONT SAVED
40 REM CHARACTER SET.  THIS ROUTINE
45 REM WILL BE WRITTEN TO DISK
50 REM OR CASSETTE IN LIST FORMAT.
60 REM FOLLOW THE IMBEDDED INSTRUC-
65 REM TIONS TO CREATE A DATA
70 REM SUBROUTINE FROM YOUR SAVED
75 REM CREATE-A-FONT FILE.
80 REM WRITE AND DEBUG YOUR PROGRAM.
90 REM ENTER YOUR DATA SUBROUTINE.
100 REM BEGIN YOUR PROGRAM WITH A
105 REM GOSUB TO YOUR SUBROUTINE.
110 REM AFTER EACH GRAPHICS CALL
120 REM POKE 756,CHSET/256 TO ACCESS
130 REM YOUR REDEFINED CHARACTER SET.
140 REM SAVE YOUR PROGRAM WITH THE
145 REM NEW SUBROUTINE INTACT.
150 REM PRESS SYSTEM RESET BETWEEN
155 REM RUNS.
160 POKE 106,PEEK(106)-5:GRAPHICS 0:PO
KE 710,0:CHSET=(PEEK(106)+1)*256:? "IN
SERT INPUT MEDIA"
170 ? "ENTER NAME OF CHARACTER SET FIL
E.":? "DEVICE:FILENAME.EXT":TRAP 570:I
NPUT INFILE$:IF INFILE$="" THEN 170
180 ? "ENTER NAME OF OUTPUT FILE .":?
"DEVICE:FILENAME.EXT":INPUT OUTFILE$:T
RAP 40000:IF OUTFILE$="" THEN 180
190 ? "ENTER BEGINNING LINE # FOR SUBR
OUTINE":TRAP 570:INPUT LINO:TRAP 40000
200 IF INFILE$(1,1)="C" THEN ? "K↖POSI
TION TAPE, PRESS PLAY.":? "PRESS RETUR
N"
210 REM CIO LOAD
220 CLOSE #1:TRAP 580:OPEN #1,4,0,INFI
LE$:IO=16
230 IOCB=832+IO:POKE IOCB+2,7
240 POKE IOCB+4,0:POKE IOCB+5,CHSET/25
6
250 POKE IOCB+8,0:POKE IOCB+9,4
260 XFR=USR(ADR("hhh◘LVd"),IO)
270 CLOSE #1:POKE 764,255
280 IF OUTFILE$(1,1)="C" THEN ? "POSIT
ION OUTPUT TAPE":? "K↖PRESS PLAY AND R
ECORD":? "PRESS RETURN":GOTO 310
290 ? "K↖INSERT OUTPUT DISK AND PRESS
A KEY"
300 IF PEEK(764)=255 THEN 300
310 CLOSE #2:TRAP 590:OPEN #2,8,0,OUTF
ILE$
320 GRAPHICS 0:POKE 710,0:? "SAVING FI
LE - DO NOT DISTURB!"
330 ? "THIS TAKES ONE MINUTE AND WILL"
:? "RETURN TO BASIC WHEN FINISHED."
340 ? #2;LINO;" POKE 106,PEEK(106)-5:G
RAPHICS 0:CHSET=(PEEK(106)+1)*256:? ";
CHR$(34);"ONE MOMENT";CHR$(34)
350 ? #2;LINO+5;" CHI=CHSET/256:CLO=0:
POKE 203,CLO:POKE 204,CHI"
360 ? #2;LINO+10;" DIM XFR$(28):RESTOR
E ";LINO+20;":FOR N=1 TO 28:READ ML:XF
R$(N,N)=CHR$(ML):NEXT N"
370 ? #2;LINO+20;" DATA 104,169,0,133,
205,168,169,224,133,206,177,205,145,20
3,200,208"
380 ? #2;LINO+30;" DATA 249,230,204,23
0,206,165,206,201,228,208,239,96"
390 ? #2;LINO+40;" XFR=USR(ADR(XFR$))"
400 ? #2;LINO+50;" RESTORE ";LINO+90
410 ? #2;LINO+60;" READ A:IF A=-1 THEN
 RETURN"
420 ? #2;LINO+70;" FOR Z=0 TO 7:READ J
:POKE CHSET+A*8+Z,J:NEXT Z"
430 ? #2;LINO+80;" GOTO ";LINO+60
440 LIN2=LINO+80
450 REM SELECT ONLY REVISED CHARACTERS
460 FOR CH=0 TO 127:FLAG=0
470 FOR J=0 TO 7
480 IF PEEK(CHSET+CH*8+J)<>PEEK(57344+
CH*8+J) THEN FLAG=1
490 NEXT J
500 IF  NOT FLAG THEN 530
510 LIN2=LIN2+10
520 ? #2;LIN2;" DATA ";:? #2;CH;:FOR J
=0 TO 7:? #2;",";PEEK(CHSET+CH*8+J);:N
EXT J:? #2
530 NEXT CH
540 ? #2;LIN2+10;" DATA -1"
550 CLOSE #2:GRAPHICS 0:POKE 710,148
560 ? "↖YOUR NEW SUBROUTINE IS":END
570 ? "K↖BAD INPUT-TRY AGAIN":INFILE$=
"":OUTFILE$="":FOR DE=1 TO 500:NEXT DE
:GOTO 30
580 ? "K↖CANNOT READ INPUT FILE-TRY AG
AIN":INFILE$="":OUTFILE$="":FOR DE=1 T
O 500:NEXT DE:TRAP 40000:GOTO 30
590 ? "K↖FILE OUTPUT PROBLEM.  REWIND
TAPE":? "OR CHANGE DISKETTE AND PRESS
A KEY":? "TO TRY AGAIN."
600 TRAP 40000:POKE 764,255
610 IF PEEK(764)=255 THEN 610
620 GOTO 280
```

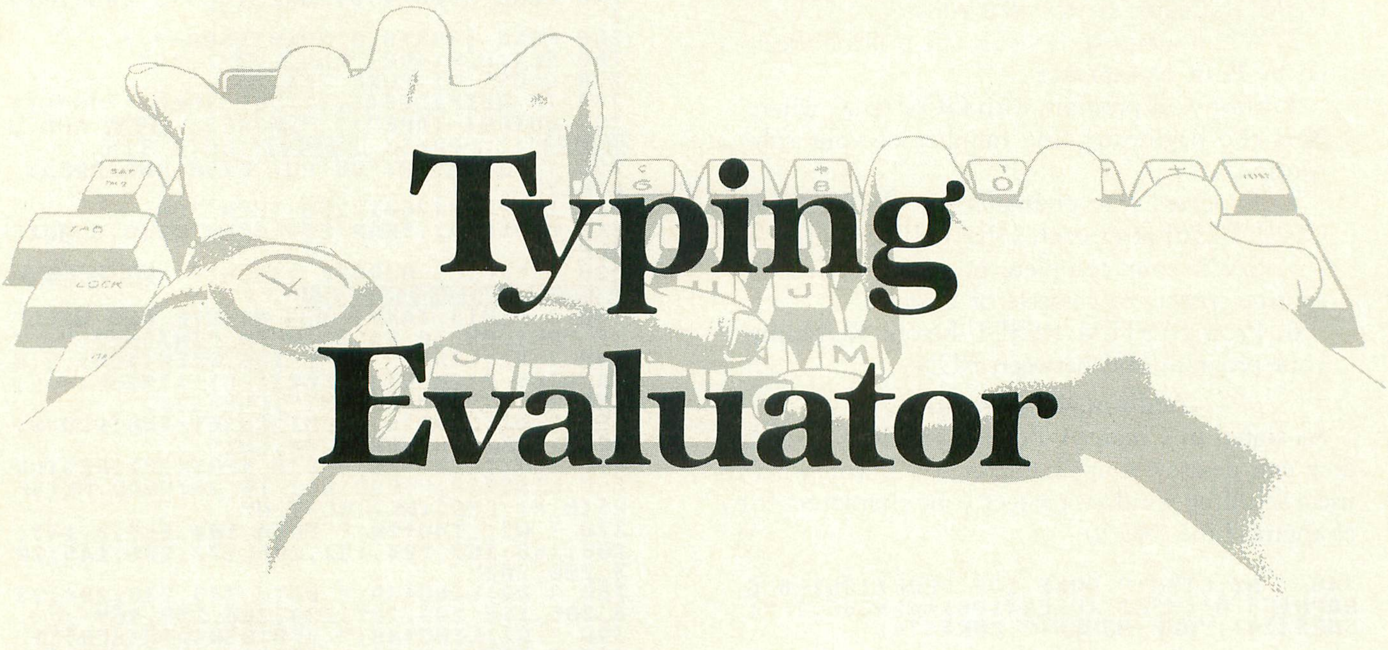
●

CHECKSUM DATA.

(see page 43)

```
10 DATA 165,123,602,514,17,832,344,193
,894,920,913,992,46,359,569,7483
105 DATA 858,276,390,318,961,405,167,6
5,677,929,781,340,984,461,872,8484
230 DATA 748,115,210,591,358,67,391,51
0,469,638,833,933,105,401,955,7324
380 DATA 500,744,630,963,107,147,393,4
11,218,314,304,763,350,321,101,6266
530 DATA 639,217,565,11,292,621,818,38
5,524,725,4797
```

●



Typing Evaluator

16K Cassette or Disk

by William Abell, Jr.

Have you ever spent *hours* typing in one of the very good programs provided in **ANALOG**, because your typing ability was at or near zero? This program may be able to help you. It provides you with a screen full of random words to allow you to practice your typing technique. The score, printed at the bottom of the screen after each time trial, provides an incentive to improve both your speed and accuracy.

How to use the program.

When the program is run, it will first display a title screen, then an introduction screen which explains how to use the program. At the bottom of the screen is the prompt to press the START button to begin. As soon as the button is pressed, the screen goes blank, while the computer makes up random length words. A few seconds later, the screen comes alive again with seven lines of text — which could be a message in secret code or a foreign language. In reality it is neither, simply characters chosen at random by your Atari computer.

The cursor is positioned directly under the first character of the first line, ready to make its journey to the bottom right of your screen. As soon as you type the first character, the computer stopwatch is started, so that your typing speed can be calculated. The computer also keeps track of the errors that you make as you type, so that your accuracy can be calculated, too. As you type the last character of each line, the cursor is automatically positioned under the first character of the next line of text.

If you make an error, do not attempt to go back and correct it. All keys are disabled except for the letter keys and the spacebar, so any attempt at correction would be fruitless. As soon as the last character is typed, the computer looks at its stopwatch, computes your speed and displays it on the screen. The percent of characters typed correctly is also displayed.

To try again, simply press the START button and, in a few seconds, a brand new screen full of text will be displayed. To end the program, you must press the BREAK key.

Program customization.

As you become proficient at typing the alphabet, you may want to expand the program to include other characters. The program is well documented, so you should have no trouble customizing it to your own desires. For example, if you wanted to practice typing numbers, you could change the second statement in Line 150 to: T=RND(0)*9+48 and Line 410 to: IF K<48 OR K>57 THEN 380. The change to Line 150 alters the ATASCII codes randomly generated to those between 48 and 57, which corresponds to all the numbers from 0 to 9. The change to Line 410 alters the values of the ATASCII codes accepted from the keyboard to numbers only, plus the spacebar which is accepted on Line 400.

Those wanting to get really ambitious might consider modification of the program, so that it randomly generates real words, rather than words made up of random characters. One way to accomplish this is to load words of different lengths into a pseudo-table, which is then accessed randomly and loaded into the string, T$. □

Program Description.

Line 50 — Selects subroutine to display title screen (optional).

Line 60 — DIMensions string variables.

Line 70 — Selects subroutine to display the introduction screen.

Line 80 — Stops program execution until START button is pushed.

Line 90 — Initializes variables.

Line 130 — Turns off DMA to allow faster character generation.

Line 140 — Generates a random length for next word.

Line 150 — For/next loop to generate random characters for the word and place them in the string, T$.

Line 160 — Provides trailing space for the word and checks for end of text.

Lines 170 - 200 — If within nine spaces of the end of the line, define length of next word to even out line and increment line number.

Lines 240 - 270 — Clear screen and turn on DMA. Print T$ to screen with two spaces between each line and position cursor under first character.

Lines 310 - 340 — Line 320 stops program execution until the first key is depressed. Line 330 resets the Atari's timer to 0 (starts the stopwatch).

Lines 400 - 410 — Rejects all keys except letters and spacebar (other keys may be included by changing the numeric values in these statements).

Lines 420 - 430 — LOADS the typed character into the string, R$, prints the character, and positions the cursor under the first character of the next line, when the end of line is reached.

Line 440 — routes program to calculation of results section when last character is typed.

Line Line 490 — Determines elapsed time in minutes by reading Atari's timer (stops stopwatch).

Line 500 — Calculates typing speed in words per minute.

Lines 510 - 530 — Compares random character string, T$, to typed character string, R$, to determine number of correct entries.

Line 540 — Calculates the percent accuracy.

Lines 550 - 560 — Prints results.

Line 580 — Stops program execution until START button is pressed.

Lines 630 - 690 — Title screen subroutine. Mixes graphics 1 and graphics 2 characters on screen by using display list manipulation in Line 640. Line 670 provides a time delay for viewing title screen.

Lines 730 - 830 — Introduction text to explain how to use program.

Note: Typing requirements can be shortened considerably by removing the REM statements and the title and introduction screen subroutines. If subroutines are omitted, then remove Lines 50, 70 and 80.

Variables description.

LENGTH	Length of random word
T	ATASCII code of random character
T$	String containing random text
LINE	Sequence number of line
I	Sequence # of random character
C	Sequence # of typed character
R$	String containing typed characters
N	Counting variable
K	ATASCII code of typed character
MIN	Elapsed time in minutes
RATE	Typing speed in words per minute
COR	Number of characters correct
ACC	Percent of characters correct
DL	Display list pointer
DELAY	Time delay loop counting variable

BASIC listing.

```
10 REM      TYPING EVALUATOR
20 REM             BY
30 REM         BILL ABELL
40 REM            1984
50 GOSUB 630
60 DIM T$(300),R$(300)
70 GOSUB 730
80 IF PEEK(53279)<>6 THEN 80
90 LINE=0:I=1:C=1
100 REM *****************************
110 REM *    GENERATE RANDOM TEXT   *
120 REM *****************************
130 POKE 559,0
140 LENGTH=INT(RND(0)*8+1)
150 FOR N=1 TO LENGTH:T=RND(0)*25+65:T
$(I,I)=CHR$(T):I=I+1:NEXT N
160 T$(I,I)=" ":IF LINE=7 THEN 240
170 I=I+1:IF I-38*LINE=29 THEN LENGTH=
9:GOTO 200
180 IF I-38*LINE<30 THEN 140
190 LENGTH=38-(I-38*LINE)
200 LINE=LINE+1:GOTO 150
210 REM *****************************
220 REM *    PRINT TEXT TO SCREEN   *
230 REM *****************************
240 ? "↖"
250 POKE 559,34
260 FOR N=0 TO 6:? T$((1+38*N),(38+38*
N)):? :NEXT N
270 POSITION 1,2:? " ";
```

```
280 REM ******************************
290 REM * CATCH FIRST KEY DEPRESSED *
300 REM ******************************
310 OPEN #1,4,0,"K:"
320 IF PEEK(764)=255 THEN 320
330 POKE 18,0:POKE 19,0:POKE 20,0:REM
**RESET CLOCK**
340 GOTO 380
350 REM ******************************
360 REM * PRINT & STORE KEYS PUSHED *
370 REM ******************************
380 IF PEEK(764)=255 THEN 380
390 GET #1,K
400 IF K=32 THEN 420
410 IF K<65 OR K>90 THEN 380
420 R$(C,C)=CHR$(K):? CHR$(K);:C=C+1:F
OR N=0 TO 5:IF C=38*N+38 THEN POSITION
 1,3*N+5:? " ";:R$(C,C)=" ":C=C+1
430 NEXT N
440 IF C=266 THEN GOTO 490
450 GOTO 380
460 REM ******************************
470 REM *  TABULATE & PRINT RESULTS *
480 REM ******************************
490 MIN=(PEEK(18)*256*256+PEEK(19)*256
+PEEK(20))/3600
500 RATE=INT(259/MIN/6+0.5):COR=0
510 FOR N=1 TO 259
520 IF T$(N,N)=R$(N,N) THEN COR=COR+1
530 NEXT N
540 ACC=INT(COR/260*100+0.5)
550 POSITION 7,21:? "RATE = ";RATE;" W
ORDS PER MINUTE"
560 POSITION 12,22:? "ACCURACY = ";ACC
;"%":CLOSE #1
570 ? "      █ PRESS START TO TRY AGAIN
";
580 IF PEEK(53279)<>6 THEN 580
590 POKE 764,255:GOTO 90
600 REM ******************************
610 REM *        TITLE SCREEN        *
620 REM ******************************
630 GRAPHICS 17:SETCOLOR 4,1,10
640 DL=PEEK(560)+256*PEEK(561)+4:POKE
DL+7,7:POKE DL+12,7
650 POSITION 2,6:? #6;"typing evaluato
r":POSITION 9,9:? #6;"BY"
660 ? #6:? #6;"      BILL ABELL"
670 FOR DELAY=1 TO 1000:NEXT DELAY
680 GRAPHICS 0
690 RETURN
700 REM ******************************
710 REM *    INTRODUCTION SCREEN     *
720 REM ******************************
730 ? "K":? :? "              TYPING EVAL
UATOR":?
740 ? "This program provides a means o
f      improving touch typing skills.
 A     group of words is presented ";
750 ? "made up ofrandom letters.  Thes
e words should becopied exactly as pre
sented ";
760 ? "on the    screen.  As soon as t
he last letter ofeach line is typed, c
ontinue with the first letter";
770 ? " of the next line."
780 ? :? "Measurement of your typing s
peed will start when the first key is
pressed.  Rate and accuracy";
790 ? " results will be    printed at
 the end of the timed trial."
800 ? :? "           Press START to begin
.":?
810 ? " █ The computer will take about
 5 █"
820 ? " █ seconds to make up the words
. ";
830 RETURN
```

•

CHECKSUM DATA.
(see page 43)

```
10 DATA 335,81,610,121,763,765,769,679
,267,778,773,784,780,366,172,8043
160 DATA 271,212,654,438,306,783,906,7
89,376,4,849,816,804,158,782,8148
310 DATA 265,520,496,727,797,145,803,5
50,581,439,668,753,753,277,732,8506
460 DATA 802,72,808,4,293,356,844,755,
613,964,329,524,866,145,788,8163
610 DATA 249,794,263,854,851,637,652,9
18,618,790,875,796,447,26,761,9531
760 DATA 540,911,205,788,674,328,307,6
04,4357
```

•

Attention Programmers!

ANALOG Computing is interested in programs, articles, and software review submissions dealing with the Atari 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.

ANALOG 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 **ANALOG Computing**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **ANALOG**. 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, **ANALOG Computing**, P.O. Box 23, Worcester, MA 01603.

Money Hungry

16K Cassette or 24K Disk

by Donald P. Murphy

This month's assembly language game, **Money Hungry**, is a game of skill for one player.

Typing it in.

Before typing anything, look at the listings accompanying this article.

Listing 1 is the BASIC data and data checking routine. This listing is used to create both cassette and disk versions of **Money Hungry**. The data statements are listed in hexadecimal (base 16), so the program will fit in 16K cassette systems. This makes typing more difficult, but it's a necessary evil.

Listing 2 is the assembly language source code for the game of **Money Hungry**, created with the OSS MAC/65 assembler. You *do not* have to type this listing to play the game! It is included for those readers interested in assembly language.

Follow the instructions below to make either a cassette or disk version of **Money Hungry**.

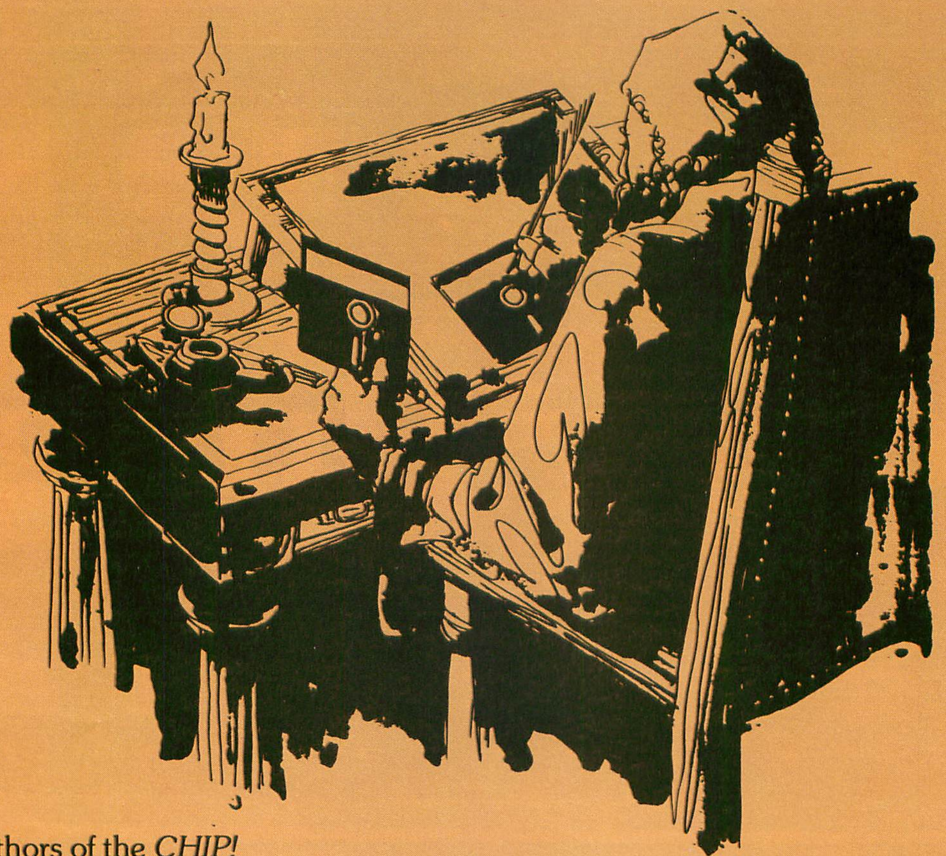
Cassette instructions.

1. Type Listing 1 into your computer using the BASIC cartridge and verify your typing with **C:CHECK** (see page 47).
2. Type RUN and press RETURN. The program will begin and ask:

```
MAKE CASSETTE (0) OR DISK (1)?
```

Type 0 and press RETURN. The program will begin checking the DATA statements, printing the line number of each as it goes. It will alert you if it finds any problems. Fix any incorrect lines and re-RUN the program, if necessary, until all errors are eliminated.

SpartaDOS and ARCHIVER II™



From the authors of the *CHIP!*

SpartaDOS is our all new command driven DOS fully compatible with all disk drives made for the Atari Home Computers.™ Supports all Atari compatible densities - both single and double sided, 5¼ and 8 inch.

UltraSpeed™ I/O allows read/write 2 to 4 times faster than standard!* Comprehensive utilities include: a sector copier, RS-232 handler, and DOS file translators.

ARCHIVER II allows UltraSpeed backup of all diskettes and compacting of multiple programs on each disk. Includes an exceptional editor, a speed check, and is fully automatic.

Available now for Atari 810 and 1050 drives! Coming soon for most other brands.

Suggested Retail Price $129.95 (plus shipping and installation)

See SpartaDOS and ARCHIVER II now at leading computer stores or order direct from:

Spartan Software
Div. of ICD, Inc.
828 Green Meadow Ave.
Rockford, IL 61107
815-229-2999 (1:00-9:00 p.m. CST)

*Speed depends on drive hardware. A chip replacement is required for most drives.

3. When all your DATA lines are correct, the computer will beep twice and prompt you to "READY CASSETTE AND PRESS RETURN." Insert a blank cassette in your recorder, press the RECORD and PLAY buttons simultaneously and hit RETURN. The message "WRITING FILE" will appear, and the program will create a machine language boot tape version of **Money Hungry**, printing each DATA line number as it goes. When the READY prompt appears, the game is recorded and ready to play. CSAVE the BASIC program onto a separate tape before continuing.

4. To play the game, rewind the tape created by the BASIC program to the beginning. Turn your computer OFF and remove all cartridges. Press the PLAY button on your recorder and turn ON your computer while holding down the START key. If you have a 600 or 800XL computer, you must hold the START and OPTION keys when you turn on the power. The computer will "beep" once. Hit the RETURN key and **Money Hungry** will load and run automatically.

Disk instructions.

1. Type Listing 1 into your computer, using the BASIC cartridge and verify your typing with **D:CHECK2** (see page 43).

2. Type RUN and press RETURN. The program will ask:

```
MAKE CASSETTE (0) OR DISK (1)?
```

Type 1 and press RETURN. The program will begin checking the DATA lines, printing the line number of each statement as it goes. It will alert you if it finds any problems. Fix incorrect lines and re-RUN the program, if necessary, until all errors are eliminated.

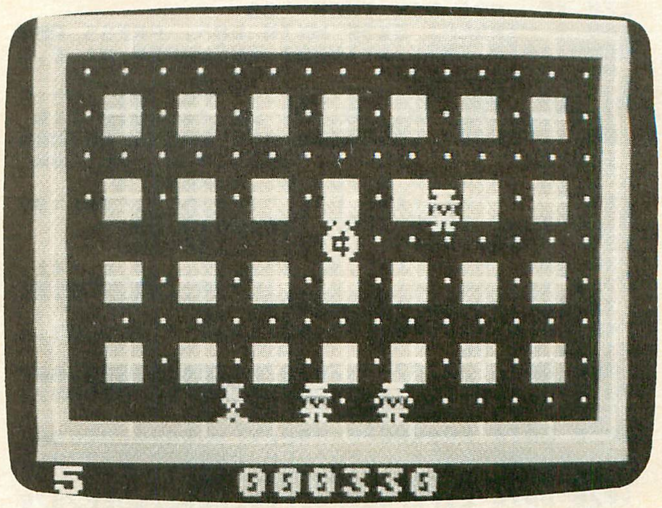
3. When all DATA lines are correct, you will be prompted to "INSERT DISK WITH DOS, PRESS RETURN." Put a disk containing DOS 2.0S into drive #1 and press RETURN. The message "WRITING FILE" will appear, and the program will create an AUTORUN.SYS file on the disk, displaying each DATA line number as it goes. When the READY prompt appears, the game is ready to play. Be sure the BASIC program is SAVEd before continuing.

4. To play the game, insert the disk containing the AUTORUN.SYS file into drive #1. Turn your computer OFF, remove all cartridges and turn the computer back ON. **Money Hungry** will load and run automatically.

Playing the game.

Money Hungry requires one joystick, plugged into port 1. The game is started by pressing START or the joystick button.

In **Money Hungry**, you play the part of a thief trying to collect as much money as possible, while making every effort to avoid the police. You are placed in a maze, with coins scattered all around. You pick up the coins by moving over them. Once on every board, a money bag appears, containing a 100 coin bonus. Your score, in coins, is shown at the bottom of the screen.



Money Hungry.

There are three police patrolling the maze, and if they catch you five times, the game is over. The number of tries you have remaining is shown at the lower left of the screen.

You can block the police by placing doors in the maze. To do this, press the joystick button while moving through the maze. A red door will appear, and the police can't get through it. You have four such doors and can reopen them at any time simply by running over them. Once you pick up a door, you can reuse it by pressing the joystick button again.

When you have picked up all of the coins in the maze, you will move to the next level. In each level, either you or the police will increase in speed. Good luck! □

Listing 1.
BASIC listing.

