SoftSide™

Now for the IBM®-PC, too!

#34

THREE DOLLARS

Microcomputers: THE SOUND OF THE FUTURE

"SOFTALK" REVIEWED HOME FINANCIAL PROGRAMS AND HERE'S WHAT THEY HAD TO SAY...

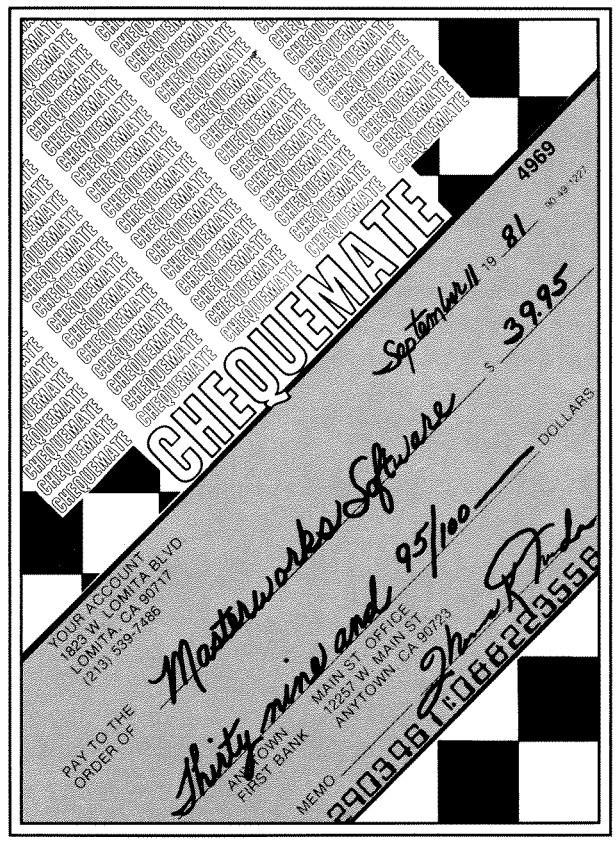
"At $39.95 ChequeMate™ is the Bargain of the Bunch."

Masterworks delivers more **value.** Our ChequeMate was reviewed in SOFTALK as the best bargain in home financial programs.

But as popular as ChequeMate is, Masterworks doesn't stop there. Our full line of home and business software programs offers you one of the most complete packages available. At the most attractive, "best selling" prices.

Here's a shopping list of our innovative programs:

ChequeMate
ChequeMate PLUS
(available for IBM PC & APPLE)
MasterDisk
MasterDOS
PAYMaster
ACCOUNTS Payable
ACCOUNTS Receivable



General LEDGER
ORDER Entry
FIXED Assets
BUSINESS Master

What's so great about Masterworks?

Plenty. Including the fact that with Masterworks Software you can get many of the features of large computer systems, previously unavailable for micros, for your APPLE II and IBM Personal Computer. This is made possible through Masterworks' unique programming system, exclusive to this complete software line.

You want value for your money — you want Masterworks software. For more information or the dealer nearest to you call (213) 539-7486.

Masterworks Software, Inc.

(213)539-7486.

25834 Narbonne Ave.
Lomita, California 90717

master charge THE INTERBANK CARD

VISA

OF ALL THE THINGS YOU BUY, HOW MANY ARE GOOD ENOUGH TO BE WARRANTED 5 YEARS?

Few disks stand the test of time. Because few are built to the precision standards or certified to the critical levels of Omni's complete line.

Each Omni disk is rated for 12 million passes without disk-related errors or significant wear. Each is certified error-free at a minimum of twice the error-threshold of your system. And built to exceed all industry specifications including those of ANSI, ECMA, ISO and virtually every drive manufacturer. So you can count on them for the long haul. We guarantee it.

Call toll-free (800 343-7620) for your nearest dealer. In Mass., call 617 799-0197.

Omni Resources, 4 Oak Pond Ave., Millbury, Mass. 01527

Dealers. Software houses.
Check our prices, services and specifications. We offer duplicating, formatting, private labeling, small minimums, fast delivery and copy protection schemes on disks for virtually any system.

OMNI

THE DISK GOOD ENOUGH TO BE WARRANTED 5 YEARS

SoftSide™

SoftSide Vol. 6, No. 1

SUBSCRIPTION INQUIRIES should be sent to *SoftSide* Publications, 100 Pine Street, Holmes, PA 19043. EDITORIAL AND ADVERTISING CORRESPONDENCE should be sent to *SoftSide* Publications, 6 South Street, Milford, NH 03055. Telephone (603) 673-0585.

SoftSide (ISSN 0274-8630) is published monthly by *SoftSide* Publications, Inc., 6 South Street, Milford, NH 03055. Printed at Lorell Press, Avon, MA. Second class postage paid at Milford, NH, Avon, MA, and at additional mailing offices. Subscription rates: US, and Canada, $30/12 issues. First Class US, APO, FPO, Mexico, $40/12 issues. Other foreign countries, $62/12 issues. Media subscription rates: US Magazine and Cassette, $75/12 issues. US Magazine and Disk $125/12 issues. APO, FPO, Canada and Mexico, add $20/12 issues. Other foreign add $50/12 issues.

Cover illustration by Alan Hashimoto

FEATURES

DEPARTMENTS

CONTENTS

PC/SIDE

TRS-80®/SIDE

ATARI®/SIDE

APPLE™/SIDE

ATTENTION AUTHORS

SoftSide Publications is actively seeking program, article and review submissions for the TRS-80®, IBM®-PC, Apple™ and ATARI® home computers.

- **Programs — SoftSide** has always been the leader in the field of BASIC software. BASIC remains our specialty. However, with the advent of Disk Version (DV), we can now also offer an outlet for Machine Language and multiple language programs which do not lend themselves to printed versions. Games, utilities and educational software, as well as any other applications for the home computer user are preferred, although we will consider virtually any type of program. Hybrid mixes of articles and programs are also welcomed.

Please be sure to include **full** documentation of subroutines and a list of variables, also a brief article describing the program.

- **Reviews** — Well written, informed reviews of all software for the systems we cover are a regular feature of **SoftSide.** Reviewers should take into consideration all aspects of a particular software package, from speed of execution to programming creativity to the estimated length of time that the product will hold the customer's interest.
- **Articles** — We welcome article submissions of all types, but prefer those specifically geared to the home computer market. We give our readers information as a first priority, but vary our content to include some humor and commentary.

All text, including documentation and descriptive articles for programs, should be typewritten and double-spaced. Extra monetary consideration will be given to articles and reviews submitted on disks (Scripsit, Super-Text II, etc.). Programs should be submitted on a good disk. TRS-80® BASIC programs should function under both Level II and Disk BASIC.

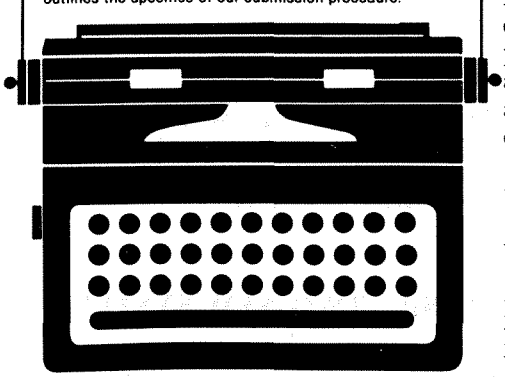
Please be sure to pack your disks carefully and to include your return address and phone number.

Send to:

SoftSide Publications
SUBMISSIONS DEPARTMENT
6 South Street
Milford, NH 03055

We regret that due to the volume we receive, we are unable to return submissions which do not include return postage.

Be sure to send for our **FREE AUTHOR'S GUIDE.** It further outlines the specifics of our submission procedure.



EDITORIAL

The New Conceptual Tool

by Randal L. Kottwitz

Editors of computer magazines get asked a lot of questions. One query which has been ringing my phone rather frequently of late is: "Johnny is going off to college next year and is not sure what he wants to study. Could you tell me how to convince him to go into computers?"

My answer to these bewildered parents is not what they expect, for I recommend that "Johnny" not go into computers unless that is *his* major area of interest. That answer does not come from a fear of "Johnny" not finding interest or a career in this field, but from a realization that the specialization of computerists is losing some of its validity. It would be foolish for us to consider computers to be an isolated field any more — for they have woven themselves into the very fabric of our everyday lives. "Johnny" doesn't need to specialize in computers to be prepared for the electronic world of the future. Instead, he should pursue *his* major interest (probably what he would have studied had computers never existed) — making sure that he has a thorough understanding of how to utilize computers within that field.

I find myself sounding like something of a fanatic as I talk to my friends in music, television, psychology, anthropology, education and other fields. Invariably, the conversation turns to a discussion of how they could utilize computers to make their job easier. In most cases, the tasks we find the computer could perform for them are some that they currently find the most tedious and inefficient. Then, we get into an area of application that asks questions with few answers. "What could the computer allow you to do that you haven't conceived of doing before?"

It is for the answer to this question that I look to the "Johnnys" of the future. We are on the verge of seeing the first generation enter the work force who have been offered the computer as a feasible tool for their approach to problem solving during all of their formal education. The mystique of the big blue boxes in the sealed, air conditioned laboratories is receding and the computer is moving into the common man's tool box. The computer will no longer be considered an "alternative solution" to problems, but a core *tool* with its own special bailiwick of applications. It's important to realize what a change this will effect in our cultural approach to problem solving. In effect, the computer is seeping from the right to left side of the problem/solution equation. It is now shaping the way we *conceive* of our problems, as well as providing solutions. The impact of that change in our conceptual process will not show itself fully for some time, but the ramifications could easily be compared to the impact electrical wiring and lighting have had on our culture. For a quick quantification of that impact, consider what we might now have instead of television (a giant of cultural impact in its own right) had its inventor not been able to consider the common availability of electricity in his conceptual process.

As with any powerful tool, improperly utilized, the computer can and probably will have some negative effects on our society. However, the freedom this new conceptual tool will bring to our lives is well worth the cost of its negative ramifications. Our natural tendency as a culture has been to pick the first solution to our problem which comes to mind and run with it. The disadvantage of this approach has been the risk of either running off a cliff or into a worse problem than we started with. With the computer in our conceptual tool box, the future offers us a strategy in problem solving which will more easily allow us to select the most attractive path from many ideas and concepts. I'd hate to try to place a price tag on our new conceptual tool if it were to be measured against its future contribution to our lives.

Randal L. Kottwitz

Editor-in-Chief

DATA PERFECT

FOR THE ATARI 400 AND 800 COMPUTERS

YOU MAKE THE COMPARISON

	D.P.	FILE MANAGER "800"
GENERAL INFORMATION		
Cost of Program	$99.95	
Cost of Utilities Program *(Included In Program)*	$00.00	
Cost of Reports Program *(Included In Program)*	$00.00	
Compatible With Letter Perfect (tm) Word Processing	**YES**	
Menu Driven *(Very User Friendly)*	YES	
Complete Documentation *(Manual Tabbed And Indexed)*	YES	
Single Load Program *(No Swapping Of Program Diskette)*	YES	
Machine Language *(Extremely Fast Operation)*	YES	
Can Use Single Disk Drive	YES	
Can Us Multiple Disk Drives	YES	
Ability To Design Screen Mask *(User Designs Arrangement Of Data)*	YES	
Full Keyboard Editing Available *(Delete/Insert A Character; Go To End/Beg. of Line; Fine 'n', TAB, ETC.)*	YES	
Compatible With Bit 3 80-Column Board *(40-Column and 80-Column Version Available)*	YES	
Works With Any Parallel Printer *(Supports Atari 850 Interface)*	YES	
Totals Of Numeric Field *(Return Total And Average Value/Field)*	YES	
Fail Safes Provided For Data Protection	YES	
Error Messages Displayed	YES	
Status Lines For Ease of Use *(Options Always Available For Reference)*	YES	
SEARCHES AND EDITING		
Multiple Searches Allowed On Same Record *(Search On 9 Criteria Per Record)*	YES	
Search On Two Criteria In Same Field *(Up To 4 Fields In Single Record)*	YES	
Wild Card Searches *(And/Or, Include, Character, Or Block)*	YES	
Search On Basis Of Record Number *(Search For An Individual Record)*	YES	
Search On Range Of Data Desired *(Dates, Numbers, Values, Greater Or Less Than, Equal To, etc.)*	YES	
Editing Of Records Individually	YES	
Editing Records Globally *(Verification Allowed)*	YES	
Delete Records Individually *(Verification Allowed)*	YES	
Deleting Records Globally *(Verification Allowed)*	YES	
UTILITIES SECTION		
Add Fields To Existing Data Base	YES	
Delete Fields From Existing Data Base	YES	
Reformat A Data Base *(Copy Format Of Existing Data Base)*	YES	
Make Additional Copies Of Data Base *(Create Data Base For Extended Records)*	YES	
Sort on Multiple Criteria *(Sort On Basis Of 4 Fields In A Sort)*	YES	
Sorts On Multiple Criteria *(Assending Or Descending)*	YES	
Depth Of Sort Can Be Changed *(Designate Number Of Charters Deep To Sort)*	YES	
Merge Information From Other Data Bases *(Merge Standard Text Files)*	YES	
Add Or Delete Fields From Data Base	YES	
Merge Previous Entered Data From Existing File	YES	
Back Up A Data Base *(Make A Back Up Of Current Source Data)*	YES	
Pack A Data Base *(Remove Deleted Records From Disk Storage)*	YES	

	D.P.	FILE MANAGER "800"
REPORT GENERATOR		
Design Report To User Specifications	YES	
Level Breaks Allowed At Users Option *(Up To 4 Level Breaks Per Report)*	YES	
Designate Font To Be Used In Report	YES	
Boldfacing Allowed In A Report *(With Dot Matrix Printer)*	YES	
Mathematical Formulas Allowed In Report *(Example, Field 'x' + Field 'y' = Field 'z')*	YES	
Auto Page Number Allowed In Report	YES	
Auto Date Entering Allowed In Report	YES	
Repeating Characters Allowed	YES	
Optional Level Breaks and Page Breaks When Sort Values Change	YES	
Up To 7 Lines Allowed For Header on Each Report	YES	
Up To 2 Lines Allowed For Detail Information On A Report	YES	
Variable Spacing Allowed Between Data On Items In A Report	YES	
Multiple Fields Allowed In A Report *(Number, Date, Alpha, Formula)*	YES	
Search Criterian Allowed On Report *(Same Criteria As In Editor)*	YES	
Ability To Have "Literal" Data *Printed In A Report*	YES	
Ability To Have "Conditional" Data Printed In A Report	YES	
Use A Default Date Field	YES	
Designate Default Value For Specific Fields	YES	
LABELS REPORT GENERATOR		
Mailing Labels Allowed *(Specifically Designed For Labels)*	YES	
User Designs Data Placement On Label *(One Across Label Design)*	YES	
Multiple Fields Allowed On Label *(Date, Alpha, Numeric, Formula)*	YES	
Repeating Characters Allowed	YES	
Front Designation Allowed	YES	
Print Labels On A Conditional Basis	YES	
Search Criteria Valid On Label *(Same Search Criteria As Editing)*	YES	
MATHEMATICAL ABILITIES		
Basic Math Calculation Addition, Substraction, Multiplication, Division	YES	
Built In Calculator (Automatic) *(Use In Editing, Or Adding Data)*	YES	
Find the Integer Value Of A Numeric Expression	YES	
Find The Log Base 'e' Of 'x'	YES	
Find The Log Base '10' Of 'x'	YES	
Find The Absolute Value Of 'n'	YES	
Exponentential Notation Used	YES	
Find The Square Root Of 'n'	YES	
Formulas Allowed Between Fields [Field × /+ − */) Field y = Field z] [Field × /+ − */) N = Field Y]	YES	
SPECIFICS		
Maximum Number Of Fields Per Record	32	
Maximum Number Of Formulas In A File	16	
Maximum Length Of A Field	127	
Maximum Record Length	511	
Maximum Number Of Level Breaks	4	
Records Per Diskette *(Depends On Length And Number Of Fields)*	VAR.	
Data Bases Allowed On Each Diskette *(Can Be Expanded To Additional Diskettes)*	ONE	
Form Letter Capability *(Compatible With Letter Perfect)*	YES	

LJK ENTERPRIES, INC.
P.O. BOX 10827
ST. LOUIS, MO. 63129
[314] 846-6124

INPUT/OUTPUT

From our readers

INPUT

A SAVING IDEA

Dear *SoftSide*,

As a subscriber, I wish to express my satisfaction with *SoftSide*.

I am a Professor of Clinical Chemistry and head of a hospital biochemistry laboratory. I have been programming computers for several years, but a year ago I bought my own TRS-80® Model III with 48K, two drives and a printer. Despite some shortcomings, I found it was a wonderful instrument. I use it primarily for statistical work. Recently, I bought programs which allow me to perform principal component analysis and discriminant analysis on up to 250 patients and 50 parameters. Incredible!

I would like to submit a small idea of mine, which I find very helpful. I now begin all my programs in the following fashion:

```
1 GOTO 10
2 SAVE "programname",A:STOP
10 actual beginning of program
```

When I have typed in a few lines, or modified something, all I have to do is type "RUN2" and my program is saved. If I type "RUN", it works normally. I've found that when you type in a long program, it's better to SAVE it often, as you never know when the power will fail.

Jacques D. Weill, M.D., Ph.D.
Veigne, FRANCE

COPY PROTECTION

Dear *SoftSide*,

I found the letter from Thorne Harris, President of Superior Software, (*Input*, Issue 31), heartwarming. I am glad to see that someone understands the user's problems with uncopyable software. Also, I feel that the locking of software does not stop that much piracy. One only has to look at the number of Disk Dupers out there, (I know of at least four) to see that locking software is a losing battle which discourages people from buying software and wastes production time and money on designing new protection schemes. Hopefully, others feel as I do, and will try to convey the message to the software companies that locking software is hurting their business, not helping.

I would very much appreciate an address for Superior Software.

Lee A. Kendter, Jr.
Philadelphia, PA

Editor's Reply: More and more manufacturers seem to be jumping on the bandwagon of unprotected software. Indeed, we may be seeing the advent of a whole new way of thinking in the way software is distributed. We know of one manufacturer who sends his software out for the cost of the raw media and shipping. When booted, a message appears on the screen asking the user to send the suggested price for the software after he has used it and justified its value. Now that's an honor system! The address for Superior Software:

Superior Software, Inc.
4312 Arizona Ave.
Kenner, LA 70062

PLEA FOR BABY BLUE

Dear *SoftSide*,

I have seen *SoftSide* at a neighbor's home.

I have recently bought an IBM® PC and am wondering whether you are planning to start putting programs for this machine in *SoftSide*.

Dickie Gall
Richardson, TX

Editor's Reply: Your wish is our command — well, not quite, but you will find software for the IBM PC in this issue of *SoftSide*. We're very happy to welcome PC/Side to our pages.

ATARI® MICROSOFT

Dear *SoftSide*,

I have Atari Microsoft BASIC and am hungering for programs to type, which is why I love your magazine. I have a suggestion, though. Double the programs in your magazine, and triple your price. I know I, and others, would pay anything for *SoftSide*.

Could you, in forthcoming issues, actually describe how to translate other Microsofts (Apple™, Pet™, etc.) to Atari. Your *Solitaire* (May, 1982) was superb.

I have seen a book called *SoftSide TRS-80* in a local book store. Will there be one for the Atari?

Thanks, you're great.

Carmen Verzillo
Fairport, NY

Editor's Reply: As we become more familiar with Microsoft BASIC on the ATARI, we will be passing on our translation knowledge to you. Watch *Hints & Enhancements* for valuable information on misprints in the manual and differences from other Microsofts. We also plan to cover the differences between Atari BASIC and Microsoft BASIC in an upcoming installment of *Exploring the Atari Frontier*. As for a book about *SoftSide* on the Atari, versions of *The Best of SoftSide* are currently under production for the Atari, Apple and TRS-80.

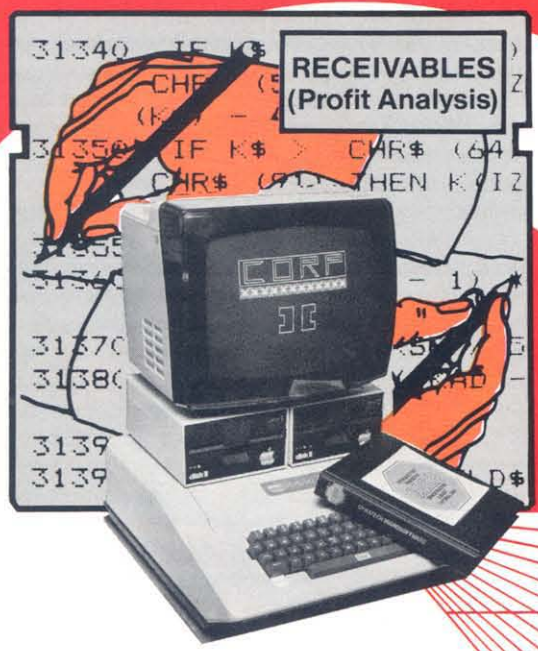
SOFTSIDE — A GOLD MINE?