```
10 REM *** MONEY HUNGRY ***
20 TRAP 20:? "MAKE CASSETTE (0), OR DI
SK (1)";:INPUT DSK:IF DSK>1 THEN 20
30 TRAP 40000:DATA 0,1,2,3,4,5,6,7,8,9
,0,0,0,0,0,0,0,10,11,12,13,14,15
40 DIM DAT$(91),HEX(22):FOR X=0 TO 22:
READ N:HEX(X)=N:NEXT X:LINE=990:RESTOR
E 1000:TRAP 120:? "CHECKING DATA"
50 LINE=LINE+10:? "LINE:";LINE:READ DA
T$:IF LEN(DAT$)<>90 THEN 220
60 DATLIN=PEEK(183)+PEEK(184)*256:IF D
ATLIN<>LINE THEN ? "LINE ";LINE;" MISS
ING!":END
70 FOR X=1 TO 89 STEP 2:D1=ASC(DAT$(X,
X))-48:D2=ASC(DAT$(X+1,X+1))-48:BYTE=H
EX(D1)*16+HEX(D2)
80 IF PASS=2 THEN PUT #1,BYTE:NEXT X:R
EAD CHKSUM:GOTO 50
90 TOTAL=TOTAL+BYTE:IF TOTAL>999 THEN
TOTAL=TOTAL-1000
```

```
100 NEXT X:READ CHKSUM:IF TOTAL=CHKSUM
 THEN 50
110 GOTO 220
120 IF PEEK(195)<>6 THEN 220
130 IF PASS=0 THEN 170
140 IF  NOT DSK THEN 160
150 PUT #1,224:PUT #1,2:PUT #1,225:PUT
 #1,2:PUT #1,68:PUT #1,33:CLOSE #1:END
160 FOR X=1 TO 107:PUT #1,0:NEXT X:CLO
SE #1:END
170 IF  NOT DSK THEN 200
180 ? "INSERT DISK WITH DOS, PRESS RET
URN";:DIM IN$(1):INPUT IN$:OPEN #1,8,0
,"D:AUTORUN.SYS"
190 PUT #1,255:PUT #1,255:PUT #1,158:P
UT #1,32:PUT #1,10:PUT #1,44:GOTO 210
200 ? "READY CASSETTE AND PRESS RETURN
";:OPEN #1,8,128,"C:":RESTORE 230:FOR
X=1 TO 40:READ N:PUT #1,N:NEXT X
210 ? :? "WRITING FILE":PASS=2:LINE=99
0:RESTORE 1000:TRAP 120:GOTO 50
220 ? "BAD DATA: LINE ";LINE:END
230 DATA 0,24,118,32,157,32,169,0,141,
47,2,169,60,141,2,211,169,0,141,231,2,
133,14,169,56,141,232,2
240 DATA 133,15,169,68,133,10,169,33,1
33,11,24,96
1000 DATA 7070707070707047B42070067007
7070707006419E2000000000002D2F2E25390028
352E27323900000000000000000,530
1010 DATA 00000000000000E2F90000000000000
000000000000646F6E616C6400006D75727068
79000000F0F2E5F3F300F3F4,992
1020 DATA E1F2F400F4EF00E2E5E7E9EE7070
7045003A050505050505050505050741042136
147F362A7F1C3E00FA8A8ABA,191
1030 DATA E6E6FFBAFF771C08143E1C1C3E00
388888EAEA38EAEA38A7A1ADA5AFB6A5B2A200
BDFD279D0038BDFD289D0039,996
1040 DATA BDFE299D003ABDFE2A9D003BCAD0
E5A9208D3102A99E8D3002A9228D2F02AD1FD0
C906F005AD8402D0F4A9048D,276
1050 DATA 3002A9218D3102A9388DF402A988
8DC402A9FC8DC502A9368DC602A9008D6F028D
08D2A9038D320220692220BB,838
1060 DATA 22206B23200E238D1ED020712620
F72220DA26A200AD0CD0F0034C0D278D0AD4AC
0BD4B900188D12D0B900198D,901
1070 DATA 13D0B9001A8D14D0B9001B8D15D0
A5CDC96B90D44C9726C6B6F0034C4426A5B785
B6A91E85B1A98085B0A91985,499
1080 DATA B3A90085B2AABD05D02904F0138D
1ED0B58CD589D0044901958CB5894901958986
C1B5B9186901B4BEC8209525,948
1090 DATA A6C1B58C2902F007A588F0074C51
22A587D00BB58C9589AD0AD22903958CB589D0
034CD525C901D0034CE825C9,323
1100 DATA 02D0034CFB254C1D26A9008DDB3A
8DDC3AA90085B0A93A85B1A006B1B0F00EC91E
F00AC91FD00A88A90091B0C8,594
1110 DATA A91191B0C8C8C02490E4A5B01869
2885B0A5B1690085B1A006C93B90D1A5B0C990
D0CBA90085CD85BCA905858F,831
1120 DATA 60A91E85B1A90085B0A891B0C8D0
FBE6B1A6B1E020D0F3A91885B1A90085B0A891
B0C8D0FBE6B1A6B1E01CD0F3,101
1130 DATA A92E8D2F02A9038D1DD0A91C8D07
D460A906A223A091205CE4A9408D0FD4A907A2
21A0F04C5CE4A98085B0A91E,311
1140 DATA 85B1A90085B2A91985B3A017A208
BD182191B0BD212191B2C8CA10F2E6B3A5B018
698085B0A5B1690085B1C920,71
1150 DATA 90DBA94385B98D01D0A98485BA8D
02D0A9B385BB8D03D0A91785BE85BF85C0A903
8589858BA901858A60A208A0,74
1160 DATA 37BD2A2199001EBD3321990018C8
CA10F0A97C85B88D00D0A93785BDA90B85CB85
CC60A9008D01D28D03D2A6B8,363
1170 DATA E8A4BDC8205C2420A624C681F003
4C5924A5808581A5CC2903C903F007A584F007
4CC723A583D004A5CC85CBAD,972
1180 DATA 7802C90FF00285CCA5CBC90FD003
4C5924C90AF008C906F004C90ED020A20AA4BD
C018B0034C5924B9001E99FF,994
1190 DATA 1DB9001899FF17C8CAD0F0C6BD4C
5924C909F008C905F004C90DD023A5BDC95790
034C59246909A8A20AB9FF1D,379
1200 DATA 99001EB9FF1799001888CAD0F0E6
BD4C5924C90BD012A6B8E044B0034C5924CA86
B88E00D04C5924A6B8E0B390,36
1210 DATA 034C5924E886B88E00D04C5FE486
C584C69838E91085C3A90885C4A90085C2203E
258583A5C285C62901D004A9,495
1220 DATA 018583A5C538E93085C3A90485C4
A90085C2203E258584A5C285C529034901F004
A9018584A5C285C560A90085,754
1230 DATA B485CAA5CB2901D00AA5CCC90BD0
02C6C6E6C6A5C685C7A92885C8205E25E6C5A5
C51865C985B4A5CA693A85B5,184
1240 DATA A000B1B4C91ED011A9268D01D2A9
148D00D2A900A891B4E68FC91FD017A9268D01
D2A9148D00D2A900A891B4C6,713
1250 DATA B491B4E6B4E68FC911F01CC91DD0
2BA9A68D01D2A9DC8D00D2A26486BCC6B49891
B4E6B491B46098 91B4E6CDA2,739
1260 DATA 0186BCA95F8D01D2A9B98D00D260
A00838E5C40826C206C32A289005E5C44C5325
65C488D0EDB00365C41826C2,31
1270 DATA 60A90085C9A20846C790031865C8
6A66C9CAD0F385CA60A005E001B00160B9BF3B
C91B9009A91299BF3B8810F1,496
1280 DATA 6018690199BF3BCAD0E060858898
38E91085C3A90885C4A90085C2203E258587A5
C22901D004A9018587A58838,718
1290 DATA E93085C3A90485C4A90085C2203E
258588A5C229034901F004A901858860B4B9C0
B390034C4426C89895B99D01,890
1300 DATA D04C4426B4B9C044B0034C442688
9895B99D01D04C4426B4BEA20AC018B0034C44
26B1B08891B0C8B1B28891B2,379
1310 DATA C8C8CAD0F0A6C1D6BE4C4426A6C1
B5BEC95790034C44266909A8A20A88B1B0C891
B088B1B2C891B288CAD0F0A6,151
1320 DATA C1F6BE4C4426A5B018698085B0A5
B1690085B1E6B3A6C1E8E003B0034C0C22A6BC
A90085BC207425AD8402D003,701
1330 DATA 2050274C62E4A903858085B78581
85B6A013A90099B83B88D0FAA9578DBA3BA000
A91299BF3BC8C006D0F860A5,425
1340 DATA B7C580F008C680D00CA9018580C6
B7D004A90185B720C82620D22720692220BB22
200E23206B2320F72220DA26,68
1350 DATA 4CC121A906A2E4A05F205CE4A907
A2E4A0624C5CE4A9268D2702A9ED8D2602A901
A000A2044C5CE4A91C8DDB3A,245
1360 DATA A91D8DDC3AA9278D2702A9048D26
024CE426A9008DDB3A8DDC3A6020C826209127
AEBA3BCAE052B0034C33278E,950
1370 DATA BA3B20BB22206B23200E2320F722
8D1ED04CC121A200BD3C219DBA3BE8E004D0F5
A200BD40219DC63BE8E004D0,986
1380 DATA F54C7021C68FD003E68F60A584F0
17A5C52903D02DA9C08D02D2A9A38D03D2A000
A91E91B460A5C62901D016A9,676
1390 DATA C08D02D2A9A38D03D2C6B4A001A9
1F91B48891B460E68F60A9A88D03D2A9648590
A2FFA014A5908D02D2CAD0F8,151
1400 DATA 88D0F5E690E690A014A590C9A0D0
E9A9888D03D2A9328D02D2A2FFA0FFCAD0FD88
D0FAA9008D02D28D03D260A9,240
1410 DATA A88D01D2A94085 91A2FFA014A591
8D00D2CAD0F888D0F5C691C691A014A591D0EB
A9008D00D28D01D260000000,363
1420 DATA 0000000000555555AA5555555595
9595AA55555555555555AA959595959595959595
95959595151515151515 1515,466
1430 DATA 959595AA959595955565656AA5656
5656555555AA565656565656AA5555555554
545454545454545656565656,90
1440 DATA 5656565555555555555555959595
9555555555555656565655555555555555959595
959555555555565656560000,333
1450 DATA 0080000000000003C666E76663C00
0018381818187E00003C660C18307E00007E0C
180C663C00000C1C3C6C7E0C,345
1460 DATA 00007E607C06663C00003C607C66
663C00007E060C18303000003C663C66663C00
003C663E060C3800220A2AA0,697
1470 DATA A2A02A0A88A0280A2A0A28A0C0C0
C0C0C0C0C0C0000000FF000000000003C666E6E
603E0000183C66667E660000,516
1480 DATA 7C667C66667C00003C666060663C
0000786C66666C7800007E607C60607E00007E
607C606060000003E60606E66,134
```

```
1490 DATA 3E000066667E66666600007E1818
18187E000006060606663C0000666C78786C66
000060606060607E00006377,850
1500 DATA 7F6B6363000066767E7E6E660000
3C666666663C00007C66667C606000003C6666
666C3600007C66667C6C6600,328
1510 DATA 003C603C06063C00007E18181818
18000066666666667E0000666666663C180000
63636B7F7763000066663C3C,964
1520 DATA 6666000066663C18181800007E0C
1830607E00001E181818181E0000406030180C
06000078181818181878000008,686
1530 DATA 1C3663000000000000000000FF00
00000000050301010103010101030101010301
010103010101030101010301,167
1540 DATA 0101030101080A00000000000005
04001100110011001100110011001100110011
001100110011001100110011,456
1550 DATA 0B0A000000000000000050400110106
00110106001101060011010600110106001101
060011010600110B0A000000,692
1560 DATA 00000000504001100110011001100
11001100110011001100110011001100110011
00110B0A000000000000000504,986
1570 DATA 0011010600110106001101060011
010600110106001101060011010600110B0A00
00000000000005040000000000,201
1580 DATA 0000000000000000000000001100
11001100110011001100110B0A000000000000
050400110106001101060011,415
1590 DATA 0106001101060011010600110106
0011010600110B0A000000000000000504001100
110011001100110011001100,684
1600 DATA 1100110011001100110011001100
110B0A0000000000000005040011010600110106
001101060011010600110106,970
```

```
1610 DATA 001101060011010600110B0A0000
00000000050400110011001100110011001100
110011001100110011001100,269
1620 DATA 11001100110B0A0000000000000005
02010101020101010201010102010101020101
010201010102010101020101,385
1630 DATA 090A000000000000570000001212
1212121A000000000000000000000000000000
000000000000000000000000,607
1640 DATA 0000000000000000000000000000
00000000000000000000000000000000000000
000000000000000000000000,607
```

•

CHECKSUM DATA.

(see page 43)

```
10 DATA 518,351,496,811,423,729,200,60
3,555,573,694,613,29,205,967,7767
160 DATA 142,198,962,857,491,30,155,11
5,413,970,124,580,333,96,738,6204
1060 DATA 594,922,912,820,44,936,974,9
70,962,182,975,902,891,112,230,10426
1210 DATA 888,616,168,897,233,917,901,
766,732,776,96,805,870,834,30,9529
1360 DATA 69,860,982,25,249,872,323,63
7,266,402,391,538,446,222,507,6789
1510 DATA 284,117,562,457,631,493,510,
488,536,526,528,487,481,325,6425
```

•

```
; MONEY HUNGRY
;
; BY DONALD MURPHY
;
; ANALOG COMPUTING
;
;VARIABLES
;
LEVEL    =   $80     ;LEVEL #
DELAY    =   $81     ;DELAY INDEX
HERE     =   $82
NOX      =   $83     ;HORIZ. MOVE FLAG
NOY      =   $84     ;VERT. MOVE FLAG
TX       =   $85     ;X HOLD AREA
DM       =   $86     ;MISC. VARIABLE
XM       =   $87     ;HORIZ. FLAG
YM       =   $88     ;VERT. FLAG
DIRC     =   $89     ;ENMY CUR DIRECTION
DIRN     =   $8C     ;NEXT ENMY DIRECTION
NDOOR    =   $8F     ;# OF DOORS
TF3      =   $90
TF4      =   $91
POINTL   =   $B0     ;GENERAL POINTER
POINTH   =   $B1
TABPTL   =   $B2     ;TABLE POINTER
TABPTH   =   $B3
VBPNTL   =   $B4     ;VBLANK POINTER
VBPNTH   =   $B5
EDELAY   =   $B6     ;ENEMY DELAY
ELEVEL   =   $B7     ;ENEMY LEVEL
XPOSPØ   =   $B8     ;HORIZ. POS. PLR Ø
XPOSP1   =   $B9     ;HORIZ. POS. PLR 1
XPOSP2   =   $BA     ;HORIZ. POS. PLR 2
XPOSP3   =   $BB     ;HORIZ. POS. PLR 3
SCOUNT   =   $BC     ;SCORE COUNT TO ADD
YPOSPØ   =   $BD     ;VERT. POS. PLR Ø
YPOSP1   =   $BE     ;VERT. POS. PLR 1
YPOSP2   =   $BF     ;VERT. POS. PLR 2
YPOSP3   =   $CØ     ;VERT. POS. PLR 3
TEMPX    =   $C1     ;VERT. POS. PLR 4
DRES     =   $C2     ;DIVISION RESULT
LDIV     =   $C3     ;LOW DIVIDEND
DIVS     =   $C4     ;DIVISOR
XPOSSC   =   $C5     ;HOR. POS. ON SCRN
YPOSSC   =   $C6     ;VERT POS. ON SCRN
MUPL     =   $C7     ;LO MULTIPLIER
MUCN     =   $C8     ;MULTIPLICAND
RESL     =   $C9     ;RESULT
RESH     =   $CA
STICKC   =   $CB     ;STICK VALUE
STICKN   =   $CC     ;NEXT STICK VALUE
NDOTS    =   $CD     ;#DOTS EATEN
;
;HARDWARE & SHADOW LOCATIONS
;
COLORØ   =   $Ø2C4   ;PLAYFIELD COLORS
COLOR1   =   $Ø2C5
COLOR2   =   $Ø2C6
SDLSTL   =   $Ø230   ;DISP LIST POINTER
SDLSTH   =   $Ø231
CHBAS    =   $Ø2F4   ;CHAR BASE ADDR
GRACTL   =   $DØ1D   ;GRAPHICS CONTROL
SDMCTL   =   $Ø22F   ;DMA CONTROL
COLPMØ   =   $DØ12   ;P/M COLORS
COLPM1   =   $DØ13
COLPM2   =   $DØ14
COLPM3   =   $DØ15
HPOSPØ   =   $DØØØ   ;PLAYER HOR. POS.
HPOSP1   =   $DØØ1
HPOSP2   =   $DØØ2
HPOSP3   =   $DØØ3
PMBASE   =   $D407   ;P/M BASE ADDRESS
VCOUNT   =   $D4ØB   ;VERT. LINE COUNTER
WSYNC    =   $D4ØA   ;WAIT FOR SYNC
PØPF     =   $DØØ4   ;COLLISION REGISTERS
PØPL     =   $DØØC
P1PF     =   $DØØ5
HITCLR   =   $DØ1E   ;COLLISION CLEAR
RANDOM   =   $D2ØA   ;RANDOM #
NMIST    =   $D4ØF   ;NMI STATUS
AUDCTL   =   $D2Ø8   ;AUDIO CONTROL
AUDC1    =   $D2Ø1   ;AUD. CHANNELS
AUDC2    =   $D2Ø3
AUDF1    =   $D2ØØ   ;AUD. FREQUENCIES
AUDF2    =   $D2Ø2
CONSOL   =   $DØ1F   ;CONSOLE KEYS
SKCTL    =   $Ø232   ;SERIAL PORT CTRL
GPRIOR   =   $Ø26F   ;P/M PRIORITY
STICKØ   =   $Ø278   ;STICK Ø
STRIGØ   =   $Ø284   ;BUTTON Ø
;
;TABLES AND RAM ASSIGNMENTS
;
CTABØ    =   $18ØØ   ;PL.Ø COLORS
CTAB1    =   $19ØØ   ;PL.1 COLORS
CTAB2    =   $1AØØ   ;PL.2 COLORS
CTAB3    =   $1BØØ   ;PL.3 COLORS
CHSET    =   $38ØØ   ;CHARACTER SET
SCRAM    =   $3AØØ   ;SCREEN ADDRESS
PMRAM    =   $1CØØ   ;P/M ADDRESS
;
;PROGRAM START
;
         *=  $2Ø9E
;
;TITLE SCREEN DISPLAY LIST
;
TSCRD    .BYTE 112,112,112,112,112
         .BYTE 112,112,$47
         .WORD TSCRW
         .BYTE 112,6,112,7,112,112,112
         .BYTE 112,6,$41
         .WORD TSCRD
;
;TITLE SCREEN DATA
;
TSCRW    .BYTE Ø,Ø,Ø,Ø,45,47,46,37
         .BYTE 57,Ø,40,53,46,39,50,57
         .BYTE Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
         .BYTE 226,249,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
         .BYTE Ø,Ø,Ø,100,111,110,97
         .BYTE 108,100,Ø,Ø,109,117,114
         .BYTE 112,104,121,Ø,Ø,Ø,240,242
         .BYTE 229,243,243,Ø,243,244,225
         .BYTE 242,244,Ø,244,239,Ø,226
         .BYTE 229,231,233,238
;
;GAME BOARD DISPLAY LIST
;
DLIST    .BYTE 112,112,112,$45
         .WORD SCRAM
         .BYTE 5,5,5,5,5,5,5
         .BYTE 5,5,5,7,65
         .WORD DLIST
;
;PLAYER COLORS & BIT MAPS
;
ENBMAP   .BYTE 54,20,127,54
         .BYTE 42,127,28,62,Ø
ENCMAP   .BYTE $FA,$8A,$8A,$BA
         .BYTE $E6,$E6,$FF,$BA,$FF
PLBMAP   .BYTE 119,28,8,20,62
         .BYTE 28,28,62,Ø
PLCMAP   .BYTE $38,$88,$88,$EA
         .BYTE $EA,$38,$EA,$EA,$38
GAME     .BYTE $A7,$A1,$AD,$A5
OVER     .BYTE $AF,$B6,$A5,$B2
;
;GAME EXECUTION STARTS HERE!
;
INIT     LDX #Ø
INILP    LDA CHDAT,X ;SET UP
         STA CHSET,X ;CHAR SET
         LDA CHDAT+256,X
         STA CHSET+256,X
         LDA SCDAT,X ;AND SCREEN
         STA SCRAM,X
         LDA SCDAT+256,X
         STA SCRAM+256,X
         DEX
         BNE INILP
         LDA # >TSCRD ;SHOW...
         STA SDLSTH   ;TITLE SCREEN
         LDA # <TSCRD
         STA SDLSTL
         LDA #$22
         STA SDMCTL
PROG     LDA CONSOL   ;GET CONSOLE
         CMP #6       ;START PRESSED?
         BEQ GO       ;YES!
         LDA STRIGØ   ;TRIGGER PRESSED?
         BNE PROG     ;NO!
GO       LDA #DLIST&255 ;SHOW...
         STA SDLSTL   ;GAME SCREEN
         LDA #DLIST/256
         STA SDLSTH
         LDA #CHSET/256 ;POINT TO
         STA CHBAS    ;CHAR SET
         LDA #$88     ;SET COLORS
         STA COLORØ
         LDA #$FC
         STA COLOR1
         LDA #$36
         STA COLOR2
         LDA #Ø       ;SET...
         STA GPRIOR   ;PRIORITY
         STA AUDCTL   ;AND AUDIO
         LDA #3       ;RESET SOUNDS
         STA SKCTL
         JSR SETSC    ;SETUP SCREEN
         JSR SETPM    ;SETUP P/M
         JSR SETPL    ;SET YOU
         JSR SETEM    ;SET ENEMIES
         STA HITCLR   ;CLEAR COLLISIONS
         JSR INT      ;MISC. INIT.
         JSR SETVB    ;START VBLANK
         JSR PRIZE    ;INIT MONEY BAG
         LDX #Ø       ;ZERO X REG
;
;MAINLINE COLOR LOOP
;
COLOR    LDA PØPL     ;HIT ANYTHING?
         BEQ CONT     ;NO, CONTINUE
         JMP DEATH    ;YOU'RE DEAD!
CONT     STA WSYNC    ;WAIT FOR SYNC
         LDY VCOUNT   ;GET VERT LINE#
```