Last month, I encountered your magazine for the first time. I have a TRS-80 Model III. One of the great pleasures of playing with micros is putting in someone's program and then playing around with it, all the while increasing the slope on the learning curve.

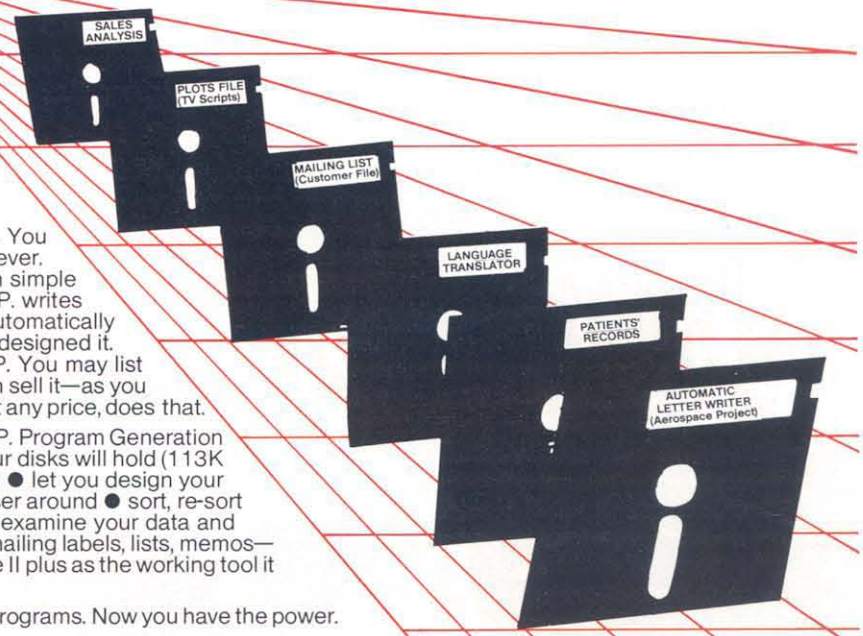
continued on page 9


When all hell breaks loose.
Pandora's box is open. And all the evils of the past are loosed upon mankind. Armed with bolts of lightning, you have the chance to recapture and return these corrupt creatures of doom to the prison of Pandora's box. Time is short. The world is already changing for the worst. Your skills and courage are needed.
PANDORA'S BOX
Pandora's Box—a new generation type game with overtones of Hi-res adventure graphics.
Only $29.95 for the Apple II*
At your computer store, or:
DATAMOST INC.
9748 Cozycroft Ave., Chatsworth, Ca 91311 (213) 709-1202
VISA/MASTERCARD accepted. $2.00 shipping/handling charge. (California residents add 6½% sales tax.)
*Apple II is a trademark of Apple Computers, Inc.

your power is ready...
software that writes software
for Apple II™



NEEDED: Apple II Plus with Auto Start, 2 drives, Applesoft in Rom, 48K.



C.O.R.P.™ writes software. Perhaps you should read that again. Not "simplifies programming" or "makes debugging easier." C.O.R.P. writes APPLESOFT BASIC—complete, stand-alone programs which run, bug-free, the very first time. You need not type a single character of basic code—ever. C.O.R.P. does that. Your answers to questions in simple English "design" the program. In minutes, C.O.R.P. writes all the program code, scrolls it to the screen and automatically saves it to your disk. Your program because you designed it. Once written, your program runs without C.O.R.P. You may list your program, examine it, modify it further or even sell it—as you wish, royalty free. No mere "data base manager," at any price, does that.

The applications are almost limitless. Your C.O.R.P. Program Generation System will: ● handle nearly as much data as your disks will hold (113K per disk!) ● find any record in less than a second ● let you design your own screens as quickly as you can move the curser around ● sort, re-sort and update your data for you ● examine and re-examine your data and then print reports, checks, invoices, statements, mailing labels, lists, memos—you name it. C.O.R.P. allows you to use your Apple II plus as the working tool it really is. Isn't that why you bought one?

Computers should write code. You sould design programs. Now you have the power.

Complete C.O.R.P. System includes: Master Disk ● Utility 1 ● Utility 2 ..... $250

Diagnostic Disc $30

User Demo Disk $20

Manual (Deductible from later order of complete system)

CORTECHS CORPORATION

7847 N. Caldwell, Chicago, Illinois 60648

See your dealer today or call toll-free 1-800-621-4109

(In Illinois, call (312) 470-0700

Dealer inquiries welcome.

C.O.R.P.™

C.O.D. welcome.

C.O.R.P. is a trademark of Maromaty Scotto Software Corporation.

Finding *SoftSide* was like finding a little gold mine. It contains the programs I enjoy, with plenty of room to innovate. Probably the most amazing surprise was that your programs are error free. Surely, this must be because *SWAT* is working for you internally.

Doug Linn
Westfield, NJ

Editor's Reply: Thanks for the roses. We're proud of the concentration we place on the quality of the software we publish. *SWAT* has, indeed, helped maintain that quality, both for us internally, and for the readers who laboriously type in our programs.

OUTPUT

by Randal L. Kottwitz

Bfrtz, zwatt, twing, deedle-deedle-dum! It's been noisy around *SoftSide* this month as we've prepared our annual issue on microcomputer sound. The auditory experience has been exhilarating as the new IBM-PC® and Apple™ have been singing a Bach Fugue, and Ataris® have been warbling Handel. Even the TRS-80s® have been putting in their two cents worth. There's no doubt that the microcomputer is bringing a whole new group of artists into the musical family.

PC/Side debuts with this issue and we proudly welcome the IBM PC to the *SoftSide* family. We've been diligently investigating all aspects of this wondrous machine and have even had a few fights over keyboard time. In many ways, we are taken back to the days of 1978 when our staff swarmed about the first TRS-80s® to come through our doors. We've found some very interesting aspects of the PC and will pass the results of our exploration on to you as fast as we can get them prepared.

As with all of the other computers we cover, we've had to make some decisions concerning the minimum system configuration we will be supporting for the IBM. The basic system requirements *SoftSide* will be adhering to for the PC will be 64K RAM, Color/Graphics adapter and Advanced BASIC. This does not mean that we will not occasionally publish programs for smaller or larger systems (*Squish,* elsewhere in this issue, is a good example of a program for a smaller system.), but we will try to stay within these parameters as much as possible. We will be checking all of our programs on a monochrome display and, when possible, adapting them for use on systems without the Color/Graphics adapter. However, as a computer entertainment magazine, the proper utilization of color and graphics is vital to our coverage of the PC. The products we review for the PC will, in most cases, fall into these same system requirements. Please write or call our editorial offices if you're interested in reviewing products for the *SoftSide* readers or are preparing programs on the PC. We're currently preparing a high percentage of this material within our walls and will be happy to accept any ideas or help you may have to offer.

Those of you with sharp eyes may have noticed a slight change in our *Attention Authors* notice in issue 33. As of this time we will no longer be accepting program submissions on tape. The reasons for this change are many, but the main motivation is the problems inherent in tape technology. Many of the tapes we receive will not load and the time required for processing a submission requiring three or four loading attempts has become prohibitive as our volume of submissions has increased. We realize that many of you are creating high quality programs on cassette based systems, but ask that you take the time to get to a disk system and down-load your files before you send them to us.

The computer trade show season is well under way and again, *SoftSide* will be attending most of the major expositions. Please come by and see us. The face to face contact that can be garnered at these shows is invaluable to all of us. We hope to see you there. Until then, Happy Hacking!

HINTS & ENHANCEMENTS

From our readers

IBM® PC HINTS

Our initial work with the IBM PC has uncovered some surprises and revealed a few points that need clarification.

1) When you calculate the number of bytes required for a graphics GET, you must remember to count all the dots in both the x and y directions. If you determine x and y by taking the difference between the starting and ending locations, you will count one dot too few, and the formula will not allocate enough memory. This is not clearly stated in the manual.

2) Clear the screen after editing a BASIC line. The screen editor tends to leave old, invisible end-of-line markers lying about and these can sneak into new text entered on the screen. The results can be maddening.

3) The manual does not state that user-defined string functions will accept only string arguments. This is consistent with normal Microsoft BASIC usage, but differs from TRS-80 Disk BASIC usage. Since IBM PC BASIC is very similar to TRS-80 Disk BASIC in other ways, this can mislead the unwary programmer.

TRS-80® INPUT MASK ROUTINE

Here is a user-defined function which takes a mixed-case string, extracts the left-most character, and masks it to upper case.

```
DEF FN I$(I$)=CHR$(ASC(I$+CHR$(13))
+32*(LEFT$(I$,1)>"Z"))
```

This simplifies screening input responses, since your program will not have to check for every possible variation.

Alan J. Zett
Merrimack, NH

EPSON MX-80 HINT FOR TRS-80® *GOTHIC LETTER PRINTER*

Elmer Wilhelm of Joliet, IL suggests the following addition to *Gothic Letter Printer.*

```
10 LPRINT CHR$(15)CHR$(27)"1"
```

CHR$(15) sets the Epson in compressed mode. CHR$(27)"1" sets line spacing to 8 lines per inch. This makes a tighter looking letter.

ATARI® MICROSOFT BASIC HINTS

As with the IBM PC, we have found some things about Atari Microsoft BASIC which need an explanation. Here are two of the more important items.

1) One of the standard features of Microsoft Basic not found in the Atari version is MID$ on the left side of an equation. Here is a programming hint to get around this deficiency. Simply add the following user-defined function to the beginning of your program.

```
DEF FN MID$(O$,B,R$)=LEFT$(O$,B-1)
+R$+MID$(O$,B+LEN(R$))
```

O$ represents the original string which you are modifying.
B is the starting position in O$ to modify.
R$ is the replacement string.

For example: If O$="0123456789", B=4, and R$="XXX", the normal Microsoft Basic command would be:

```
MID$(O$,B)=R$
```

In our defined function you would use:

```
O$=FN MID$(O$,B,R$)
```

The result after either statement is executed would be:

O$="012XXX6789", B=4, R$="XXX"

2) There is an error in the latest version of the Atari Microsoft BASIC manual. The INPUT AT function will not function properly as shown. The correct syntax is as follows:

Format: INPUT/#iocb/ /"prompt____string"/,/AT(s,b)/;variable____name,/variable____name/
INPUT#6 /"prompt____string"/, /AT(x,y) variable____name
Examples: 120 INPUT "TYPE YOUR NAME";A$
350 INPUT "ACCOUNT NO., NAME";NUM,B$
300 INPUT#5, AT(9,7) X

There is a required SPACE following the AT(x,y) option, not the semi-colon shown in the manual. The example listed in the second paragraph of the explanation of INPUT should also show a space instead of a semi-colon.

Alan J. Zett
Merrimack, NH

APPLE™ DISK PEEKER/POKER ENHANCEMENT

The March 1982 Apple DV Bonus, *Disk PEEKER/POKER* by Mike Westerfield, is definitely an asset and a welcome utility program. My one gripe is the lack of internal instructions. I don't need the program often enough to memorize the commands and it's annoying to have to locate paper instructions when I do. Here are some enhancements to correct this.

```
500  VTAB 1: PRINT "    TRACK "; PEEK
     (TK);"    SECTOR "; PEEK (S
     K): VTAB 21: CALL  - 958: PRINT
     " (Q)UIT, (R)EAD, (W)RITE, (
     E)DIT": PRINT : PRINT "OPTIO
     N?";: GET A$
540  IF A$ = "Q" THEN  TEXT : HOME :
     END
550  IF A$ = "R" THEN 1000
560  IF A$ = "W" THEN 1500
570  IF A$ = "E" THEN 2000


2000  VTAB 21: HTAB 1: CALL  - 95
     8: PRINT " (Q)UIT, I=UP, M=D
     OWN, J=RIGHT, L=LEFT"
2010  POKE 34,0: VTAB 3: HTAB 5


3000  PRINT : PRINT " '/' = SAME,
      '-' = -1, '+' OR ';' = +1":
      PRINT : PRINT "CURRENT TRAC
     K,SECTOR: "; PEEK (TK);","; PEEK
     (SK)
3040  INPUT "        TRACK,SECTOR
     : ";I$,J$: IF I$ = ";" THEN
     I$ = "+"
```

continued on page 12

SoftSide DV, the magazine of the future, is here!

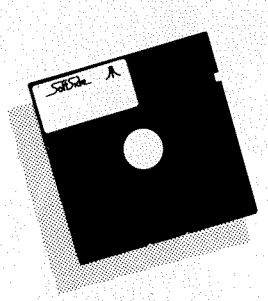
If your computer could pick a magazine, wouldn't it prefer one in its own language? Now there's one available. **SoftSide DV** is an enhancement of the **SoftSide** you have in your hands.

SoftSide DV contains not only the complete programs listed in every month's issue of **SoftSide,** but additional programs of every conceivable type, as well — multiple and Machine Language programs, modified languages, ongoing modular programs and software so extensive, it would take an entire issue of **SoftSide** just to print the code. Only the documentation for these programs will appear in **SoftSide** Magazine, **NOT** the code.

Feel as though you're missing something? You are! But, you needn't miss out on another issue. **SoftSide DV** is now available for Apple™, ATARI® and the TRS-80® . The cost to you — $125 for 12 magazines and 12 disks, packed with some of the best software available, all delivered to your home in the next year. For orders outside the USA, please add $36. For your convenience, we offer an installment payment plan for VISA and MasterCard holders: You pay only $32.50 per month for four months (a total of $130, which includes a $5 billing charge). Please use the special DV-CV bind-in card in this issue to order.

Computerists are offered the rare opportunity of marching into a new frontier. Advance to the front of the parade by subscribing to **SoftSide DV**, the magazine of the future, available today!

Translation of the Month Contest



You could win a FREE *SoftSide* DV or CV Subscription!

SoftSide's* Translation of the Month** has been so well received by our readers, we're offering a greater author incentive than ever before. No, we can't give you a job at the U.N., but we will award a one-year subscription to ***SoftSide DV or an 18-month subscription to ***SoftSide CV*** for a high-quality translation of one of our past programs. That's a value of $125 for the ***Disk Version*** or $112.50 for the ***Cassette Version*** — you'll be rewarded every month for your translation efforts!

Here are some of the most important qualifications we look for in a translation winner.

Your entry must be a translation of one of the featured programs from a past issue of ***SoftSide***. (We're particularly interested in Apple™ and ATARI® translations of some of our older TRS-80® only issues. Write for a list of suggested candidates.) In general, we're looking for translations of programs which are a CHALLENGE to translate. Some of the programs we publish are written in more or less "generic" BASIC, which can be typed into another computer with very few changes. Although these programs require the least effort to translate, they are also the least likely candidates for contest winners.

Your translation should be thoroughly tested and completely bug-free. Just converting program lines doesn't automatically ensure a workable translation. Be sure to use-test your translation as carefully as you would test a program you had written entirely from scratch.

Your translation should fully utilize the unique features of the computer for which it is written. The objective of a translation is to "fit" the capability and convention of its host computer, not simply mechanically duplicate the operation of the original program. This is especially true of programs which use graphics, and should be kept in mind for such minor features as keyboard layout (use of such special keys as arrows, ESC, CTRL, CLEAR, etc.). Also be careful with screen formatting; a word that spills over into the next line because of a PRINT statement that wasn't properly rewritten betrays such carelessness that we'll probably reject your translation automatically.

Your entry should incorporate any improvements and enhancements you can add to the original program. Don't feel that you have to limit yourself to the boundaries of the original. (On the other hand, don't go overboard and destroy the character of the original by completely rewriting it!) An enhanced translation is much more likely to catch our attention than a line-for-line duplicate, and it will have more value to our readers.

It's not necessary to include extensive documentation with your translation, only that which is different from the original. If most of the originally published documentation applies to your translation, simply say so. You should, however, include descriptions and explanations of any changes or enhancements you've made.

All **Translation Contest** entries must be submitted on disk, with documentation in printed or typed form. Disks will be returned only if accompanied by a self-addressed, stamped envelope. Send your entries to:

SoftSide

Translation Contest
6 South Street, Milford, NH 03055

Hints ***continued***

```
3041  IF  ASC (I$) < 43 OR  ASC (
    I$) = 44 OR  ASC (I$) = 46 OR
     ASC (I$) > 57 THEN 3000
3042  IF J$ = ";" THEN J$ = "+"
3043  IF  ASC (J$) < 43 OR  ASC (
    J$) = 44 OR  ASC (J$) = 46 OR
     ASC (J$) > 57 THEN 3000
3044 I =  VAL (I$):J =  VAL (J$):
     IF I$ = "/" THEN I =  PEEK
    (TK)
3045  IF I$ = "+" THEN I =  PEEK
    (TK) + 1
3046  IF I$ = "-" THEN I =  PEEK
    (TK) - 1
3048  IF J$ = "/" THEN J =  PEEK
    (SK)
3050  IF J$ = "+" THEN J =  PEEK
    (SK) + 1
3052  IF J$ = "-" THEN J =  PEEK
    (SK) - 1
3054  IF I <  - 1 OR J <  - 1 OR
    I > 35 OR J > 16 THEN  CALL
     - 198: HOME : GOTO 3000
3056  IF J =  - 1 THEN J = 15:I =
    I - 1
3058  IF I =  - 1 THEN I = 34:J =
    15
3060  IF J = 16 THEN J = 0:I = I +
    1
3062  IF I = 35 THEN I = 0:J = 0
3070  POKE TK,I: POKE SK,J: RETURN
```

Line 500 is a simple way of adding command prompts. By changing "END" to "QUIT", I was able to avoid conflicts with "EDIT". This allowed me to use GET instead of INPUT for single-key command input.

Lines 2000-2010 display cursor-movement and exit key prompts.

Lines 3000 to 3070 simplify track and sector entry. You may enter "/" to specify the same track or sector, ";" or "+" to increase track or sector count by one, or "-" to decrease track or sector count by one. Track and sector counts now "wrap around." When a sector count increments past 15, it goes to 0 and the track count is increased by one. Likewise, when the sector count decrements past 0, it goes to 15 and the track count decreases by one. Track counts wrap from 0 to 34 and from 34 to 0. This makes it possible to scan through a disk sector by sector just by entering "/,+" repeatedly.

Now that I have listed my enhancements and my reasons, let me close by saying that I have great respect and admiration for Mike Westerfield and his capabilities as a programmer. The actual formula for his program is excellent.

Richard A. Bryant
Brunswick, ME

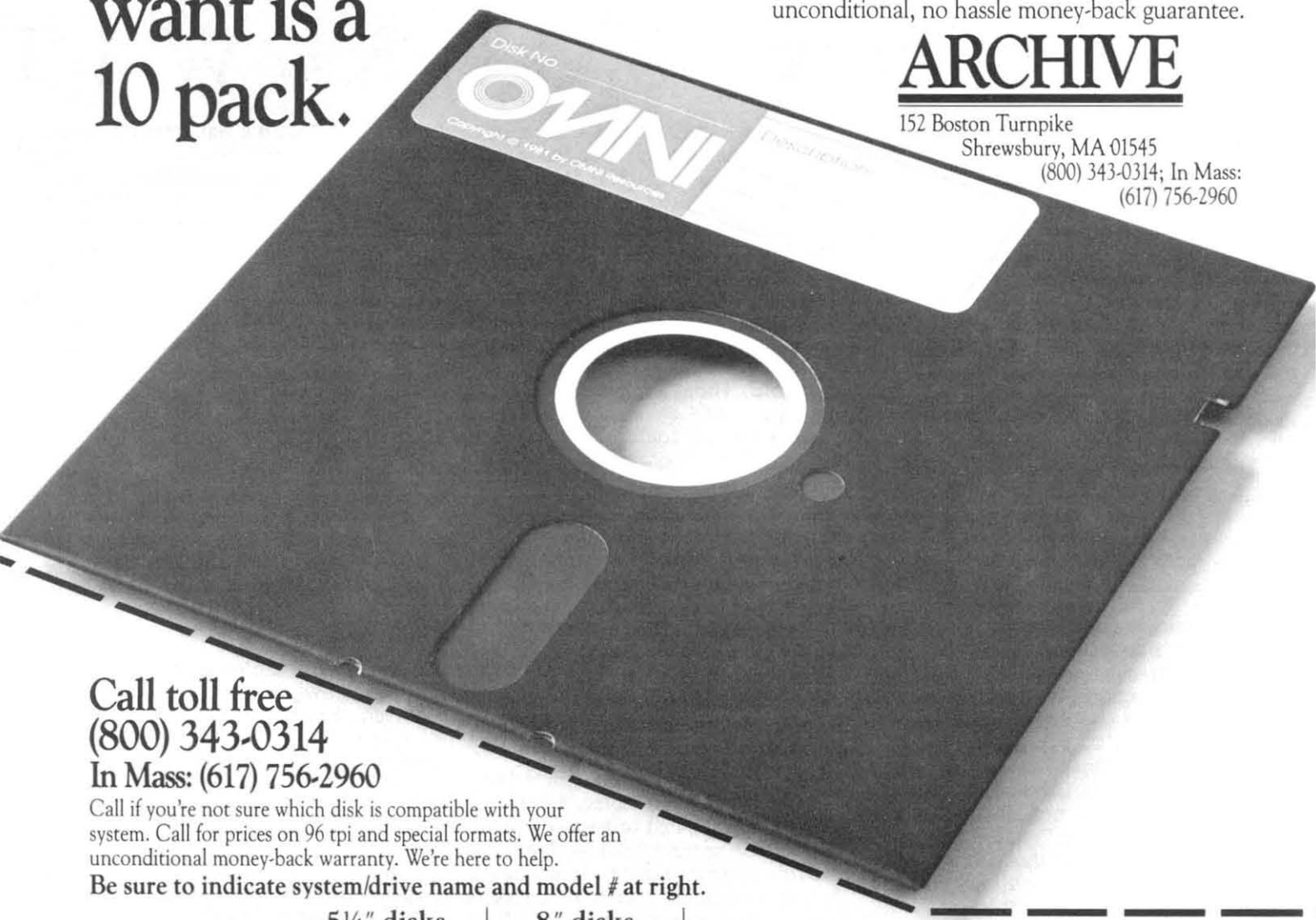
Get Omni quality for as little as $1.99...

even if all you want is a 10 pack.

Call toll-free for great savings on Omni's complete line of 5¼" and 8" premium disks. Each is certified error-free at a minimum of twice the error threshold of your system. Each is rated for more than 12 million passes without disk-related errors or significant wear. And each is precision fabricated to exceed all ANSI specifications with such standard features as reinforced hub rings and Tyvec sleeves. Get same day shipment and an unconditional, no hassle money-back guarantee.

ARCHIVE

152 Boston Turnpike
Shrewsbury, MA 01545
(800) 343-0314; In Mass:
(617) 756-2960



Call toll free
(800) 343-0314
In Mass: (617) 756-2960

Call if you're not sure which disk is compatible with your system. Call for prices on 96 tpi and special formats. We offer an unconditional money-back warranty. We're here to help.

Be sure to indicate system/drive name and model # at right.

	5¼" disks		8" disks		
	Cost per 10 pack	Quantity	Cost per 10 pack	Quantity	Total Cost
Single side/single density	$19.90	_____	$24.90	_____	$_____
Single side/double density	$23.90	_____	$31.90	_____	$_____
Double side/single density	——	_____	$34.90	_____	$_____
Double side/double density	$37.50	_____	$37.50	_____	$_____
Flip/Floppy reversible	$39.90	_____	$39.90	_____	$_____
Plastic library case (in lieu of soft storage box)	$ 2.99	_____	$ 3.49	_____	$_____
Shipping and handling ($2.00 first 10 pack, 40¢ additional 10 packs. Continental U.S. only.)					$_____
5% sales tax (Mass only)					$_____
				Total	$_____

☐ Check ☐ Master Card
☐ COD ☐ VISA

Card # _____ Exp._____

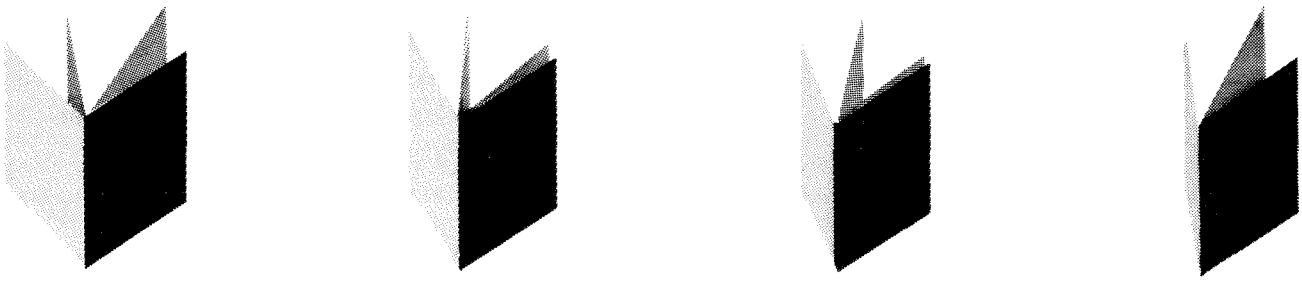
System/drive model # _____

Name _____

Address _____

_____

Tel. _____



A BOOK WIT

by Allen L. Wold

"That "book" is actually a tiny computer, not much bigger than a paperback, with a flat screen, a few control keys, and a huge memory containing millions of pages of text."

A friend of mine was worried about her child's involvement in computer games. Her concern was that with all the time her child was spending at the computer, she was not socializing with other children. "After all," she said, "it's such a solitary business, playing games. She just sits there, staring at the screen, not talking, not moving."

"Just like reading," I said.

Which isn't to say that there might not still be a problem with her child's fascination with computer games, but her concern was with the solitary nature of the activity. The game player is all alone, only interacting with the computer. When I brought it to her attention that books had the same effect, her particular worry vanished.

It seems appropriate that books be considered in a column on the future of electronic entertainment. After all, that's why most people read books, for recreation. I read science fiction, my mechanic reads mysteries, my wife reads romances, and so on. But where does the computer come in?

The Hitchhiker's Guide to the Galaxy, by Douglas Adams, is a story about the last Earthman's adventures in an insane universe. He has only a towel and a book, "*The Hitchhiker's Guide*" of the title, to help him. That "book" is actually a tiny computer, not much bigger than a paperback, with a flat screen, a few control keys, and a huge memory containing millions of pages of text. Though to some that might seem the ultimate in portable libraries, it is, in fact, not the last word. Though it is supposedly the product of a super-advanced technology, we could almost produce such an item today...almost.

Though the technology is not yet here, it is coming fast. Various forms of flat screens are rapidly being developed. For our computerized book, we will need something about 4 inches by 7 inches (the size of a paperback). It should be thin enough that, when closed, it will be no more than about an inch and a half thick. When the computerbook is opened, one side will be the screen, the other will contain the few control buttons.

The keyboard for the computerbook would not be very complex. After all, it will not be intended to be a computer, will not be programmed, and will not provide communications. It will be only a data terminal, used only to receive information. Other functions will be served by other devices. The controls need be no more than on/off (default off if the cover is closed), scroll/stop, back up (reverse), scroll speed control, hold, display/suppress graphics, image size control, hold/find place, keyword search/index, and so on. Further input will not be necessary.

The biggest problem is the screen display. Though TV screens can now be made with a diagonal measurement of less than 3 inches (Panasonic Micro Color TV, 2.6 inch diagonal measure), the depth of the CRT is still a problem. To be truly portable, as a paperback book is, we need a flat screen of some kind.

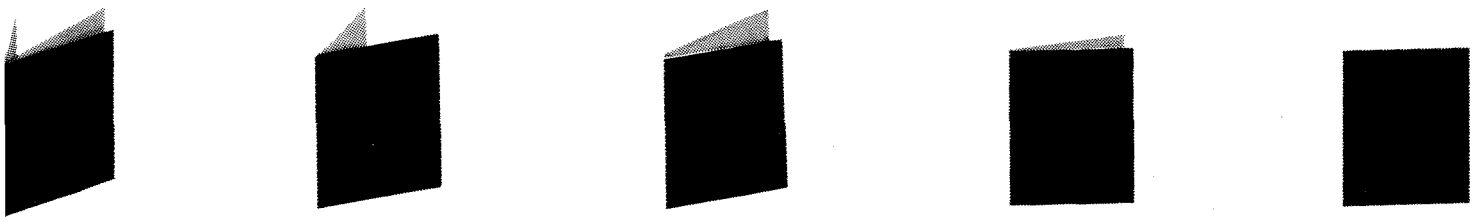
Some of the possibilities include light emitting diodes, electroluminescent display, gas discharge, electrophoretics, electrochromics, magnetic particle displays, and liquid crystal displays, according to Bernard Conrad Cole, writing about portable computers in *Interface Age*, January, 1982.

Both in the United States and Japan, television and computer companies are busy exploring these and other technologies, developing visual displays that meet certain criteria: They are flat (less than 3/4 inch thick), take very little energy to operate, remain visible after the power is off, are visible in low light or even complete darkness, and possibly even produce full color.

Given that the problem of a small, cheap, low-powered, flat screen is solved, the next problem is memory. In order to be a real book, the computerbook should hold at least 200,000 words. There are various possibilities, none of which are completely satisfactory at the moment.

First, we don't need the absolute speed of ROM chips. We're not running a program, just reading text. Bubble memories, which are about as fast as floppy disks, can store masses of data, but are still quite bulky and expensive. Magnetic media are even bulkier.

Optical data storage sounds promising, at first. A conventional laser disk could hold something in the neighborhood of a billion words of text. If we reduced the disk to only a three inch diameter, still accounting for the central hole, it could hold over sixteen million words. The only problem with a disk is that, like a record, you have to spin it, and such



H NO PAGES

mechanical devices would take up more space than our computerbook would allow. They would also draw more energy than a small battery would provide. The scanning lasers themselves need not be a problem as, for applications similar to this one, they are being developed to be no bigger than a grain of rice.

Another possibility is the laser card, being developed at SRI International. According to John Douglas, in an article on micro-memories in the July/August issue of *Science 82*, this card, no bigger than a credit card, would be able to hold up to 100,000 words of text. He does not explain how it would be read, but I assume it would use "micro-lasers." Such a card could be carried in the pocket without fear of damage, or in a compartment in the computerbook. An entire library could be contained in a shoebox.

The foregoing indicates that the technology necessary to create such a computerbook, while not yet available, is in the offing. If the development of pocket calculators and microcomputers is any example, it will be available sooner than we might imagine. I don't wish to discuss the technology, *per se*, any further, but to examine how such technology might affect our lives.

Let's see what we'd have. First, you would only need to buy one computerbook. The text cards would be quite inexpensive when mass produced in typical publishing quantities. The computerbook could be carried anywhere, but more importantly, a whole library could be carried along with it.

The computerbook will not degrade, as paperbacks will. The text could be adjusted to any size the reader found desirable. Those with sharp eyes could read print the size of that found in *The Compact Edition* of the Oxford English Dictionary, while those with vision problems could read characters an inch tall or more. The screen would not have to scroll right to left, it would merely justify the margins to be compatible with the type size.

Needless to say, the text would scroll only as fast as you desired, and would pause whenever you wished. The computerbook would automatically keep your place if you put it down for a moment. In fact, it could keep several places, so that several people could read the same text, or different texts, without losing their places. You could read at night, in dim light, or under a bright sun, without eyestrain.

Books with illustrations frequently present the problem of placing the photo or drawing in proximity to the pertinent text. With the computerized book, this problem would be eliminated. Unless you choose to suppress graphics, the picture would appear at the top of the screen with the text at the bottom. The picture would remain for as long as the text referred to it. You could also go back to previous pictures, if you wished.

All well and good, so far. While rather different from paperback publishing of today, the effect on our reading habits does not seem too profound. However, consider this.

Suppose the text card, instead of being ROM-like, was EPROM-like. Instead of each card containing an indelible text, it merely contained the capacity to store text, which could be changed at will. You might want to have some cards recorded permanently, such as favorite novels. But what about newpapers?

We are assuming that, by the projected time, the home computer phenomenon will have progressed to the point that everybody will have their phone and television connected to a computer or computer terminal, which will, in turn, be part of a larger computer network. Your newspaper subscription will not provide you with a physical, tree-pulp paper, but with the right to plug a text card into the terminal, and have the newspaper office "program" it with the latest edition. Instead of being faced with an edition deadline every day, the publisher would merely update, add, or delete, on a cyclic basis. For an extra fee, perhaps, you would have access to all the older editions.

I thought for a while about having the newspaper broadcast to your computer via radio signals, but that does not seem feasible. Each paper publisher would have to have its own frequency, which isn't too bad, but when I started to think about libraries doing something similar, the problem got out of hand.

Here's what I mean. Today, the gross publishing industry can be roughly, and simplistically, broken down this way. There is the author, who creates the material. There is the publisher, who puts the text into an accessible format, such as a book. There are the library and the book store, which make the book available to the reader. Libraries, especially the Library of Congress, archive the material for posterity. Finally, there is a reader. Of course, in many cases, these functions overlap.

With our computerbook, many of these functions will change, some of them radically. Authorship will remain fundamentally the same, though the text may be written on a word processor and submitted to a publisher via a modem, instead of typed and submitted through the mail.

The publisher will no longer be responsible for printing and binding, or for distribution and storage, which today comprise the major portion of a publisher's budget. The text will still have to be put into the standard computer-readable format, but the functions of selection and editing will be emphasized. In fact, the publisher,

per se, will no longer exist, but will be replaced by the editor, whose preference or subject, quality of selection and editing, and encouragement of authors will be deciding factors when selecting texts. Instead of *Pocket Books*, you will have *Hartwell Selections*. With the reduction of production costs, those functions, and promotion, will receive more attention.

The bookstore will no longer have to provide massive storage space for multiple copies of texts which may, or may not, sell. Libraries will not be limited by shelf space for the number of different titles they have available. In fact, bookstores and libraries, as we know them today, will no longer exist. They will resemble local radio or television stations, providing the public with accessibility to any texts "in print." If you want a book, you contact your library, via the nearest computer terminal, and the library transmits the text to your computerbook card. They will also serve the editor and author by collecting the royalties.

Any text, once published, will be permanently stored in the Library of Congress. It will be the Library of Congress which will make the texts available to the local library store. You, as the reader, will then have access to every text which has ever been published — any time, anywhere.

Let us look at our computerbook again. It is sometime in the future, let's say a hundred years from now. You own a computerbook data terminal, as does each member of your family. You carry it with you, as you do your wallet. It is three inches wide, seven inches high, one and a half inches thick. It opens like a book, with the high resolution color screen filling one side, and a few large keys on the other. There is a slot which holds three or four text cards, any of which can be accessed at any time. A jack permits you to plug into any public or private computer terminal. One card carries the daily paper, updated at any time. Another carries the novel you are currently reading. Another is a textbook on computer telescopy, and the fourth is blank, just in case you decide to add something new.

Other novels you have read need not be kept on cards. Your library store knows which ones you have bought, and you may receive copies of them again at no further charge. The newspaper will let you access earlier releases, should you wish to trace a story backwards, or refresh your memory. Your text is there for you to study whenever you wish, and like your novels, has been paid for in advance. If you need a textbook on accounting instead, you can easily exchange it.

When I was writing this column, I often needed to refer to back issues of various magazines. This was a problem, because I couldn't remember which magazine I needed, what issue, nor even the article titles for which I was looking. As a subscriber to a computerized magazine service, you would need only to plug in, and use the publisher's indexing service to find any article, story or ad that you wished. If you didn't subscribe, the library store would charge you a very small fee to see the article in question. Once you had paid for it, it would be yours forever. The one time fee would pay the store, the editor, and the author of the article.

I don't know about you, but I want my computerbook today.

COMPUTER Nightmare

By David Cohen

Editor's Note: SoftSide's editorial department received a desperate letter from the author of the following article. He is thoroughly convinced that his computer and those of users throughout the world carry a death wish for their "masters." We include the scenario here as an interesting psychological study of the effects of prolonged exposure to the pixels of a CRT display.

After ten continuous hours of intensive flowcharting, coding and debugging "Zap The Xenophobes," my head was throbbing and my vision blurred. I decided to take a hot shower and sleep for a while before beginning to type in my masterpiece.

I awoke the next morning, refreshed and ready to start entering the first piece of code. I picked up my messy sheaf of program notes and approached the computer sitting unobtrusively on my desk. I carefully laid my notes on the machine's chassis and pulled out a box of new diskettes. Removing four, I popped them into the awaiting disk drives. I put the blank cassette into the recorder, plugged the BASIC cartridge into its slot, closed the slot cover, and powered up the console.

The prompt, "READY", was displayed on the sky-blue screen. Then it started. A few seconds later, as I was looking down at my notes, I heard the familiar new-line "beep." Looking up, I saw words appearing one at a time on the screen, with no instructions from me, until the finished text read, "READY OR NOT, HERE I COME!"

Instinctively, I hit the Reset key before I was even sure I had really seen the words my mind had discerned. "Oh, come on, stop day dreaming," I told myself, with a slap across the face. I noticed I had forgotten to format the new disks so I removed the first blank and inserted the master diskette in the empty disk drive.

The drives growled, and the red indicator lights flashed. Fine, but the disk operating system's prompt wasn't on the screen. Then, there it was in capital letters — "DOS." It continued printing, in time to strains of *The Doors* emanating from the speaker..."IS DE END MY ONLY FRIEND, DE END..."

Inexplicably, the disk crashed, with a bizarre pattern appearing on the screen and the formatting process prematurely terminated. Damn, now I'd have to reboot the system and start all over again. Fortunately, I had made a back-up of the master diskette, so I inserted it and reinitiated the formatting sequence.

The four disk drive lights lit up in unison as the DOS was magnetically etched on the blank disks. Good. I began keying in the code. At the tenth line, I got an ERROR 257 — Bad Typing. Dutifully, I retyped the line and hit Return. The error handler came back with, "HA, HA, HA — WHAT A STUPID MISTAKE, YOU FOOL. AND YOU CALL YOURSELF A PROGRAMMER, HA! ERROR 500 — GETTING UP THIS MORNING!" Malicious laughter was broadcasting from the speaker. Now I was sure *something* was going on.

I hit the Reset only to have the machine respond, "OH, NO YOU DON'T — YOU'RE NOT GOING TO SCRAMBLE MY CIRCUITS WITH YOUR BUNGLING PROGRAM!" I had no idea what was going on. I tried Break, Control A through Z...nothing worked. I tried to pull the plug, but it seemed glued into the outlet. The laughter grew louder and more ominous. Finally, I remembered that opening the cartridge slot cover would shut off the machine.

I sat in front of the console until my breathing relaxed before carefully closing the slot cover again. I rubbed my eyes and stared long and hard at the machine — nothing but the DOS prompt. After a pregnant pause, I went into BASIC and again began entering the program. Then, without warning, I heard the switch of the voice synthesizer flick on. A thundering monotone filled the room: "YOU THOUGHT YOU'D PULL MY PLUG, STUPID HUMAN — NOW YOU WILL PAY THE PRICE FOR YOUR FOLLY!
HA,HA,HA,HA,HA!"

As the maniacal laughter reverberated through the room, I suddenly smelled smoke. Looking wildly about, I saw the disaster. There, on top of the machine — my notes! Ten hours of painstaking writing and planning were burning to ashes on the overheating chassis!

With a mind not totally my own, I reached under the bed and pulled out an ax. I swung it high over my head, my eyes blazing wildly with but one thought — vengeance! The maniacal laughter ceased, replaced by the hideous, high pitched whine of the cassette recorder loading data at maximum volume. As I dropped the ax and clutched my ears in agony, chaos thundered from the peripherals' speakers in quadraphonic sound.

My knees turned to jelly, and as I fell back into the bookcase I heard the disk drives angrily grinding, like buzzsaws about to cut me down. Scarcely was the thought subvocalized when, to my horror, the disk drives opened simultaneously and four, razor sharp diskettes flew at me like ninja throwing stars. I dropped to the floor (no major effort, I assure you), and the deadly disks whistled past, burying themselves inches deep in the bookcase behind my head.

I shielded my eyes from a blinding light glaring from the CRT. The smoke grew thicker as the machine's guts melted — strained past their limits by the monster's 6502 brain.

I was snapped out of my shock by the click of the cartridge slot cover sliding open to reveal, not the BASIC cartridge, but a towering, fully armed, BASIC class, short range nuclear missile smoking in its silo, prepared for launching in 20 seconds!