```
        LDA CTABØ,Y ;GET COLOR FOR
        STA COLPMØ  ;PLAYER Ø
        LDA CTAB1,Y ;GET COLOR FOR
        STA COLPM1  ;PLAYER 1
        LDA CTAB2,Y ;GET COLOR FOR
        STA COLPM2  ;PLAYER 2
        LDA CTAB3,Y ;GET COLOR FOR
        STA COLPM3  ;PLAYER 3
        LDA NDOTS   ;# OF DOTS...
        CMP #1Ø7    ;=1Ø7?
        BCC COLOR   ;NO, GET NEXT COLOR
        JMP NEXTL   ;NEW LEVEL!
;
;DEFERRED VBLANK
;
DVBL    DEC EDELAY  ;TIME FOR EN. MOVE?
        BEQ EDO     ;YES!
        JMP END     ;NO, EXIT
EDO     LDA ELEVEL  ;RESET MOVE TIMER
        STA EDELAY
        LDA #PMRAM/256+2 ;POINT TO
        STA POINTH  ;PLAYERS
        LDA #$8Ø
        STA POINTL
        LDA #CTAB1/256 ;GET COLOR...
        STA TABPTH  ;TABLE OFFSET
        LDA #Ø
        STA TABPTL
        TAX         ;ZERO X
MAIN    LDA P1PF,X  ;HIT ANYTHING?
        AND #4      ;GET COLL. BITS
        BEQ NHDR    ;NO COLLISIONS
        STA HITCLR  ;RESET COLLISIONS
        LDA DIRN,X  ;NEXT DIRECTION
        CMP DIRC,X  ;= CURRENT DIR?
        BNE EOR1    ;NO, CHANGE IT.
        EOR #1      ;REVERSE IT,
        STA DIRN,X  ;STORE IT.
EOR1    LDA DIRC,X  ;GET CURRENT
        EOR #1      ;REVERSE IT,
        STA DIRC,X  ;AND STORE.
NHDR    STX TEMPX   ;SAVE X
        LDA XPOSP1,X ;GET PLAYER X'S
        CLC         ;HORIZ. POS,
        ADC #1      ;ADD 1,
        LDY YPOSP1,X ;GET VERT POS,
        INY         ;ADD 1
        JSR PMOVE   ;MOVE PLAYER
        LDX TEMPX   ;GET X AGAIN
        LDA DIRN,X  ;GET NEXT DIRECTION
        AND #2      ;GET 1ST 2 BITS
        BEQ TXM     ;TRY HORIZ. MOVE
        LDA YM      ;OK TO MOVE VERT?
        BEQ CCHD    ;YES, CHANGE DIR
        JMP KCD     ;NO, KEEP CURRENT
TXM     LDA XM      ;HORIZ MOVE OK?
        BNE KCD     ;NO, KEEP CURRENT
CCHD    LDA DIRN,X  ;CHANGE DIRECTION
        STA DIRC,X  ;STORE NEXT DIR
        LDA RANDOM  ;GET RANDOM #
        AND #3      ;MASK TO Ø-3
        STA DIRN,X  ;SAVE FOR NEXT
KCD     LDA DIRC,X  ;GET CURRENT DIR
        BNE TA1     ;NOT RIGHT
        JMP MPXR    ;MOVE RIGHT!
TA1     CMP #1      ;IS IT LEFT?
        BNE TA2     ;NO!
        JMP MPXL    ;MOVE LEFT!
TA2     CMP #2      ;IS IT UP?
        BNE TA3     ;NO!
        JMP MPXU    ;MOVE UP!
TA3     JMP MPXD    ;MOVE DOWN!
;
;SET UP SCREEN
;
SETSC   LDA #Ø      ;ERASE MONEY BAG
        STA SCRAM+219
         STA SCRAM+22Ø
        LDA #SCRAM&255 ;POINT...
        STA POINTL  ;TO SCREEN
        LDA #SCRAM/256
        STA POINTH
        LDY #6      ;SCRN BORDER OFFSET
L1      LDA (POINTL),Y ;GET SCRN BYTE
        BEQ NFL     ;EMPTY-PLOT DOT!
        CMP #3Ø     ;IS IT DOT?
        BEQ NFL     ;YES, PLOT AGAIN!
        CMP #31     ;WALL?
        BNE PL1     ;YES, DON'T PLOT
        DEY         ;PREVIOUS POS.
        LDA #Ø      ;ZERO IT!
        STA (POINTL),Y
        INY         ;BACK TO THIS POS.
NFL     LDA #17     ;PLOT A DOT.
        STA (POINTL),Y
PL1     INY         ;NEXT POSITION
        INY
        CPY #36     ;END OF LINE?
        BCC L1      ;NO!
        LDA POINTL  ;INCREMENT...
        CLC         ;POINTER...
        ADC #4Ø     ;TO NEXT LINE
        STA POINTL
        LDA POINTH
        ADC #Ø
        STA POINTH
        LDY #6      ;LAST LINE?
        CMP #SCRAM/256+1
        BCC L1      ;NO!
        LDA POINTL  ;LAST LINE?
        CMP #144
        BNE L1      ;NO!
        LDA #Ø      ;RESET...
        STA NDOTS   ;# OF DOTS
        STA SCOUNT  ;AND SCORE COUNT
        LDA #5      ;RESET # DOORS
        STA NDOOR
        RTS         ;ALL DONE!
;
;INIT PLAYER-MISSILE GRAPHICS
;
SETPM   LDA #PMRAM/256+2 ;POINT...
        STA POINTH  ;TO...
        LDA #Ø      ;PLAYER MEMORY
        STA POINTL
        TAY         ;ZERO Y REG.
L3      STA (POINTL),Y ;ZERO P/M AREA
        INY
        BNE L3
        INC POINTH
        LDX POINTH  ;DONE CLEARING?
        CPX #PMRAM/256+4
        BNE L3      ;NO!
        LDA #CTABØ/256 ;POINT...
        STA POINTH  ;TO...
        LDA #Ø      ;COLOR TABLES
        STA POINTL
        TAY         ;ZERO Y REG.
Q1      STA (POINTL),Y ;ZERO COLOR TABLES
        INY
        BNE Q1
        INC POINTH
        LDX POINTH  ;DONE CLEARING?
        CPX #CTAB3/256+1
        BNE Q1      ;NO!
        LDA #46     ;TURN ON P/M
        STA SDMCTL
        LDA #3
        STA GRACTL
        LDA #PMRAM/256 ;POINT TO...
        STA PMBASE  ;P/M AREA
        RTS         ;ALL DONE!
;
;RUN VBLANK ROUTINES
;
SETVB   LDA #6      ;SET IMMEDIATE!
        LDX #VBLANK/256
        LDY #VBLANK&255
        JSR $E45C
        LDA #64     ;TURN ON
        STA NMIST
        LDA #7      ;SET DEFERRED!
        LDX #DVBL/256
        LDY #DVBL&255
        JMP $E45C   ;ALL DONE!
;
;DRAW ENEMIES ON SCREEN
;
SETEM   LDA #$8Ø    ;POINT TO P/M
        STA POINTL
        LDA #PMRAM/256+2
        STA POINTH
        LDA #Ø      ;POINT TO COLORS
        STA TABPTL
        LDA #CTAB1/256
        STA TABPTH
L4      LDY #23     ;P/M VERT OFFSET
        LDX #8      ;P/M HEIGHT
L5      LDA ENBMAP,X ;DRAW ONE!
        STA (POINTL),Y
        LDA ENCMAP,X
        STA (TABPTL),Y
        INY
        DEX
        BPL L5
        INC TABPTH  ;NEXT COLOR TABLE
        LDA POINTL  ;NEXT PLAYER
        CLC
        ADC #$8Ø
        STA POINTL
        LDA POINTH
        ADC #Ø
        STA POINTH
        CMP #PMRAM/256+4 ;LAST PLAYER?
        BCC L4      ;NO!
        LDA #$43    ;PL 1 HORIZONTAL
        STA XPOSP1
        STA HPOSP1
        LDA #$84    ;PL 2 HORIZONTAL
        STA XPOSP2
        STA HPOSP2
        LDA #$B3    ;PL 3 HORIZONTAL
        STA XPOSP3
        STA HPOSP3
        LDA #23     ;ALL VERTICAL
        STA YPOSP1
        STA YPOSP2
        STA YPOSP3
        LDA #3      ;P1&3 GOING DOWN
        STA DIRC
        STA DIRC+2
        LDA #1      ;P2 GOING LEFT
        STA DIRC+1
        RTS         ;ALL DONE!
;
;SET UP YOUR PLAYER
;
SETPL   LDX #8      ;8 BYTES TALL
        LDY #55     ;GET VERT OFFSET
L6      LDA PLBMAP,X ;DRAW HIM!
        STA PMRAM+512,Y
        LDA PLCMAP,X
        STA CTABØ,Y
        INY
        DEX
        BPL L6
        LDA #124    ;SET HORIZ POS
        STA XPOSPØ
        STA HPOSPØ
        LDA #55     ;AND VERT POS
        STA YPOSPØ
        LDA #11     ;GOING LEFT!
        STA STICKC
        STA STICKN
        RTS         ;BYE!
;
;IMMEDIATE VBLANK
;
VBLANK  LDA #Ø      ;TURN OFF...
        STA AUDC1   ;SOUND 1
        STA AUDC2   ;SOUND 2
        LDX XPOSPØ  ;GET PLAYER X
        INX         ;OFFSET FOR SCRN
        LDY YPOSPØ  ;GET PLAYER Y
        INY         ;OFFSET IT TOO
        JSR PMTGR   ;CONVERT TO SCRN LOC
        JSR XYTOME  ;THEN TO ADDRESS
        DEC DELAY   ;OK TO MOVE?
        BEQ DO      ;YUP!
        JMP ENDVB   ;NO, EXIT.
DO      LDA LEVEL   ;RESET TIMER
        STA DELAY
        LDA STICKN  ;GET NEXT DIRECTION
        AND #3      ;MASK IT,
        CMP #3      ;HORIZ MOVEMENT?
        BEQ TNM     ;YES!
        LDA NOY     ;CHECK IF VERT CLEAR
        BEQ CHD     ;IT'S CLEAR!
        JMP CD      ;KEEP CURRENT DIR
TNM     LDA NOX     ;HORIZ CLEAR?
        BNE CD      ;NO! KEEP CURRENT
CHD     LDA STICKN  ;GET NEXT MOVE
        STA STICKC  ;SAVE IN CURRENT
CD      LDA STICKØ  ;GET STICK
        CMP #15     ;MOVED?
        BEQ NC      ;NO!
        STA STICKN  ;SAVE NEW MOVE
NC      LDA STICKC  ;GET CURRENT
        CMP #15     ;MOVED?
        BNE N15     ;YES!
        JMP ENDVB   ;NO, EXIT.
N15     CMP #1Ø     ;UP?
        BEQ MU      ;YES!
        CMP #6      ;UP?
        BEQ MU      ;YES!
        CMP #14     ;UP?
        BNE VBP1    ;NO!
MU      LDX #$ØA    ;1Ø BYTES TO MOVE
        LDY YPOSPØ  ;GET VERT OFFSET
        CPY #24     ;AT TOP?
        BCS VBL1    ;NO, MOVE IT!
        JMP ENDVB   ;BYE!
VBL1    LDA PMRAM+512,Y ;MOVE PLAYER
        STA PMRAM+511,Y
        LDA CTABØ,Y
        STA CTABØ-1,Y
        INY
        DEX
        BNE VBL1
        DEC YPOSPØ  ;DEC VERT POS.
MUE     JMP ENDVB   ;BYE!
VBP1    CMP #9      ;DOWN?
        BEQ MD      ;YES!
        CMP #5      ;DOWN?
        BEQ MD      ;YES!
        CMP #13     ;DOWN?
        BNE VBP2    ;NO!
MD      LDA YPOSPØ  ;AT BOTTOM?
        CMP #87
        BCC OKMD    ;NO, MOVE IT!
        JMP ENDVB   ;;BYE!
OKMD    ADC #9      ;GET LAST PLAYER BYTE
        TAY         ;PUT IN Y
        LDX #$ØA    ;1Ø BYTES TO MOVE
VBL2    LDA PMRAM+511,Y ;MOVE PLAYER
        STA PMRAM+512,Y
        LDA CTABØ-1,Y
        STA CTABØ,Y
```

```
        DEY
        DEX
        BNE VBL2
        INC YPOSPØ  ;INC VERT POS.
MDE     JMP ENDVB   ;BYE!
VBP2    CMP #11     ;LEFT?
        BNE VBP3    ;NO!
        LDX XPOSPØ  ;TOO FAR LEFT?
        CPX #68
        BCS OKML    ;NO!
        JMP ENDVB   ;BYE!
OKML    DEX         ;DEC HORIZ POS
        STX XPOSPØ
        STX HPOSPØ
MLE     JMP ENDVB   ;BYE!
VBP3    LDX XPOSPØ  ;TOO FAR RIGHT?
        CPX #179
        BCC OKMR    ;NO!
        JMP ENDVB   ;BYE!
OKMR    INX         ;INC HORIZ POS
        STX XPOSPØ
        STX HPOSPØ
ENDVB   JMP $E45F   ;LEAVE VBLANK!
;
;P/M TO SCREEN COORDS SUBROUTINE
;
PMTGR   STX XPOSSC  ;SAVE HORIZ
        STY YPOSSC  ;SAVE VERT
        TYA         ;GET VERT IN A
        SEC         ;SUB SCREEN OFFSET
        SBC #16
        STA LDIV    ;SAVE IN DIVIDER
        LDA #8      ;DIV. BY 8
        STA DIVS
        LDA #Ø      ;ZERO RESULT
        STA DRES
        JSR DIVIDE  ;DIVIDE IT!
        STA NOX     ;STORE REMAINDER
        LDA DRES    ;GET RESULT
        STA YPOSSC  ;SAVE SCREEN Y
        AND #1      ;EVEN OR ODD?
        BNE OD1     ;ODD!
        LDA #1      ;PROHIBIT ANY
        STA NOX     ;HORIZONTAL MOVE
OD1     LDA XPOSSC  ;GET HORIZ.
        SEC         ;SUB SCREEN OFFSET
        SBC #48
        STA LDIV    ;PUT IN DIVIDER
        LDA #4      ;DIVIDE BY 4
        STA DIVS
        LDA #Ø      ;ZERO RESULT
        STA DRES
        JSR DIVIDE  ;DIVIDE IT!
        STA NOY     ;SAVE REMAINDER
        LDA DRES    ;GET RESULT
        STA XPOSSC  ;SAVE SCREEN X
        AND #3      ;IS VERT MOVE OK?
        EOR #1
        BEQ OD2     ;YES!
        LDA #1      ;PROHIBIT ALL
        STA NOY     ;VERTICAL MOVES
OD2     LDA DRES    ;GET RESULT
        STA XPOSSC  ;SAVE SCREEN X
        RTS         ;ALL DONE!
;
;X&Y COORDS TO MEMORY ADDRESS
;
XYTOME  LDA #Ø      ;ZERO...
        STA VBPNTL  ;VBLANK POINTER
        STA RESH    ;AND RESULT
        LDA STICKC  ;GET CURRENT DIR.
        AND #1      ;LEFT?
        BNE NI      ;NO!
        LDA STICKN  ;GET NEXT DIR
        CMP #11     ;LEFT?
        BNE INCY    ;NO!
        DEC YPOSSC  ;CANCEL NEXT INC
INCY    INC YPOSSC  ;INC VERT POS
NI      LDA YPOSSC  ;GET VERT
        STA MUPL    ;SAVE MULTIPLIER
        LDA #40     ;TIMES 40
        STA MUCN
        JSR MULT    ;MULTIPLY IT!
        INC XPOSSC  ;INC HORIZ POS
        LDA XPOSSC  ;GET IT
        CLC         ;ADD RESULT
        ADC RESL
        STA VBPNTL  ;SAVE IN POINTER
        LDA RESH
        ADC #SCRAM/256 ;ADD SCRN BASE
        STA VBPNTH
        LDY #Ø      ;ZERO Y
        LDA (VBPNTL),Y ;GET BYTE
        CMP #30     ;IS IT DOOR?
        BNE DCHD    ;NO!
        LDA #38     ;MAKE DOOR SOUND
        STA AUDC1
        LDA #20
        STA AUDF1
        LDA #Ø
        TAY
        STA (VBPNTL),Y ;ERASE DOOR
        INC NDOOR   ;1 MORE DOOR
DCHD    CMP #31     ;IS IT DOOR?
        BNE CVDA    ;NO!
        LDA #38     ;MAKE DOOR SOUND
        STA AUDC1
        LDA #20
        STA AUDF1
        LDA #Ø
        TAY
        STA (VBPNTL),Y ;ERASE DOOR
        DEC VBPNTL  ;(2 BYTES)
        STA (VBPNTL),Y
        INC VBPNTL
        INC NDOOR   ;1 MORE DOOR
CVDA    CMP #17     ;IS IT A DOT?
        BEQ CC      ;YES!
        CMP #29     ;MONEY BAG?
        BNE R       ;NO!
        LDA #166    ;MAKE SOUND
        STA AUDC1
        LDA #220
        STA AUDF1
        LDX #100    ;ADD 100 TO SCORE
        STX SCOUNT
        DEC VBPNTL  ;ERASE MONEY BAG
        TYA
        STA (VBPNTL),Y
        INC VBPNTL
        STA (VBPNTL),Y
        RTS         ;AND EXIT!
CC      TYA         ;ERASE DOT
        STA (VBPNTL),Y
        INC NDOTS   ;1 MORE DOT TAKEN
        LDX #1      ;ADD 1 POINT
        STX SCOUNT
        LDA #95     ;MAKE SOUND
        STA AUDC1
        LDA #185
        STA AUDF1
R       RTS         ;AND EXIT!
;
;GENERAL-PURPOSE DIVIDE ROUTINE
;
DIVIDE  LDY #8      ;GET # OF BITS
        SEC         ;SUBTRACT DIVISOR
        SBC DIVS
DILO    PHP         ;SAVE PROC. STATUS
        ROL DRES    ;ROTATE RESULT
        ASL LDIV    ;SHIFT DIVIDER
        ROL A       ;ROTATE DIVIDER
        PLP         ;GET STATUS BACK
        BCC ADD     ;BRANCH IF CLEAR
        SBC DIVS    ;SUBTRACT DIVISOR
        JMP NEXT    ;CONTINUE
ADD     ADC DIVS    ;ADD DIVISOR
NEXT    DEY         ;LAST BIT?
        BNE DILO    ;NO!
        BCS LAST    ;BRANCH IF LAST
        ADC DIVS    ;ADD DIVISOR
        CLC         ;CLEAR CARRY
LAST    ROL DRES    ;ROTATE RESULT
        RTS         ;AND EXIT.
;
;MULTIPLY SUBROUTINE
;
MULT    LDA #Ø      ;ZERO RESULT
        STA RESL
        LDX #8      ;8 BITS
MLOP    LSR MUPL    ;SHIFT MULTIPLIER
        BCC NOADD   ;HI BIT EMPTY
        CLC         ;ADD MULTIPLICAND
        ADC MUCN
NOADD   ROR A       ;ROTATE RESULT
        ROR RESL    ;AND LOW RESULT
        DEX         ;1 MORE BIT
        BNE MLOP    ;MORE TO COME!
        STA RESH    ;SAVE RESULT HI
        RTS         ;AND EXIT
;
;SCOREKEEPING SUBROUTINE
;
SCORE   LDY #5      ;6TH SCORE DIGIT
        CPX #1      ;ANY INCREASE?
        BCS GET     ;YES!
        RTS         ;NO, RETURN
GET     LDA SCRAM+447,Y ;GET SCORE
        CMP #27     ;>=9?
        BCC NOB     ;NO
        LDA #18     ;WRAP TO ZERO
        STA SCRAM+447,Y
        DEY         ;NEXT CHARACTER
        BPL GET
        RTS         ;ALL DONE!
NOB     CLC         ;INCREMENT THE
        ADC #1      ;SCORE
        STA SCRAM+447,Y
        DEX         ;NEXT ADD
        BNE SCORE   ;GO ADD IT!
        RTS         ;ALL DONE!
;
;PLAYER MOVEMENT
;
PMOVE   STA YM      ;SAVE ACCUM
        TYA         ;MOVE VERT TO A
        SEC         ;SCREEN OFFSET
        SBC #16
        STA LDIV    ;PUT IN DIVIDER
        LDA #8      ;DIV BY 8
        STA DIVS
        LDA #Ø      ;CLEAR RESULT
        STA DRES
        JSR DIVIDE  ;DIVIDE IT!
        STA XM      ;SAVE REMAINDER
        LDA DRES    ;GET RESULT
        AND #1      ;ODD?
        BNE OD1M    ;YES!
        LDA #1      ;DON'T PERMIT
        STA XM      ;HORIZ. MOVE
OD1M    LDA YM      ;GET VERT
        SEC         ;SUB SCREEN OFFSET
        SBC #48
        STA LDIV    ;STORE IN DIVIDER
        LDA #4      ;DIV BY 4
        STA DIVS
        LDA #Ø      ;CLEAR RESULT
        STA DRES
        JSR DIVIDE  ;DIVIDE IT!
        STA YM      ;SAVE REMAINDER
        LDA DRES    ;GET RESULT
        AND #3      ;VERT MOVE OK?
        EOR #1
        BEQ OD2M    ;YES!
        LDA #1      ;DON'T ALLOW
        STA YM      ;VERT MOVEMENT
OD2M    RTS         ;EXIT!
MPXR    LDY XPOSP1,X ;CAN PLAYER
        CPY #179    ;MOVE RIGHT?
        BCC CMR     ;YES!
        JMP END     ;NO!
CMR     INY         ;INCREMENT HORIZ
        TYA         ;PUT IN ACCUM
        STA XPOSP1,X ;SET THE POS
        STA HPOSP1,X
        JMP END     ;ALL DONE!
MPXL    LDY XPOSP1,X ;CAN PLAYER
        CPY #68     ;MOVE LEFT?
        BCS CML     ;YES!
        JMP END     ;NO!
CML     DEY         ;DEC HORIZ
        TYA         ;PUT IN ACCUM
        STA XPOSP1,X ;CHANGE POS
        STA HPOSP1,X
        JMP END     ;ALL DONE
MPXU    LDY YPOSP1,X ;GET VERT
        LDX #$ØA    ;10 BYTES TO MOVE
        CPY #24     ;AT TOP?
        BCS CMU     ;NO, CAN MOVE
        JMP END     ;CAN'T GO UP!
CMU     LDA (POINTL),Y ;MOVE UP!
        DEY
        STA (POINTL),Y
        INY
        LDA (TABPTL),Y ;MOVE COLORS
        DEY
        STA (TABPTL),Y
        INY
        INY
        DEX
        BNE CMU
        LDX TEMPX   ;RESTORE X
        DEC YPOSP1,X ;DEC VERT POS
        JMP END     ;ALL DONE
MPXD    LDX TEMPX   ;GET PLAYER #
        LDA YPOSP1,X ;CAN HE...
        CMP #87     ;GO DOWN?
        BCC CMD     ;YES!
        JMP END     ;NO!
CMD     ADC #9      ;GET OFFSET...
        TAY         ;TO PLAYER END
        LDX #$ØA    ;10 BYTES
L       DEY
        LDA (POINTL),Y ;MOVE HIM!
        INY
        STA (POINTL),Y
        DEY
        LDA (TABPTL),Y ;MOVE COLOR
        INY
        STA (TABPTL),Y
        DEY
        DEX
        BNE L
        LDX TEMPX   ;RESTORE X
        INC YPOSP1,X ;INC VERT POS
        JMP END     ;ALL DONE
;
;VBLANK FINISH
;
END     LDA POINTL  ;ADD 128 TO
        CLC         ;P/M POINTER
        ADC #$8Ø    ;TO INDICATE
        STA POINTL  ;NEXT PLAYER
        LDA POINTH
        ADC #Ø
        STA POINTH
        INC TABPTH
        LDX TEMPX   ;GET PLAYER #
        INX         ;INCREMENT IT
```

```
         CPX #3           ;ALL DONE?
         BCS ENDDVB       ;YES!
         JMP MAIN         ;NO, LOOP BACK!
ENDDVB   LDX SCOUNT       ;GET SCORE ADD
         LDA #0           ;ZERO IT
         STA SCOUNT
         JSR SCORE        ;ADD TO SCORE
         LDA STRIG0       ;BUTTON PRESSED?
         BNE PQW          ;NO!
         JSR DOOR         ;PLACE A DOOR
PQW      JMP $E462        ;VBLANK DONE!
;
;INITIALIZATION SEQUENCE
;
INT      LDA #3           ;SET UP...
         STA LEVEL        ;INITIAL SPEEDS
         STA ELEVEL
         STA DELAY
         STA EDELAY
         LDY #19          ;CLEAR SCORE LINE
         LDA #0
CG       STA SCRAM+440,Y
         DEY
         BNE CG
         LDA #$57         ;RESET # MEN
         STA SCRAM+442
         LDY #0           ;ZERO SCORE
         LDA #18
L2       STA SCRAM+447,Y
         INY
         CPY #6
         BNE L2
         RTS              ;ALL DONE!
;
;CHANGE LEVELS
;
NEXTL    LDA ELEVEL       ;GET ENEMY SPEED
         CMP LEVEL        ;SAME AS PLAYER?
         BEQ NDS          ;YES!
         DEC LEVEL        ;SPEED UP PLAYER
         BNE NOC          ;BUT DON'T ALLOW
         LDA #1           ;LEVEL < 1
         STA LEVEL
NDS      DEC ELEVEL       ;SPEED UP ENEMY
         BNE NOC          ;BUT DON'T ALLOW
         LDA #1           ;ELEVEL < 1
         STA ELEVEL
NOC      JSR CLRVB        ;CLEAR VBLANK
         JSR SOUND2       ;MAKE A SOUND
         JSR SETSC        ;SET UP SCREEN
         JSR SETPM        ;SET UP P/M
         JSR SETEM        ;SET ENEMIES
         JSR SETPL        ;SET UP PLAYER
         JSR SETVB        ;SET VBLANK
         JSR PRIZE        ;SET UP MONEYBAG
         JMP COLOR        ;DISPLAY COLORS
;
;CLEAR VBLANK VECTORS
;
CLRVB    LDA #6           ;IMMEDIATE OFF
         LDX #$E4
         LDY #$5F
         JSR $E45C
         LDA #7           ;DEFERRED OFF
         LDX #$E4
         LDY #$62
         JMP $E45C        ;AND EXIT
;
;MONEY BAG HANDLER
;
PRIZE    LDA #MONEY/256 ;SET UP...
         STA $0227        ;TIMER 1 VECTOR...
         LDA #MONEY&255 ;TO POINT TO...
         STA $0226        ;BAG PRINTER
SETT1    LDA #1           ;SET UP TIMER!
         LDY #0
         LDX #4
         JMP $E45C        ALL DONE!
;
;MONEY BAG DISPLAY ROUTINE
;
MONEY    LDA #28          ;PUT BAG CHAR 1
         STA SCRAM+219 ;ON SCREEN
         LDA #29          ;PUT BAG CHAR 2
         STA SCRAM+220 ;ON SCREEN
         LDA #MGONE/256 ;REPOINT TIMER
         STA $0227        ;TO BAG ERASE
         LDA #MGONE&255 ;ROUTINE
         STA $0226
         JMP SETT1        ;GO SET IT
MGONE    LDA #0           ;ERASE BAG
         STA SCRAM+219
         STA SCRAM+220
         RTS              ;AND EXIT
DEATH    JSR CLRVB        ;TURN OFF VBLANK
         JSR SOUND1       ;DO DEATH SOUND
         LDX SCRAM+442 ;1 LESS LIFE
         DEX
         CPX #$52         ;MORE LIVES?
         BCS STILL        ;YES!
         JMP GOVER        ;GAME OVER!
STILL    STX SCRAM+442 ;ZERO LIVES
         JSR SETPM        ;SET UP P/M
         JSR SETPL        ;AND PLAYER
         JSR SETEM        ;AND ENEMIES
         JSR SETVB        ;AND VBLANK
         STA HITCLR       ;CLEAR COLLISIONS
         JMP COLOR        ;AND RESTART!
;
;GAME OVER ROUTINE
;
GOVER    LDX #0           ;SHOW MESSAGE:
PR1      LDA GAME,X     ;'GAME'
         STA SCRAM+442,X
         INX
         CPX #4
         BNE PR1
         LDX #0
PR2      LDF OVER,X     ;'OVER'
         STA SCRAM+434,X
         INX
         CPX #4
         BNE PR2
         JMP PROG         ;AND RESTART!
;
;OPEN DOOR
;
DOOR     DEC NDOOR        ;1 LESS DOOR
         BNE RED          ;DOORS LEFT
         INC NDOOR        ;NO DOORS!
         RTS              ;EXIT
RED      LDA NOY          ;VERTIVAL DOOR?
         BEQ VD           ;YES!
         LDA XPOSSC       ;OK FOR HOR. DOOR?
         AND #3
         BNE ENDD         ;NO!
         LDA #$C0         ;MAKE DOOR SOUND
         STA AUDF2
         LDA #$A3
         STA AUDC2
         LDY #0           ;DRAW DOOR
         LDA #30
         STA (VBPNTL),Y
         RTS              ;AND EXIT
VD       LDA YPOSSC       ;VERT DOOR OK?
         AND #1
         BNE ENDD         ;NO!
         LDA #$C0         ;MAKE DOOR SOUND
         STA AUDF2
         LDA #$A3
         STA AUDC2
         DEC VBPNTL       ;BACK 1 CHAR
         LDY #1           ;DRAW DOOR
         LDA #31
         STA (VBPNTL),Y
         DEY
         STA (VBPNTL),Y
         RTS              ;AND EXIT
ENDD     INC NDOOR        ;NO DOOR DISPLAYED,
         RTS              ;RESET & RETURN
;
;SOUND SUBROUTINES
;
SOUND1   LDA #168         ;INIT SOUND
         STA AUDC2
         LDA #100         ;INIT FREQ
         STA TF3
         LDX #$FF         ;LOW DELAY COUNT
         LDY #20          ;HI DELAY COUNT
SL1      LDA TF3          ;GET FREQ
         STA AUDF2        ;STORE IT
         DEX              ;DELAY 1
          BNE SL1          ;LOOP IF NOT READY
         DEY              ;DELAY 2
         BNE SL1          ;LOOP AGAIN
         INC TF3          ;INC FREQUENCY
         INC TF3
         LDY #20          ;RESET HI DELAY
         LDA TF3          ;GET FREQ
         CMP #$A0         ;LAST FREQ?
         BNE SL1          ;NO!
         LDA #136         ;END OF SOUND
         STA AUDC2
         LDA #50
         STA AUDF2
         LDX #255         ;LO DELAY COUNT
         LDY #255         ;HI DELAY COUNT
SL2      DEX              ;DELAY LOOP
         BNE SL2
         DEY
         BNE SL2
         LDA #0           ;SOUND OFF
         STA AUDF2
         STA AUDC2
         RTS              ;BYE!
SOUND2   LDA #168         ;INITIAL SOUND
         STA AUDC1
         LDA #$40         ;INITIAL FREQ
         STA TF4
         LDX #$FF         ;LO DELAY COUNT
         LDY #20
SL3      LDA TF4          ;GET FREQ
         STA AUDF1        ;STORE IT
         DEX              ;DELAY 1
         BNE SL3
         DEY              ;DELAY 2
         BNE SL3
         DEC TF4          ;NEXT FREQ
         DEC TF4
         LDY #20          ;RESET DELAY
         LDA TF4          ;LAST FREQ?
         BNE SL3          ;NO!
         LDA #0           ;TURN OFF SOUND
         STA AUDF1
         STA AUDC1
         RTS              ;AND EXIT
;
;CHARACTER SET DATA
;
CHDAT
         .BYTE 0,0,0,0,0,0,0,0,85,85
         .BYTE 85,170,85,85,85,85,149,149,149,170
         .BYTE 85,85,85,85,85,85,85,170,149,149
         .BYTE 149,149,149,149,149,149,149,149,149,149
         .BYTE 21,21,21,21,21,21,21,21,149,149
         .BYTE 149,170,149,149,149,149,86,86,86,170
         .BYTE 86,86,86,86,85,85,85,170,86,86
         .BYTE 86,86,86,86,86,170,85,85,85,85
         .BYTE 84,84,84,84,84,84,84,84,86,86
         .BYTE 86,86,86,86,86,86,85,85,85,85
         .BYTE 85,85,85,85,149,149,149,149,85,85
         .BYTE 85,85,86,86,86,86,85,85,85,85
         .BYTE 85,85,85,149,149,149,149,149,85,85
         .BYTE 85,86,86,86,86,86,0,0,0,128
         .BYTE 0,0,0,0,0,60,102,110,118,102
         .BYTE 60,0,0,24,56,24,24,24,126,0
         .BYTE 0,60,102,12,24,48,126,0,0,126
         .BYTE 12,24,12,102,60,0,0,12,28,60
         .BYTE 108,126,12,0,0,126,96,124,6,102
         .BYTE 60,0,0,60,96,124,102,102,60,0
         .BYTE 0,126,6,12,24,48,48,0,0,60
         .BYTE 102,60,102,102,60,0,0,60,102,62
         .BYTE 6,12,56,0,34,10,42,160,162,160
         .BYTE 42,10,136,160,40,10,42,10,40,160
         .BYTE 192,192,192,192,192,192,192,192,0,0
         .BYTE 0,255,0,0,0,0,0,60,102,110
         .BYTE 110,96,62,0,0,24,60,102,102,126
         .BYTE 102,0,0,124,102,124,102,102,124,0
         .BYTE 0,60,102,96,96,102,60,0,0,120
         .BYTE 108,102,102,108,120,0,0,126,96,124
         .BYTE 96,96,126,0,0,126,96,124,96,96
         .BYTE 96,0,0,62,96,96,110,102,62,0
         .BYTE 0,102,102,126,102,102,102,0,0,126
         .BYTE 24,24,24,24,126,0,0,6,6,6
         .BYTE 6,102,60,0,0,102,108,120,120,108
         .BYTE 102,0,0,96,96,96,96,96,126,0
         .BYTE 0,99,119,127,107,99,99,0,0,102
         .BYTE 118,126,126,110,102,0,0,60,102,102
         .BYTE 102,102,60,0,0,124,102,102,124,96
         .BYTE 96,0,0,60,102,102,102,108,54,0
         .BYTE 0,124,102,102,124,108,102,0,0,60
         .BYTE 96,60,6,6,60,0,0,126,24,24
         .BYTE 24,24,24,0,0,102,102,102,102,102
         .BYTE 126,0,0,102,102,102,102,60,24,0
         .BYTE 0,99,99,107,127,119,99,0,0,102
         .BYTE 102,60,60,102,102,0,0,102,102,60
         .BYTE 24,24,24,0,0,126,12,24,48,96
         .BYTE 126,0,0,30,24,24,24,24,30,0
         .BYTE 0,64,96,48,24,12,6,0,0,120
         .BYTE 24,24,24,24,120,0,0,8,28,54
         .BYTE 99,0,0,0,0,0,0,0,0,0
         .BYTE 255,0,0
;
;SCREEN DATA
;
SCDAT
```

```
 .BYTE 0,0,0,5,3,1,1,1,3,1
 .BYTE 1,1,3,1,1,1,3,1,1,1
 .BYTE 3,1,1,1,3,1,1,1,3,1
 .BYTE 1,1,3,1,1,8,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,1,6,0
 .BYTE 17,1,6,0,17,1,6,0,17,1
 .BYTE 6,0,17,1,6,0,17,1,6,0
 .BYTE 17,1,6,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,1,6,0
 .BYTE 17,1,6,0,17,1,6,0,17,1
 .BYTE 6,0,17,1,6,0,17,1,6,0
 .BYTE 17,1,6,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,0,0,0,0
 .BYTE 0,0,0,0,0,0,0,0,0,0
 .BYTE 0,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,1,6,0
 .BYTE 17,1,6,0,17,1,6,0,17,1
 .BYTE 6,0,17,1,6,0,17,1,6,0
 .BYTE 17,1,6,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,1,6,0
 .BYTE 17,1,6,0,17,1,6,0,17,1
 .BYTE 6,0,17,1,6,0,17,1,6,0
 .BYTE 17,1,6,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,4,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,0,17,0,17,0
 .BYTE 17,0,17,0,17,11,10,0,0,0
 .BYTE 0,0,0,5,2,1,1,1,2,1
 .BYTE 1,1,2,1,1,1,2,1,1,1
 .BYTE 2,1,1,1,2,1,1,1,2,1
 .BYTE 1,1,2,1,1,9,10,0,0,0
 .BYTE 0,0,87,0,0,0,0,18,18,18
 .BYTE 18,18,26,0,0,0,0,0,0,0
 .BYTE 0,0,0,0,0,0,0,0,0,0
 .BYTE 0,0,0,0,0,0,0,0,0,0
 .BYTE 0,0,0,0,0,0,0,0,0,0
 .BYTE 0,0,0,0,0,0,0,0,0,0
 .BYTE 0,0,0,0,0,0,0,0,0,0
 .BYTE 0,0,0
;
;RUN ADDRESS
;
 *= $02E0
 .WORD INIT
 .END
```

(Reader Comment
continued from page 6)

yours, I never would have bought an Atari. Furthermore, present Atari owners might be discouraged to the point of selling their machines. This serves only to weaken not only your reader base, but the Atari community in general.

In closing, I am disappointed in what I considered to be the best Atari-specific magazine now available. I hope you will re-think your position and institute an editorial policy that better serves the interests of your readership.

Yours truly,
C.A. Castravelli
Montreal, Quebec, Canada

We are an Atari-exclusive magazine because we consider the equipment the finest available in its price range. However, rather than promotion, we feel that our job is to inform our readers on both positive and negative aspects of Atari and its associated products.

—*Ed.*

Keypad nightmares.

Hear! Hear! **ANALOG** readers, the nightmare is over. We have just found a solution to allow you to use the SYSTEM RESET button while using the Atari numerical keypad.

The format:

POKE 580,0:A=USR(1536)

Add this line to your BASIC program or use it in immediate mode.

Also, as a note of reference, the keypad handler is stored in page six of memory.

JTM Software

I am writing to you for some help or advice. Your magazine, to which I subscribe, seems most likely to be able to answer my *cri de coeur.*

The problem is as follows: I own an Atari 800 with interface and an Epson MX80FT III and would very much like to access the bit-image capability. This, according to the Epson manual, involves passing hex numbers to the printer—something I have been unable to do. Do you know where I can get a subroutine that will solve the difficulty and allow me to dump the screen to the printer?

Yours sincerely,
Andrew W. Kerr
Lincoln, U.K.

You can expect a review of screen dumping programs in an upcoming issue of **ANALOG**, *but, for now, try the subroutine that follows. It is designed to print a GRAPHICS 8 or 24 screen to an Epson printer and is put into operation with this simple GOSUB statement. Feel free to renumber the subroutine, as there are no line number references within it.*

—*TH*

```
1000 REM GRAPHICS 8 DUMPER
1020 SC=PEEK(88)+PEEK(89)*256:LS=159:REM LS=191 FOR GRAPHICS 24
1040 LH=INT(LS/256):LL=LS-LH*256
1050 OPEN #1,8,0,"P:":PRINT #1;"€1":FOR XC=0 TO 39
1070 PRINT #1;"€K";CHR$(LL);CHR$(LH);
1090 FOR YC=LS TO 0 STEP -1:PRINT #1;CHR$(PEEK(SC+YC*40+XC));
1100 NEXT YC:? #1:NEXT XC:CLOSE #1:RETURN
```

Sound FX

16K Cassette or Disk

by John Carmody

Creating sound effects with the Atari computers is fun! Creating just the right effect can, however, be frustrating. Most of the sound manipulation utilities are powerful but complicated. What most of us need is a program that is powerful but simple, one that is most useful for BASIC programming, or just exploring the amazing sound potential of our omnipotent computers.

Sound FX is that program. The screen displays:

VOICE/PLAY :
ATTACK :
SUSTAIN :
DECAY :
FREQUENCY :
DISTORTION :
VOLUME :

for all four voices. A joystick, plugged into port 1, moves the cursor. Pressing the button on most lines increases the appropriate value. Pressing the button on the VOICE/PLAY line plays the sound. Pressing START at any time plays all four voices with the time values of the current voice. When any value reaches its maximum, it is reset to zero. When playing a single voice, the BASIC SOUND command appropriate to the SUSTAIN portion of the note is displayed at the bottom of the screen. □

Line	Function
0	Skip to initialization
6000	Begin single voice play
6006	Skip if ATTACK=0
6009	Skip if SUSTAIN=0
6019	Skip if DECAY=0
6030	Turn off voice
6050	Return to joystick entry
7000	If OPTION then end
7001	Begin multiple voice play
7002	Skip if ATTACK=0
7009	Skip if SUSTAIN=0
7019	Skip if DECAY=0
7030	Return to joystick entry
8000	Initialize
8015	Read initial sound values
8020	Read screen display
8050	Set up screen and colors
8100	Read X,Y positions for display
8900	Print bottom of screen
9000	Print sound matrix on screen
9010	Print initial sound values
9020	Position cursor for start
10000	Main entry loop
10001	Check for console key pressed
10007	If no entry, loop
10008	Get rid of cursor at current location
10010	Convert stick to X,Y
10060	Decide where to move cursor
10150	Put cursor at new location
10200	Loop back
15000	Button pressed if on VOICE/PLAY then play
15010	Increment/reset value

15020 Jump back to cursor printing routine
19000 Date for hi-lo of values
20000 Date for initial values

```
0 GOTO 8000
10 REM
20 REM          SOUND DEMO
30 REM              by
40 REM        John Carmody
50 REM
6000 SETCOLOR 2,(VOICE=0)*14+(VOICE=1)
*3+(VOICE=2)*10+(VOICE=3)*8,(VOICE=0)*
14:POSITION 2,22:? "During SUSTAIN:";
6001 POSITION 2,23:? "BASIC COMMANDƐ→Ɛ
→ SOUND ";VOICE;:FOR I=3 TO 5:? ",";SN
D(VOICE,I);:NEXT I:? "     ";
6006 POSITION XY(VOICE,0)+12,XY(VOICE,
1)+YA:? "Ɛ∢";:IF SND(VOICE,0)=0 THEN 6
009
6007 FOR I=0 TO DD STEP DD/SND(VOICE,0
)
6008 SOUND VOICE,SND(VOICE,3),SND(VOIC
E,4),SND(VOICE,5)*(I/DD):NEXT I
6009 IF SND(VOICE,1)=0 THEN 6019
6010 FOR I=0 TO 10 STEP 10/SND(VOICE,1
)
6015 SOUND VOICE,SND(VOICE,3),SND(VOIC
E,4),SND(VOICE,5):NEXT I
6019 IF SND(VOICE,2)=0 THEN 6030
6020 FOR I=0 TO 10 STEP 10/SND(VOICE,2
)
6025 SOUND VOICE,SND(VOICE,3),SND(VOIC
E,4),SND(VOICE,5)*((10-I)/10):NEXT I
6030 IF STRIG(0)<>0 THEN SOUND VOICE,0
,0,0
6050 GOTO 10000
7000 IF PEEK(53279)=3 THEN ? "← ";:POS
ITION 2,22:POKE 752,0:? "↑";:POKE 16
,192:POKE 53774,192:END
7001 POSITION 2,23:? "
                              ";
7002 SETCOLOR 2,7,14:IF SND(VOICE,0)=0
 THEN 7009
7003 FOR I=0 TO DD STEP 3*DD/SND(VOICE
,0):FOR VO=0 TO 3
7005 SOUND VO,SND(VO,3),SND(VO,4),SND(
VO,5)*(I/DD):NEXT VO:NEXT I
7009 IF SND(VOICE,1)=0 THEN 7019
7010 FOR I=0 TO DD STEP 3*DD/SND(VOICE
,1):FOR VO=0 TO 3
7015 SOUND VO,SND(VO,3),SND(VO,4),SND(
VO,5):NEXT VO:NEXT I
7019 IF SND(VOICE,2)=0 THEN 7029
7020 FOR I=0 TO DD STEP 3*DD/SND(VOICE
,2):FOR VO=0 TO 3
7025 SOUND VO,SND(VO,3),SND(VO,4),SND(
VO,5)*((DD-I)/DD):NEXT VO:NEXT I
7029 POKE 53279,8:IF PEEK(53279)=7 THE
N FOR I=0 TO 3:SOUND I,0,0,0:NEXT I
7030 GOTO 10000
8000 DIM A$(90),XY(3,1),MESS$(255),LSF
T$(30):LSFT$(1)="←":LSFT$(30)="←":LSFT
$(2)=LSFT$:DD=100
8015 RESTORE 20000:DIM SND(3,5):FOR I=
0 TO 3:FOR J=0 TO 5:READ X:SND(I,J)=X:
NEXT J:NEXT I
8020 RESTORE 8030:FOR I=0 TO 6:READ A$
:MESS$(LEN(MESS$)+1)=A$:MESS$(LEN(MESS
$)+1)=LSFT$(1,LEN(A$)-1):NEXT I
8030 DATA VOICE/PLAY:↓,ATTACK    :↓,SU
STAIN   :↓,DECAY     :↓,FREQUENCY :↓,D
ISTORTION:↓,VOLUME    :↓
8050 GRAPHICS 0:SETCOLOR 1,12,8:SETCOL
OR 2,12,0:SETCOLOR 4,3,2:POKE 752,1:PO
KE 82,1
8055 POKE 16,64:POKE 53774,64
8100 RESTORE 8110:FOR I=0 TO 3:READ X,
Y:XY(I,0)=X:XY(I,1)=Y:NEXT I
8110 DATA 1,0,23,0,1,8,23,8
8900 POSITION 2,15:? "Use JOYSTICK 1 t
o position cursorƐ↑.":? "Press button
to change value."
8901 ? "To hear sound, press button wh
en":? "cursor is on VOICE/PLAY line.":
? "Press START for combined sounds,"
8902 ? "using time values of current v
oice.":? "OPTION ends the program."
9000 FOR I=0 TO 3:POSITION XY(I,0),XY(
I,1):? MESS$:POSITION XY(I,0)+11,XY(I,
1):? I;
9010 FOR J=0 TO 5:POSITION XY(I,0)+11,
XY(I,1)+J+1:? SND(I,J);:NEXT J:NEXT I:
? "Ɛ∢";
9020 VOICE=3:YA=6
10000 TRAP 10000:X=STICK(0):Y=STRIG(0)
:IF Y<>1 THEN 15000
10001 POKE 53279,8:IF PEEK(53279)<>7 T
HEN 7000
10003 SETCOLOR 2,12,0
10007 IF X=15 THEN 10000
10008 POSITION XY(VOICE,0)+11,XY(VOICE
,1)+YA:? "     ←←←←";:IF YA>0 THEN ? SN
D(VOICE,YA-1);
10009 IF YA=0 THEN ? VOICE;
10010 POKE 77,0:YPOINT=0:XPOINT=0:IF X
=14 THEN YPOINT=-1
10020 IF X=13 THEN YPOINT=+1
10030 IF X=11 THEN XPOINT=-1
10040 IF X=7 THEN XPOINT=+1
10050 YA=YA+YPOINT:XA=XPOINT
10060 IF VOICE=0 AND YA>6 THEN YA=0:VO
ICE=2
10070 IF VOICE=1 AND YA>6 THEN YA=0:VO
ICE=3
10080 IF VOICE=3 AND YA<0 THEN YA=6:VO
ICE=1
10090 IF VOICE=2 AND YA<0 THEN YA=6:VO
ICE=0
10100 IF VOICE=0 AND XA>0 THEN VOICE=1
10110 IF VOICE=1 AND XA<0 THEN VOICE=0
10120 IF VOICE=3 AND XA<0 THEN VOICE=2
10130 IF VOICE=2 AND XA>0 THEN VOICE=3
10140 YA=YA*((YA>=0) AND (YA<=6))
10150 POSITION XY(VOICE,0)+11,XY(VOICE
,1)+YA:? "     ←←←←";:IF YA>0 THEN ? SN
D(VOICE,YA-1);"Ɛ∢";
10160 IF YA=0 THEN ? VOICE;"Ɛ∢";
10200 GOTO 10000
15000 IF YA=0 THEN GOTO 6000
15010 SND(VOICE,YA-1)=SND(VOICE,YA-1)+
1:RESTORE (YA-1)+19000
15012 READ XA,I:IF SND(VOICE,YA-1)>XA
THEN SND(VOICE,YA-1)=I
15015 IF SND(VOICE,YA-1)<I THEN SND(VO
ICE,YA-1)=XA
15017 IF YA-1=4 THEN SND(VOICE,YA-1)=2
*INT(SND(VOICE,YA-1)/2+0.5)
15020 GOTO 10140
19000 DATA 50,0
19001 DATA 50,0
19002 DATA 50,0
19003 DATA 255,0
19004 DATA 14,0
19005 DATA 15,0
20000 DATA 0,1,5,100,10,6
20001 DATA 5,25,15,20,8,10
20002 DATA 15,8,15,55,2,8
20003 DATA 0,5,30,16,4,6
```

•

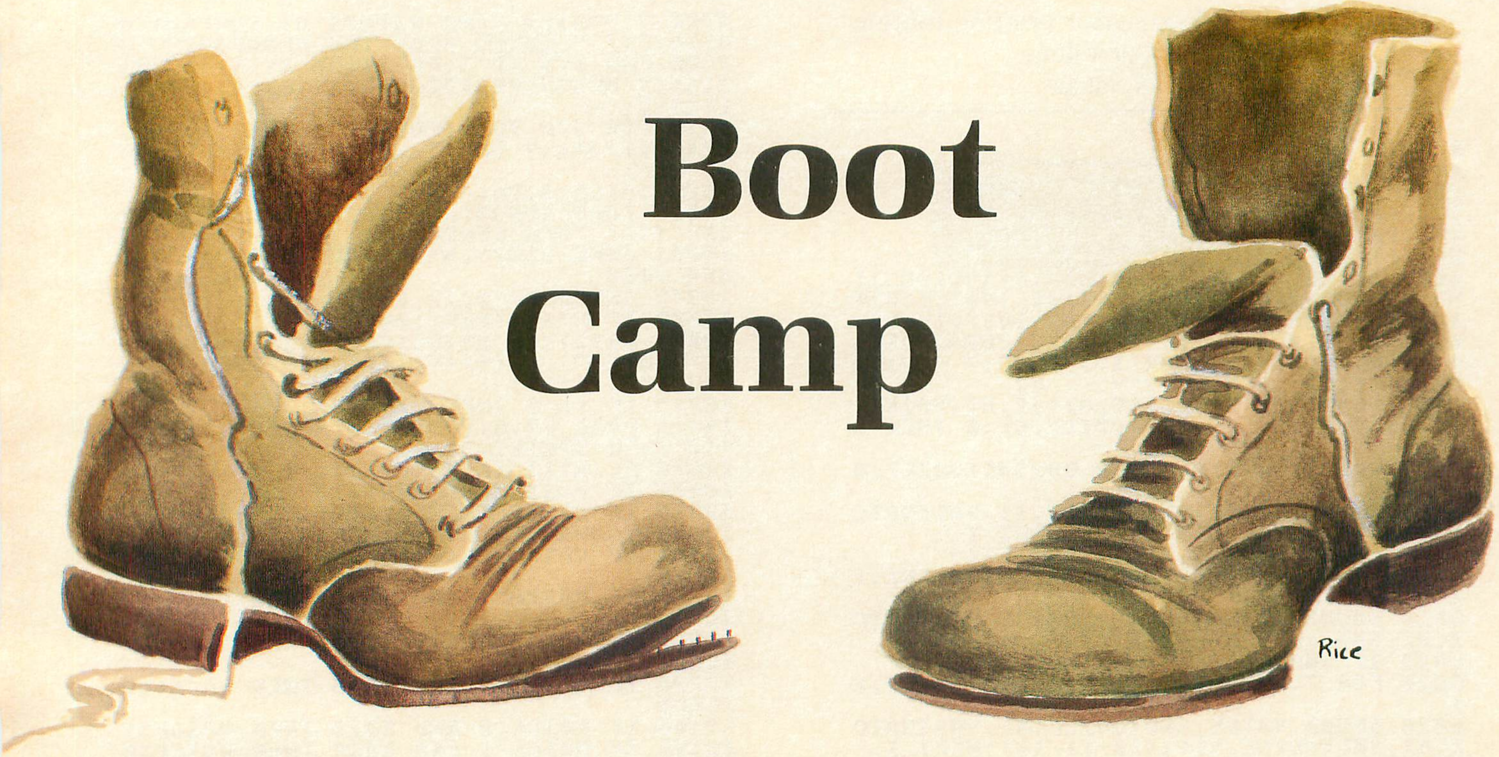
CHECKSUM DATA.

(see page 43)

```
0 DATA 427,672,604,75,331,705,314,567,
900,198,861,854,13,96,834,7451
6020 DATA 17,259,256,914,200,158,49,28
3,216,859,280,873,864,283,190,5701
7029 DATA 45,914,666,232,114,459,287,4
77,406,3,228,652,23,521,226,5253
9020 DATA 289,131,761,782,951,149,593,
451,896,893,733,382,816,822,828,9477
10090 DATA 826,863,862,874,879,995,501
,944,179,758,655,524,633,107,203,9803
19000 DATA 869,872,875,159,885,890,962
,823,1,747,7083
```

•

Boot Camp



by Tom Hudson

Welcome to **Boot Camp**, the beginner's assembly language column. With this issue, we will have completed our introduction to the world of 6502 assembly operation codes. Starting next issue, we'll find out exactly how to apply these instructions in BASIC subroutines, games, utilities and other programs.

Fun with subroutines.

Last issue's homework was for you to write a subroutine that would add the X and Y registers, placing the result in the accumulator. If the result of the add was greater than 255, you were to put the value $FF in the X register. If not, you were to set the X register to zero. Figure 1 shows one possible solution. Let's step through it and see how it works.

```
10 ;SUBROUTINE "ADDXY"
12 ;
14 ;ADDS X REGISTER TO Y REGISTER
16 ;PLACING RESULT IN ACCUMULATOR
18 ;IF RESULT  > 255, X REG = $FF
20 ;IF RESULT <= 255, X REG = $00
22 ;
24          *=  $0600
26 ADDXY    CLD         ;BINARY MATH
28          STX TEMP    ;SAVE X REG.
30          TYA         ;PUT Y IN ACC.
32          CLC         ;CLEAR FOR ADD
34          ADC TEMP    ;ADD X REGISTER
36          BCS GTR255  ;BRANCH IF > 255
38          LDX #$00    ;ZERO X REGISTER
40          RTS         ;AND RETURN!
42 GTR255   LDX #$FF    ;SET X REGISTER
44          RTS         ;AND RETURN!
46 TEMP     *=*+1
48          .END
```

Figure 1.

Lines 10 - 22 are the subroutine documentation lines. They tell what the subroutine does and how to use it. This can help refresh your memory if you need to change a program several years after you write it.

Line 26 is the entry point for the subroutine. I have labeled this one ADDXY, for "Add X and Y registers." It's a good idea to use descriptive labels in your programs. I could have called the subroutine DOG, but this wouldn't help me remember what the subroutine does. This line clears the decimal mode, so that we're sure the subroutine is operating in binary math mode.

Line 28 stores the X register at the location TEMP, a temporary hold area.

Line 30 transfers the Y register to the accumulator with the TYA instruction. This is done because the 6502 add instruction (ADC) only works with the accumulator.

Line 32 clears the carry flag for the add operation.

Line 34 adds the accumulator (which now contains the Y-register value) to the location TEMP (which contains the X-register value). After this instruction executes, we have completed the first part of the homework, adding the X and Y registers with the result in the accumulator.

Line 36 branches to the label GTR255 (Greater than 255) if the carry flag is set (BCS). If the carry is not set, execution continues at Line 38. Remember that the carry flag is set if the result of an add operation is greater than 255. Review the issue 17 **Boot Camp** if you're not sure of the carry flag's function.

Line 38 places a zero in the X register if the add result was not greater than 255. The X register in this case is used as an indicator to tell the code which called the subroutine that the addition result fits in the accumulator. If the carry flag had been set, the result was greater than 255 and would not have fit in the 8-bit accumulator.

Line 40 is an RTS instruction. This will return control to the code which called the subroutine.

Line 42, labeled GTR255, is the code that will be executed if the add result is too large for the accumulator. It loads the X register with the value $FF. Once again, after the subroutine has been executed, the calling routine can test the X register. If the X register contains $FF, the calling routine can take the appropriate action.

Line 44 is another RTS instruction, and will return control to the calling code.

Line 46 defines a one-byte temporary storage location, labeled TEMP.

How would we use this subroutine? Figure 2 shows an example of the code necessary to call the subroutine ADDXY.