Almost unconscious from the overwhelming assault on my senses, I crawled to the ax on the floor and feebly raised it over my head to destroy the microcircuited menace trying to eliminate me from the great equation. With a sudden burst of strength borne of desperation, I brought the ax down hard, smashing the chassis open. Again, I raised the ax and let it bite deep into the computer's bowels. My limbs stiffened and my mouth opened in a silent scream of agony as 120 volts of electricity, boosted by the power supply, surged through me, nearly searing my brain in a split second. With a scream scarcely audible above the din, I fell back to the floor, barely alive. The chips just couldn't take it! Sparks and fire flew from the dying machine and acrid fumes of smoldering semiconductors reducing to slag filled the room. Gasping, I realized I had conquered!

My heart fell as a tremor shook the floor and a dark shadow consumed the room. To my horror, I realized the Doomsday Maneuver had been activated — the towering BASIC missile, slowly emerging from its silo, was about to be launched and would impact in ten seconds. My face twisted in fear and despair as I realized no power in the world could stop it now... 5...4...it was coming closer and closer...3...2...1...Yaaaaaaaaaaaaa!

I flew out of bed onto the floor, rolling up into a tight ball to await the end. Suddenly, I realized it had only been a nightmare.

I stood up slowly and shivered in my sweat-drenched pajamas, the blood wildly pounding in my temples. "A nightmare, yeah, that's all it was. Just a bad dream, nothing to get excited about. You're safe now," I told myself as I prepared to eat breakfast.

After some eggs, cereal, and a large Bloody Mary, I slowly approached the computer and hit the power switch. "READY..." Smiling with relief, I prepared to get started..."OR NOT, HERE I COME!"

My eyes jerked open wide as I flew from my seat. Screaming wildly, I crashed out the window, wicked laughter pealing behind me. I scrambled to my feet and took off like a shot, refusing to look back. Suddenly, a tremor shook the earth and threw me to the ground. Fearfully turning my head, I saw a mushroom cloud gently wafting skyward from where my house had once stood...

From Whence We Came—
MUSIC FROM THE MACHINE

by Randal L. Kottwitz

"Inside the door lies the deepest, purest joy of life. It is the joy of music. To millions that door has been locked. They have been barred out, not by a lack of music feeling — for that is instinctive to everyone — simply by a lack of music's mechanics — of mere note knowledge."

The Aeolian Company
Advertisement, 1915

I play the piano — poorly. Most people don't play at all. This evening, Artur Rubinstein played the greatest piano masterpieces of the world in my living room. I sat, enraptured, as the strains of Liszt, Brahms and Debussy wafted through my ears. Ah! What I wouldn't sacrifice to attain his mastery of the keyboard. No, I don't know Mr. Rubinstein, nor has he ever graced my home with his physical presence. However, through the miracle of modern recording and reproduction technology, I can listen to his artistry whenever I wish. Indeed, I can invite Pavarotti, The Boston Symphony, Judy Garland or the entire cast of *A Chorus Line* to join me for the evening and perform my favorite music. Even if I possessed all of the musical instruments necessary and had the voice of an angel, it would be impossible for me to enjoy the spectrum of music that recordings put at my disposal.

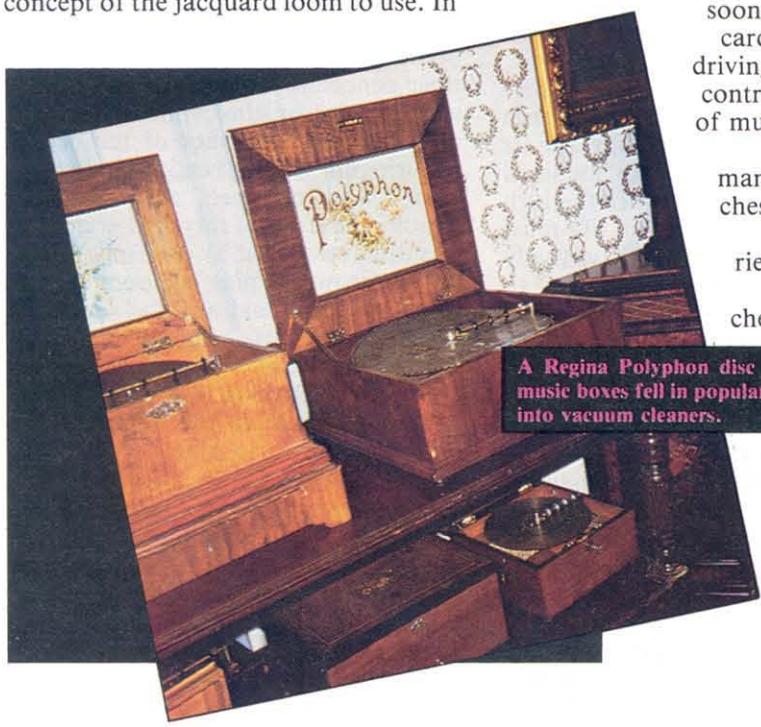
Music is a vital part of our everyday lives. It soothes, entertains, provides "background noise" and even makes us buy more groceries at the supermarket. We take its constant presence for granted as the technology for its production has become an assumed luxury. Reflect on life prior to the 19th century and realize how dull life must have sounded. Before 1800, virtually the only way people could listen to music was as it was performed by live musicians. Although labor was still relatively inexpensive, the inconvenience and comparative cost of "live" entertainment made it a luxury for only the very, very wealthy.

Enter, the first of the mechanical music machines, the cylinder music box. With its advent, music started the march to the place of constancy it now holds. The principle behind the music box's operation was relatively simple — a metal cylinder with pins mounted in it was rotated by a spring mechanism. The pins were positioned to "pluck" the tines of a metal comb placed parallel to the cylinder, each tine graduated in size to produce a different note of the musical scale. The comb could easily be compared to a graduated row of flexible tuning forks placed side by side. Although simple in principle, the genius required to develop and "program" such a system was extensive. In many cases, the craftsmen drilling and positioning the pins in the cylinder may have been working from a musical score they had never heard performed.

It was not long before the "simple" cylinder music box became not so simple. Lead was added to the underside of some of the longer tines of the steel comb to allow lower notes and a fuller sound. More combs were added for additional "voices" and greater volume. Soon, the Victorians were exercising their penchant for mechanical gadgets and the pins on the cylinder were not only plucking the tines of the comb, but ringing bells, beating miniature drums and castanets and even activating the stops of tiny reed organs. It was discovered that by carefully offsetting the rows of pins surrounding the cylinder, additional musical selections could be added to a single cylinder. All that was necessary was a mechanism to calibrate the position of the cylinder by a distance equal to the offset of the pins. Of course, a single music box could have interchangeable cylinders to support even more selections. The picture above shows cylinder music boxes of all sizes and shapes in the collection of The Musical Wonder House. "The Geneva Titan" orchestral music box, also in their collection, incorporated *all* of the above features in a mammoth box almost five feet in length with six interchangeable cylinders containing 72 selections! "Orchestral" indeed.

The next development in the technology of the musical machine came in the mechanism doing the "plucking." The brass cylinders were difficult to produce and did not easily lend themselves to the production line manufacturing necessary for marketing to the masses. The resultant change brought about the disc music box. A large metal disc, approx. 18 to

20 inches in diameter, was stamped on a large press. In the more common implementation of the technology, metal teeth were die-cut in the disc and bent downward to perform a function similar to the pins on the cylinder type music boxes. The rest of the mechanism on these boxes was very similar to that of the cylinder boxes, simply repositioned to be "plucked" by the teeth of the disc as it rotated. One, very rare, type of disc music box provided a disc punched with a gridwork of holes and hundreds of loose metal pins. The owner could then place the pins in the appropriate holes to arrange his own selections. The "language" of this programming was in simple binary — either a note was plucked, or it was not. This concept provided a versatile musical instrument, but the tedium of "programming" a single selection kept the idea from gaining any great popularity. There was one other type of disc music box which brought the "punch-card" concept of the jacquard loom to use. In this type of music box, the disc had no teeth projecting from its holes, but provided the "pattern" for spring-loaded teeth located on the music box to pop up through. When released into a hole, each of the teeth would trigger the plucking of a tine of a comb, ringing of a bell, etc.



A Regina Polyphon disc music box. When music boxes fell in popularity, Regina moved into vacuum cleaners.

The punch-card technology Joseph Marie Jacquard developed for his famous looms was one of the first practical applications of binary "language." His ideas reached their ultimate development in the punch-card computer systems we know so well from our telephone bills. However, along the way, they had a dynamic impact on the musical machine. Organs were the first place the on/off condition created by a punched card came into use, because their existing technology was already dependent on controlling the on/off conditions of the air passing to the pipes or reeds. The cards were simply chained together, pulled through a central compressed air source and given control of the "stops" of the organ formerly governed by the keyboard and other controls on the manual. Prior to the application of steam technology, foot pedals were pumped by the person playing the organ to provide the compressed air and drive the mechanism pulling the cards.

Again, the Victorians applied their desire for "all things mechanical" and soon the holes in the cards were not only driving the organ, but controlling every sort of musical instrument imaginable. The mammoth street orchestrions of northern Europe carried the equivalent of an entire orchestra, with all of the instruments "played" by seemingly endless chains of punched cards driven by steam. The most common example of an orchestrion is the mechanical orchestra in an old fashioned carousel.

The piano is said to be the most universal of musical instruments and it was not long before the "automatic player" was brought to its keyboard. The first playing devices consisted of a separate box to be pushed up to a standard keyboard. Felt covered wooden "fingers" were driven by hand-cranked, pegged barrels (like the cylinder music boxes). I can't help but compare this "kluged" technology to that of the boxes fitted over the keyboards of electric typewriters to create letter-quality "printers" for the computer industry of today.

Soon, Jacquard's technology came to the piano, as well, and players were driven by foot pedal power and perforated paper music rolls. External players generally did not encompass more than 65 playing notes but, by 1910, the self-player was built into the piano itself and many covered the full 88 note range of the keyboard. The first player pianos lacked dynamic variations in the musical sounds produced. By judicious use of hand operated levers for dampers, hammer rail lift and tempo, as well as pedaling force and speed, the operator of a foot-pumped player piano could impart a measure of expression to the music produced from the roll but the interpretation was most generally the operator's own.

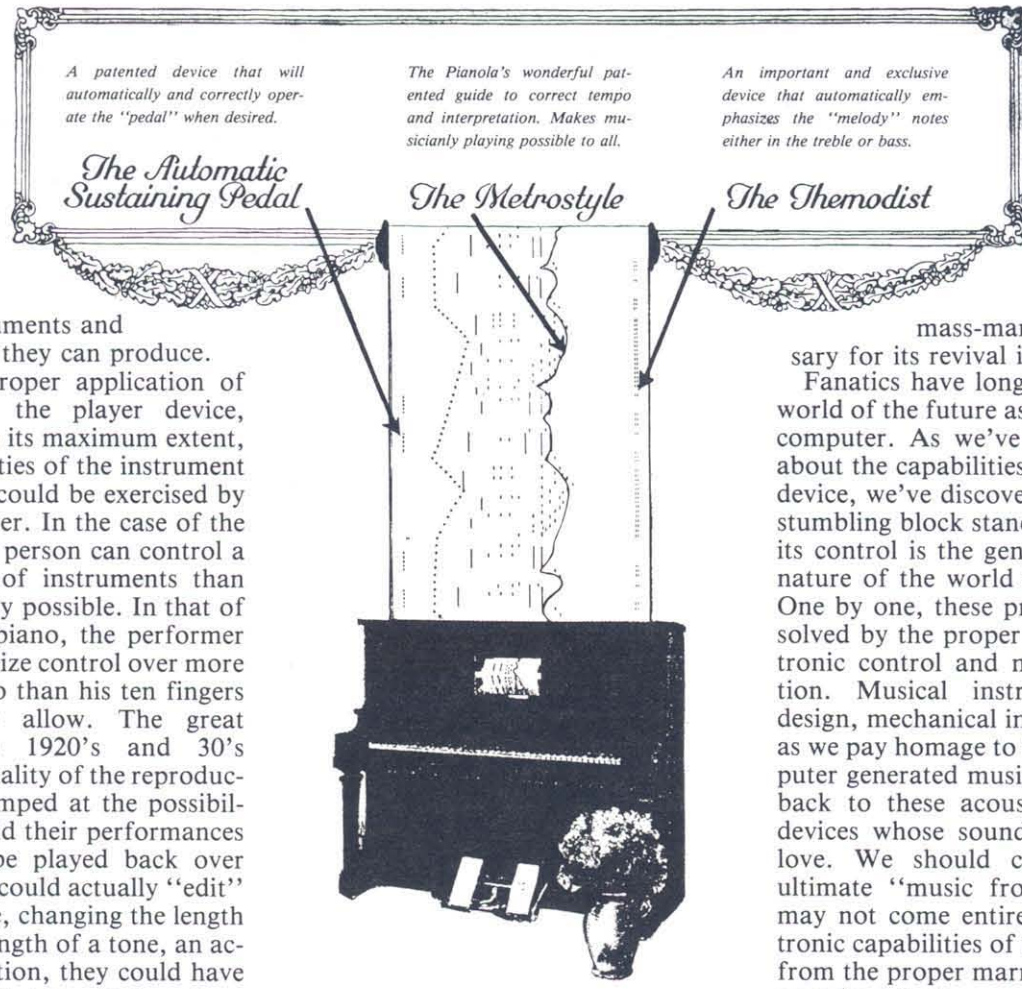
Around 1904, Edwin Welte of Germany developed a player mechanism which operated from paper rolls coded with binary control instructions for operation of the sustaining and soft pedals of the piano, as well as the striking force of individual keys. A piano was designed to record the dynamics from a live performance, special pneumatic devices were built to recreate these dynamics in the player, and the "reproducing piano" was born. As the reproducing piano gained popularity, many manufacturers jumped on the bandwagon. In the end, three major companies dominated the market: Duo-Art (earlier versions known as Pianolas), Welte (Welte-Mingnon in the U.S.), and Ampico. The mechanism could be driven either by an electric air compressor or foot pedals, but the devices actually controlling the mechanical elements of the piano were largely pneumatic.

The "musical computer" driving a reproducing piano is truly amazing. The complexity of its programming is not at all unlike that necessary to control the music synthesizer as we have come to know it. The major difference is in the device actually producing the sound. The music synthesizer electronically creates sounds to either mimic traditional musical instruments or produce tones unique to itself. The reproducing piano or any of the other "automatic" instruments described above produce the same sounds as they would if played by a human. The

Player device, in its simplest application, provides a means for those not blessed with the "note knowledge" or manual dexterity to operate the instruments and enjoy the sounds they can produce.

As with the proper application of any technology, the player device, when exercised to its maximum extent, takes the capabilities of the instrument far beyond what could be exercised by a human performer. In the case of the orchestrions, one person can control a greater quantity of instruments than otherwise humanly possible. In that of the reproducing piano, the performer /technician can seize control over more parts of the piano than his ten fingers would otherwise allow. The great pianists of the 1920's and 30's recognized this quality of the reproducing piano and jumped at the possibility. Not only could their performances be recorded to be played back over eternity, but they could actually "edit" their performance, changing the length of a note, the strength of a tone, an accent, etc. In addition, they could have as many as all 88 keys of the piano sounding at one time and control the sustain, accent and other characteristics of each key individually.

Regrettably, the advent of quality phonographs and the Great Depression brought an end to virtually all but the standard 88-note player piano industry. The reproducing piano has become a relic of our past, too costly to produce and having too many musical rivals for it to possibly have the mass-market appeal necessary for its revival in today's society.



The rolls for reproducing pianos contained "control" instructions to operate many more functions of the piano than a standard player mechanism.

Fanatics have long been touting the world of the future as controlled by the computer. As we've found out more about the capabilities of this *electronic* device, we've discovered that the main stumbling block standing in the way of its control is the generally *mechanical* nature of the world in which we live. One by one, these problems are being solved by the proper marriage of electronic control and mechanical operation. Musical instruments are, by design, mechanical in nature. Perhaps, as we pay homage to the world of computer generated music, we should look back to these acoustical, mechanical devices whose sounds we've come to love. We should consider that the ultimate "music from the machine" may not come entirely from the electronic capabilities of the computer, but from the proper marriage of its *unique* sound production qualities and its *controlling* capabilities with the lovely tones of the mechanical musical instruments our civilization has spent centuries refining. With that marriage, the new art form of "electronic music" would lose its sterility, and reach for our very souls.

Reprinted from "*Through You I Live Forever*" by permission of Beckett Productions, Dallas, Texas.

The Musical Wonder House

Much of the material contained in this article was garnered during the author's recent visit to The Musical Wonder House in Wiscasset, Maine. This museum is one of the musical wonders of the world, with a collection of mechanical musical instruments ranging from a small 18th century mechanical pipe organ to the resonant music boxes of the Victorians...from mechanical reed organs to the "talking machines" and Grand Player-Pianos of the early 20th century. A visit to this grand Georgian sea captain's mansion is an experience in music and history never to be forgotten. Many thanks go to Danilo Konvalinka, Douglas Henderson and Timothy Wallace of The Musical Wonder House for their generosity and hospitality.

If you are interested in knowing more about these musical instruments, the Automatic Musical Instrument Collectors' Association (AMICA) publishes a newsletter. Many of the members of this organization are electronic engineers and computerists as well. For more information, write:

AMICA
c/o Dorothy Bromage
P.O. Box 387
La Habra, CA 90631

How about a nice vacation on the beaches of France – for just $39.95?

At SSI, we think that our latest software for the Atari,® TRS-80® and Apple® — BATTLE FOR NORMANDY™ — is more than a great strategy game. We think of it as a great vacation package. After all, we are whisking you off to the northern coast of France for 25 days of fun and excitement (June 6 to 30, 1944) — all for just $39.95!

Of course, we're not promising peace and quiet. There is a war going on, you know....World War II!

We'll be taking you to the beaches of Normandy in style. Not on a mere jumbo 747 or an ocean liner, but an LCI (that's Landing Craft Infantry). Let's see the Joneses top that!

Like any good traveler, you need to plan ahead. What's the weather going to be like out there? Rough and stormy? Calm and gorgeous? It's hard to say, so you'd better be prepared for all kinds.

How about supplies? Well, we're a little tight on luggage space, so you'll have to juggle among the things you really need: fuel, general, and combat supplies.

How about some friendly sea bombardment to let the natives know you're coming? After all, you don't want trouble once you hit the beaches. The natives are a bit hostile at first, and a little naval artillery fire really helps to loosen them up.

And our service? Have we got good service — and plenty of it! Whole divisions and regiments of infantry, paratroopers, tank units, and commandoes are at your every beck and call. We do have to apologize for their different leadership and combat ratings. It's so hard to get good, consistent help these days.

You want sightseeing? You've got sightseeing! Nice historical towns like St. Lo, Cherbourg, and Caen — which you just have to take in (or take over, as the case may be.)

We know there's a lot that goes into preparing for a vacation, and we don't want you to worry about the details one bit.

SSI is one of the most advanced companies around because we're totally computerized. Our great computer program takes care of all the dirty work so all you do is enjoy. After all, you're on this trip for the fun and games, not work. And if you have a hard time finding friends to play with, who needs friends? You can play solitaire against the computer any time.

For complete details see the travel agent at your nearest game or computer store today!

SSI

STRATEGIC SIMULATIONS INC

As part of our demanding standards of excellence, we use maxell floppy discs.

If there are no convenient stores near you, VISA and M/C holders can order direct by calling **800-227-1617, ext. 335 (toll free)**. In California, call 800-772-3545, ext. 335. To order by mail, send your check to: Strategic Simulations Inc, 465 Fairchild Drive, Suite 108, Mountain View, CA 94043. (California residents, add 6.5% sales tax.)

Available in 32K cassette or 40K disc for the Atari® 400/800; 16K cassette for the TRS-80® Level II, Models I & III; and 48K disc for the Apple® II with Applesoft ROM card. Please specify which version when ordering.

WRITE FOR A FREE COLOR CATALOG OF ALL OUR GAMES.

GENERAL INFORMATION

Concerning SoftSide line listings, SWAT & Magnetic Media

Follow these procedures unless otherwise instructed by the documentation in the magazine. Back issues may differ in some details.

SWAT TABLES

At the conclusion of each line listing of a ***SoftSide*** program, we include a *SWAT* (*Strategic Weapon Against Typos*) Table. *SWAT* was published in issue #30 of ***SoftSide*** and is available as a free reprint. Please send a self-addressed, stamped envelope to ***SoftSide*** Publications, Inc., Dept. *SWAT*, 6 South Street, Milford, NH 03055.