```
       LDX ADD1    ;GET ADD #1
       LDY ADD2    ;GET ADD #2
       JSR ADDXY   ;ADD X & Y
       CPX #$00    ;ADD OK?
       BNE BADADD  ;NO!
       STA RESULT  ;ADD OK!
       JMP OK      ;JUMP ELSEWHERE
BADADD JMP NOTOK   ;HANDLE ERROR
```

Figure 2.

As you can see, this code first loads the X and Y registers with the desired add values, then JSRs to the subroutine.

The first instruction after the JSR tests the X register to see if it's zero. If not, the add was too large for the accumulator, and we branch to the label BADADD. If the add was okay, we store the accumulator in the location labeled RESULT and jump to another part of the program, labeled OK.

Of course, the use of the X register as an overflow flag was not really necessary in this problem. We could have simply tested the carry flag after the JSR and taken the appropriate action then. Still, I thought this would be a good time to introduce you to the technique of using subroutine result indicators.

So there you have it. Just one of the many ways in which the homework assignment can be solved. I'm sure most of you came up with other ways to accomplish the objective, and — as long as they work — it doesn't matter which approach you take. Just remember to thoroughly test each subroutine you write, to be sure they'll return the proper results.

Getting pushy.

Up till now, all our stack usage has been handled by the 6502 itself, in the JSR and RTS instructions. Now we're going to find out how to use the stack for our own purposes.

The first two stack instructions we're going to investigate are the PHA (Push accumulator onto stack) and PLA (Pull accumulator from stack). The format of the PHA instruction is:

PHA (NO ADDRESSING)

The PHA instruction is used to place the accumulator on the "top" of the stack. It doesn't affect any status flags. Let's see what happens when a PHA instruction executes.

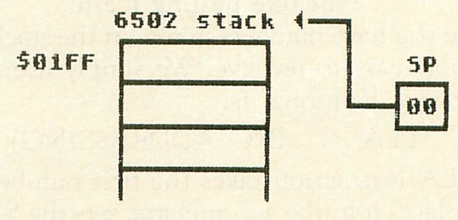


Figure 3.

Figure shows how the stack looks when it's empty. The stack pointer (SP) contains $00. As you recall from the last two **Boot Camp** installments, the 6502 stack resides in the memory from $0100-01FF.

Let's assume the following two instructions are executed:

```
LDA #$40
PHA
```

The first instruction loads the accumulator with the value $40. The second instruction "pushes" this value onto the stack. The 6502 decrements the stack pointer (to $FF), then stores the accumulator's contents at the indicated memory location. Figure 4 shows how the stack looks after the PHA instruction.

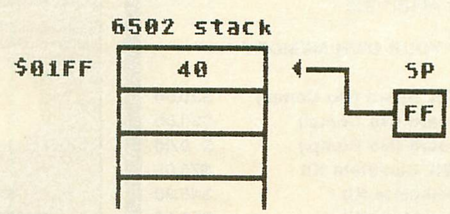


Figure 4.

If we like, we can push another value onto the stack. Let's push the value $6D onto the stack this time. Here's the code:

```
LDA #$6D
PHA
```

This time, the stack pointer will be decremented (to $FE), and the value $6D stored at the indicated location. Figure 5 shows how the stack looks now.

(continued on next page)

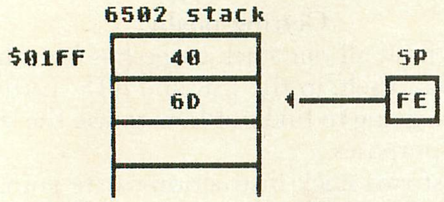


Figure 5.

See how simple the PHA instruction is? No registers except the stack pointer are affected, and the numbers are sitting on the stack, ready for you to use them. How do we get them back? With the PLA instruction, of course!

Not like pulling teeth.

Once you have numbers stored on the stack, they're incredibly easy to retrieve. We simply use the PLA instruction. Its format is:

PLA (NO ADDRESSING)

The PLA instruction takes the first number on the stack, places it in the accumulator, sets the SIGN and ZERO flags accordingly, and increments the stack pointer so that the next value is ready to be pulled from the stack. Let's see how this works with the numbers we placed on the stack earlier.

Figure 5 shows the stack as it appears now. We want to pull a value off the stack, so we write the following code:

```
PLA
```

The 6502 loads the accumulator from the indicated byte of the stack ($6D) and increments the stack pointer. At this point, the accumulator contains $6D, and the stack looks like Figure 6.

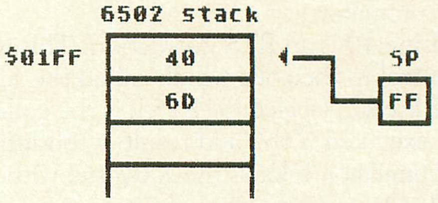


Figure 6.

Simple, right? We've just retrieved the last number placed on the stack. Let's do it again. We use the code:

```
PLA
```

When complete, the accumulator contains $40, and the stack looks like Figure 7.

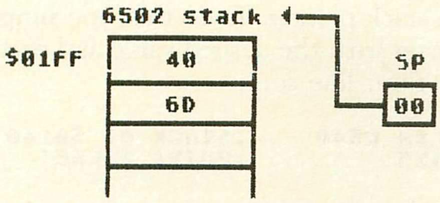


Figure 7.

Now you see how easy stack usage is. All you need to do is push and pull the desired values, and the computer takes care of all necessary overhead. However, there are a few things you need to remember when using the stack.

Stack logic.

The first thing you must remember about the stack is that it is a LIFO (Last-In, First-Out) structure. That is, the *last* number you place onto the stack will be the *first* number that you pull off. This sometimes takes getting used to, but you'll get the hang of it if you diagram your stack logic on paper first.

Second, the stack can only hold up to 256 numbers, and some space on the stack is used by the system. A good rule of thumb is to use the stack only when you need to, like in BASIC USR calls or when you're running out of memory (a PHA only takes one byte; an STA can take up to three bytes).

Using the stack.

What can you use the stack for? Most people use it to store numbers temporarily or as a small table that automatically maintains pointers.

Here's an example of using the stack to save the accumulator's contents when a subroutine is executed. Remember that when a subroutine is executed, if it uses any registers, the values that were in those registers are lost.

Figure 8 shows how to save the accumulator so that you can be sure it is unchanged after a subroutine executes.

```
10      PHA             ;SAVE ACCUMULATOR
20      JSR  SUBRTN     ;PERFORM SUBROUTINE
30      PLA             ;RESTORE ACCUMULATOR
```

Figure 8.

Line 10 pushes the accumulator's contents onto the stack. Now, no matter what the subroutine does with the accumulator, we can always restore the accumulator to its original value.

Line 20 calls the subroutine SUBRTN with the JSR instruction. We assume that the subroutine manipulates the accumulator, changing it to some unknown value.

Line 30 pulls the old accumulator value off the stack, making sure that we have the accumulator restored to the desired value.

Unfortunately, the designers of the 6502 did not allow for the PUSHing of the X and Y registers, so we have to write a little extra code.

To push the X register, we use the code:

```
TXA             ;MOVE X TO ACCUM.
PHA             ;AND PUSH IT!
```

This transfers the X register to the accumulator, then pushes the value onto the stack.

Similarly, the Y value register can be pushed with the sequence:

```
TYA             ;MOVE Y TO ACCUM.
PHA             ;AND PUSH IT!
```

To pull the X or Y registers from the stack, use one of the following code sequences:

```
PLA             ;PULL THE VALUE,
TAX             ;AND PUT IN X!

PLA             ;PULL THE VALUE,
TAY             ;AND PUT IN Y!
```

These routines are simple enough, but you should remember that the accumulator will be lost in all of these operations unless you save it somewhere first.

Saving your status.

Sometimes you'll want to save the processor status register before a subroutine or comparison operation so that you can test certain flags later. This can be done by using the PHP (Push processor status register onto stack) and PLP (Pull processor status register from stack) instructions. Their formats are:

PHP (NO ADDRESSING)
PLP (NO ADDRESSING)

The PHP and PLP instructions work just like the PHA and PLA instructions, except that they push and pull the status flags instead of the accumulator.

The PHP instruction does not affect any flags, but the PLP instruction changes *all* the flags, since it is actually loading the flags from the stack.

We'll explore the use of the PHP in more detail later, when the need arises.

Which way to the stack?

Someday, you may need to know where the stack pointer is currently pointing, or you may need to change the stack pointer to point to a particular location. This is usually a rare occurrence, but I needed to do this in my debug utility, HBUG, in issue 18.

The 6502 has two instructions that will allow us to examine and change the stack pointer. These are TSX (Transfer stack pointer to X) and TXS (Transfer X to stack pointer). The formats of these instructions are:

TSX (NO ADDRESSING)
TXS (NO ADDRESSING)

The TSX instruction simply loads the X register with whatever happens to be in the stack pointer at the time. The sign and zero flags reflect the result of the load.

Figure 9 shows an example of the use of the TSX instruction.

```
10          *=  $0600
12          LDA #$F0      ;PUT # IN ACCUM.
14          TSX           ;GET STACK PTR
16          STX STACK1    ;SAVE STACK #1
18          PHA           ;PUSH ACCUM.
20          TSX           ;GET STACK PTR
22          STX STACK2    ;SAVE STACK #2
24          PLA           ;PULL ACCUM.
26          TSX           ;GET STACK PTR
28          STX STACK3    ;SAVE STACK #3
30          BRK           ;ALL DONE!
32 STACK1   *=*+1
34 STACK2   *=*+1
36 STACK3   *=*+1
38          .END
```

Figure 9.

Let's walk through this code and see what happens.

Line 12 loads the accumulator with $F0.

Line 14 transfers the current contents of the stack pointer to the X register.

Line 16 stores the X register (which now contains the stack pointer value) in the location STACK1. This records the original stack location, so we can observe it later.

Line 18 pushes the accumulator onto the stack. As we now know, the stack pointer will be decremented by 1 after this operation.

Line 20 transfers the stack pointer to the X register again.

Line 22 stores the X register (containing the stack pointer value) in the location STACK2. This will record the stack's position after the PHA instruction.

Line 24 pulls the accumulator from the stack.

Line 26 transfers the stack pointer to the X register a final time.

Line 28 stores the stack pointer contained in the X register at the location STACK3.

Line 30 stops the program's execution.

Type this program into your computer and assemble it. Note the locations of STACK1, STACK2 and STACK3 during the assembly. When the program is assembled, execute it.

After execution, examine the memory locations at STACK1, STACK2 and STACK3. STACK1 contains the stack's location at the beginning of the program. STACK2 contains the stack's location after the PHA instruction. Since the PHA *decrements* the stack pointer, STACK2 should be one less than STACK1. STACK3 contains the stack pointer's contents after the PLA instruction. A PLA instruction *increments* the stack pointer, so STACK3 will be one more than STACK2.

The TXS instruction does the opposite of TSX. That is, you can move the contents of the X register to the stack pointer. To do this, you simply load the X register with the desired value and execute a TXS instruction, like so:

```
LDX #$40      ;STACK AT $0140,
TXS           ;POINT THERE!
```

I strongly suggest that you leave this instruction alone for the time being. Incorrect setting of the stack pointer can cause a system lockup, so hold on until we get a chance to use it safely in a **Boot Camp** program.

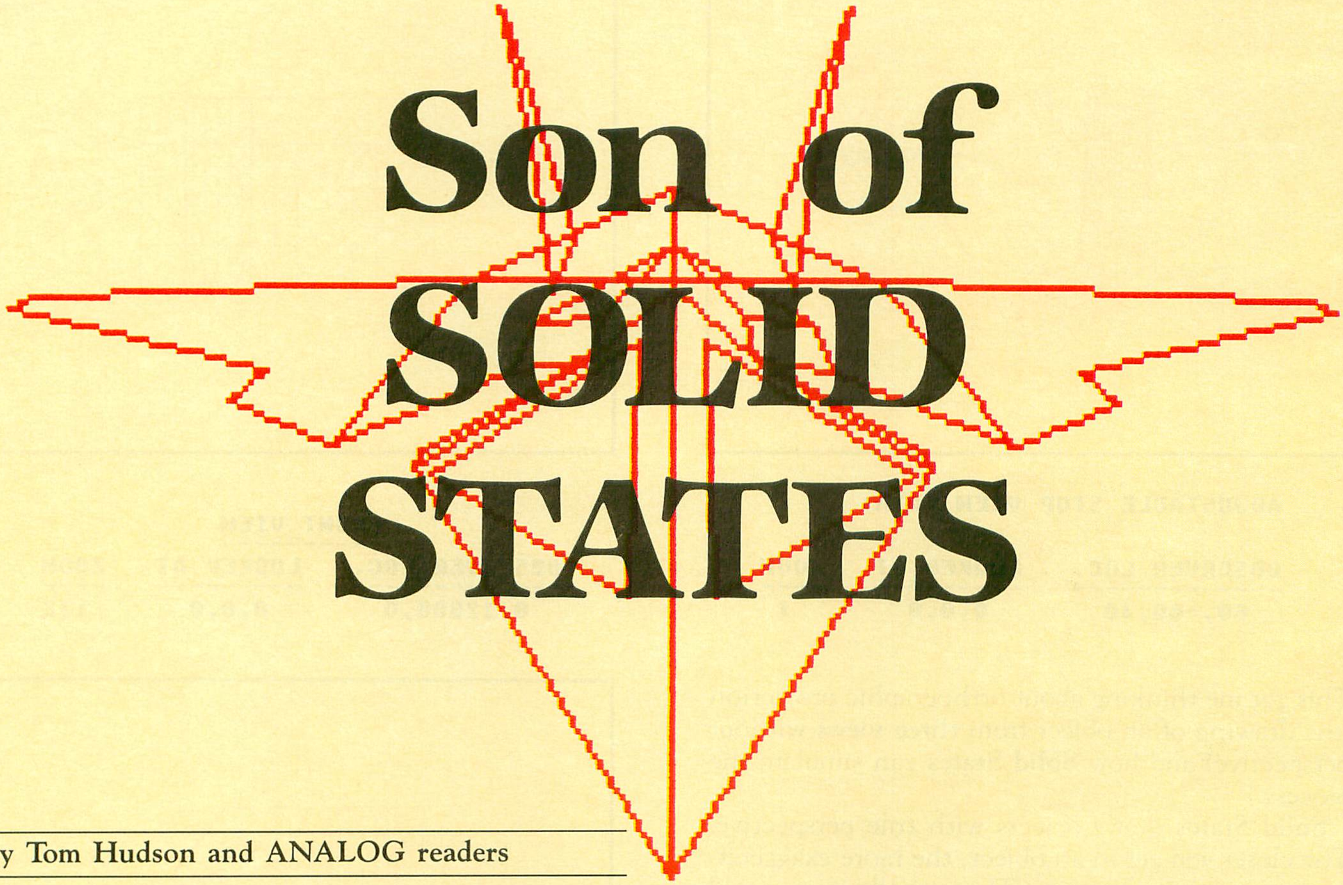
All for now.

Well, we've covered all the major 6502 instructions, and we're ready to learn some system-specific material. Starting next issue, we'll go full speed ahead into the world of the Atari's innards. □

Send all letters to:

Boot Camp
c/o ANALOG Computing
P.O. Box 23
Worcester, MA 01603

Son of SOLID STATES



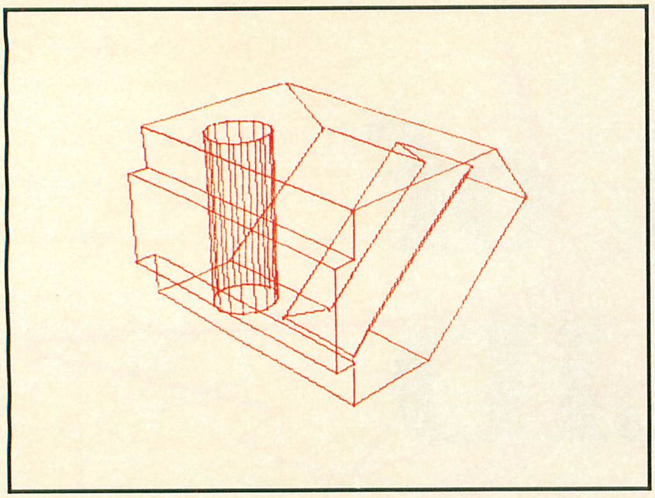
by Tom Hudson and ANALOG readers

Okay, **Solid States** fans, hang onto your hats! This issue, we've got program modifications and 3-D objects galore from yours truly and **ANALOG** readers around the world. Let's get started, shall we?

Far out!

Our first object was sent in by Vinette DePhillipe, of Hampton, Virginia. It's called an "adjustable stop" and reminds me of my days back in good ol' Glendale High School. My general drafting teacher, Ernie Belden (and heaven help you if you ever called him Ernie), was always coming up with weird objects for us to draw, and we never could figure out what they were used for. Here is the data for the adjustable stop:

```
POINTS:58

POINT 1: -13      -12      -9
POINT 2: -13      -12      -5
POINT 3: -13      -14      -5
POINT 4: -13      -14      4
POINT 5: -13      -12      4
POINT 6: -13      -12      8
POINT 7: 13       -12      8
POINT 8: 13       -12      4
POINT 9: 13       -14      4
POINT 10: 13      -14      -5
POINT 11: 13      -12      -5
POINT 12: 13      -12      -9
POINT 13: -13     -3       -9
POINT 14: 1       -3       -9
POINT 15: -1      -7       -9
POINT 16: 8       -7       -9
POINT 17: 6       -3       -9
POINT 18: 13      -3       -9
POINT 19: -1      8.25     4
POINT 20: 8       8.25     4
POINT 21: 6       10       2
POINT 22: 1       10       2
POINT 23: -13     10       2
POINT 24: -13     5        8
POINT 25: 13      5        8
POINT 26: 13      10       2
POINT 27: -6      -4       -9
POINT 28: -4.86   -4.23    -9
POINT 29: -3.88   -4.88    -9
POINT 30: -3.23   -5.86    -9
POINT 31: -3      -7       -9
POINT 32: -3.23   -8.14    -9
POINT 33: -3.88   -9.12    -9
POINT 34: -4.86   -9.77    -9
POINT 35: -6      -10      -9
POINT 36: -7.14   -9.77    -9
POINT 37: -8.12   -9.12    -9
POINT 38: -8.77   -8.14    -9
POINT 39: -9      -7       -9
POINT 40: -8.77   -5.86    -9
POINT 41: -8.12   -4.88    -9
POINT 42: -7.14   -4.23    -9
POINT 43: -6      -4       8
POINT 44: -4.86   -4.23    8
POINT 45: -3.88   -4.88    8
POINT 46: -3.23   -5.86    8
POINT 47: -3      -7       8
POINT 48: -3.23   -8.14    8
POINT 49: -3.88   -9.12    8
POINT 50: -4.86   -9.77    8
POINT 51: -6      -10      8
POINT 52: -7.14   -9.77    8
POINT 53: -8.12   -9.12    8
POINT 54: -8.77   -8.14    8
POINT 55: -9      -7       8
POINT 56: -8.77   -5.86    8
POINT 57: -8.12   -4.88    8
POINT 58: -7.14   -4.23    8

LINES:88

LLINE 1: 1 TO 2
LINE 2: 2 TO 3
LINE 3: 3 TO 4
LINE 4: 4 TO 5
LINE 5: 5 TO 6
LINE 6: 6 TO 7
LINE 7: 7 TO 8
LINE 8: 8 TO 9
LINE 9: 9 TO 10
LINE 10: 10 TO 11
LINE 11: 11 TO 12
LINE 12: 12 TO 1
LINE 13: 2 TO 11
LINE 14: 3 TO 10
LINE 15: 4 TO 9
LINE 16: 5 TO 8
LINE 17: 6 TO 7
LINE 18: 1 TO 13
LINE 19: 13 TO 14
LINE 20: 14 TO 15
LINE 21: 15 TO 16
LINE 22: 16 TO 17
LINE 23: 17 TO 18
LINE 24: 18 TO 12
LINE 25: 15 TO 19
LINE 26: 19 TO 20
LINE 27: 20 TO 21
LINE 28: 21 TO 17
LINE 29: 20 TO 16
LINE 30: 14 TO 22
LINE 31: 19 TO 22
LINE 32: 13 TO 23
LINE 33: 23 TO 24
LINE 34: 24 TO 25
LINE 35: 25 TO 26
LINE 36: 26 TO 18
LINE 37: 26 TO 21
LINE 38: 23 TO 22
LINE 39: 24 TO 6
LINE 40: 25 TO 7
LINE 41: 27 TO 28
LINE 42: 28 TO 29
LINE 43: 29 TO 30
LINE 44: 30 TO 31
LINE 45: 31 TO 32
LINE 46: 32 TO 33
LINE 47: 33 TO 34
LINE 48: 34 TO 35
LINE 49: 35 TO 36
LINE 50: 36 TO 37
LINE 51: 37 TO 38
LINE 52: 38 TO 39
LINE 53: 39 TO 40
LINE 54: 40 TO 41
LINE 55: 41 TO 42
LINE 56: 42 TO 27
LINE 57: 27 TO 43
LINE 58: 43 TO 44
LINE 59: 44 TO 45
LINE 60: 45 TO 46
LINE 61: 46 TO 47
LINE 62: 47 TO 48
LINE 63: 48 TO 49
LINE 64: 49 TO 50
LINE 65: 50 TO 51
LINE 66: 51 TO 52
LINE 67: 52 TO 53
LINE 68: 53 TO 54
LINE 69: 55 TO 54
LINE 70: 55 TO 56
LINE 71: 56 TO 57
LINE 72: 57 TO 58
LINE 73: 58 TO 43
LINE 74: 28 TO 44
LINE 75: 29 TO 45
LINE 76: 30 TO 46
LINE 77: 31 TO 47
LINE 78: 32 TO 48
LINE 79: 33 TO 49
LINE 80: 34 TO 50
LINE 81: 35 TO 51
LINE 82: 36 TO 52
LINE 83: 37 TO 53
LINE 84: 38 TO 54
LINE 85: 39 TO 55
LINE 86: 40 TO 56
LINE 87: 41 TO 57
LINE 88: 42 TO 58
```



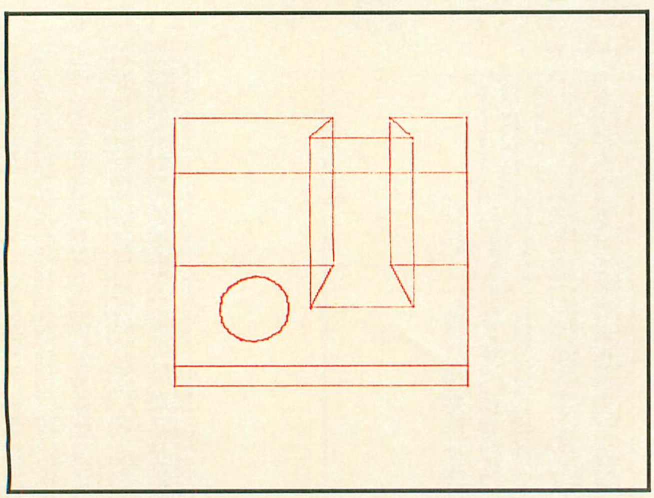
ADJUSTABLE STOP VIEW DATA:

OBSERVER LOC.	LOOKED AT	ZOOM
60,-60,40	0,0,0	1

This got me thinking about orthographic projection (the drawing of an object from three views without perspective) and how **Solid States** can simulate the process.

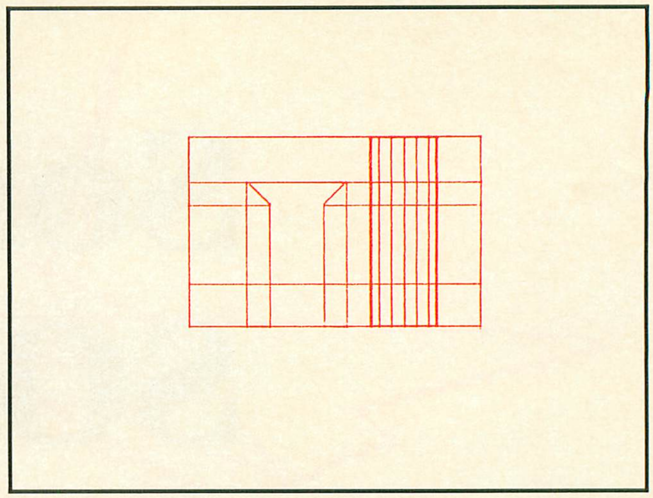
Solid States draws objects with true perspective. The closer you get to an object, the more exaggerated the perspective becomes. Theoretically, if you could look at an object from an infinite distance through a large telescope, you would see the object without the perspective distortion.

We can produce orthographic views with **Solid States** by placing our viewpoint very far from the object and using a large ZOOM factor. The orthographic views of the adjustable stop are shown below, along with the coordinate and ZOOM information.



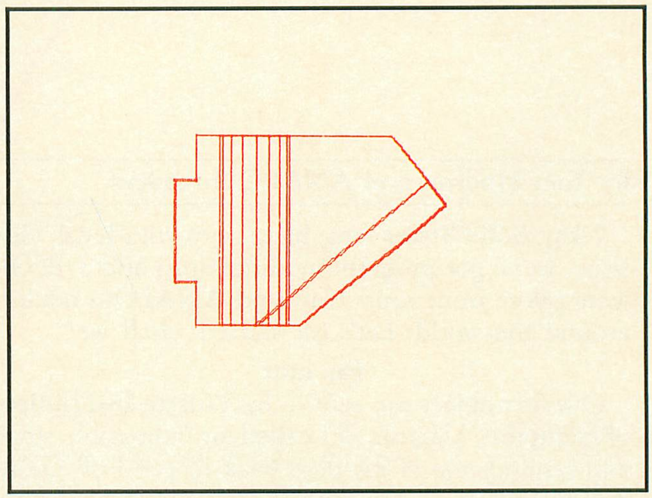
TOP VIEW

OBSERVER LOC.	LOOKED AT	ZOOM
0,0,12000	0,0,0	132



FRONT VIEW

OBSERVER LOC.	LOOKED AT	ZOOM
0,12000,0	0,0,0	132



SIDE VIEW

OBSERVER LOC.	LOOKED AT	ZOOM
12000,0,0	0,0,0	132

Conversely, if we place our viewpoint very *close* to the object and use a very small ZOOM factor, we will see the object severely distorted by perspective. Try the following view with the adjustable stop, and you'll see what I mean.

ADJ STOP EXAGGERATED PERSPECTIVE

OBSERVER LOC.	LOOKED AT	ZOOM
15,0,0	0,0,0	.04

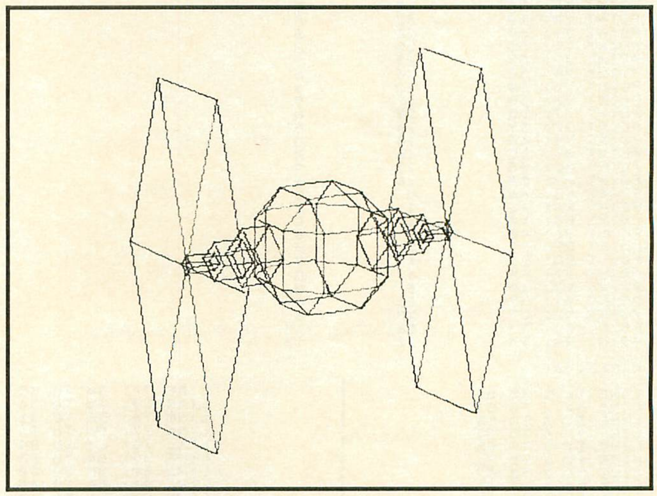
Those readers interested in mechanical drawing applications of **Solid States** should find this information very useful. Mr. Belden would be proud.

TIE one on.

Robert Ashcraft of Amarillo, Texas, sent in the data for a TIE fighter, the famous Imperial craft from the *Star Wars* movies.

Here's the data for the TIE fighter:

```
POINTS:84

POINT 1: 5       -2      6
POINT 2: 5       2       6
POINT 3: 5       4       0
POINT 4: 5       2       -6
POINT 5: 5       -2      -6
POINT 6: 5       -4      0
POINT 7: -5      -2      6
POINT 8: -5      2       6
POINT 9: -5      4       0
POINT 10: -5     2       -6
POINT 11: -5     -2      -6
POINT 12: -5     -4      0
POINT 13: -1     -2      1
POINT 14: 1      -2      1
POINT 15: 1      -2      -1
POINT 16: -1     -2      -1
POINT 17: -1     1       2
POINT 18: 1      1       2
POINT 19: 1      -1      2
POINT 20: -1     -1      2
POINT 21: -1     2       -1
POINT 22: 1      2       -1
POINT 23: 1      2       1
POINT 24: -1     2       1
POINT 25: -1     -1      -2
POINT 26: 1      -1      -2
POINT 27: 1      1       -2
POINT 28: -1     1       -2
POINT 29: -2     1       1
POINT 30: -2     -1      1
POINT 31: -2     -1      -1
POINT 32: -2     1       -1
POINT 33: 2      1       -1
POINT 34: 2      -1      -1
POINT 35: 2      -1      1
POINT 36: 2      1       1
POINT 37: 2      0       1
POINT 38: 2      1       0
POINT 39: 2      0       -1
POINT 40: 2      -1      0
POINT 41: 3      0       1
POINT 42: 3      1       0
POINT 43: 3      0       -1
POINT 44: 3      -1      0
POINT 45: -2     0       1
POINT 46: -2     1       0
POINT 47: -2     0       -1
POINT 48: -2     -1      0
POINT 49: -3     0       1
POINT 50: -3     1       0
POINT 51: -3     0       -1
POINT 52: -3     -1      0
POINT 53: 3      0       0.67
POINT 54: 3      0.67    0
POINT 55: 3      0       -0.67
POINT 56: 3      -0.67   0
POINT 57: 4      0       0.67
POINT 58: 4      0.67    0
POINT 59: 4      0       -0.67
POINT 60: 4      -0.67   0
POINT 61: -3     0       0.67
POINT 62: -3     0.67    0
POINT 63: -3     0       -0.67
POINT 64: -3     -0.67   0
POINT 65: -4     0       0.67
POINT 66: -4     0.67    0
POINT 67: -4     0       -0.67
POINT 68: -4     -0.67   0
POINT 69: 4      0       0.33
POINT 70: 4      0.33    0
POINT 71: 4      0       -0.33
POINT 72: 4      -0.33   0
POINT 73: 5      0       0.33
POINT 74: 5      0.33    0
POINT 75: 5      0       -0.33
POINT 76: 5      -0.33   0
POINT 77: -4     0       0.33
POINT 78: -4     0.33    0
POINT 79: -4     0       -0.33
POINT 80: -4     -0.33   0
POINT 81: -5     0       0.33
POINT 82: -5     0.33    0
POINT 83: -5     0       -0.33
POINT 84: -5     -0.33   0

LINES:138

LINE 1: 1 TO 2
LINE 2: 2 TO 3
LINE 3: 3 TO 4
LINE 4: 4 TO 5
LINE 5: 5 TO 6
LINE 6: 6 TO 1
LINE 7: 1 TO 4
LINE 8: 2 TO 5
LINE 9: 3 TO 6
LINE 10: 7 TO 8
LINE 11: 8 TO 9
LINE 12: 9 TO 10
LINE 13: 10 TO 11
LINE 14: 11 TO 12
LINE 15: 12 TO 7
LINE 16: 7 TO 10
LINE 17: 8 TO 11
LINE 18: 9 TO 12
LINE 19: 13 TO 20
LINE 20: 20 TO 17
LINE 21: 17 TO 24
LINE 22: 24 TO 21
LINE 23: 21 TO 28
LINE 24: 28 TO 25
LINE 25: 25 TO 16
LINE 26: 16 TO 13
LINE 27: 14 TO 19
LINE 28: 19 TO 18
LINE 29: 18 TO 23
LINE 30: 23 TO 22
LINE 31: 22 TO 27
LINE 32: 27 TO 26
LINE 33: 26 TO 15
LINE 34: 15 TO 14
LINE 35: 13 TO 14
LINE 36: 14 TO 35
LINE 37: 35 TO 36
LINE 38: 36 TO 23
LINE 39: 23 TO 24
LINE 40: 24 TO 29
LINE 41: 29 TO 30
LINE 42: 30 TO 13
LINE 43: 16 TO 15
LINE 44: 15 TO 34
LINE 45: 34 TO 33
LINE 46: 33 TO 22
LINE 47: 22 TO 21
LINE 48: 21 TO 32
LINE 49: 32 TO 31
LINE 50: 31 TO 16
LINE 51: 17 TO 18
LINE 52: 18 TO 36
LINE 53: 36 TO 33
LINE 54: 33 TO 27
LINE 55: 27 TO 28
LINE 56: 28 TO 32
LINE 57: 32 TO 29
LINE 58: 29 TO 17
LINE 59: 20 TO 19
LINE 60: 19 TO 35
LINE 61: 35 TO 34
LINE 62: 34 TO 26
LINE 63: 26 TO 25
LINE 64: 25 TO 31
LINE 65: 31 TO 30
LINE 66: 30 TO 20
LINE 67: 37 TO 38
LINE 68: 38 TO 39
LINE 69: 39 TO 40
LINE 70: 40 TO 37
LINE 71: 41 TO 42
LINE 72: 42 TO 43
LINE 73: 43 TO 44
LINE 74: 44 TO 41
LINE 75: 37 TO 41
LINE 76: 38 TO 42
LINE 77: 39 TO 43
LINE 78: 40 TO 44
LINE 79: 45 TO 46
LINE 80: 46 TO 47
LINE 81: 47 TO 48
LINE 82: 48 TO 45
LINE 83: 49 TO 50
LINE 84: 50 TO 51
LINE 85: 51 TO 52
LINE 86: 52 TO 49
LINE 87: 45 TO 49
LINE 88: 46 TO 50
LINE 89: 47 TO 51
LINE 90: 48 TO 52
LINE 91: 53 TO 54
LINE 92: 54 TO 55
LINE 93: 55 TO 56
LINE 94: 56 TO 53
LINE 95: 57 TO 58
LINE 96: 58 TO 59
LINE 97: 59 TO 60
LINE 98: 60 TO 57
LINE 99: 53 TO 57
LINE 100: 54 TO 58
LINE 101: 55 TO 59
LINE 102: 56 TO 60
LINE 103: 61 TO 62
LINE 104: 62 TO 63
LINE 105: 63 TO 64
LINE 106: 64 TO 61
LINE 107: 65 TO 66
LINE 108: 66 TO 67
LINE 109: 67 TO 68
LINE 110: 68 TO 65
LINE 111: 61 TO 65
LINE 112: 62 TO 66
LINE 113: 63 TO 67
LINE 114: 64 TO 68
LINE 115: 69 TO 70
LINE 116: 70 TO 71
LINE 117: 71 TO 72
LINE 118: 72 TO 69
LINE 119: 73 TO 74
LINE 120: 74 TO 75
LINE 121: 75 TO 76
LINE 122: 76 TO 73
LINE 123: 69 TO 73
LINE 124: 70 TO 74
LINE 125: 71 TO 75
LINE 126: 72 TO 76
LINE 127: 77 TO 78
LINE 128: 78 TO 79
LINE 129: 79 TO 80
LINE 130: 80 TO 77
LINE 131: 81 TO 82
LINE 132: 82 TO 83
LINE 133: 83 TO 84
LINE 134: 84 TO 81
LINE 135: 77 TO 81
LINE 136: 78 TO 82
LINE 137: 79 TO 83
LINE 138: 80 TO 84
```



TIE FIGHTER VIEW DATA:

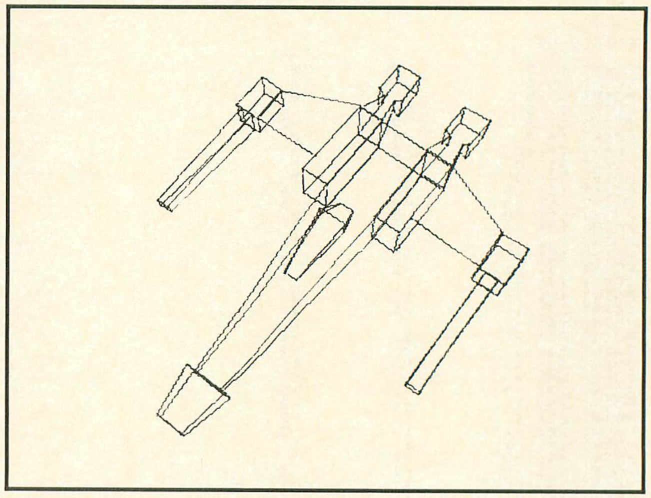
OBSERVER LOC.	LOOKED AT	ZOOM
100,173,50	0,0,0	6

Interestingly, a few days after receiving the TIE fighter, I got a letter from Jason Leigh, of Kowloon, Hong Kong. He sent in — you guessed it — a 3-D X-Wing fighter, the spacecraft used by the rebel forces in the *Star Wars* films!

Here's the X-Wing data:

```
POINTS:94

POINT 1: 0       26      11
POINT 2: 0       22      11
POINT 3: 7       21      12
POINT 4: 7       27      12
POINT 5: 7       26      11
POINT 6: 7       22      11
POINT 7: 37      28      13
POINT 8: 37      32      13
POINT 9: 40      32      13
POINT 10: 40     43      12
POINT 11: 40     44      12
POINT 12: 22     44      11
POINT 13: 22     46      11
POINT 14: 40     46      12
POINT 15: 40     47      12
POINT 16: 46     47      12
POINT 17: 46     43      12
POINT 18: 50     32      13
POINT 19: 55     31      11
POINT 20: 55     32      12
POINT 21: 60     32      12
POINT 22: 60     28      12
POINT 23: 55     28      12
POINT 24: 55     29      11
POINT 25: 50     28      13
POINT 26: 50     20      13
POINT 27: 55     19      11
POINT 28: 55     20      12
POINT 29: 60     20      12
POINT 30: 60     16      12
POINT 31: 55     16      12
```

```
POINT 32: 55        17      11
POINT 33: 50        16      13
POINT 34: 46        5       12
POINT 35: 46        1       12
POINT 36: 40        1       12
POINT 37: 40        2       12
POINT 38: 22        2       11
POINT 39: 22        4       11
POINT 40: 40        4       12
POINT 41: 40        5       12
POINT 42: 40        16      13
POINT 43: 37        16      13
POINT 44: 37        20      13
POINT 45: 24        23      12
POINT 46: 24        25      12
POINT 47: 33        26      15
POINT 48: 33        22      15
POINT 49: 36        25      13
POINT 50: 36        23      13
POINT 51: 0         26      9
POINT 52: 0         22      9
POINT 53: 7         21      8
POINT 54: 7         27      8
POINT 55: 7         26      9
POINT 56: 7         22      9
POINT 57: 37        28      6
POINT 58: 37        32      6
POINT 59: 40        43      8
POINT 60: 40        44      8
POINT 61: 22        44      9
POINT 62: 22        46      9
POINT 63: 40        46      8
POINT 64: 40        47      8
POINT 65: 46        47      8
POINT 66: 46        43      8
POINT 67: 50        32      6
POINT 68: 55        31      9
POINT 69: 55        32      8
POINT 70: 60        32      8
POINT 71: 60        28      8
POINT 72: 55        28      8
POINT 73: 55        29      9
POINT 74: 50        28      6
POINT 75: 50        20      6
POINT 76: 55        19      9
POINT 77: 55        20      8
POINT 78: 60        20      8
POINT 79: 60        16      8
POINT 80: 55        16      8
POINT 81: 55        17      9
POINT 82: 50        16      6
POINT 83: 46        5       8
POINT 84: 46        1       8
POINT 85: 40        1       8
POINT 86: 40        2       8
POINT 87: 22        2       9
POINT 88: 22        4       9
POINT 89: 40        4       8
POINT 90: 40        5       8
POINT 91: 37        16      6
POINT 92: 37        20      6
POINT 93: 33        26      13
POINT 94: 33        22      13

LINES:161

LINE 1: 1 TO 51
LINE 2: 2 TO 52
LINE 3: 3 TO 53
LINE 4: 4 TO 54
LINE 5: 5 TO 55
LINE 6: 6 TO 56
LINE 7: 7 TO 57
LINE 8: 8 TO 58
LINE 9: 9 TO 10
LINE 10: 10 TO 59
LINE 11: 11 TO 60
LINE 12: 12 TO 61
LINE 13: 13 TO 62
LINE 14: 14 TO 63
LINE 15: 15 TO 64
LINE 16: 16 TO 65
LINE 17: 17 TO 66
LINE 18: 18 TO 67
LINE 19: 19 TO 68
LINE 20: 20 TO 69
LINE 21: 21 TO 70
LINE 22: 22 TO 71
LINE 23: 23 TO 72
LINE 24: 24 TO 73
LINE 25: 25 TO 74
LINE 26: 26 TO 75
LINE 27: 27 TO 76
LINE 28: 28 TO 77
LINE 29: 29 TO 78
LINE 30: 30 TO 79
LINE 31: 31 TO 80
LINE 32: 32 TO 81
LINE 33: 33 TO 82
LINE 34: 34 TO 83
LINE 35: 35 TO 84
LINE 36: 36 TO 85
LINE 37: 37 TO 86
LINE 38: 38 TO 87
LINE 39: 39 TO 88
LINE 40: 40 TO 89
LINE 41: 41 TO 90
LINE 42: 42 TO 41
LINE 43: 43 TO 91
LINE 44: 44 TO 92
LINE 45: 10 TO 17
LINE 46: 17 TO 18
LINE 47: 33 TO 34
LINE 48: 34 TO 41
LINE 49: 12 TO 13
LINE 50: 13 TO 14
LINE 51: 14 TO 11
LINE 52: 11 TO 12
LINE 53: 61 TO 62
LINE 54: 62 TO 63
LINE 55: 63 TO 60
LINE 56: 60 TO 61
LINE 57: 10 TO 15
LINE 58: 15 TO 16
LINE 59: 16 TO 17
LINE 60: 17 TO 10
LINE 61: 59 TO 64
LINE 62: 64 TO 65
LINE 63: 65 TO 66
LINE 64: 66 TO 59
LINE 65: 38 TO 39
LINE 66: 39 TO 40
LINE 67: 40 TO 37
LINE 68: 37 TO 38
LINE 69: 87 TO 88
LINE 70: 88 TO 89
LINE 71: 89 TO 86
LINE 72: 86 TO 87
LINE 73: 41 TO 34
LINE 74: 34 TO 35
LINE 75: 35 TO 36
LINE 76: 36 TO 41
LINE 77: 90 TO 83
LINE 78: 83 TO 84
LINE 79: 84 TO 85
LINE 80: 85 TO 90
LINE 81: 46 TO 45
LINE 82: 46 TO 47
LINE 83: 47 TO 49
LINE 84: 49 TO 50
LINE 85: 50 TO 48
LINE 86: 48 TO 45
LINE 87: 45 TO 94
LINE 88: 94 TO 50
LINE 89: 46 TO 93
LINE 90: 93 TO 49
LINE 91: 47 TO 48
LINE 92: 47 TO 93
LINE 93: 48 TO 94
LINE 94: 1 TO 2
LINE 95: 2 TO 3
LINE 96: 3 TO 4
LINE 97: 4 TO 1
LINE 98: 51 TO 52
LINE 99: 52 TO 53
LINE 100: 53 TO 54
LINE 101: 54 TO 51
LINE 102: 5 TO 6
LINE 103: 6 TO 44
LINE 104: 44 TO 26
LINE 105: 26 TO 25
LINE 106: 25 TO 7
LINE 107: 7 TO 5
LINE 108: 55 TO 56
LINE 109: 56 TO 92
LINE 110: 92 TO 75
LINE 111: 75 TO 74
LINE 112: 74 TO 57
LINE 113: 57 TO 55
LINE 114: 25 TO 18
LINE 115: 18 TO 8
LINE 116: 8 TO 7
LINE 117: 7 TO 25
LINE 118: 74 TO 67
LINE 119: 67 TO 58
LINE 120: 58 TO 57
LINE 121: 57 TO 74
LINE 122: 44 TO 43
LINE 123: 43 TO 33
LINE 124: 33 TO 26
LINE 125: 26 TO 44
LINE 126: 92 TO 91
LINE 127: 82 TO 75
LINE 128: 75 TO 92
LINE 129: 18 TO 19
LINE 130: 19 TO 20
LINE 131: 20 TO 21
LINE 132: 21 TO 22
LINE 133: 22 TO 23
LINE 134: 23 TO 24
LINE 135: 24 TO 25
LINE 136: 25 TO 18
LINE 137: 67 TO 68
LINE 138: 68 TO 69
LINE 139: 69 TO 70
LINE 140: 70 TO 71
LINE 141: 71 TO 72
LINE 142: 72 TO 73
LINE 143: 73 TO 74
LINE 144: 74 TO 67
LINE 145: 26 TO 33
LINE 146: 33 TO 32
LINE 147: 32 TO 31
LINE 148: 31 TO 30
LINE 149: 30 TO 29
LINE 150: 29 TO 28
LINE 151: 28 TO 27
LINE 152: 27 TO 26
LINE 153: 75 TO 82
LINE 154: 82 TO 81
LINE 155: 81 TO 80
LINE 156: 80 TO 79
LINE 157: 79 TO 78
LINE 158: 78 TO 77
LINE 159: 77 TO 76
LINE 160: 76 TO 75
LINE 161: 91 TO 82
```



X-WING VIEW DATA:

OBSERVER LOC.	LOOKED AT	ZOOM
0,0,100	30,25,5	.7

Now you can stage your own space battles using **Solid States** and your Atari computer!

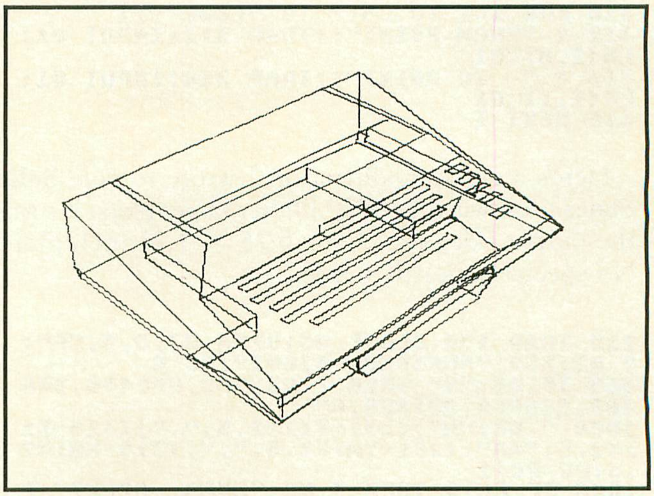
Atari self-portrait.

Our last object this time out is an Atari 800 computer, generated by Robert Groves, of Columbus, Ohio. This graphic has a great deal of detail, and I cringe just thinking of all the time Robert must have spent getting it right.

Here's the data for the Atari 800:

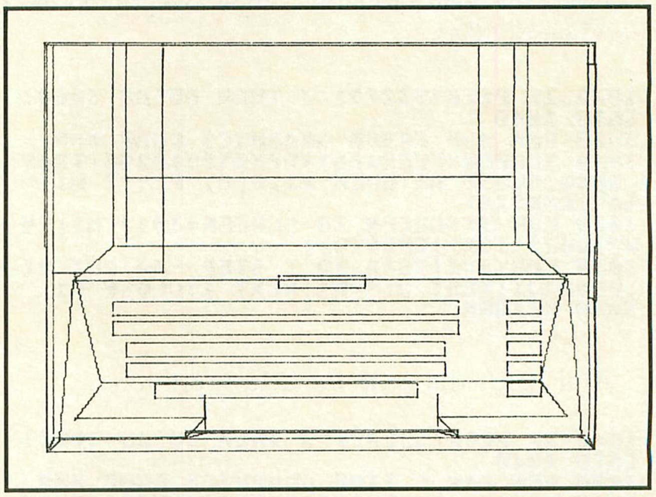
```
POINTS:112

POINT 1: 4          3       1.5
POINT 2: 4          -3      0
POINT 3: -4         -3      0
POINT 4: -4         3       1.5
POINT 5: -3         3       1.5
POINT 6: -3         0       1.5
POINT 7: -3.6       -0.5    0.9
POINT 8: -3.8       -2.8    0.3
POINT 9: 3.8        -2.8    0.3
POINT 10: 3.6       -0.5    0.9
POINT 11: 3         0       1.5
POINT 12: 3         3       1.5
POINT 13: 2.8       3       1.5
POINT 14: 2.8       1       1.5
POINT 15: 2.8       0       1.5
POINT 16: -2.8      0       1.5
POINT 17: -2.8      1       1.5
POINT 18: -2.8      3       1.5
POINT 19: -2.8      -0.5    0.9
POINT 20: -0.6      -0.5    0.9
POINT 21: 0.6       -0.5    0.9
POINT 22: 2.8       -0.5    0.9
POINT 23: 0.6       -0.4    1
POINT 24: -0.6      -0.4    1
POINT 25: 0.6       -0.4    0.8
POINT 26: -0.6      -0.4    0.8
POINT 27: 0.6       -0.6    0.8
POINT 28: -0.6      -0.6    0.8
POINT 29: -3        -0.8    1
POINT 30: 2         -0.8    1
POINT 31: 2.7       -0.8    1
POINT 32: 3.2       -0.8    1
POINT 33: -3        -1      1
POINT 34: 2         -1      1
POINT 35: 2.7       -1      1
POINT 36: 3.2       -1      1
POINT 37: -3        -1.1    0.9
POINT 38: 2         -1.1    0.9
POINT 39: 2.7       -1.1    0.9
POINT 40: 3.2       -1.1    0.9
POINT 41: -3        -1.3    0.9
POINT 42: 2         -1.3    0.9
POINT 43: 2.7       -1.3    0.9
POINT 44: 3.2       -1.3    0.9
POINT 45: -2.8      -1.4    0.8
POINT 46: 1.8       -1.4    0.8
POINT 47: 2.7       -1.4    0.8
POINT 48: 3.2       -1.4    0.8
POINT 49: 3.2       -1.6    0.8
POINT 50: 2.7       -1.6    0.8
```

```
POINT 51: 1.8    -1.6   Ø.8
POINT 52: -2.8   -1.6   Ø.8
POINT 53: -2.8   -1.7   Ø.7
POINT 54: 1.8    -1.7   Ø.7
POINT 55: 2.7    -1.7   Ø.7
POINT 56: 3.2    -1.7   Ø.7
POINT 57: 3.2    -1.9   Ø.7
POINT 58: 2.7    -1.9   Ø.7
POINT 59: 1.8    -1.9   Ø.7
POINT 6Ø: -2.8   -1.9   Ø.7
POINT 61: -2.4   -2     Ø.6
POINT 62: 1.4    -2     Ø.6
POINT 63: 2.7    -2     Ø.5
POINT 64: 3.2    -2     Ø.5
POINT 65: 3.2    -2.2   Ø.5
POINT 66: 2.7    -2.2   Ø.5
POINT 67: 1.4    -2.2   Ø.6
POINT 68: -2.4   -2.2   Ø.6
POINT 69: Ø      Ø      Ø
POINT 7Ø: 4      -Ø.8   Ø
POINT 71: 4      -Ø.8   Ø.2
POINT 72: 4      2.7    Ø.2
POINT 73: 4      2.7    Ø
POINT 74: 3.9    2.7    -Ø.2
POINT 75: 3.9    -Ø.8   -Ø.2
POINT 76: 4      3      Ø
POINT 77: 3.9    3      -Ø.3
POINT 78: 3.9    3      -Ø.7
POINT 79: 3.6    -Ø.4   -Ø.7
POINT 8Ø: 3.2    -2     -Ø.7
POINT 81: 3.6    -2.5   -Ø.3
POINT 82: 3.9    -2.8   -Ø.3
POINT 83: 3.9    -Ø.4   -Ø.3
POINT 84: 2.3    -Ø.4   -Ø.7
POINT 85: 2.3    -Ø.4   -Ø.3
POINT 86: 2.3    3      -Ø.7
POINT 87: 1.7    -2.7   -Ø.7
POINT 88: -1.7   -2.7   -Ø.7
POINT 89: -3.2   -2     -Ø.7
POINT 9Ø: -3.6   -2.5   -Ø.3
POINT 91: -2     -2.8   -Ø.3
POINT 92: 2      -2.8   -Ø.3
POINT 93: 2      -2.7   -Ø.1
POINT 94: -2     -2.7   -Ø.1
POINT 95: -1.7   -2.15  -Ø.7
POINT 96: 1.7    -2.15  -Ø.7
POINT 97: 1.7    -2.5   -Ø.3
POINT 98: -1.7   -2.5   -Ø.3
POINT 99: -3.9   -2.8   -Ø.3
POINT 1ØØ: -2    -2.7   -Ø.3
POINT 1Ø1: -2    -2.7   -Ø.1
POINT 1Ø2: 2     -2.7   -Ø.1
POINT 1Ø3: 2     -2.7   -Ø.3
POINT 1Ø4: 2.3   3      -Ø.3
POINT 1Ø5: -3.9  3      -Ø.3
POINT 1Ø6: -4    3      Ø
POINT 1Ø7: -2.3  3      -Ø.3
POINT 1Ø8: -2.3  3      -Ø.7
POINT 1Ø9: -2.3  -Ø.4   -Ø.7
POINT 11Ø: -2.3  -Ø.4   -Ø.3
POINT 111: -3.6  -Ø.4   -Ø.7
POINT 112: -3.9  -Ø.4   -Ø.3

LINES:122

LINE 1: 1 TO 2
LINE 2: 2 TO 3
LINE 3: 3 TO 4
LINE 4: 4 TO 1
LINE 5: 5 TO 6
LINE 6: 6 TO 7
LINE 7: 7 TO 8
LINE 8: 8 TO 9
LINE 9: 9 TO 1Ø
LINE 1Ø: 1Ø TO 11
LINE 11: 11 TO 12
LINE 12: 6 TO 6
LINE 13: 6 TO 11
LINE 14: 7 TO 1Ø
LINE 15: 13 TO 15
LINE 16: 16 TO 18
LINE 17: 14 TO 17
LINE 18: 19 TO 2Ø
LINE 19: 21 TO 22
LINE 2Ø: 23 TO 25
LINE 21: 25 TO 27
LINE 22: 8Ø TO 89
LINE 23: 24 TO 26
LINE 24: 26 TO 28
LINE 25: 27 TO 28
LINE 26: 24 TO 23
LINE 27: 29 TO 3Ø
LINE 28: 31 TO 32
LINE 29: 31 TO 35
LINE 3Ø: 35 TO 36
LINE 31: 36 TO 32
LINE 32: 3Ø TO 34
LINE 33: 33 TO 34
LINE 34: 33 TO 29
LINE 35: 37 TO 38
LINE 36: 38 TO 42
LINE 37: 42 TO 41
LINE 38: 41 TO 37
LINE 39: 39 TO 4Ø
LINE 4Ø: 4Ø TO 44
LINE 41: 44 TO 43
LINE 42: 43 TO 39
LINE 43: 45 TO 46
LINE 44: 47 TO 48
LINE 45: 48 TO 49
LINE 46: 49 TO 5Ø
LINE 47: 5Ø TO 47
LINE 48: 46 TO 51
LINE 49: 51 TO 52
LINE 5Ø: 52 TO 45
LINE 51: 53 TO 54
LINE 52: 55 TO 56
LINE 53: 56 TO 57
LINE 54: 57 TO 58
LINE 55: 58 TO 55
LINE 56: 59 TO 54
LINE 57: 59 TO 6Ø
LINE 58: 6Ø TO 53
LINE 59: 61 TO 62
LINE 6Ø: 63 TO 64
LINE 61: 64 TO 65
LINE 62: 65 TO 66
LINE 63: 66 TO 63
LINE 64: 62 TO 67
LINE 65: 67 TO 68
LINE 66: 68 TO 61
LINE 67: 73 TO 76
LINE 68: 71 TO 72
LINE 69: 72 TO 74
LINE 7Ø: 74 TO 75
LINE 71: 75 TO 71
LINE 72: 77 TO 82
LINE 73: 3 TO 8
LINE 74: 84 TO 86
LINE 75: 8Ø TO 81
LINE 76: 79 TO 83
LINE 77: 84 TO 85
LINE 78: 85 TO 83
LINE 79: 84 TO 79
LINE 8Ø: 79 TO 8Ø
LINE 81: 85 TO 1Ø4
LINE 82: 82 TO 92
LINE 83: 96 TO 97
LINE 84: 97 TO 81
LINE 85: 97 TO 92
LINE 86: 92 TO 87
LINE 87: 92 TO 93
LINE 88: 93 TO 94
LINE 89: 94 TO 91
LINE 9Ø: 91 TO 88
LINE 91: 88 TO 87
LINE 92: 88 TO 95
LINE 93: 95 TO 98
LINE 94: 98 TO 91
LINE 95: 91 TO 99
LINE 96: 9Ø TO 89
LINE 97: 9Ø TO 98
LINE 98: 87 TO 96
LINE 99: 3 TO 1Ø6
LINE 1ØØ: 1Ø6 TO 1Ø5
LINE 1Ø1: 1Ø7 TO 1Ø8
LINE 1Ø2: 2 TO 82
LINE 1Ø3: 1Ø8 TO 1Ø9
LINE 1Ø4: 1Ø9 TO 11Ø
LINE 1Ø5: 11Ø TO 112
LINE 1Ø6: 112 TO 111
LINE 1Ø7: 111 TO 1Ø9
LINE 1Ø8: 111 TO 89
LINE 1Ø9: 99 TO 1Ø5
LINE 11Ø: 77 TO 76
LINE 111: 1Ø4 TO 86
LINE 112: 86 TO 1Ø8
LINE 113: 1Ø7 TO 1Ø8
LINE 114: 77 TO 1Ø5
LINE 115: 76 TO 1
LINE 116: 1Ø6 TO 4
LINE 117: 1Ø7 TO 11Ø
LINE 118: 1Ø1 TO 1Ø2
LINE 119: 1Ø3 TO 1Ø2
LINE 12Ø: 3 TO 99
LINE 121: 9 TO 2
LINE 122: 7Ø TO 2
```



800 VIEW DATA:

OBSERVER LOC.	LOOKED AT	ZOOM
-12,-12,12	0,0,0	1



800 TOP VIEW DATA:

OBSERVER LOC.	LOOKED AT	ZOOM
0,0,1500	0,0,0	100

Well, that just about does it for 3-D images this month. We received several others, some of which were just too large to print. If you've got a 3-D object you'd like to send, try to keep it under 250 points and lines total.

No more boo-boos.

Vinette DePhillipe and Ted Talay, of Hampton, Virginia, submitted this short program modification which traps errors in the initial data entry process. This eliminates the annoyance of having to re-enter a whole object's data when an entry error is made.