APPLE™

Disks are in 16-sector format, created under DOS 3.3. To use, just boot the disk. A cover/menu program will run automatically.

Tapes LOAD in the normal manner. Advance the tape to the beginning of the lead-in tone; stop the tape; insert the plug into the EAR jack; type LOAD; start the tape; and press RETURN. Side two of the tape is a duplicate of side one, unless one or more Integer BASIC programs are included, in which case side two contains the Integer programs.

ATARI®

Line Listings use the following conventions in representing unprintable characters, unless otherwise noted:

Characters (including blank spaces) which are underlined should be typed in inverse video.

When graphics or control characters are to be included in a string (between quotation marks), it will be noted in a nearby REMark. In such cases, graphics characters are represented by the corresponding lower-case letter, and control characters are represented by the corresponding unshifted key symbol. For example: The lower-case letter s represents a graphics cross, entered by holding down the CTRL key and then pressing the S key. The symbol = represents a control-down-arrow, entered by first pressing and releasing the ESC key, then holding down the CTRL key and pressing the = key. (See Appendix F, and the back cover, of the *ATARI® BASIC Reference Manual.)*

The one exception to the above practice is that a clear-screen character (ESC CTRL-↰) is represented in listings by a right-hand brace, which looks like this: }

A shifted = is represented in the listings by a vertical line with a small gap in it: ¦

SWAT — Before appending *SWAT* to a program in memory, the program to be SWATed must first be LISTed to disk or cassette (using LIST "D:FILENAME" for disk or LIST "C:" for tape). Next, turn the computer off, then on again, to clear the system and ENTER the program back into memory (using ENTER "D:filename" for disk or ENTER "C:" for tape). Because of the unique method in which *ATARI® BASIC* stores variables in a program, the variable table must always be in the same order to produce accurate *SWAT* codes. LISTing and ENTERing the program is the only known way to rebuild the variable table in a specific order so that *SWAT* codes can match.

Disks do not contain DOS.SYS files, and are therefore not bootable by themselves. First boot a disk which contains any version of DOS, then insert the ***SoftSide*** disk and RUN "D:COVER" (*Adventure of the Month* — RUN "D:INTRO").

Tapes CLOAD in the normal manner. If you have difficulty, try this procedure:
(1) Type POKE 54018,54 and press RETURN.
(2) Turn up the volume on your TV.
(3) Type CLOAD and press RETURN once.
(4) Press the PLAY button and listen.
(5) When you hear a steady lead-in tone, press RETURN again.
Side two of the tape is a duplicate of side one.

IBM® PC

DV is available on individual order. There is no CV at this time.

TRS-80®

Disks are available in Model I or Model III format. They contain the DOS PLUS operating system, and a cover program which automatically runs upon booting. Back issues prior to May, 1982, are available only in Model I format, and may be converted using the TRSDOS CONVERT utility on a two-drive Model III. Older back issues (with Model I TRSDOS) require you to enter BASIC and then type RUN "COVER".

Tapes CLOAD in the normal manner on Model I's, and at low speed (500 baud) on Model III's. The first program is a cover/menu program. Side two of the tape is a duplicate of side one.

NOTES ABOUT MAGNETIC MEDIA

SoftSide disks and tapes are duplicated by reliable, professional duplication services; bad copies are very rare. However, the trip through the mail occasionally wreaks havoc with sensitive magnetic media. If, after a reasonable number of tries and a careful check and cleaning of your equipment, you are not able to load a program from a tape or disk, please return it to us with an exact description of the problem. If we cannot duplicate the problem on our systems, we will advise you when we send the replacement copy.

We use no copy-protection on our media. We urge you to make a backup copy of every disk or tape as soon as you receive it (and at the same time resist the urge to give copies to friends). Our replacement policy does not extend beyond 30 days.

PC/SIDE page 25

TRS-80®/SIDE page 40

ATARI®/SIDE page 54

APPLE™/SIDE page 83

appleFest®

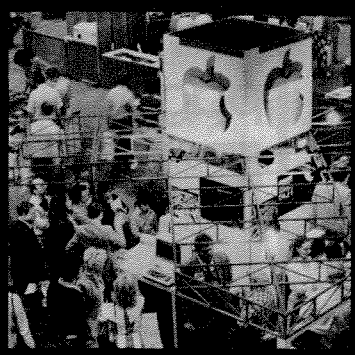
APPLE EUPHORIA

"Absolutely the best show I've ever been to in my life."

"It was incredible—I never knew there were so many products available for the Apple."

These are some of the things people are saying about Applefest, the world's largest exposition of Apples and Apple compatible products.

At Applefest, you can see and try out everything new for your Apple. Software for every conceivable application from arcade games to investment programs, music to machine language, teaching systems to accounting packages, word processors to graphics processors. Hundreds of peripherals, including printers, hard disks, modems, memory cards, video displays and synthesizers, plus accessories, publications, support services and more! Over 5,000 products are displayed and available for purchase at super show prices.



Get a taste of Apple Euphoria. Don't miss Applefest when it comes to a city near you.

Applefest/Minneapolis
Thursday-Sunday
September 16-19, 1982
Minneapolis Auditorium
Admission: Adult $5 per day, Child $3 per day
Show Hours: Thurs. 11 AM to 7 PM, Fri. and Sat. 11 AM to 7 PM, Sun. 12 NOON to 6 PM

Applefest/Houston
Thursday-Sunday
October 28-31, 1982
Houston Civic Center
Admission: Adult $5 per day, Child $3 per day
Show Hours: Thurs. 11 AM to 7 PM, Fri. and Sat. 11 AM to 7 PM, Sun. 12 NOON to 6 PM

Applefest/San Francisco
Thursday-Sunday
November 18-21, 1982
Brooks Hall
Admission: Adult $6 per day, Child $3 per day
Show Hours: Thurs. 11 AM to 7 PM, Fri. and Sat. 11 AM to 9 PM, Sun. 12 NOON to 6 PM

For information on exhibit space or tickets, call or write:
Northeast Expositions, 824 Boylston Street, Suite 202, Chestnut Hill, MA 02167 (617-739-2000)

PC BLUES BOX

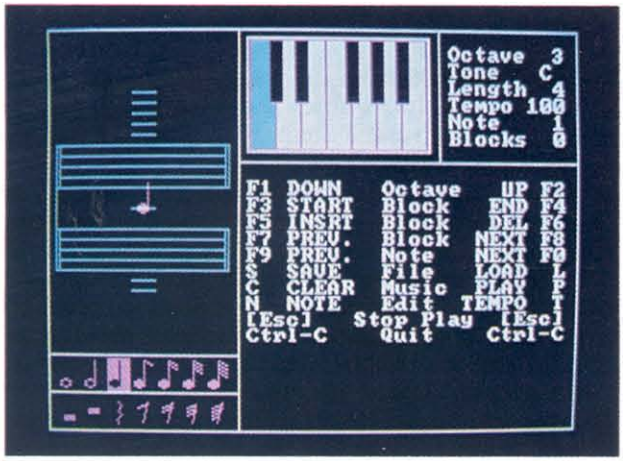
by Fred J. Condo and Alan J. Zett

PC Blues Box is a music editor program for an IBM® PC with 64K RAM, Advanced BASIC, a color graphics adapter, and one disk drive.

One of the first things the ***SoftSide*** programming staff discovered on its new IBM PC was its musical feature. At first, we laboriously typed in long strings of the quasi-musical notation required by Advanced BASIC's PLAY command. Wouldn't it be nice, we thought, to have an easy way to enter sheet music into our PC? Fortunately, ***SoftSide*** now supports the IBM PC, so we can share the results of our wish-fulfillment with you, our readers.

This is the first installment of a two-part series. This month, we have the basic note-entry system for storage of up to 2000 notes, as well as facilities for moving among previously stored notes and editing them. We also have the graphics display system, which consists of a piano keyboard, a musical staff, and text displays of information about the current note. And, of course, there is a means to store and retrieve your music to and from a disk drive.

The first thing that happens when the *PC Blues Box* is running is a pause while the PC allocates array space. Then the screen clears, and various shapes flash rapidly in the upper



left hand corner of the screen. After a few seconds, the main screen appears. The upper right hand corner contains information on the current note. This is called the information square. At first, this square contains the default values that occur whenever the program is started up. At the left is a musical staff with a quarter note positioned at middle C. Below the staff, in the lower left hand corner of the screen, is the note/rest duration indicator. The large rectangle at the lower right tells you the purposes of the PC's function keys, as well as those of a few other keys on the keyboard. Space limitations prevent the listing of all the significant keys on the screen, but all functional keys are described here. The blank area in this block is reserved for communication between you and the computer.

Instructions

First of all, keys F3 through F8 won't work until next month's installment. Go ahead. Push one of them. Their only effect is to cause the display of the message, "Not yet implemented."

Most of the function keys operate in pairs. F1 and F2 are, respectively, the Octave Down and Octave Up keys. They move the current note exactly one octave in the specified direction. Moving down from octave 1 sends you to octave 5; conversely, moving up from octave 5 sends you back down to octave 1. Thus, you can quickly move over the whole range of available notes. F9 and F10 (F10 appears on the screen as F0) move you around the buffer. They are valid only when you have stored notes. The note buffer does not wrap around: if you try to move below note 1 or beyond the first empty position in the buffer, the *PC Blues Box* ignores the instruction.

Striking the "C" key tells the program to clear any stored music from memory. Before it takes this irrevocable action, it asks you if you are sure you want to do it. You don't need to clear the memory if you've just started the program.

The "T" key allows you to set the tempo. The allowable range for tempo is 32 to 255, inclusive. The tempo, quite simply, controls the speed at which the music is played; it leaves the relative lengths of notes unchanged.

The "P" key causes the program to play the music stored in memory. If you decide not to listen to the whole tune, press the escape key, labeled "Esc" on the keyboard. A few more notes will play after you press it, but the music will

stop. Pressing "Esc" when music is not playing does nothing.

Now, to enter music into memory. It is quite simple. The left and right arrow keys on the numeric keypad control the length of the rest or note. Moving left selects longer durations; moving right, shorter durations. Durations wrap around in both directions. The space bar toggles between rests and notes. The up and down arrow keys control the pitch of the note. You can change the pitch even if you are currently entering rests — just watch the information square in the upper right hand corner. In order to enter the currently displayed note, press the RETURN key (the one next to the numeric keypad labeled with a bent arrow). When you press RETURN, whatever is displayed on the screen is accepted as the note, and its number appears next to the word "Note" in the information square. Upon acceptance, the note sounds through the PC's speaker. Note: if you get no response to the arrow keys, press the "Num Lock" key once. It is located directly above the numeric keypad.

You will probably want to save music you've entered on diskette. To do so, simply press "S". The computer will ask you to type a filename for the music, and will then save it under that name. In order to retrieve music from the disk, press "L" (Load). Again, the computer will ask, "File?" You must type the same name that you typed when you saved the music. When either saving or loading a file, be sure to specify the drive and filename, i.e., A:filename. It is important to note that loading a music file will erase any music in memory at that time. After loading a file, you may play it by pressing the "P" key as usual. When you load a file, you are positioned at the first empty location after the music, so you can edit or add to music you've stored on disk. Just remember to save the edited version.

Editing

If you have entered some notes into the *PC Blues Box*, you may wish to alter some of them. To alter a note, you must position yourself at its location in the music memory. This is quite easy — just use the "N" key to move to the precise location you want, entering the number of the note you wish to edit, or use the F9 and F10 keys to step through the notes one at a time. As an alternative, you can use "N" to move to the general location of the note you want to change, and use F9 and F10 to position yourself precisely. When you move onto a previously entered note, its characteristics are extracted from memory, and appear on the musical staff, on the piano keyboard, on the rest/note indicator, and in the information square. At this point, you may change any of these characteristics. When you hit RETURN, the current information about the note replaces the old information at that position. There is one important thing to remember. The tempo, which is stored as a characteristic of each note, is not read back during editing. In order to change the tempo of a piece, simply return to the beginning and go back through it, hitting the RETURN key repeatedly. Each time you hit RETURN, the note is accepted, and you automatically move to the next note.

Next month, we will be adding provisions allowing you to generate MERGE files so you can include music in your own programs, block-repeat editing functions which will

keep you from having to continually re-type repetitive musical passages, and the ability to play only certain parts of the music stored in memory. We will also consider including features suggested to us by our readers. Next month's *PC/Side* will also have complete *SWAT* tables for the *PC Blues Box.*

One special note about the F10 key: it lets you move to the empty position just after the last entered note. This is, of course, where new notes are added. When you move to this location with F10, the same defaults are set as those at the beginning of the program: middle C, quarter note, tempo 100. If you want to continue with the parameters of the last entered note, move to that last note with F9, then re-enter it by hitting RETURN. This will place you at the first unused position, but will preserve the characteristics of the last entered note. This may make it easier to continue adding to a piece which you've recalled from diskette.

Variables

AC$: When equal to " ", "#", or "-", indicates a natural, sharp, or flat.
BCOUNT: Block counter. Reserved for future use.
BK(i): Array of screen positions for the black piano keys.
BLOCK(i,j): Block pointer stack. Reserved for future use.
BPOS: Current block occupied. Reserved for future use.
C7$: Temporary variable for reading the tone from memory.
CN$: The current note, without accidental.
DOT(i): Array containing the shape table for the dot on dotted notes.
DOTTED: Boolean flag denoting whether the current note is dotted.
F$: File name for save and load.
FALSE: Boolean constant equal to zero.
FL7: Temporary Boolean flag for reading the flatness of a note from memory.
FLAT: Boolean flag denoting the flatness of the current note.
FLAT(i): Array containing the shape table for the musical flat symbol.
FNL2!: Function that returns the logarithm to the base 2 of its argument.
FNS$: Function that strips the first character from a string. Used to remove the leading space on numbers.
FNU$: Function that returns the upper case version of a letter.
IN: ASCII code of IN$.
IN$: The key the user strikes.
INSERTING: Boolean flag that determines whether a block is being inserted. Reserved for future use.
KBX,KBY: Screen coordinates of the piano keyboard.
LZ: The number of leading zeroes in a number to be inserted into an element of M$(i).
M$(i): Array of stored notes. Its elements are the arguments to the PLAY command.
M$: Temporary variable used in constructing elements of M$(i).
MAX: The number of elements of M$(i).
MIDC: Offset from STAFFY for position of the middle-C line on the staff.
N$: The full name of the current note, for example, "C#". This string is always two characters long.
N2$: Same as N$, but with "b" replacing "-" as the symbol for a flatted note. Used in the information square.
N7$: Temporary version of N$ used in reading a note back from memory.
COUNT: The number of the first empty position in M$(i).
NCURS(i): Contains the shape table for the highlight cursor on the time indicator.
NDX: X coordinate at which to draw a note on the staff.
NDY(i): Array of y coordinates for drawing notes on the staff.
NN: Pointer into NDY(i).
NOTE: Boolean flag that determines whether to use notes or rests.
NOTE1(i) to NOTE64(i): Contain shape tables for the notes of the corresponding lengths.
NX,NY: Screen coordinates at which to draw the whole note on the time indicator.
OC: The current octave number.
OC7: Temporary octave number used to read a note back from M$(i).
OLDTX: Previous x coordinate on the time indicator; used to erase the highlight cursor.
PCC(i),PCL(i),PCR(i): Shape tables used to highlight and un-highlight the white keys.
PIANOCC(i),PIANOCL(i), PIANOCR(i): Shape tables to initially draw the white keys.
PREVDOT: Previous value of DOTTED; used to determine whether the dot should be erased.
PREVNOTE: Previous value of NOTE; determines whether to erase a note or a rest.
PREVPTR: Previous value of PTNPTR; determines precise note to be erased.
PREVTIME: Previous value of time; determines the length of the note or rest to be erased.
PSCALES$: Patterned scale that determines PTNPTR.
PT$: Temporary string used to build PTN$.
PTN$: Long, patterned string from which N$ is extracted.
PTNPTR: Pattern pointer. Points to the part of PTN$ from which to extract the name of a note.
RDX,RDY: Offsets from STAFFX,STAFFY for drawing rests on the staff.
REST1(i) to REST64(i): Shape tables for rests.
RX,RY: Screen coordinates for the whole rest on the time indicator.
SCALE$: Scale that determines the value of NN.
SH7: Temporary Boolean sharpness flag.
SHARP: Boolean flag for the sharpness of the current note.
SHARP(i): Shape table for the sharp symbol drawn on the staff.
STAFF(i): Shape table for the staff itself.
STAFF$: DRAW commands used to create the staff shape.
STAFFX,STAFFY: Screen coordinates for the staff.
TEMPO: The actual tempo value.
TIME: A power of two from 1 to 64, representing the length of the current note or rest.
TRUE: Boolean constant, equal to -1.
TX: Current x coordinate for the time indicator highlight.
X: General loop and work variable.
X$: A PLAY string. Used to generate the scales played at the start of the program.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                           SS
SS   IBM PC  ADVANCED BASIC   SS
SS       'PC Blues Box'      SS
SS          Authors:         SS
SS Fred J. Condo/Alan J. Zett SS
SS     Copyright (c) 1982    SS
SS SoftSide Publications, Inc SS
SS                           SS
SS SS SS SS SS SS SS SS SS SS SS
```

Initialize, and set start-up defaults.

```
10 CLS:KEY OFF:DEFINT A-Z:MAX=2000:DIM N
CURS(38),SHARP(38),FLAT(38),STAFF(1884),
NOTE1(38),NOTE2(38),NOTE4(38),NOTE8(38),
NOTE16(38),NOTE32(38),NOTE64(38),BLOCK(9
9,2),M$(MAX),PCL(104),PCC(104),PCR(104)
50 DIM REST1(38),REST2(38),REST4(38),RES
T8(38),REST16(38),REST32(38),REST64(38),
PIANOL(104),PIANOR(104),PIANOC(104),DOT(
38),NDY(35),BK(5):DEF FNU$(A$)=CHR$(ASC(
LEFT$(A$,1))+32*(LEFT$(A$,1)>"Z"))
70 DEF FNL2!(X)=LOG(X)/LOG(2):DEF FNS$(N
$)=RIGHT$(N$,LEN(N$)-1):TRUE=-1:FALSE=0:
FLAT=FALSE:SHARP=FALSE:OC=3:CN$="C":SCAL
E$="CDEFGAB":NN=7*OC-6
90 STAFFX=5:STAFFY=55:PTN$="":X$="C C#D
E-E F F#G A-A B-B ":FOR X=1 TO 5:PTN$=PT
N$+X$:NEXT:PSCALE$="C.D.EF.G.A.B":PTNPTR
=(12*(OC-1)+INSTR(PSCALE$,CN$)-SHARP+FLA
T)*2-1
100 KBX=123:KBY=6:NX=6:NY=160:RX=6:RY=18
0:MIDC=30:NDX=STAFFX+49:RDX=NDX-2:RDY=ST
AFFY
110 NCOUNT=0:TEMPO=100:BCOUNT=0:DOTTED=F
ALSE:NOTE=TRUE:PREVNOTE=NOTE:TIME=4:PREV
TIME=TIME:PREVDOT=DOTTED:PREVPTR=PTNPTR:
BCOUNT=0:BPOS=0:NCOUNT=1:NPOS=1:INSERTIN
G=FALSE:GOSUB 13000:CLS:X$="Just a momen
t...":LOCATE 12,20-LEN(X$)/2:PRINT X$
140 GOSUB 16170:GOSUB 2000:GOSUB 15000:G
OSUB 14000
```

Main Input Loop: Accept valid characters; reject invalid ones; branch to appropriate subroutines.

```
1000 IN$=INKEY$:IF IN$="" THEN 1000
1010 IF FNU$(IN$)="S" THEN GOSUB 18000:G
OTO 1150
1020 IF FNU$(IN$)="L" THEN GOSUB 17000:G
OTO 1150
1030 IF FNU$(IN$)="P" THEN GOSUB 19000:G
OTO 1150
1040 IF FNU$(IN$)="T" THEN GOSUB 20000:G
OTO 1150
1045 IF FNU$(IN$)=" " THEN GOSUB 11000:N
OTE=NOT NOTE:GOTO 1150
1050 IF FNU$(IN$)="N" THEN GOSUB 21000:G
OTO 1150
1055 IF FNU$(IN$)="C" THEN GOSUB 22000:G
OTO 1150
1060 IF IN$=CHR$(13) THEN GOSUB 16000:GO
TO 1150
1065 IF IN$=CHR$(3) THEN GOSUB 23000:GOT
O 1150
1070 IF ASC(LEFT$(IN$,1))<>0 THEN 1000 E
LSE IN=ASC(RIGHT$(IN$,1))
1080 IF IN<59 OR (IN>68 AND IN<72) OR IN
=73 OR IN=79 OR IN>80 THEN 1000
1090 IF IN=72 THEN GOSUB 3000:GOTO 1150
1100 IF IN=75 THEN GOSUB 6000:GOTO 1150
1110 IF IN=77 THEN GOSUB 5000:GOTO 1150
1120 IF IN=80 THEN GOSUB 4000:GOTO 1150
1140  ON IN-58 GOSUB 8000,7000,50000,500
00,50000,50000,50000,50000,9000,10000
1150 GOSUB 2000
1160 GOSUB 12000:GOTO 1000
```

Calculation of new values for OC, SHARP, FLAT, NN, N$, and AC$. The value of PTNPTR is the sole parameter of this routine.

```
2000 N$=MID$(PTN$,PTNPTR,2):CN$=LEFT$(N$
,1):AC$=RIGHT$(N$,1):IF AC$=" " THEN SHA
RP=FALSE:FLAT=FALSE
2030 IF AC$="#" THEN SHARP=TRUE:FLAT=FAL
SE
2040 IF AC$="-" THEN SHARP=FALSE:FLAT=TR
UE
2060 OC=((PTNPTR-1)/2-INSTR(PSCALE$,CN$)
)/12+1:NN=7*OC+INSTR(SCALE$,CN$)-7:RETUR
N
```

Increment pitch of note one half-step.

```
3000 GOSUB 11000:PTNPTR=PTNPTR+2:IF PTNP
TR>119 THEN PTNPTR=1
3010 RETURN
```

Decrement pitch one half-step.

```
4000 GOSUB 11000:PTNPTR=PTNPTR-2:IF PTNP
TR<1 THEN PTNPTR=119
4010 RETURN
```

Move time indicator to the right. This shortens the length of the note or rest.

```
5000 GOSUB 11000:IF NOTE THEN 5050
5020 IF TIME=64 THEN TIME=1 ELSE TIME=TI
ME*2
5030 RETURN
5050 IF DOTTED THEN DOTTED=FALSE:RETURN
5060 DOTTED=TRUE:IF TIME=64 THEN TIME=1
ELSE TIME=TIME*2
5070 RETURN
```

Move time indicator to the left.

```
6000 GOSUB 11000:IF NOTE THEN 6040
6010 IF TIME=1 THEN TIME=64 ELSE TIME=TI
ME/2
6020 RETURN
6040 IF NOT DOTTED THEN DOTTED=TRUE:RETU
RN
6050 DOTTED=FALSE:IF TIME=1 THEN TIME=64
 ELSE TIME=TIME/2
6060 RETURN
```

Move up an octave.

```
7000 GOSUB 11000:PTNPTR=PTNPTR+24:IF PTN
PTR>119 THEN PTNPTR=PTNPTR-120
7010 RETURN
```

Move down an octave.

```
8000 GOSUB 11000:PTNPTR=PTNPTR-24:IF PTN
PTR<1 THEN PTNPTR=PTNPTR+120
8010 RETURN:GOSUB 11000:IF NPOS=1 THEN R
ETURN
```

Move to the previous note in the buffer.

```
9000 GOSUB 11000:IF NPOS=1 THEN RETURN
9010 NPOS=NPOS-1:IF BPOS>1 THEN IF NPOS=
BLOCK(BPOS-1,2) THEN BPOS=BPOS-1
9020 C7$=MID$(M$(NPOS),10,2):N7$=LEFT$(C
7$,1):OC7=VAL(MID$(M$(NPOS),6,1)):DOTTED
=(RIGHT$(M$(NPOS),1)="."):TIME=VAL(MID$(
M$(NPOS),8,2)):NOTE=NOT (ASC(N7$)=80)
9060 SH7=(RIGHT$(C7$,1)="#"):FL7=(RIGHT$
(C7$,1)="-"):IF NOTE THEN PTNPTR=(12*(OC
7-1)+INSTR(PSCALE$,N7$)-SH7+FL7)*2-1:RET
URN ELSE PTNPTR=49:RETURN
```

Move to the next note in the buffer.

```
10000 GOSUB 11000:IF NPOS>=NCOUNT-1 THEN
 NPOS=NCOUNT:PTNPTR=49:TIME=4:DOTTED=FAL
SE:NOTE=TRUE:RETURN
10010 NPOS=NPOS+1:IF NPOS=BLOCK(BPOS+1,1
) THEN BPOS=BPOS+1
10020 C7$=MID$(M$(NPOS),10,2):N7$=LEFT$(
C7$,1):OC7=VAL(MID$(M$(NPOS),6,1)):DOTTE
D=(RIGHT$(M$(NPOS),1)="."):TIME=VAL(MID$
(M$(NPOS),8,2)):NOTE=NOT (ASC(N7$)=80)
10060 SH7=(RIGHT$(C7$,1)="#"):FL7=(RIGHT
$(C7$,1)="-"):IF NOTE THEN PTNPTR=(12*(O
C7-1)+INSTR(PSCALE$,N7$)-SH7+FL7)*2-1:RE
TURN ELSE PTNPTR=49:RETURN
```

Save the previous state of the screen for proper erasure of symbols.

```
11000 PREVTIME=TIME:PREVDOT=DOTTED:PREVN
OTE=NOTE:PREVPTR=PTNPTR:RETURN
```

Display routine. Re-create the data of the previous note, using the information preserved in line 11000.

```
12000 PREVN$=MID$(PTN$,PREVPTR,2):PREVAC
$=RIGHT$(PREVN$,1):PREVOC=((PREVPTR-1)/2
-INSTR(PSCALE$,LEFT$(PREVN$,1)))/12+1:PR
EVNN=7*PREVOC+INSTR(SCALE$,LEFT$(PREVN$,
1))-7:N2$=N$:IF AC$="-" THEN MID$(N2$,2,
1)="b"
```

Update the information square.

```
12050 LOCATE 3,31:IF DOTTED THEN PRINT U
SING "Tone  \ \";N2$+"." ELSE PRINT USIN
G "Tone   \\";N2$
12060 LOCATE 2,31:PRINT USING "Octave  #
";OC:LOCATE 6,31:PRINT USING "Note ####"
;NPOS:LOCATE 5,31:PRINT USING "Tempo ###
";TEMPO:LOCATE 7,31:PRINT USING "Blocks
##";BCOUNT:LOCATE 4,31:PRINT USING "Leng
th ##";TIME
```

Update the time indicator.

```
12100 OLDTX=FNL2!(PREVTIME)*15+RX:IF PRE
VNOTE AND PREVDOT THEN PUT (OLDTX,NY),DO
T
12110 PUT (OLDTX,RY+20*PREVNOTE),NCURS:T
X=RX+15*FNL2!(TIME):PUT (TX,RY+20*NOTE),
NCURS:IF NOTE AND DOTTED THEN PUT (TX,RY
+20*NOTE),DOT
```

Update the piano keys.

```
12160 X=INSTR(SCALE$,LEFT$(PREVN$,1)):XQ
=INT(X-X/3+.5):IF PREVAC$<>" " THEN PAIN
T (KBX+BK(XQ),KBY+5),0,2 ELSE IF X=2 OR
X=5 OR X=6 THEN PUT (KBX+15*(X-1),KBY),P
CC ELSE IF X=1 OR X=4 THEN PUT (KBX+15*(
X-1),KBY),PCL ELSE PUT (KBX+15*(X-1),KBY
),PCR
12180 X=INSTR(SCALE$,LEFT$(N$,1)):XQ=INT
(X-X/3+.5):IF AC$<>" " THEN PAINT (KBX+B
K(XQ),KBY+5),1,2 ELSE IF X=2 OR X=5 OR X
=6 THEN PUT (KBX+15*(X-1),KBY),PCC ELSE
IF X=1 OR X=4 THEN PUT (KBX+15*(X-1),KBY
),PCL ELSE PUT (KBX+15*(X-1),KBY),PCR
12190 LINE (0,0)-(319,199),3,B
```

Update the staff.

```
12210 IF NOT PREVNOTE THEN 12440
12220 ON FNL2!(PREVTIME)+1 GOTO 12230,12
240,12250,12260,12270,12280,12290
12230 PUT (NDX,NDY(PREVNN)),NOTE1:GOTO 1
2300
12240 PUT (NDX,NDY(PREVNN)),NOTE2:GOTO 1
2300
12250 PUT (NDX,NDY(PREVNN)),NOTE4:GOTO 1
2300
12260 PUT (NDX,NDY(PREVNN)),NOTE8:GOTO 1
2300
12270 PUT (NDX,NDY(PREVNN)),NOTE16:GOTO
12300
12280 PUT (NDX,NDY(PREVNN)),NOTE32:GOTO
12300
12290 PUT (NDX,NDY(PREVNN)),NOTE64
12300 IF PREVDOT THEN PUT (NDX,NDY(PREVN
N)),DOT
12310 IF PREVAC$="-" THEN PUT (NDX-2,NDY
(PREVNN)),FLAT ELSE IF PREVAC$="#" THEN
PUT (NDX-2,NDY(PREVNN)),SHARP
12320 IF NOT NOTE THEN 12520 ELSE ON FNL
2!(TIME)+1 GOTO 12330,12340,12350,12360,
12370,12380,12390
12330 PUT (NDX,NDY(NN)),NOTE1:GOTO 12400
12340 PUT (NDX,NDY(NN)),NOTE2:GOTO 12400
12350 PUT (NDX,NDY(NN)),NOTE4:GOTO 12400
12360 PUT (NDX,NDY(NN)),NOTE8:GOTO 12400
12370 PUT (NDX,NDY(NN)),NOTE16:GOTO 1240
0
12380 PUT (NDX,NDY(NN)),NOTE32:GOTO 1240
0
12390 PUT (NDX,NDY(NN)),NOTE64
12400 IF DOTTED THEN PUT (NDX,NDY(NN)),D
OT
12410 IF AC$="-" THEN PUT (NDX-2,NDY(NN)
),FLAT ELSE IF AC$="#" THEN PUT (NDX-2,N
DY(NN)),SHARP
12420 RETURN
12440 IF PREVNOTE THEN 12320 ELSE ON FNL
2!(PREVTIME)+1 GOTO 12450,12460,12470,12
480,12490,12500,12510
12450 PUT (RDX,RDY),REST1:GOTO 12320
12460 PUT (RDX,RDY),REST2:GOTO 12320
12470 PUT (RDX,RDY),REST4:GOTO 12320
12480 PUT (RDX,RDY),REST8:GOTO 12320
12490 PUT (RDX,RDY),REST16:GOTO 12320
12500 PUT (RDX,RDY),REST32:GOTO 12320
12510 PUT (RDX,RDY),REST64:GOTO 12320
12520 ON FNL2!(TIME)+1 GOTO 12530,12540,
12550,12560,12570,12580,12590
12530 PUT (RDX,RDY),REST1:RETURN
12540 PUT (RDX,RDY),REST2:RETURN
12550 PUT (RDX,RDY),REST4:RETURN
12560 PUT (RDX,RDY),REST8:RETURN
12570 PUT (RDX,RDY),REST16:RETURN
12580 PUT (RDX,RDY),REST32:RETURN
12590 PUT (RDX,RDY),REST64:RETURN
```

Set up the shape tables.

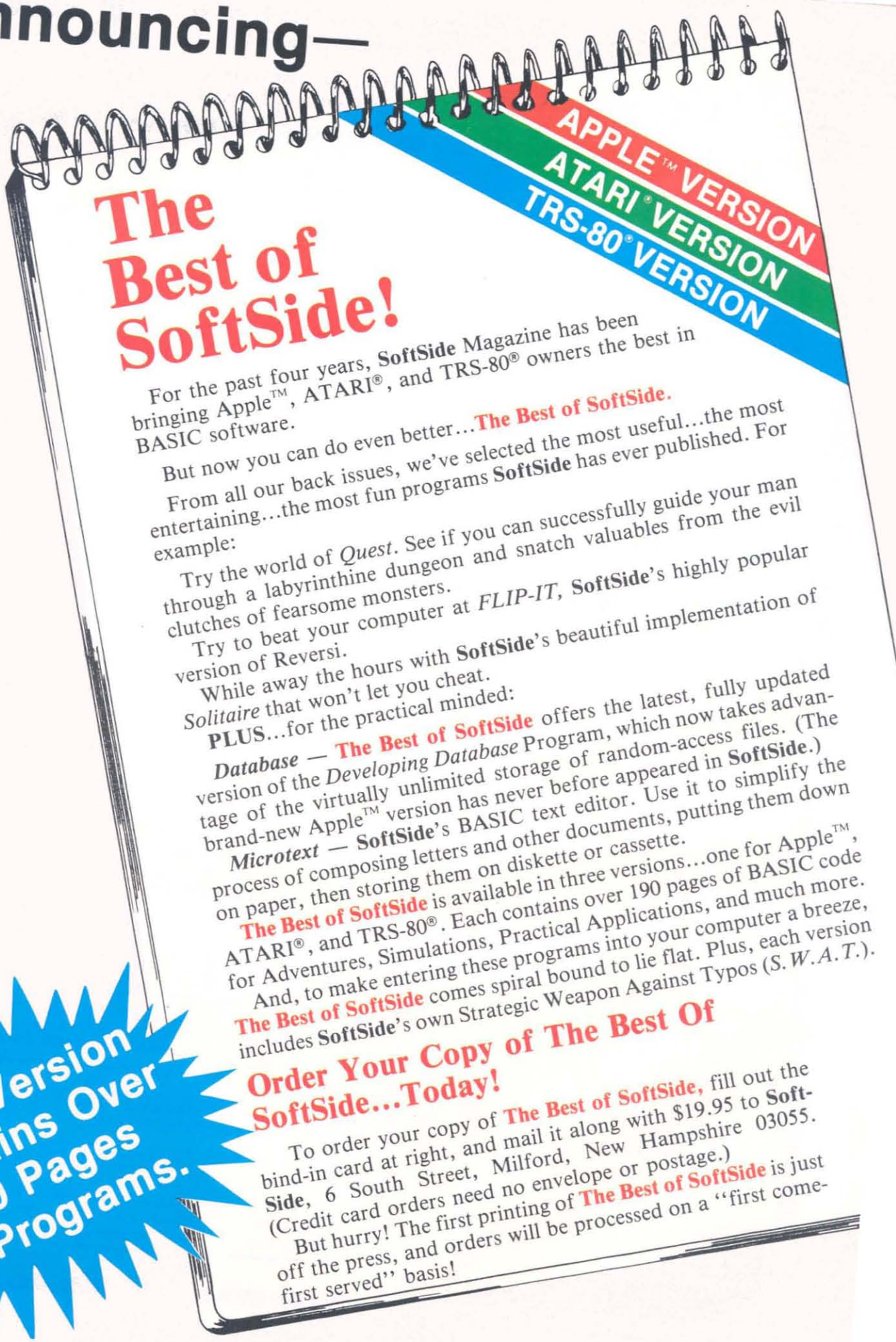
```
13000 SCREEN 1:CLS:COLOR 0,1:STAFF$="S10
 A000 BM000,000 C1 D8 R1 U8 L1 R1 BR40 D
8 R1 U8 L1 L40 D2 R40 D2 L40 D2 R40 D2 L
40":STAFF$=STAFF$+" L1 D16 R1 U8 L1 R1 B
R40 D8 R1 U8 L1 L40 D2 R40 D2 L40 D2 R40
 D2 L40":DRAW STAFF$:STAFF$=""
13030 GET (0,0)-(104,70),STAFF:CLS
13050 CIRCLE (4,13),3,2,,,.55:GET (0,0)-
(13,17),NOTE1:LINE (7,13)-(7,1),2:GET (0
,0)-(13,17),NOTE2:PAINT (4,13),2,2:GET (
0,0)-(13,17),NOTE4:LINE (7,1)-(12,3),2:G
ET (0,0)-(13,17),NOTE8:LINE (7,3)-(12,5)
,2:GET (0,0)-(13,17),NOTE16
13060 LINE (7,5)-(12,7),2:GET (0,0)-(13,
17),NOTE32:LINE (7,7)-(12,9),2:GET (0,0)
-(13,17),NOTE64:CLS
13080 LINE(2,0)-(2,6),2:LINE (4,0)-(4,6)
,2:LINE (1,2)-(5,2),2:LINE (1,4)-(5,4),2
:GET (0,0)-(13,17),SHARP:CLS:LINE (2,0)-
(2,6),2:LINE (2,6)-(6,4),2:LINE (6,4)-(2
,2),2:GET (0,0)-(13,17),FLAT
13100 CLS:LINE (3,11)-(9,13),2,BF:GET (0
,0)-(13,17),REST1:CLS:LINE (3,9)-(9,7),2
,BF:GET (0,0)-(13,17),REST2:CLS:LINE (6,
3)-(8,5),2:LINE (8,5)-(6,8),2:LINE (6,8)
-(8,10),2:LINE (8,10)-(5,13),2:LINE (5,1
3)-(6,15),2:GET (0,0)-(13,17),REST4
13110 CLS:LINE (6,14)-(9,3),2:LINE (9,4)
-(3,6),2:GET (0,0)-(13,17),REST8:LINE (9
,6)-(3,8),2:GET (0,0)-(13,17),REST16:LIN
E (7,8)-(3,10),2:GET (0,0)-(8,16),REST32
:LINE (7,10)-(3,12),2:GET (0,0)-(13,17),
REST64
13130 CLS:LINE (0,0)-(9,30),3,BF:LINE (0
,30)-(12,50),3,BF:GET (0,0)-(12,50),PIAN
OL
13131 PAINT (5,5),2,0:GET (0,0)-(12,50),
PCL
13132 CLS:LINE (4,0)-(9,30),3,BF:LINE(0,
30)-(12,50),3,BF:GET (0,0)-(12,50),PIANO
C
13133 PAINT (5,5),2,0:GET (0,0)-(12,50),
PCC
13134 CLS:LINE (4,0)-(12,30),3,BF:LINE (
0,30)-(12,50),3,BF:GET (0,0)-(12,50),PIA
NOR
13135 PAINT (5,5),2,0:GET (0,0)-(12,50),
PCR
13140 CLS:LINE (0,0)-(13,17),2,BF:GET (0
,0)-(13,17),NCURS:CLS:PSET (10,14),2:PSE
T (10,15),2:PSET (11,14),2:PSET (11,15),
2:GET (0,0)-(13,17),DOT:RETURN
```

Draw the initial screen.