```
270 FOR I=1 TO P5:? "POINT ";I
272 TRAP 272:INPUT Q1,Q2,Q3
274 X(I)=Q1:Y(I)=Q2:Z(I)=Q3:NEXT I
```

```
310 FOR I=1 TO L5:? :? "LINE ";I
312 ? "FROM POINT";:TRAP 312:INPUT Q1:
LN(I,0)=Q1
314 ? "  TO POINT";:TRAP 314:INPUT Q1:
LN(I,1)=Q1
316 NEXT I
```

Here's a quick modification I wrote to give **Solid States** users with the 1020 Plotter larger plots. It turns the output sideways, giving 6.25 × 3.75 inch plots. No special action is needed.

```
210 TRAP 190:CLOSE #3:OPEN #3,8,0,"P":
? #3;EG$;"*H*I*M0,-820*I":PC=0
860 IF O$="Y" THEN ? #3;"M0,0*D480,0*D
480,800*D0,800*D0,0"
1000 ? #3;"M";(191-Y1)*2.5;",";(319-X1
)*2.5;"*D";(191-YW)*2.5;",";(319-XW)*2
.5:PC=PC+1
1020 IF PC>0 THEN ? #3;"H*M0,-820*I":P
C=0
```

Printer output!

I've received quite a few letters from readers wanting hard-copy output on their Epson, Gemini 10X and C.Itoh printers. You asked for it, and you've got it!

Here's the code for Epson-compatible printers (such as Gemini 10X):

```
1045 IF PEEK(53279)=3 THEN GOSUB 3000:
GOTO 1040
3000 REM *** EPSON GRAPHICS DUMP ***
3010 SCREEN=PEEK(88)+PEEK(89)*256:TRAP
 3040:CLOSE #1:OPEN #1,8,0,"P:":? #1;"
␛A";CHR$(8)
3020 FOR I=SCREEN TO SCREEN+39:? #1;"␛
K";CHR$(192);CHR$(0);
3030 FOR J=I+7640 TO I STEP -40:PUT #1
,PEEK(J):NEXT J:? #1:NEXT I:CLOSE #1
3040 RETURN
```

And here's the code for C.Itoh printers:

```
1045 IF PEEK(53279)=3 THEN GOSUB 3000:
GOTO 1040
3000 REM *** C.ITOH GRAPHICS DUMP ***
3010 SCREEN=PEEK(88)+PEEK(89)*256:TRAP
 3040:CLOSE #1:OPEN #1,8,0,"P:":? #1;"
␛T16"
3020 FOR I=SCREEN+39 TO SCREEN STEP -1
:? #1;"␛S0192";
3030 FOR J=I TO I+7640 STEP 40:PUT #1,
PEEK(J):NEXT J:? #1:NEXT I:CLOSE #1
3040 RETURN
```

After **Solid States** has completed the picture on the screen, you'll hear a short tone. Pressing OPTION will print the screen on your printer (the process takes about 2.5 minutes). It's as easy as that! If you don't want to print the screen, pressing START will return you to the coordinate entry point, just like the original version of the program.

If your printer isn't ready, the program will go back and wait for the START or OPTION keys again. Simply ready your printer and press OPTION to print the screen.

Remember, your printer *must* be equipped with graphics ability in order for the screen-print function to work.

Supply and demand.

As long as I receive your 3-D object data and suggestions for modifications, we'll be running **Solid States** updates like this indefinitely.

One thing I'd like to do is equip **Solid States** with hidden-line removal, so the drawings won't be cluttered with lines that should be invisible. If you know of a good hidden-line removal algorithm, *please* let me know. I'm checking my resources, but, so far, I haven't found anything specific on the process. If you want to see your name in lights (well, at least, in the pages of ANALOG), send me anything you can find!

I'm also working on converting **Solid States** to a 100% machine language program for much faster operation. I'm going to wait until I find a hidden-line algorithm before I do this, though.

Keep those 3-D objects coming! Send them to:

Solid States
c/o ANALOG Computing
P.O. Box 23
Worcester, MA 01603

If you don't have the **Solid States** program, ANALOG issue 16 is still available as a back issue. See the ad elsewhere in this issue. □

Mathman

16K Cassette or 24K Disk

by Francisco R. Moncada

Mathman is an educational game which is useful in helping children learn their math in a variation of Hangman. When a child enters the correct answer, the computer congratulates him or her with a musical tune, and the screen turns different colors. If the answer is incorrect, then the computer emits an angry buzz and turns the screen red. This also causes a part of the man to be drawn on the screen.

The object of the game is to answer as many problems as possible correctly before being hanged. If your man is hung, the game is over. **Mathman** has six different levels to choose from. The bigger the number entered, the higher the level. **Mathman** has problems ranging from addition and subtraction to multiplication, with numbers from 0 through 99.

Answers are entered by typing from right to left. For example, to type the answer "23," first type in the "3" and then the "2." There is no need to push the return key; the computer enters the number once the last number has been entered. Another thing to note is that **Mathman** gives you only fifteen seconds to enter your answer. This means that, if an answer is not typed in within fifteen seconds, the computer will read the blank as a wrong answer.

One warning: **Mathman** has no delete feature. Once a number is entered, it can't be erased. I hope that you enjoy this program and find it useful. □

```
1 REM ***************************
2 REM *         MATHMAN         *
3 REM * BY FRANCISCO MONCADA *
4 REM *    ANALOG COMPUTING    *
5 REM ***************************
11 HS=25:GOTO 5000
100 A=INT(11*RND(0)):B=INT(11*RND(0)):
SIGN=1:RETURN
200 A=INT(26*RND(0)):B=INT(26*RND(0)):
SIGN=1:RETURN
250 A=INT(11*RND(0)):B=INT(10*RND(0)):
SIGN=3:RETURN
300 A=INT(51*RND(0)):B=INT(51*RND(0)):
SIGN=2:IF B>A THEN 300
305 RETURN
400 A=INT(21*RND(0)):B=INT(21*RND(0)):
SIGN=2:IF B>A THEN 400
405 RETURN
450 AB=INT(3*RND(0)+1):ON AB GOSUB 200
,250,300:RETURN
600 SCOR=SCOR+1:POKE 87,1:IF SCOR>99 T
HEN SCOR=0:POSITION 18,0:? #6;" "
610 POSITION 17,0:? #6;SCOR:IF SCOR>HS
 THEN HS=SCOR:POSITION 7,0:? #6;HS
620 POKE 87,2:FOR SO=30 TO 98 STEP 3:K
=PEEK(53279):IF K=5 THEN 5000
625 SOUND 0,0,14,8:FOR I=0 TO 10:NEXT
I:SOUND 1,SO+5,10,7:POKE 712,98:IF K=6
 THEN 6000
630 POSITION 9,4:POKE 711,PEEK(53770):
? #6;" right;":NEXT SO
640 SOUND 0,0,0,0:SOUND 1,0,0,0:POSITI
ON 9,4:? #6;"          "
650 RETURN
800 POKE 87,2:FOR I=0 TO 40:SOUND 0,10
8,12,10:K=PEEK(53279):IF K=6 THEN 6000
805 POSITION 9,4:? #6;" WRONG!":IF K=5
 THEN 5000
810 NEXT I:POSITION 9,4:? #6;"         "
:SOUND 0,0,0,0:RETURN
```

```
1000 GRAPHICS 17:POS=64:P=PEEK(16):IF
P>127 THEN P=P-128:POKE 16,P:POKE 5377
4,P
1010 BE=PEEK(560)+PEEK(561)*256+4:F=1
1020 POKE BE-1,71:POKE BE+2,7:POKE BE+
3,7:POKE BE+4,7
1030 FOR I=5 TO 24:POKE BE+I,6:NEXT I
1040 POKE BE+19,65:POKE BE+20,PEEK(560
):POKE BE+21,PEEK(561):POKE 87,2:POKE
710,10:POKE 711,38:POKE 712,144
1045 POSITION 5,0:? #6;"HIGH SCORE":PO
SITION 9,1:? #6;HS:SC=PEEK(88)+256*PEE
K(89):GOSUB 1130
1050 POKE 87,1:POSITION 2,5:? #6;"r:TO
 PICK LEVEL":POSITION 4,6:? #6;"PUSH s
elect."
1055 POSITION 2,8:? #6;"-:TO START GAM
E":POSITION 4,9:? #6;"PUSH start.":O=0
1100 O=O+1:FOR I=0 TO 7:NEXT I:POKE 70
8,PEEK(53770):POKE 709,PEEK(53770):IF
O=150 THEN 5000
1115 IF PEEK(53279)=5 THEN GOSUB 20000
:GOTO 6000
1117 IF PEEK(53279)=6 THEN 6000
1120 GOTO 1100
1130 RESTORE 1150
1135 POS=POS+1:READ M:IF M=-1 THEN RET
URN
1140 POKE SC+POS,M:FOR I=0 TO 40:NEXT
I:GOTO 1135
1150 DATA 39,97,45,101,0,47,118,37,114
,-1
2000 WRONG=WRONG+1:POKE 712,48:POKE 71
0,248
2010 POKE 87,7:ON WRONG GOSUB 2050,206
0,2070,2080,2090,2100,2110,2120,2130
2020 RETURN
2050 COLOR 3:PLOT 9,6:DRAWTO 17,6:PLOT
 18,6:DRAWTO 18,15
2053 PLOT 8,6:DRAWTO 8,15:DRAWTO 17,15
:PLOT 12,16:DRAWTO 12,18:PLOT 14,16:DR
AWTO 14,18
2055 GOSUB 800:RETURN
2060 PLOT 7,18:DRAWTO 19,18:DRAWTO 19,
29
2065 DRAWTO 7,29:DRAWTO 7,18:GOSUB 800
:RETURN
2070 PLOT 7,30:DRAWTO 7,40:DRAWTO 10,4
0:DRAWTO 10,34:DRAWTO 11,34
2075 PLOT 19,30:DRAWTO 19,40:DRAWTO 16
,40:DRAWTO 16,34:DRAWTO 15,34
2078 DRAWTO 11,34:GOSUB 800:RETURN
2080 PLOT 4,18:DRAWTO 4,26:DRAWTO 5,26
:DRAWTO 5,21
2085 PLOT 6,21:PLOT 4,18:DRAWTO 6,18:P
LOT 6,27
2087 GOSUB 800:RETURN
2090 PLOT 22,18:DRAWTO 22,26:DRAWTO 21
,26:DRAWTO 21,21
2095 PLOT 20,21:PLOT 19,18:DRAWTO 21,1
8:PLOT 20,27
2097 GOSUB 800:RETURN
2100 PLOT 5,41:DRAWTO 10,41:PLOT 5,42:
DRAWTO 10,42
2105 GOSUB 800:RETURN
2110 PLOT 16,41:DRAWTO 21,41:PLOT 16,4
2:DRAWTO 21,42
2115 GOSUB 800:RETURN
2120 COLOR 2:PLOT 11,8:PLOT 15,8:PLOT
13,9
2125 PLOT 12,10:DRAWTO 14,10:PLOT 12,1
2:DRAWTO 14,12
2127 PLOT 11,13:PLOT 15,13:GOSUB 800:R
ETURN
2130 PLOT 8,4:DRAWTO 13,4:PLOT 7,5:DRA
WTO 18,5
2135 PLOT 7,6:DRAWTO 7,11:PLOT 19,6:DR
AWTO 19,10:GOSUB 800:RETURN
5000 GRAPHICS 17:P=PEEK(16):IF P>127 T
HEN P=P-128:POKE 16,P:POKE 53774,P
5010 BE=PEEK(560)+PEEK(561)*256+4:SOUN
D 0,0,0,0:SOUND 1,0,0,0:F=1
5020 POKE BE-1,71:POKE BE+3,7:POKE BE+
4,7
5030 FOR I=5 TO 24:POKE BE+I,6
5040 NEXT I:POKE 710,10:POKE 711,38
5050 POKE BE+19,65:POKE BE+20,PEEK(560
):POKE BE+21,PEEK(561):POKE 712,144
5070 POKE 87,2:POSITION 6,0:? #6;"MaTh
MaN"
5080 POSITION 2,2:? #6;"cOpYrIgHt 1984
":POSITION 2,3:? #6;"FrAnCiScO mOnCaDa
"
5090 POKE 87,1:POSITION 2,6:? #6;"r:TO
 PICK LEVEL":POSITION 4,7:? #6;"PUSH s
elect."
5100 POSITION 2,9:? #6;"-:TO START GAM
E":POSITION 4,10:? #6;"PUSH start.":O=
0
5110 O=O+1:FOR I=0 TO 7:NEXT I:POKE 70
8,PEEK(53770):POKE 709,PEEK(53770):IF
O=150 THEN 1000
5115 IF PEEK(53279)=5 THEN GOSUB 20000
:GOTO 6000
5117 IF PEEK(53279)=6 THEN 6000
5120 GOTO 5110
6000 GRAPHICS 23:POKE 77,0:P=PEEK(16):
IF P>127 THEN P=P-128:POKE 16,P:POKE 5
3774,P
6002 BE=PEEK(560)+PEEK(561)*256+4:WRON
G=0:SCOR=0:POKE 708,12:SOUND 0,0,0,0:S
OUND 1,0,0,0
6005 POKE BE-1,70:POKE BE+2,6:FOR I=3
TO 7:POKE BE+I,7:NEXT I:POKE 710,198:P
OKE 709,44
6010 POKE 87,1:POSITION 2,0:? #6;"high
 ";HS;"  score ";SCOR:POSITION 2,1:? #
6;"L=";F
6012 POKE 87,2:POSITION 5,5:? #6;"____
"
6015 POKE 19,0:POKE 20,0:POKE 712,148:
IF WRONG=9 THEN FOR I=0 TO 200:NEXT I:
GOTO 1000
6020 ON F GOSUB 100,400,250,200,300,45
0:POSITION 7,3:? #6;"  ":POSITION 7,4:
? #6;"  ":POSITION 7,6:? #6;"  "
6025 POSITION 8,3:IF A>9 THEN POSITION
 7,3
6027 ? #6;A:POSITION 8,4:IF B>9 THEN P
OSITION 7,4
6028 ? #6;B:ON SIGN GOTO 6030,6200,630
0
6030 PR=A+B:POSITION 5,4:? #6;CHR$(11)
:IF PR>9 THEN 6040
6035 FOR I=1 TO 1:GOSUB 6500:NEXT I:PO
SITION 8,6:? #6;ANS:GOSUB 6100:GOTO 60
10
6040 FOR I=1 TO 2:GOSUB 6500:POSITION
9-I,6:IF I=1 THEN AN=ANS:? #6;AN
6045 NEXT I:POSITION 7,6:? #6;ANS:ANS=
(ANS*10)+AN:GOSUB 6100:GOTO 6010
6100 IF ANS<>PR THEN GOSUB 2000:RETURN
6105 IF ANS=PR THEN GOSUB 600:RETURN
6200 POSITION 5,4:? #6;CHR$(13):PR=A-B
:IF PR>9 THEN 6210
6205 FOR I=1 TO 1:GOSUB 6500:NEXT I:PO
SITION 8,6:? #6;ANS:GOSUB 6100:GOTO 60
10
6210 FOR I=1 TO 2:GOSUB 6500:POSITION
9-I,6:IF I=1 THEN AN=ANS:? #6;AN
6215 NEXT I:POSITION 7,6:? #6;ANS:ANS=
(ANS*10)+AN:GOSUB 6100:GOTO 6010
6300 POSITION 5,4:? #6;"x":PR=A*B:IF P
R>9 THEN 6310
6305 FOR I=1 TO 1:GOSUB 6500:NEXT I:PO
SITION 8,6:? #6;ANS:GOSUB 6100:GOTO 60
10
6310 FOR I=1 TO 2:GOSUB 6500:POSITION
9-I,6:IF I=1 THEN AN=ANS:? #6;AN
6315 NEXT I:POSITION 7,6:? #6;ANS:ANS=
(ANS*10)+AN:GOSUB 6100:GOTO 6010
6500 POKE 764,255
6505 K=PEEK(53279):IF PEEK(764)=50 THE
N ANS=0:RETURN
6510 IF PEEK(764)=31 THEN ANS=1:RETURN
6515 IF PEEK(764)=30 THEN ANS=2:RETURN
6518 IF K=6 THEN POP :GOTO 6000
6520 IF PEEK(764)=26 THEN ANS=3:RETURN
6521 IF PEEK(19)=3 AND PEEK(20)>138 TH
EN POP :GOSUB 2000:GOTO 6010
6525 IF PEEK(764)=24 THEN ANS=4:RETURN
6530 IF PEEK(764)=29 THEN ANS=5:RETURN
6535 IF PEEK(764)=27 THEN ANS=6:RETURN
6540 IF PEEK(764)=51 THEN ANS=7:RETURN
6543 IF K=5 THEN POP :GOTO 5000
6545 IF PEEK(764)=53 THEN ANS=8:RETURN
```

```
6550 IF PEEK(764)=48 THEN ANS=9:RETURN
6555 GOTO 6505
19000 END
20000 POKE 77,0:F=F+1:IF F>6 THEN F=1
20010 SOUND 0,74+F,14,7:POSITION 4,14:
? #6;"level ";F
20012 FOR I=0 TO 7:NEXT I:POKE 708,PEE
K(53770):POKE 709,PEEK(53770):IF PEEK(
53279)=5 THEN 20012
20013 SOUND 0,0,0,0
20015 IF PEEK(53279)=6 THEN RETURN
20025 IF PEEK(53279)=5 THEN 20000
20030 FOR I=0 TO 7:NEXT I:POKE 708,PEE
K(53770):POKE 709,PEEK(53770):IF PEEK(
53279)=5 THEN 20000
20035 GOTO 20015
```

•

CHECKSUM DATA.

(see page 43)