```
14000 CLS:LINE (0,0)-(319,199),3,B:LINE
(115,0)-(115,199),3:LINE (115,63)-(319,1
99),3,B:LINE (234,0)-(234,63),3
14030 PUT (KBX,KBY),PIANOL:PUT (KBX+15,K
BY),PIANOC:PUT (KBX+30,KBY),PIANOR:PUT (
KBX+45,KBY),PIANOL:PUT (KBX+60,KBY),PIAN
OC:PUT (KBX+75,KBY),PIANOC:PUT (KBX+90,K
BY),PIANOR:LINE (KBX-3,KBY-2)-(KBX+105,K
```

continued on page 33

Announcing—



The Best of SoftSide!

For the past four years, **SoftSide** Magazine has been bringing Apple™, ATARI®, and TRS-80® owners the best in BASIC software.

But now you can do even better…**The Best of SoftSide.**

From all our back issues, we've selected the most useful…the most entertaining…the most fun programs **SoftSide** has ever published. For example:

Try the world of *Quest*. See if you can successfully guide your man through a labyrinthine dungeon and snatch valuables from the evil clutches of fearsome monsters.

Try to beat your computer at *FLIP-IT*, **SoftSide**'s highly popular version of Reversi.

While away the hours with **SoftSide**'s beautiful implementation of *Solitaire* that won't let you cheat.

PLUS…for the practical minded:

Database **— The Best of SoftSide** offers the latest, fully updated version of the *Developing Database* Program, which now takes advantage of the virtually unlimited storage of random-access files. (The brand-new Apple™ version has never before appeared in **SoftSide**.)

Microtext **— SoftSide**'s BASIC text editor. Use it to simplify the process of composing letters and other documents, putting them down on paper, then storing them on diskette or cassette.

The Best of SoftSide is available in three versions…one for Apple™, ATARI®, and TRS-80®. Each contains over 190 pages of BASIC code for Adventures, Simulations, Practical Applications, and much more. And, to make entering these programs into your computer a breeze, **The Best of SoftSide** comes spiral bound to lie flat. Plus, each version includes **SoftSide**'s own Strategic Weapon Against Typos (*S.W.A.T.*).

Order Your Copy of The Best Of SoftSide…Today!

To order your copy of **The Best of SoftSide,** fill out the bind-in card at right, and mail it along with $19.95 to **SoftSide**, 6 South Street, Milford, New Hampshire 03055. (Credit card orders need no envelope or postage.)

But hurry! The first printing of **The Best of SoftSide** is just off the press, and orders will be processed on a "first come-first served" basis!

Each Version Contains Over 190 Pages Of Programs.

SAVE 33 1/3 %

ORDER TODAY!

Use this Postage-Paid card to order the next 12 issues of *SoftSide* Magazine **AND SAVE $12.00 OFF THE NEWSSTAND PRICE**

SUBSCRIBE TO SoftSide Magazine

Please enroll me as a ***SoftSide*** subscriber.
☐ 6 issues for $15 ☐ 12 issues for $24
I own an ☐ Apple™ ☐ IBM®-PC ☐ ATARI® ☐ TRS-80®

Name _____
Address _____
City/State _____ Zip _____

☐ Check or Money Order (Please enclose this card and your payment in an envelope.)

☐ **MasterCard** ☐ **VISA**
Name of Cardholder _____
MC# and Interbank#/VISA# _____
Exp. Date _____ Signature _____

SAVE $12.00 OFF THE NEWSSTAND PRICE

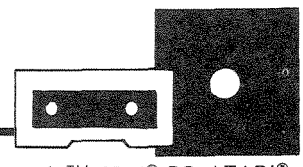
Prices for USA only. For foreign orders, see page 80. Please include exact postage on all foreign orders.

Prices subject to change without notice. Apple™, IBM®, ATARI® and TRS-80® are registered trademarks of The Apple Computer Company, International Business Machines, Inc., Warner Communications, and The Tandy Corporation respectively.

For Even Faster Service

Call Toll-Free
1-800-345-8112
(in PA 1-800-662-2444)
MASTERCARD AND VISA ONLY
To Order SoftSide

SUBSCRIBE TO SoftSide on Disk or Cassette



With ***SoftSide*** CV or DV you'll get the magazine plus each month's Apple™, IBM®-PC, ATARI®, or TRS-80® programs delivered on cassette or disk. All programs are tested and ready to run the day you receive them!

6 Issues	12 Issues
☐ *SoftSide* CV — $40	☐ *SoftSide* CV — $75
☐ *SoftSide* DV — $70	☐ *SoftSide* DV — $125

☐ 12 issues ***SoftSide*** DV — Four monthly payments of $32.50 (includes $5 billing charge).

Which computer? ☐ Apple™ ☐ IBM®-PC **(Disk Only)**
☐ ATARI® ☐ TRS-80® Mod. I ☐ TRS-80® Mod. III

Name _____
Address _____
City/State _____ Zip _____

Prices for USA only. For foreign orders, see page 80. Please include exact postage on all foreign orders.

☐ Check or Money Order (Please enclose this card and payment in an envelope.)

☐ **MasterCard** ☐ **VISA**
Name of Cardholder _____
MC# and Interbank#/VISA# _____
Exp. Date _____
Signature _____

☐ Convert the remainder of my current ***SoftSide*** subscription to ☐ CV ☐ DV. I understand that I'll be billed $4.25 per cassette or $8.42 per disk for remaining issues.
Please allow 4-6 weeks for conversion.

Prices subject to change without notice. Apple™, IBM®, ATARI® and TRS-80® are registered trademarks of The Apple Computer Company, International Business Machines, Inc., Warner Communications, and The Tandy Corporation respectively.

Over 190 Pages Of The Best Programs From SoftSide's Past...

In One Spiral Bound Book!

The Best of SoftSide

☐ **YES...**I want to enjoy the best programs — Games, Utilities, Adventures — from SoftSide Magazine's past. Please rush my order for The Best of SoftSide to:

ONLY $19.95 EACH

Name _____
Address _____
City _____ State _____ Zip _____

Here's my order For The Best of SoftSide

NUMBER OF COPIES	PRICE	SYSTEM
	$19.95 EACH	APPLE™ VERSION (over 190 pgs. of programs)
	$19.95 EACH	ATARI® VERSION (over 190 pgs. of Programs)
	$19.95 EACH	TRS-80® VERSION (over 190 pgs. of Programs)
TOTAL COPIES	**TOTAL ENCLOSED $**	PAY BY ☐ Check, ☐ M.O. ☐ VISA ☐ MasterCard (Foriegn order please include $5 postage. PAY IN USA FUNDS.)

VISA/MasterCard# _____ Exp. Date _____
Name of Cardholder _____
Signature _____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 36 MILFORD, NEW HAMPSHIRE 03055

POSTAGE WILL BE PAID BY ADDRESSEE

SoftSide™

100 Pine Street
Holmes, PA 19043

SS34

SAVE 33⅓%

ORDER TODAY!

Use this
Postage-Paid card to order
the next 12 issues of
SoftSide Magazine
AND SAVE $12.00 OFF
THE NEWSSTAND PRICE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 36 MILFORD, NEW HAMPSHIRE 03055

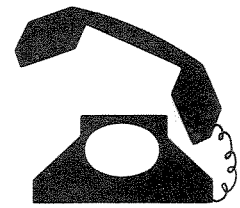
POSTAGE WILL BE PAID BY ADDRESSEE

SoftSide™

100 Pine Street
Holmes, PA 19043

SS34

For Even Faster Service



Call Toll-Free

1-800-345-8112

(in PA 1-800-662-2444)

MASTERCARD AND VISA ONLY

To Order

SoftSide™

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 36 MILFORD, NEW HAMPSHIRE 03055

POSTAGE WILL BE PAID BY ADDRESSEE

SoftSide™

6 South Street
Milford, NH 03055

SS34

Over 190 Pages Of The Best Programs From SoftSide's Past... In One Spiral Bound Book!

```
BY+52),3,B:PAINT (KBX+11,KBY+1),2,3
14050 FOR X=KBX+10 TO KBX+25 STEP 15:LIN
E (X+1,KBY)-(X+7,KBY+28),0,BF:NEXT X:FOR
 X=KBX+55 TO KBX+55+30 STEP 15:LINE (X+1
,KBY)-(X+7,KBY+28),0,BF:NEXT X
14060 PUT (KBX,KBY),PCL
14070 LINE (0,NY-2)-(115,NY-2),3:PUT (NX
,NY),NOTE1:PUT (NX+15,NY),NOTE2:PUT (NX+
30,NY),NOTE4:PUT (NX+45,NY),NOTE8:PUT (N
X+60,NY),NOTE16:PUT (NX+75,NY),NOTE32:PU
T (NX+90,NY),NOTE64:TX=RX+15*FNL2!(TIME)
:PUT (TX,NY),NCURS
14090 LINE (0,RY-2)-(115,RY-2),3:PUT (RX
,RY),REST1:PUT (RX+15,RY),REST2:PUT (RX+
30,RY),REST4:PUT (RX+45,RY),REST8:PUT (R
X+60,RY),REST16:PUT (RX+75,RY),REST32:PU
T (RX+90,RY),REST64:PUT (STAFFX,STAFFY),
STAFF
14120 FOR X=1 TO 5:LINE (STAFFX+46,STAFF
Y-X*5)-(STAFFX+60,STAFFY-X*5),1:LINE (ST
AFFX+46,STAFFY+X*5+45)-(STAFFX+60,STAFFY
+X*5+45),1:NEXT X:LINE (STAFFX+46,STAFFY
+MIDC)-(STAFFX+60,STAFFY+MIDC),1:PUT (ND
X,NDY(NN)),NOTE4
14160 GOSUB 12000:LOCATE 12,22:DEF SEG:P
OKE 78,1:PRINT"PC Blues Box":LOCATE 14,2
1:DEF SEG:POKE 78,2:PRINT "A Music Edito
r":LOCATE 15,21:PRINT "For The IBM PC"
14162 POKE 78,3:LOCATE 17,20:PRINT "by F
red J. Condo":LOCATE 18,20:PRINT "and Al
an J. Zett"
14165 X$="CDEFGAB":FOR X=0 TO 6:PLAY "MB
 O=X; T100 L32 XX$; XX$;":NEXT:FOR X=18
TO 12 STEP -1:LOCATE X,16:PRINT STRING$(
24,32):NEXT
14170 LOCATE 10,16:PRINT "F1 DOWN   Octa
ve   UP F2"
14190 LOCATE 11,16:PRINT "F3 START  Bloc
k   END F4"
14200 LOCATE 12,16:PRINT "F5 INSRT  Bloc
k   DEL F6"
14210 LOCATE 13,16:PRINT "F7 PREV.  Bloc
k  NEXT F8"
14220 LOCATE 14,16:PRINT "F9 PREV.  Note
   NEXT F0"
14230 LOCATE 15,16:PRINT "S  SAVE   File
   LOAD  L"
14240 LOCATE 16,16:PRINT "C  CLEAR  Musi
c  PLAY  P"
14250 LOCATE 17,16:PRINT "N  NOTE   Edit
  TEMPO  T"
14260 LOCATE 18,16:PRINT "[Esc]   Stop P
lay  [Esc]"
14265 LOCATE 19,16:PRINT "Ctrl-C    Quit
    Ctrl-C"
14270 RETURN
```

Set up the y coordinates for drawing notes and the x coordinates for drawing piano keys.

```
15000 RESTORE:FOR X=1 TO 35:READ NDY(X):
NDY(X)=NDY(X)+STAFFY:NEXT:FOR X=1 TO 5:R
EAD BK(X):NEXT:RETURN
15010 DATA 57,55,52,50,47,45,42,40,37,35
,32,30,27,25,17,10,7,5,2,0,-3,-5,-8,-10,
-13,-15,-18,-20,-23,-25,-28,-30,-33,-35,
-38,15,30,60,75,90
```

Accept a note into the buffer.

```
16000 GOSUB 11000:M$="T00000L00Na ":LZ=-
((TEMPO<10)+(TEMPO<100)):MID$(M$,2+LZ)=F
NS$(STR$(TEMPO)):MID$(M$,6)=FNS$(STR$(OC
)):LZ=-(TIME<10):MID$(M$,8+LZ)=FNS$(STR$
(TIME))
16060 IF NOT NOTE THEN MID$(M$,10)="P"+F
NS$(STR$(TIME)) ELSE MID$(M$,10)=N$:IF D
OTTED THEN MID$(M$,12)="."
16070 M$(NPOS)=M$:IF NOTE THEN PLAY M$
16090 NPOS=NPOS+1:IF NPOS>NCOUNT THEN NC
OUNT=NCOUNT+1
16110 IF NPOS=NCOUNT THEN RETURN
16120 C7$=MID$(M$(NPOS),10,2):N7$=LEFT$(
C7$,1):OC7=VAL(MID$(M$(NPOS),6,1)):DOTTE
D=(RIGHT$(M$(NPOS),1)="."):TIME=VAL(MID$
(M$(NPOS),8,2)):NOTE=NOT (ASC(N7$)=80)
16160 SH7=(RIGHT$(C7$,1)="#"):FL7=(RIGHT
$(C7$,1)="-"):IF NOTE THEN PTNPTR=(12*(O
C7-1)+INSTR(PSCALE$,N7$)-SH7+FL7)*2-1:RE
TURN ELSE PTNPTR=49:RETURN
```

Un-define the function keys.

```
16170 FOR X=1 TO 10:KEY X, "":NEXT X:RET
URN
```

File operations (Load and Save).

```
17000 GOSUB 11000:LOCATE 22,16:LINE INPU
T "File?";F$:LOCATE 22,16:PRINT STRING$(
24,32):OPEN F$ FOR INPUT AS #1:INPUT #1,
BCOUNT:INPUT #1,NCOUNT:IF BCOUNT<>0 THEN
 FOR X=1 TO BCOUNT:INPUT #1,BLOCK(X,1):I
NPUT #1,BLOCK(X,2):NEXT X
17060 BPOS=BCOUNT:FOR X=1 TO NCOUNT-1:IN
PUT #1,M$(X):NEXT X:CLOSE #1:NPOS=NCOUNT
:TIME=4:DOTTED=FALSE:NOTE=TRUE:PTNPTR=49
:RETURN
18000 GOSUB 11000:LOCATE 22,16:LINE INPU
T "File?";F$:LOCATE 22,16:PRINT STRING$(
24,32):OPEN F$ FOR OUTPUT AS #1:PRINT #1
,BCOUNT:PRINT #1,NCOUNT:IF BCOUNT<>0 THE
N FOR X=1 TO BCOUNT:PRINT #1,BLOCK(X,1):
PRINT #1,BLOCK(X,2):NEXT X
18060 BPOS=BCOUNT:FOR X=1 TO NCOUNT-1:PR
INT #1,M$(X):NEXT X:CLOSE #1:NPOS=NCOUNT
:TIME=4:DOTTED=FALSE:NOTE=TRUE:PTNPTR=49
:RETURN
```

Play the music!

```
19000 GOSUB 11000:PLAY "MB":FOR X=1 TO N
COUNT-1:IF INKEY$=CHR$(27) THEN RETURN E
LSE PLAY M$(X)
19020 NEXT:RETURN
```

Set tempo.

```
20000 GOSUB 11000:LOCATE 22,16:LINE INPU
T "Tempo (32-255)?";X$:TEMPO=VAL(X$):IF
TEMPO<32 OR TEMPO>255 THEN TEMPO=100
20010 LOCATE 22,16:PRINT STRING$(24,32):
RETURN
```

Move to a note to edit.

```
21000 GOSUB 11000:LOCATE 22,16:LINE INPU
T "Edit Which Note #?";X$:LOCATE 22,16:P
RINT STRING$(24,32):IF VAL(X$)<1 OR VAL(
X$)>=NCOUNT THEN RETURN
21010 NPOS=VAL(X$):C7$=MID$(M$(NPOS),10,
2):N7$=LEFT$(C7$,1):OC7=VAL(MID$(M$(NPOS
),6,1)):DOTTED=(RIGHT$(M$(NPOS),1)="."):
TIME=VAL(MID$(M$(NPOS),8,2)):NOTE=NOT (A
SC(N7$)=80)
21060 SH7=(RIGHT$(C7$,1)="#"):FL7=(RIGHT
$(C7$,1)="-"):IF NOTE THEN PTNPTR=(12*(O
C7-1)+INSTR(PSCALE$,N7$)-SH7+FL7)*2-1:RE
TURN ELSE PTNPTR=49:RETURN
```

Clear the music buffer.

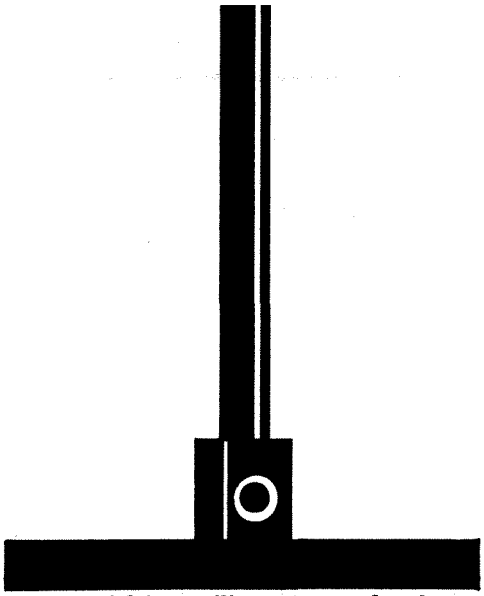
```
22000 GOSUB 11000:LOCATE 22,16:LINE INPU
T "Sure? (Y/N)";X$:LOCATE 22,16:PRINT ST
RING$(24,32)
22010 IF X$="" THEN RETURN
22020 IF FNU$(X$)<>"Y" THEN RETURN
22030 BCOUNT=0:NCOUNT=1:NPOS=1:TIME=4:NO
TE=TRUE:DOTTED=FALSE:PTNPTR=49:TIME=4:RE
TURN
```

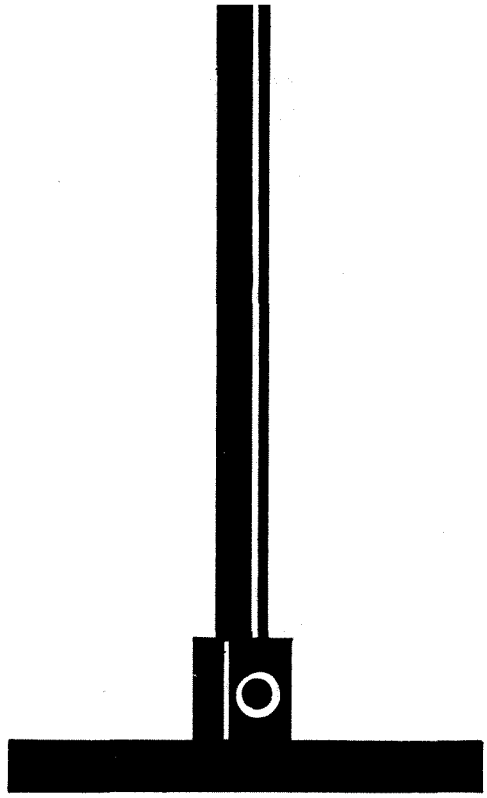
Quit.

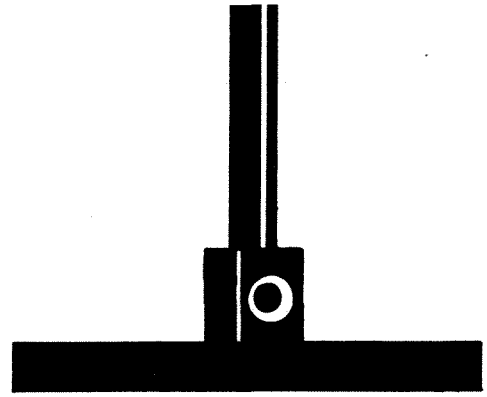
```
23000 LOCATE 22,16:LINE INPUT "Sure? (Y/
N)";X$:LOCATE 22,16:PRINT STRING$(24,32)
:IF X$="" THEN RETURN
23010 IF FNU$(LEFT$(X$,1))<>"Y" THEN RET
URN ELSE SCREEN 0:WIDTH 80:END
```

Temporary line to handle unimplemented commands.