```
1 DATA 3,97,991,851,11,416,760,787,780
,476,590,465,592,534,999,8352
610 DATA 509,654,331,373,616,606,442,9
13,274,879,506,243,62,699,967,8074
1050 DATA 663,674,401,848,893,702,179,
594,460,310,688,288,783,121,630,8234
2055 DATA 122,206,182,45,468,572,329,9
19,129,720,204,130,204,120,474,4824
2115 DATA 121,292,460,595,59,496,992,4
96,851,890,221,36,628,462,677,7276
5100 DATA 540,402,856,901,724,118,804,
401,31,173,543,381,668,443,618,7603
6030 DATA 830,993,12,109,293,109,790,9
96,15,112,383,999,18,115,28,5802
6505 DATA 662,990,999,145,8,928,14,22,
28,9,133,27,33,771,558,5327
20000 DATA 676,820,231,568,341,910,221
,212,3979
```

•

SEE SPECIAL OFFER BELOW!

YOUR ATARI SOURCE!

No Penalty For Using Your Credit Cards!

COMPUTER PALACE WE KNOW ATARI!

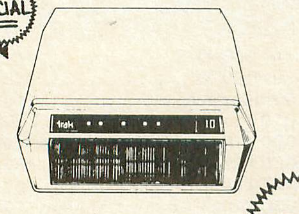
TRAK DISK DRIVES

These drives are some of our best sellers. They all offer true double-density operation, and are dependable and quiet. The AT-D4 is a double-sided drive that packs a whole lot of storage into a compact package (includes parallel printer port & print buffer). The AT-D2 is identical to the AT-D4, but is only a single-sided drive. The CHAMP is a stripped down version of the AT-D2 (without printer port, buffer, or LED readouts). The AT-S1 is a single-sided slave drive for use with any of the above drives.

AT-D4	$529
AT-D2	398
CHAMP	339
AT-S1	289

PLUS FRT.

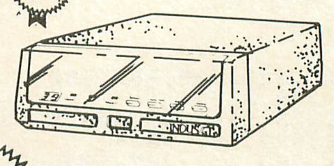
SPECIAL



INDUS GT DISK DRIVE

We recommend and sell more of these drives than any other. It offers true double-density, and is so quiet and dependable that you forget that you are using a disk drive. A Beautiful Jet Black finish with flip-up smoked Plexiglass dust cover and LED readout make it a visual delight. Add to this, the FREE SOFTWARE (Word Processing, Spread-Sheet and Data-Base) and a Full One Year Warranty and you have a winner. Includes DOS-XL.

SPECIAL



Only $398
PLUS FRT.

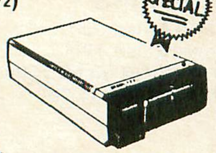
- Double Density
- Free Software
- 1-Year Warranty

ATARI 1050

- Matches XL Computers
- Dual (1½) Density

Only $379
PLUS FRT.

SPECIAL



SPECIAL OFFER

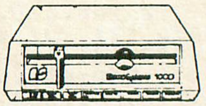
BUY ANY DRIVE AND RECEIVE UP TO $70 IN SOFTWARE ABSOLUTELY FREE!*

*Subject to supplies on hand. We reserve the right to substitute program(s) of equivalent value should your choice be out of stock.
OFFER VALID FROM 7-1-84 to 9-30-84.

CHOOSE ANY TWO (2) OF THESE PROGRAMS FREE WITH YOUR DRIVE PURCHASE:

- Repton • Wayout • Dr. Goodcode's Cavern • Pathfinder • Match Racer • Vocabulary Builder 1 • Vocabulary Builder 2 • Pre-School IQ Builder 1 • Pre-School IQ Builder 2

SPECIAL

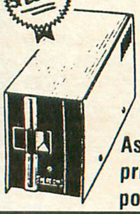


RANA 1000

- Double Density
- 1-Year Warranty

$349 PLUS FRT.

SPECIAL



PERCOM 88-S1PD

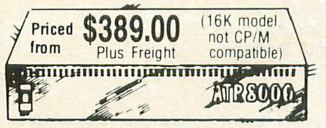
New Low Price!

$369 PLUS FRT.

- Double Density
- Printer Port

PERCOM 40-S1
As above, but without printer port. **$339** PLUS FRT.

ATR-8000 Expander



- 64K Z80 computer
- Runs any combination of 5¼" and 8" drives in single or double density
- Built-in serial and parallel ports
- Built-in printer buffer

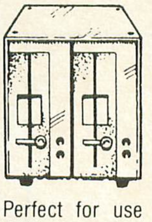
PLUS FRT.

64K CP/M model	$539.00
64K upgrade to 16K model	199.00
MYDOS	29.95

USE WITH RCP DRIVES AT RIGHT.

RCP Add-On Drives

SPECIAL



- Double Density
- Double-Sided

Perfect for use with the ATR-8000 (left), or with any other system that contains a drive controller.

RCP-100 SINGLE DRIVE	RCP-200 DOUBLE DRIVE
$379 PLUS FRT.	$595 PLUS FRT.

RC SYSTEMS 600XL Memory Upgrades

48K	$79.95
64K	99.95

KOALA WARE SOFTWARE

Instant Programmer's Guide (48K D)	$15.00
Koalagrams Spelling I (48K D)	35.10
Spider Eater (48K D)	26.90
Coloring Series I (48K D)	26.90
Coloring Series II (48K D)	26.90
LOGO Design Master (48K D)	35.10

POWER PAD SOFTWARE

Leo's Electric Paintbrush (Cart.)	$26.90
Micro Illustrator (Cart)	44.90
Leonardo's LOGO (Cart)	44.90
Micro Maestro (Cart)	26.90
Logic Master (Cart)	35.10
Bearjam (Cart)	31.50

OTHER GRAPHICS SOFTWARE

Graphics Master (40K D)	$31.50
Fun With Art (Cart)	35.10
Micropainter (48K D)	31.50
Print Wiz (48K D)	26.95
Movie Maker (48K D)	52.50
Graphics Magician (48K D)	35.10
Graphic Generator (32K D)	22.50
Video Easel (Cart)	31.50
Paint (48K D)	35.10

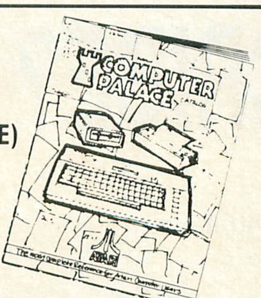
GRAPHICS BOOKS

Atari Sound & Graphics	$10.95
1st Book of Atari Graphics	12.95
Atari Player-Missile Graphics	14.95
Understanding Atari Graphics	2.95
The Creative Atari	15.95

FREE CATALOG

WITH ANY PURCHASE OR SEND $1.00 (REFUNDABLE WITH PURCHASE)

This is the most comprehensive Atari-Only reference catalog ever produced. It contains over 3000 software and hardware listings with illustrations and descriptions.



If you don't have our catalog, you're missing out!

ATARI TOUCH TABLET

- Incl. program on Cartridge.

$79.10 PLUS FRT.

Chalk Board POWER PAD

- Large dwg. surface (no dwg. prog.)

$89.90 PLUS FRT.

Koala Technologies KOALA PAD

(incl. dwg. prog.)

16K Cart	$99.90
48K Disk	89.90

PLUS FRT.

COMPUTER PALACE

OPEN M-F, 9-6 Sat. 10-4 (Pacific Time)
2160 W. 11th Avenue Eugene, Oregon 97402

MasterCard VISA AMERICAN EXPRESS

USE YOUR CREDIT CARD & CALL
Toll Free 1-800-452-8013
★ ORDERS ONLY, PLEASE ★
There's never a penalty for using your credit card!
For Information, Call (503) 683-5361

SHIPPING INFO: Minimum $2.90 Ground, $4.75 Air. Actual Cost depends on weight. Call (503) 683-5361 for information. **WARRANTY INFO:** Everything that we sell is warrantied by the manufacturer. If any item purchased from us fails to perform properly when you receive it, call us at (503) 683-5361 so that we can assist you. No returned merchandise accepted without authorization. Defective software will be replaced with another copy of the same program, otherwise, no software is returnable.

GYRUSS
PARKER BROTHERS
P.O. Box 1012
Beverly, MA 01915
8K Cartridge appr. $40.00 retail

by Tom Hudson

Gyruss is one of those games I never got around to playing at the arcade. Oh, I had seen it, all right, but either someone else was playing it, or I had blown all my quarters on the **Tempest** machines.

Interestingly, the thing that originally drew my attention to **Gyruss** is the fact that it is somewhat similar to **Tempest** in gameplay. In **Tempest**, your "shooter" is perched on the rim of a tunnel-like structure, shooting down at the things climbing up toward it. At the end of each round, your shooter flies down the tunnel at high speed, onto the next level.

In **Gyruss**, you're controlling a spaceship which is flying from Neptune to Earth. The ship can move in a circular pattern on the screen and fires toward the center of the screen. This aspect of the program is primarily what reminds me of **Tempest**.

The opposition.

Flying from Neptune to Earth in **Gyruss** is no picnic. The intervening space is crowded with several types of enemy spacecraft. You must destroy as many as possible in order to reach Earth safely.

The first type of spacecraft is the enemy plane. These are high speed attack vehicles which fly in several different formations. They may appear from far out in front of you or they can launch a surprise attack and come screaming past you from behind. These ships can shoot missiles at you, but they don't stop there. If they get half a chance, they'll fly themselves into you!

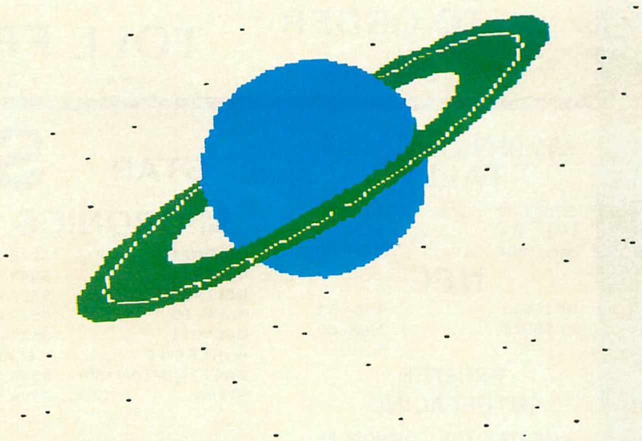
There are also "molecular satellites," so named because they look like giant molecules. These appear in threes, fly in formation and shoot missiles at you. If you can destroy the middle satellite, you fire double bullets.

The third type of spacecraft is the electromagnetic wave satellite. These come in pairs, shooting a deadly wave of energy between them. As if these enemy spacecraft weren't enough danger, there are also meteors coming from out of the distance, which you must avoid, too.

A breather.

Each time you destroy a wave of enemy spacecraft, your vessel "warps" through space, getting closer and closer to Earth. On your way there, you pass the outer planets: Neptune, Uranus, Saturn, Jupiter and Mars. Each time you reach a planet, you play a "chance stage."

In these stages, you get a chance to destroy enemy ships for bonus points—without them shooting back!



You can rack up over 10,000 points during the chance stage, so shooting skill is a definite plus here.

You are awarded a bonus ship at 60,000 points, and every 100,000 points after that. Apparently, there is no limit to the number of ships you can accumulate.

The final judgment.

If you like Atari's arcade hit, **Tempest**, I think you'll enjoy **Gyruss**. I'm not afraid to admit that I like shoot-'em-ups, and this one's a lot of fun and very challenging.

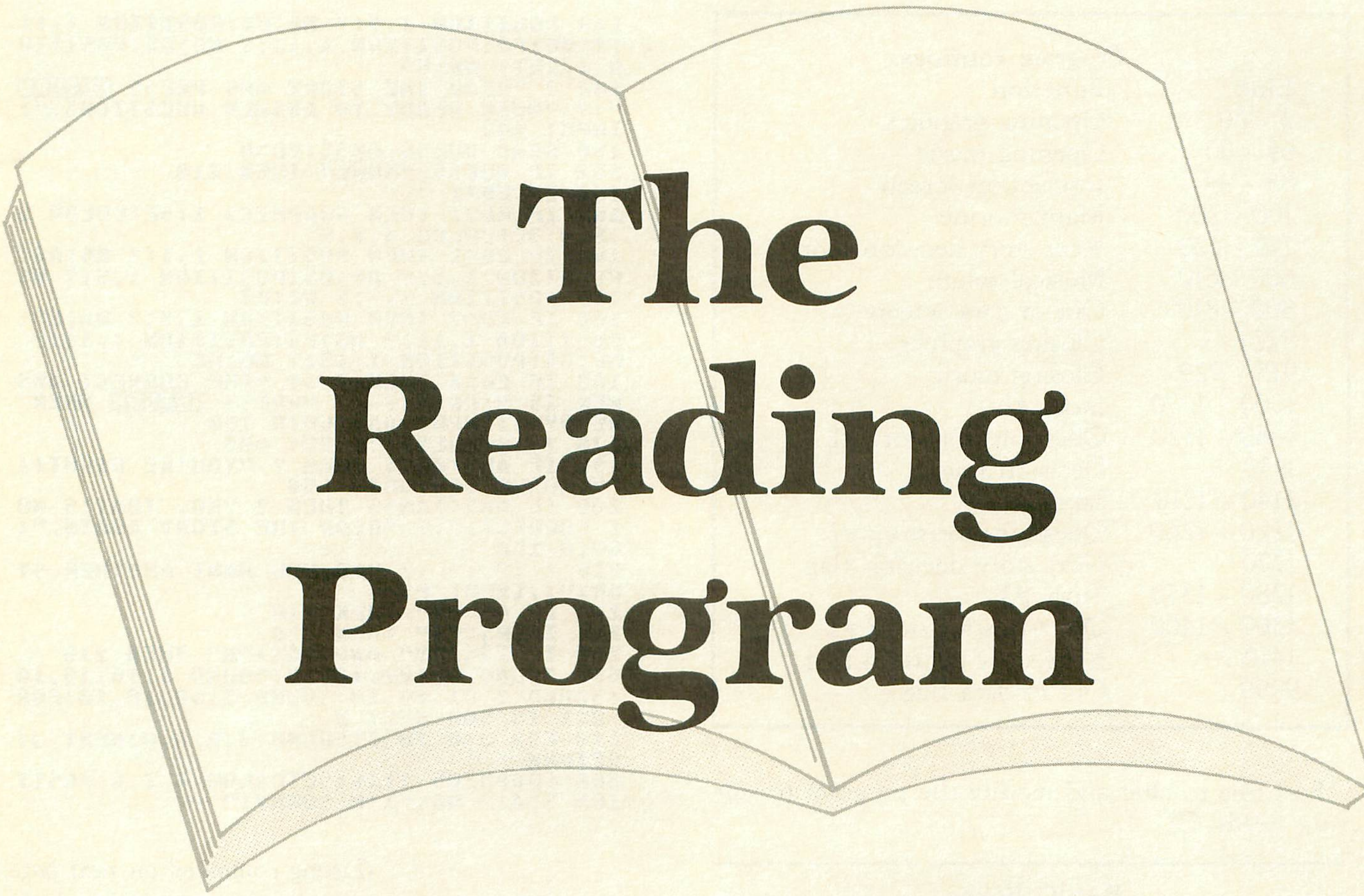
Gyruss.

Lately, **Gyruss** has become a lunchtime staple in the **ANALOG** programming office. Kyle Peacock, Charles Bachand and I have been competing for high scores (Charlie even bought a rapid-fire joystick adapter to give him a speed-shooting edge). So far, we've all reached Mars, but nobody has made it to Earth. The action really speeds up after Mars, and we've all been wiped out by kamikaze aliens!

We've been playing **Gyruss** so much that I think the rest of the **ANALOG** staff is going to go crazy listening to Bach's *Toccata and Fugue in D Minor*, the music played during the game. Well, they'd better get used to it. I have the feeling they're going to be hearing it for some time to come. □

CIRCLE #135 ON READER SERVICE CARD

The Reading Program



16K Cassette or 24K Disk

by Ed Rybczyk

Reading comprehension can be improved through drill and practice, especially for primary grade school children. The computer, when matched with appropriate software, can be used for this purpose. Reinforcement to correct responses can be immediate, with sound added to include another sensory reinforcer.

One of the problems with programs on the market is their inability to alter the program, once the initial routines are learned. The **Reading Program** is designed to be altered by simply changing the DATA statements.

Be creative.

We, as parents, often make up short stories for our children and then ask questions to check for understanding. Place those stories and questions in the DATA statements, and your children can enjoy them again and again.

After the opening screen and music, the instruction screen is shown. Next the short story screen comes up and remains in place until RETURN is pressed. This allows the child to take whatever time is necessary for reading.

The screen wipes clean, and the first question is shown in the text window. If the response is correct, the next question is shown. However, if the response is incorrect, the story is shown again, and the question repeated. After a third incorrect answer, the correct answer is shown. Correct responses are rewarded with a musical salute. There is no scoring routine in this program, but one could be added by placing a counter in the musical salute and adding a screen before the closing routine. The BREAK key is turned off during the main loop to prevent accidental interruption.

A bit of caution is necessary about placing your short story in the DATA statements. Exactly eight DATA lines are needed for correct program operation, and no single DATA line can be more than nineteen characters long. If less than eight DATA lines are necessary to tell the story, fill the remainder of the eight with blank DATA statements. Immediately after the eighth DATA statement, place your question, followed by a comma and then the correct response. You can ask as many or few questions as you desire. The questions and answers should not be more than 35 characters. However, the last question must be followed by DATA NONE,NONE. This statement is used as a flag for the next story decision loop. Leave the Line 9999 DATA statement as is. This statement is the flag for the OUT OF DATA routine. This should be the last line in your program.

Program routines.

Line	Function
2 - 60	Opening graphics
65 - 80	Opening music
85 - 99	Instruction screen
100 - 200	Main routine
210 - 220	Next story decision loop
600 - 610	Musical salute
800 - 810	Out of data screen
820	Closing graphics
915 - 929	Closing music
1000 - 1070	Story #1
1080 - 1120	Questions for story #1
1130	Decision flag
1140 - 1210	Story #2
1220 - 1260	Questions for story #2
1270	Next story decision flag
1280 - 1350	Story #3
1360 - 1400	Questions for story #3
1410	Next story decision flag
9999	Out of data flag

Feel free to alter and modify the program to suit your needs. □

BASIC listing.

```
1 REM THE READING PROGRAM
2 REM BY ED RYBCZYK
3 REM ANALOG COMPUTING
4 DIM P$(1)
5 FOR G=23 TO 18 STEP -1
10 GRAPHICS G
20 POSITION 6,3:? #6;"welcome"
25 FOR L=1 TO 20:NEXT L
30 POSITION 8,5:? #6;"to"
35 FOR L=1 TO 20:NEXT L
40 POSITION 4,7:? #6;"THE READING"
45 FOR L=1 TO 20:NEXT L
50 POSITION 6,9:? #6;"PROGRAM"
55 FOR L=1 TO 20:NEXT L
60 NEXT G
65 FOR NOTE=1 TO 16:READ P,D:SOUND 0,P
,10,10:FOR W=1 TO D/1.5:NEXT W:NEXT NO
TE:REM P=PITCH,D=DURATION
70 DATA 121,240,81,480,91,60,96,60,108
,60,60,240,81,480,91,60,96,60
75 DATA 108,120,60,240,81,480,91,60,96
,60,91,60,108,720
80 SOUND 0,0,0,0
85 GRAPHICS 1:SETCOLOR 4,12,0:SETCOLOR
 2,12,0
86 POSITION 1,4:? #6;"YOU READ THE STO
RY.":POSITION 1,8:? #6;"i'll ask quest
ions."
90 POSITION 1,12:? #6;"YOU TYPE ONE WO
RD   ANSWERS."
95 ? "PRESS RETURN WHEN READY.":INPUT
P$
99 RESTORE 1000
100 DIM A$(20),B$(20),C$(20),D$(20),E$
(20),F$(20),G$(20),H$(20),QUES$(40),AN
S$(10),AN$(10),AA$(1)
110 GRAPHICS 1:SETCOLOR 2,8,0:SETCOLOR
 4,8,0:ED=0
112 POKE 16,112:POKE 53774,112
120 READ A$,B$,C$,D$,E$,F$,G$,H$
124 IF A$="END" THEN 800
130 POSITION 1,1:? #6;A$:POSITION 1,3:
? #6;B$:POSITION 1,5:? #6;C$:POSITION
1,7:? #6;D$
140 POSITION 1,9:? #6;E$:POSITION 1,11
:? #6;F$:POSITION 1,13:? #6;G$:POSITIO
N 1,15:? #6;H$
144 ? "READ THE STORY AND PRESS RETURN
":? "WHEN READY TO ANSWER QUESTIONS.":
INPUT AA$
150 READ QUES$,ANS$:ED=0
152 IF QUES$="NONE" THEN 210
160 ED=ED+1
162 IF ED=1 THEN GRAPHICS 1:SETCOLOR 2
,8,0:SETCOLOR 4,8,0
164 IF ED=2 THEN POSITION 1,1:? #6;A$:
POSITION 1,3:? #6;B$:POSITION 1,5:? #6
;C$:POSITION 1,7:? #6;D$
166 IF ED=2 THEN POSITION 1,9:? #6;E$:
POSITION 1,11:? #6;F$:POSITION 1,13:?
#6;G$:POSITION 1,15:? #6;H$
168 IF ED=4 THEN ? :? "THE CORRECT ANS
WER IS ";ANS$:? :? "PRESS RETURN WHEN
READY";:INPUT AA$:GOTO 150
170 ? :? QUES$;:INPUT AN$
190 IF AN$=ANS$ THEN ? "YOU'RE RIGHT!!
!":GOSUB 600:GOTO 150
200 IF AN$<>ANS$ THEN ? "NO, THAT'S NO
T CORRECT.":? "READ THE STORY AGAIN.":
GOTO 160
210 ? :? :? :? "DO YOU WANT ANOTHER ST
ORY";:INPUT P$
212 IF P$="Y" THEN 110
214 IF P$="N" THEN 820
220 IF P$<>"Y" AND P$<>"N" THEN 210
600 SOUND 0,121,10,10:SOUND 1,96,10,10
:SOUND 2,81,10,10:SOUND 3,60,10,10:FOR
 X=1 TO 500:NEXT X
610 FOR J=0 TO 3:SOUND J,0,0,0:NEXT J:
RETURN
800 GRAPHICS 1+16:SETCOLOR 4,3,0:POSIT
ION 5,4:? #6;"I'M SORRY!"
```

(Listing continued on next page.)

```
810 POSITION 4,8:? #6;"I DON'T KNOW":P
OSITION 2,12:? #6;"ANY MORE STORIES.":
FOR X=1 TO 1500:NEXT X
820 ? #6;"K":POSITION 5,16:? #6;"BYE B
YE !!"
915 RESTORE 927
916 READ DURATION:IF DURATION=-1 THEN
GRAPHICS 0:END
917 DURATION=INT(DURATION*30)
918 READ PITCH:IF PITCH=0 THEN 920
919 PITCH=PITCH*3
920 SOUND 0,PITCH,10,8
921 SOUND 1,PITCH+1,10,8
922 FOR W=1 TO DURATION:NEXT W
923 SOUND 0,0,0,0
924 SOUND 1,0,0,0
925 FOR W=1 TO 3:NEXT W
926 GOTO 916
927 DATA 2,47,2,60,1,81,3,60,2,47,2,60
,1,53,3,53,2,47,2,60,1,45,2,45,1,45
928 DATA 1,47,1,47,1,53,1,53,3,60,1,0,
1.5,47,.5,53,1,60,1,53,1,47,1,47,2,47,
1,53,1,53,2,53
929 DATA 1,47,1,40,2,40,1.5,47,.5,53,1
,60,1,53,1,47,1,47,2,47,1,53,1,53,1,47
,1,53,3,60,1,0,-1
1000 DATA BILL AND MARY WENT
1010 DATA TO THE STORE TO BUY
1020 DATA APPLES. THEIR DOG
1030 DATA KING WENT ALONG.
1040 DATA THEIR MOTHER WANTED
1050 DATA TO BAKE A PIE. BILL
1060 DATA LIKED APPLE PIE BUT
1070 DATA MARY DID NOT.
1080 DATA WHAT IS THE DOG'S NAME,KING
1090 DATA WHERE DID BILL AND MARY GO,S
TORE
1100 DATA WHAT DID BILL AND MARY BUY,A
PPLES
1110 DATA WHO DID NOT LIKE APPLE PIE,M
ARY
1120 DATA WHO WENT WITH BILL AND MARY,
KING
1130 DATA NONE,NONE
1140 DATA THE CIRCUS CAME TO
1150 DATA TOWN. DAD AND ADAM
1160 DATA WENT TO THE CIRCUS.
1170 DATA ADAM LIKED THE LION
1180 DATA BUT WAS AFRAID OF
1190 DATA THE BEARS. ADAM ATE
1200 DATA CANDY. THEY HAD FUN
1210 DATA THAT SUMMER DAY.
1220 DATA WHAT CAME TO TOWN,CIRCUS
1230 DATA WHO WENT WITH ADAM TO THE CI
RCUS,DAD
1240 DATA WHAT WAS ADAM AFRAID OF,BEAR
S
1250 DATA WHAT DID ADAM LIKE,LION
1260 DATA WHAT DID ADAM EAT,CANDY
1270 DATA NONE,NONE
1280 DATA KURT'S SOCCER TEAM
1290 DATA IS THE WILDCATS.
1300 DATA HIS POSITION IS
1310 DATA STRIKER. THE TEAM
1320 DATA BEAT THE LIONS 2-0.
1330 DATA BRETT SCORED BOTH
1340 DATA GOALS. THEY HAD A
1350 DATA PARTY AFTERWARDS.
1360 DATA WHAT SPORT DID KURT PLAY,SOC
CER
1370 DATA WHO SCORED THE GOALS,BRETT
1380 DATA WHAT POSITION DID KURT PLAY,
STRIKER
1390 DATA WHAT DID THEY HAVE AFTERWARD
S,PARTY
1400 DATA WHO DID THEY BEAT,LIONS
1410 DATA NONE,NONE
9999 DATA END,A,A,A,A,A,A,A
```

•

CHECKSUM DATA.

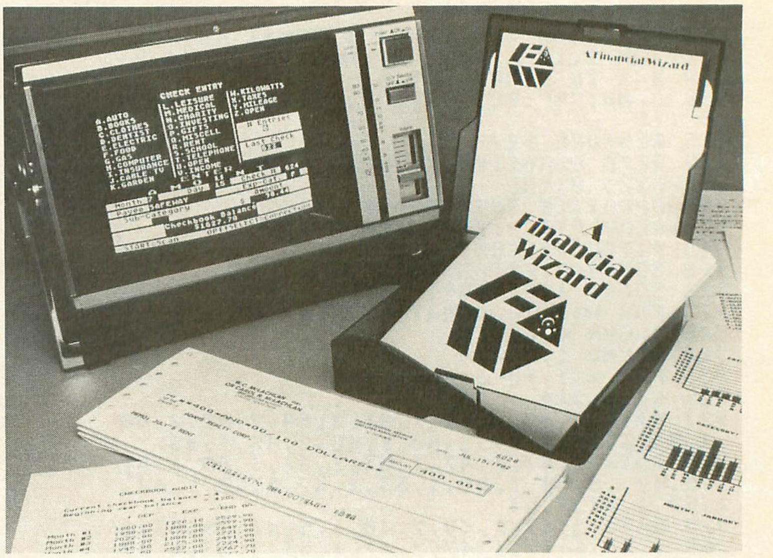
(see page 43)

```
1 DATA 874,89,465,361,972,198,585,966,
638,968,667,970,130,972,380,9235
65 DATA 83,119,29,281,995,670,63,331,1
39,76,597,719,464,41,707,5314
140 DATA 35,368,293,739,432,381,238,73
0,888,538,288,558,77,801,814,7180
220 DATA 197,380,931,83,408,298,222,21
9,2,746,404,148,161,242,94,4535
924 DATA 98,75,736,709,211,452,91,815,
762,453,947,688,843,88,207,7175
1090 DATA 616,163,592,719,861,950,894,
977,912,637,692,700,551,818,333,10415
1240 DATA 324,525,493,868,86,401,603,7
18,694,770,471,863,812,976,527,9131
1390 DATA 408,494,868,367,2137
```

•

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
- **Printouts**–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.

CIRCLE #134 ON READER SERVICE CARD

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. ________

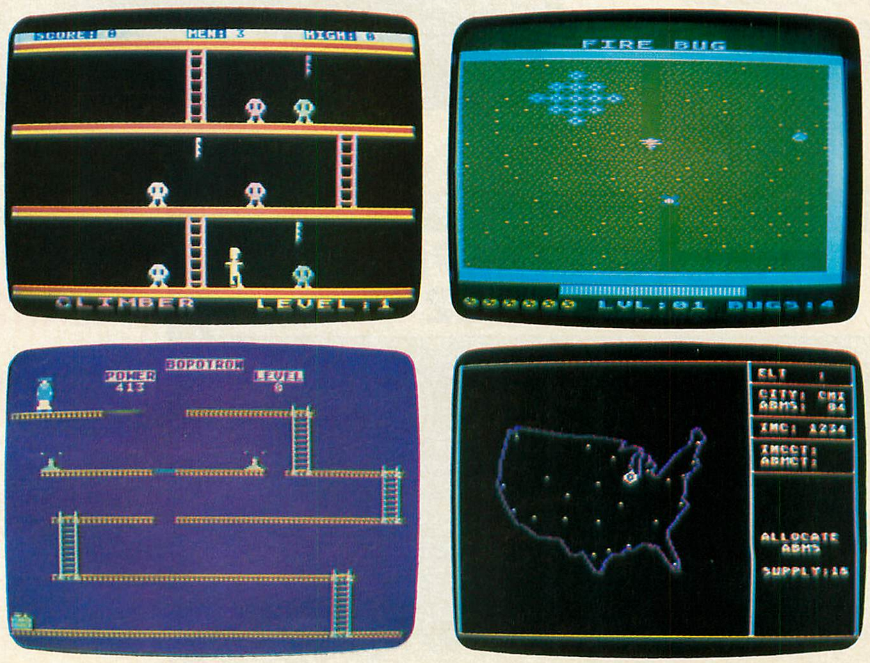
Signature ____________________

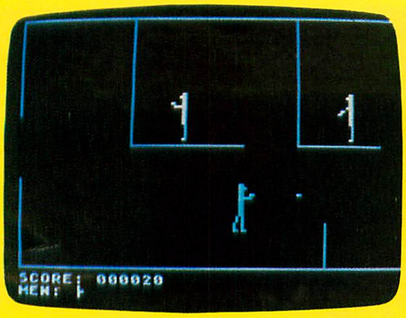
INDEX TO ADVERTISERS

READER SERVICE #	ADVERTISER	PAGE #
133	Advanced Interface	93
122	Allen Macroware	58
110	ALOG Computing	30
153	ANALOG Publishing	90, IBC
123	Astra Systems	58
101	Atari Inc.	IFC
103	Axion	8
117	C.A.P. Software	32
120	CDY Consulting	54
112	Computability	33
134	Computari	94
104	Computer Creations	12
136	Computer Games Plus	5
140	Computer Image	70
131	Computer Palace	88
109	Datasoft	24
108	Eastern House	20
121	Eclipse	56
116	Future Tech	46
125	Happy Computing	35
102	Infocom	2
129	Lateral Software	84
137	Lotsa Bytes	6
135	Lyco	90
132	Micca Enterprises	92
147	Mighty Byte/Spartan	64
106	Miles Computing	17
141	New Horizon Software	71
107	Non-Standard Magic	20
130	PC Gallery	87
105	Programmers Workshop	16
113	RAM Computer Center	38
114	Sar-An	40
117	Soft Sectre	51
118	Southern Software	51
119	Superware	6, 51
127	Tiny Tek	76
127	Walling Co.	76
128	Wedgwood Rental	78
115	Xlent Software	46

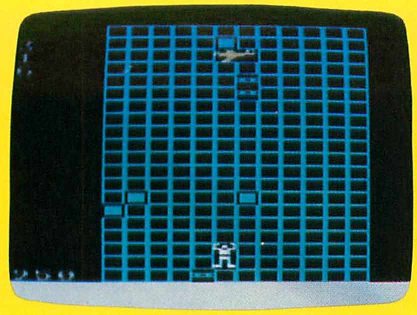
This index is an additional service. While every effort is made to provide a complete and accurate listing, the publisher cannot be responsible for inadvertent errors.

Coming soon in future issues of **ANALOG Computing!**



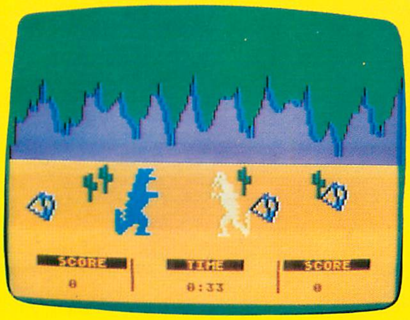


Maniac!

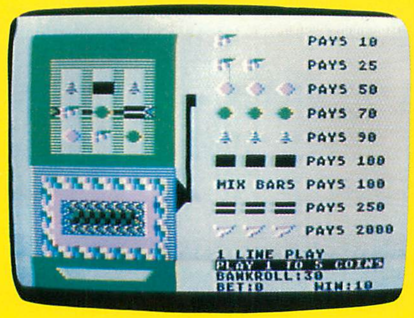


Stuntman

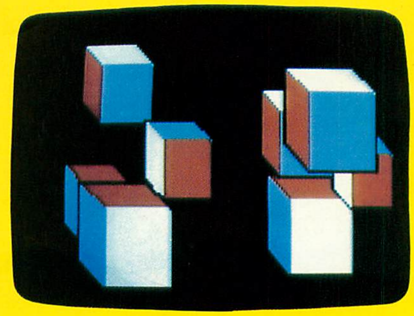
Fill 'Er Up



Dino Battle

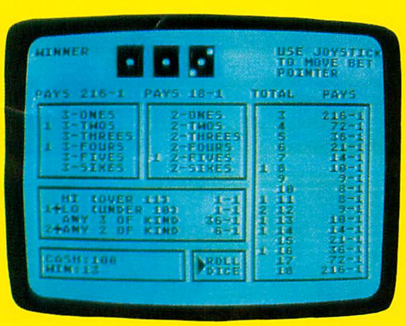


Color Slot Machine

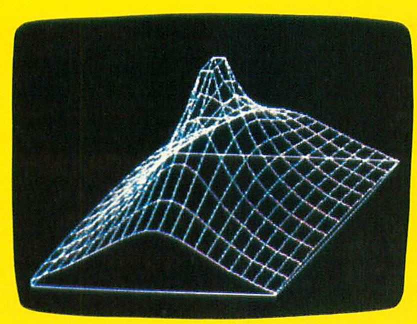


Cubes

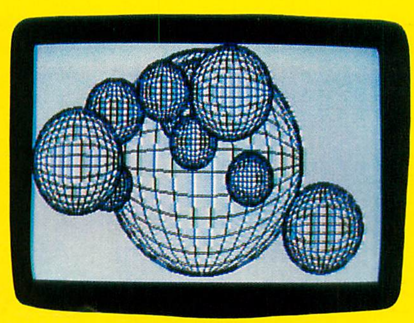
Where can you get all of these programs (and dozens more!) for only $14.95?



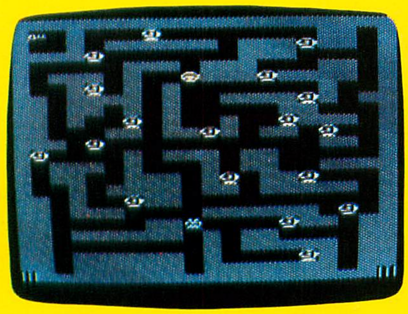
Triple Threat Dice



3-D Graphs

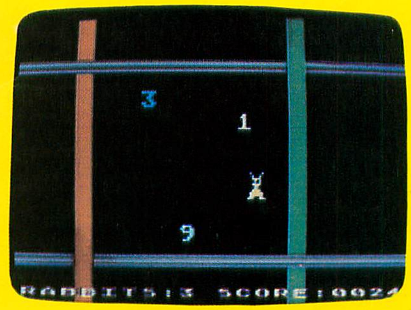


Sphere Demo



Leprechaun King

Sketch Pad



Harvey Wallbanger