```
50000 GOSUB 11000:LOCATE 22,16:PRINT "No
t yet implemented.":FOR X=1 TO 1000:NEXT
 X:LOCATE 22,16:PRINT STRING$(24,32):RET
URN
```







SQUISH

By Dave Archibald ■ IBM® translation by Alan J. Zett

***Squish* is a utility program for the IBM PC with 32K, color/graphics adapter, 1 disk drive, and BASIC.**

While working on the program *PC Blues Box* (see page 25), it became apparent that the finished code would need a little cleaning up. Programmers tend to use quite a few REMarks when programming, add extra spaces to make listings more readable, as well as putting only a few statements on each line. The result is a program that is easier to read and debug, but not very space or memory efficient. The ideal utility would remove extra spaces and REMs, and combine lines, when possible without altering the structure of the BASIC program.

Enter the TRS-80® program *Squish*. (*SoftSide*, December, 1980) Here was a program that would solve all of our problems if only it were translated to the IBM. We've done so, and it is presented here for the benefit of all our IBM subscribers. It may be of interest to note that TRS-80 Disk BASIC and IBM BASIC bear a startling resemblance to each other. Both were written by Microsoft (most microcomputer BASICs are). Most of the advanced graphics commands of IBM BASIC are surprisingly similar to the Microsoft Level III BASIC package for the TRS-80. All in all, IBM translations from the TRS-80 are a snap.

While translating, I came across several bugs in the original TRS-80 version. These bugs will be compiled for a future *Bugs, Worms and Other Undesirables* column. Other than the bugs, the only important revision was that IBM BASIC is fussy about spaces because it supports multi-character variable names. "GOTO100" and "GOTO 100" are two distinct processes. The first is a variable name (reserved words are allowed), and the second, an unconditional branch. The problem is solved by eliminating only extra spaces following certain punctuation. Originally, IBM *Squish* would remove all spaces that were not actually needed. This led to very messy listings when certain combinations of keywords and punctuation created a line that was literally *Squished* beyond all recognition. After punctuation was carefully selected, a trade-off was achieved between readability and compactness.

To use *Squish*, first LOAD the program you wish to pack. After it is LOADed, SAVE it with the ASCII option to a new file. For example: LOAD "MYPROG" and then reSAVE it as SAVE "TEST.TXT",A. After the program has been SAVEd with the ASCII option, LOAD in *Squish* and type RUN.

Answer the first question with the name of the file SAVEd with the ASCII option ("TEST.TXT" in our example above).

You will then be given a series of options. You can remove extra spaces, delete REMarks, and combine lines. The fourth option gives you the ability to protect a line or group of lines from being *Squished*. Lines entered under this option are totally ignored by *Squish*.

Squish will then read in the program to check for lines that are referenced and proceed to work on it. The display will update constantly to show you, in color, what *Squish* is doing, and where.

When *Squish* is done, it will allow you to enter the *Squished* program for examination. I suggest you then do a quick RENUMber and reSAVE the PROGRAM under another filename.

When you are finished, you will have a neatly RENUMbered program that is not only much shorter than the original (usually by a few K), but will execute faster. All in all, *Squish* is quite a useful utility.

Variables:

A: Temporary numeric storage.
AZ: Screen line on which current program line is displayed.
A$: Current program line being *Squished*.
C$: Contains lines that have been combined.
D: Length of a reserved word being searched for.

continued on page 36

SHOWS FOR YOU IN '82

THE NATIONAL COMPUTER SHOWS are the largest public computer expositions in the country. They feature hardware and software for business, industry, government, education, home, and personal use.

Under one roof you'll be able to test the new computers, desk top computers, data and word processing equipment, and a huge array of computer peripherals, computer services and computer supplies.

At each show you'll see all the major brands, all the major manufacturers, the big distributors and the local dealers and retailers. Everything together and for sale at super show prices.

Don't miss the National Computer Show coming to a city near you. Admission is $5 per person per day.

TICKET INFORMATION:
Send $5 (payable to National Computer Shows) for each day-long ticket along with the name of the Show you plan to attend to the address below. Tickets can also be purchased at the show.

The National Computer Shows,
822 Boylston St., Chestnut Hill, MA 02167
Telephone: 617-739-2000

BOSTON
THE NORTHEAST PERSONAL COMPUTER SHOW
Friday-Monday October 8-11 (Columbus Day Weekend)
Hynes Auditorium/Prudential Center
Show Hours: Friday 11 am to 9 pm,
Saturday & Sunday 12 noon to 9 pm,
Monday 11 am to 6 pm

WASHINGTON, D.C.
THE MID-ATLANTIC COMPUTER SHOW
Thursday-Sunday October 28-31
D.C. Armory/Starplex
Show Hours: 11 am to 6 pm daily

BOSTON
THE NORTHEAST BUSINESS COMPUTER SHOW
Thursday-Sunday November 11-14
Hynes Auditorium/Prudential Center
Show Hours: 11 am to 6 pm daily

ATLANTA
THE SOUTHEAST COMPUTER SHOW
Thursday-Sunday December 9-12
Atlanta Civic Center
Show Hours: 11 am to 6 pm daily

THE NATIONAL COMPUTER SHOWS

822 Boylston St., Chestnut Hill, MA 02167

IBM®-PC

DS: The maximum number of user protected lines.
DT: Set to 1 if current line contains a DATA statement.
G1-G6: A position in A$ at which INSTR starts searching.
HH: Miscellaneous.
I$: Name of a DEF FN to convert a string input into a single upper case letter.
IP$: Set to "Y" if REM statements are to be deleted.
J$: Storage for A$ as it is being reconstructed.
LN: Current line number being processed.
L$, L1$: Current character of A$ being processed.
N$: Set to STR$(LN).
P: Set to 1 the first time a quote is encountered in a PRINT statement. Set to zero when second quote or end of line is encountered. While P equals 1, all compression functions are cancelled.
PJ: Points to the current line in PRO(*) being tested.
PP: Set to 1 minus the location of the first character in a program line following the line number.
PV: Number of protected lines entered.
PRO(*): Optional protected line numbers specified by user.
Q$: Used in an INKEY$ loop.
R: Number of line numbers referenced in REF(*).
RD: Total number of REM statements deleted.
RE: Total number of lines combined.
REF(*): Number of program lines that can be referenced.
S, S1: Miscellaneous. Used in sort routine.
SD: Total number of spaces deleted.
SQ$: SAVE filename for *Squished* program.
SV$: SAVE filename for *Squished* program.
T, T1-T2: Miscellaneous.
V$: The current program line, stripped of its line number.
X: Set equal to PP.
XC$: Set to "Y" if lines are to be combined.
XS$: Set to "Y" if extra spaces are to be deleted.
XP$: Set to "Y" if user wishes to protect lines.
ZC: Position on line AZ to print L$.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                             SS
SS        IBM PC BASIC         SS
SS          'Squish'           SS
SS   Author: Dave Archibald    SS
SS Translation:  Alan J. Zett SS
SS    Copyright (c) 1982       SS
SS SoftSide Publications, Inc SS
SS                             SS
SS SS SS SS SS SS SS SS SS SS SS
```

Initialize variables, DIMension arrays, DEFine Function I$, and select run time options.

```
10 DEF FNI$(A$)=CHR$(ASC(LEFT$(A$,1))+32
*(LEFT$(A$,1)>"Z")):DEFINT B-K,S-Z:A=0:A
Z=0:A$="":C$="":D=0:DS=100:DT=0:G1=0:G2=
0:G3=0:G4=0:G5=0:G6=0:HH=0:I$="":IP$="":
J$="":LN=0:L$="":L1$="":N$="":P=0:PJ=0:P
P=0:PV=0:Q$="":R=0:RD=0:RE=0:S=0:S1=0
20 SD=0:SQ$="":SV$="":T=0:T1=0:T2=0:V$="
":X=0:XC$="":XS$="":XP$="":ZC=0:DIM REF(
DS*2),PRO(DS):SCREEN 0,0,0:WIDTH 80:COLO
R 11,0:KEY OFF:CLS:LINE INPUT"ENTER THE
NAME OF THE PROGRAM TO BE SQUISHED: ";SQ
$
30 XS$="N":IP$="N":XC$="N":XP$="N":PRINT
:LINE INPUT"ENTER THE NAME FOR THE FINAL
 SQUISHED PROGRAM: ";SV$:PRINT:LINE INPU
T"WOULD YOU LIKE EXTRA SPACES DELETED? (
Y/N) ";XS$:IF XS$="" THEN XS$="N"
40 PRINT:LINE INPUT"WOULD YOU LIKE REM S
TATEMENTS DELETED? (Y/N) ";IP$:IF IP$=""
 THEN IP$="N"
50 PRINT:LINE INPUT"WOULD YOU LIKE TO CO
MBINE LINES? (Y/N) ";XC$:IF XC$="" THEN
XC$="N"
60 PRINT:LINE INPUT"WOULD YOU LIKE TO PR
OTECT ANY LINES? (Y/N) ";XP$:IF XP$="" T
HEN XP$="N"
```

Set error trap and convert INPUT variables.

```
70 ON ERROR GOTO 560:XS$=FNI$(XS$):IP$=F
NI$(IP$):XC$=FNI$(XC$):XP$=FNI$(XP$)
```

If nothing is to be done, reRUN the program.

```
80 IF XS$="N" AND IP$="N" AND XC$="N" AN
D XP$="N" THEN RUN
```

Store user protected lines.

```
90 IF XP$="Y" THEN INPUT"ENTER LINE NUMB
ER TO PROTECT (0 TO EXIT) ";PRO(PV):IF P
RO(PV)>0 AND PV<DS THEN PV=PV+1:GOTO 90
```

OPEN source file for input.

```
100 OPEN SQ$ FOR INPUT AS #1
```

Check for End Of File and PRINT error if current line does not start with a line number.

```
110 IF EOF(1) THEN 240
120 LINE INPUT #1,A$:IF ASC(A$)>58 THEN
COLOR 12,0:PRINT:PRINT"#### '";SQ$;"' IS
 NOT AN ASCII FILE ####":PRINT:COLOR 11,
0:END
```

Search for reserved words that reference other program lines.

```
130 G1=1:G2=1:G3=1:G4=1:G5=1:G6=1

140 D=4:T=INSTR(G1,A$,"THEN"):IF T THEN
G1=T+D:GOTO 210

150 T=INSTR(G2,A$,"GOTO"):IF T THEN G2=T
+D:GOTO 210

160 T=INSTR(G3,A$,"ELSE"):IF T THEN G3=T
+D:GOTO 210

170 T=INSTR(G4,A$,"GOSUB"):IF T THEN D=5
:G4=T+D:GOTO 210

180 T=INSTR(G5,A$,"RESUME"):IF T THEN D=
6:G5=T+D:GOTO 210

190 T=INSTR(G6,A$,"RUN"):IF T THEN D=3:G
6=T+D:GOTO 210

200 GOTO 110

210 A=VAL(MID$(A$,T+D)):IF A THEN FOR HH
```

Store all referenced lines into an array.

```
210 A=VAL(MID$(A$,T+D)):IF A THEN FOR HH
=1 TO R:IF REF(HH)<>A THEN NEXT:R=R+1:RE
F(R)=A

220 IF A>0 THEN T=T+D:D=1:T1=INSTR(T,A$,
","):T2=INSTR(T,A$,":"):IF T1>0 AND (T2=
0 OR T1<T2) THEN T=T1:GOTO 210

230 GOTO 140
```

Sort all referenced and protected lines in ascending order.

```
240 CLOSE:FOR S=1 TO R:FOR S1=S TO R:IF
REF(S)<REF(S1) THEN SWAP REF(S),REF(S1)
250 NEXT S1,S:FOR S=0 TO PV:FOR S1=S TO
PV:IF PRO(S)>PRO(S1) THEN SWAP PRO(S),PR
O(S1)
```

ReOPEN source file for INPUT and OPEN destination for OUPUT.

```
260 NEXT S1,S:OPEN SQ$ FOR INPUT AS #1:O
PEN SV$ FOR OUTPUT AS #2:CLS
```

Get next program line to be processed.

```
270 IF EOF(1) THEN 380
280 LINE INPUT #1,A$:FOR HH=INSTR(A$," "
) TO LEN(A$)-1:IF MID$(A$,HH+1,1)=" " TH
EN NEXT
```

Set up pointer variables and update display.

```
290 PP=HH:X=PP:LN=VAL(A$):LOCATE 1,1:COL
OR 11,0:PRINT"SCANNING LINE:";:COLOR 12,
0:PRINT LN:PRINT:PRINT STRING$(255,32):L
OCATE 3,1:COLOR 14,0:PRINT A$:LOCATE 8,1
:COLOR 11,0:PRINT"SCANNING POSITION:
":PRINT


300 PRINT"NUMBER OF LINES COMBINED:";:CO
LOR 12,0:PRINT RE:COLOR 11,0:PRINT:PRINT
"NUMBER OF SPACES DELETED:";:COLOR 12,0:
PRINT SD:COLOR 11,0:PRINT:PRINT"NUMBER O
F REM STATEMENTS DELETED:";:COLOR 12,0:P
RINT RD:COLOR 11,0:GOTO 410
```

If combining lines is not allowed then write new line.

```
310 IF XC$<>"Y" THEN PRINT #2,A$:GOTO 27
0
```

Set up C$ to start combining lines.

```
320 IF C$="" THEN C$=A$:GOTO 270
```

Checks if current line is referenced.

```
330 IF R>0 THEN IF LN=REF(R) THEN R=R-1:
GOTO 370 ELSE IF LN>REF(R) THEN R=R-1:GO
TO 330
```

Never combine lines with IF or RETURN statements.

```
340 IF INSTR(C$,"IF") OR INSTR(C$,"RETUR
N") THEN 370
```

Combine two program lines and go for more.

```
350 V$=RIGHT$(A$,LEN(A$)-X):IF LEN(C$)+L
EN(V$)<240 THEN C$=C$+":"+V$:RE=RE+1 ELS
E 370
360 GOTO 270
```

Not enough space in C$ to combine another line so write it out first, then continue.

```
370 PRINT #2,C$:C$=A$:GOTO 270
```

Write out last program line and update display.

```
380 PRINT #2,C$:CLOSE:COLOR 12,0:LOCATE
8,19:PRINT T:LOCATE 10,26:PRINT RE:LOCAT
E 12,26:PRINT SD:LOCATE 14,34:PRINT RD
```

Prompt for optional LOADing of Squished program.

```
390 LOCATE 3,1:PRINT STRING$(255,32):LOC
ATE 3,1:COLOR 14,0:PRINT"PRESS 'L' TO LO
AD THE SQUISHED PROGRAM":SOUND 1000,6:SO
UND 660,5:COLOR 11,0
400 Q$=INKEY$:IF Q$="" THEN 400 ELSE CLS
:IF Q$="L" THEN LOAD SV$ ELSE END
```

Change color of current character being scanned.

```
410 N$=LEFT$(A$,PP):ZC=160+PP:PP=PP+1:P=
0:J$="":DT=0:FOR T=PP TO LEN(A$):L$=MID$
(A$,T,1):AZ=INT(ZC/80):LOCATE AZ+1,ZC-AZ
*80+1:COLOR 10,0:PRINT MID$(A$,T,1);:ZC=
ZC+1:COLOR 12,0:LOCATE 8,19:PRINT T
```

Set P equal to 1 on the first quote mark in a PRINT statement ELSE set P equal to 0 on second.

```
420 COLOR 11,0:IF L$=CHR$(34) THEN IF P
THEN P=0 ELSE P=1
```

If the current scan position is within a set of quote marks, skip all normal Squish processing.

```
430 IF P THEN 520
```

Branch to line 520 if DATA is found in the current program line.

```
440 IF MID$(A$,T,4)="DATA" THEN DT=1 ELS
E IF L$=":" THEN DT=0
450 IF DT THEN 520
```

Remove all spaces when safe to do so and update Spaces Deleted counter.

```
460 IF L$<>" " OR XS$<>"Y" THEN 500 ELSE
 IF J$<>"" THEN L1$=RIGHT$(J$,1):IF L1$="
^" OR (L1$>")" AND L1$<"0") OR (L1$>"9"
AND L1$<"A") THEN L$=""
470 L1$="X":IF T<LEN(A$) THEN L1$=MID$(A
$,T+1,1)
480 IF L1$="^" OR L1$=CHR$(34) OR L1$="
" OR (L1$>")" AND L1$<"0") OR (L1$>"9" A
ND L1$<"A") THEN L$=""
490 IF L$="" THEN SD=SD+1
```

Check for a user protected line.

```
500 IF PV>PJ THEN IF LN=PRO(PJ) THEN PJ=
PJ+1:GOTO 540 ELSE IF LN>PRO(PJ) THEN PJ
=PJ+1
```

Search for REMarks and remove if allowed.

```
510 IF MID$(A$,T,3)="REM" OR L$="'" THEN
 IF IP$<>"Y" THEN A$=N$+J$+MID$(A$,T,255
):GOTO 540 ELSE RD=RD+1:IF LN=REF(R) THE
N R=R-1:A$=N$+J$+"'":GOTO 540 ELSE IF J$
="" THEN 270 ELSE 530
```

Construct a new Squished version of the current line in J$. Also add a trailing quote mark if none found after a PRINT statement.

```
520 J$=J$+L$:NEXT:IF P THEN J$=J$+CHR$(3
4)
```

Add the current program line number and jump to line 310.

```
530 A$=N$+J$:GOTO 310
```

If lines have been combined then save them.

```
540 IF C$<>"" THEN PRINT #2,C$:C$=""
```

Otherwise write new program line.

```
550 PRINT #2,A$:GOTO 270
```

Error Trapping Done Here.

```
560 IF ERR=53 THEN RUN ELSE ON ERROR GOT
O 0
```

HARDWARE SPECIALS

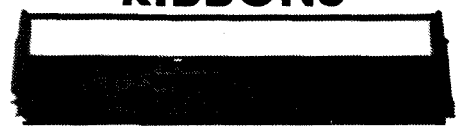
LYNX MODEM

Auto Answer - Auto Dial

Mod I & III
No RS-232
Board Required

Shipping
from Stock
$218.00

RIBBONS



MX-80 Replacement Cartridges
2 for $15.00
MX-100 Replacement Cartridges
2 for $20.00
Offer Good thru Dec. 31, 1982

ORCHESTRA 85/90

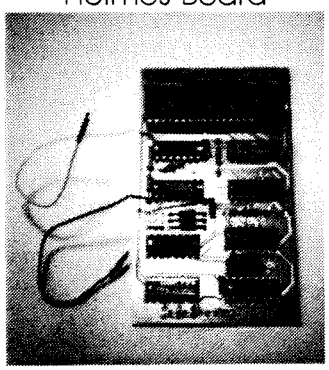
SUPER DEAL

Stereo music synthesizer with percussion for your Mod I or III.
Was $149.95
SPECIAL $99.95
Please state Mod I or II & Cass. or Disk version
Offer Good thru Dec. 31, 1982

Our Catalog is Now Available!

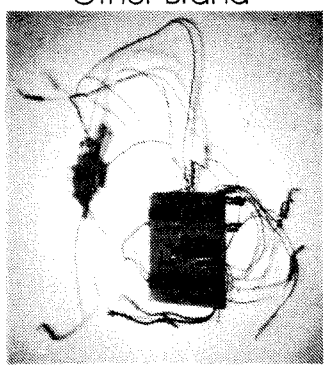
There Are Only 2 Ways to Increase the Processing Speed of Your Model III

Holmes Board



The Right Way

Other Brand



The Wrong Way

They both list for $99.95 assembled and Tested.
They both speed up your machine.

	Holmes Board	Other Brand
Installation Time	15 min.	4 hrs.
Requires Soldering	NO	YES
Requires Cutting Traces ...	NO	YES
Easily Removed if Logic Board Requires Repair ...	YES	NO

We Offer Only The Quality Product
But the Choice is Yours

SAVE $10.00

List $99.95
Only **$89.95**
Plus Shipping
Offer Good Until Dec. 31, 1982

- DEALER INQUIRIES WELCOME -

SSM SOFT SECTOR MARKETING, INCORPORATED
P.O. Box 340 • Garden City, Michigan 48135
Order Line **800-521-6504**
Michigan Orders & Questions 313-425-4020

VISA MasterCard AMERICAN EXPRESS

PAYMENT – payment accepted by charge, personal check or C.O.D. Only under the following conditions: **Charges** processed when shipped, usually within 48 hours. **Personal Checks** delay shipping, pending 3 weeks to clear. **C.O.D.** orders are certified check or cash only, add $1.50. MI residents must add 4% sales tax.
SHIPPING & HANDLING – Shipping Charges: Send the larger amount: 2% or $2.50, unless stipulated otherwise. Any order received without shipping and handling will be shipped **freight collect. Air Mail Shipping** outside of North America, please send the larger amount: 10% or $10.00. Overpayment will be refunded.

CAVERN

Cavern Quest is an arcade-style strategy game for a TRS-80® Model I or III with at least 32K RAM and one disk drive. This program was compiled using the ZBASIC 2.2 compiler, copyright Simutek Computer Products Inc. and Andrew Gariepy. It is included as the bonus program on issue 34 TRS-80 DV.

You've been in bad situations before, but this one takes the prize. A chance encounter with a rock fragment in space has crippled your mining survey rocket and forced you to land near a mountain on a small asteroid. The damage to your ship can be repaired, but you have lost most of your rocket fuel. Unless you can find a fuel source on this barren asteroid, you'll never see home again.

Luckily, your ship's sensors have detected a fuel ore deposit directly below your landing site, though at considerable depth. Your only hope is to use a remote-controlled robot miner equipped with an excavation energy-blaster to tunnel through the mountain and retrieve the ore. However, you have only one such robot miner and the mountain contains countless dangers. Make one mistake in your quest and your ship will become your tomb!

Keep the following points in mind during your quest:

1. Black areas on the screen are open areas and tunnels. Solid white areas are dirt (soil), checkered areas are stationary rock formations, and black-edged white blocks with specks inside are individual rocks. The moving objects trapped in the asteroid are monsters. See Sections 7 and 8 for details of their behavior.

2. To move your robot miner, use the up, down, left, and right arrow keys. If the direction that you indicate is blocked by dirt, the robot's mining blaster will disintegrate the dirt and allow you to move in that direction.

3. Your robot's blaster has no effect on rocks or formations, crystals, monsters, your rocket ship, or the fuel ore. It will disintegrate only dirt.

4. The blaster has an initial charge of 100 uses. Capturing a crystal (indicated by an asterisk on the screen) supplies an additional 25 uses per crystal captured. If you should ever return the robot miner to the ship with fewer than 50 uses remaining, the blaster will be given an automatic recharge of an additional 50 uses.

5. If you ever exhaust your supply of blaster uses (keep an eye on the counter in the upper right-hand screen corner) your blaster will cease to function. You will then have but a few seconds to recharge it before it self-destructs and destroys the robot miner. A time counter will appear in the upper left-hand corner of the screen to show you how much time remains before the blaster self-destructs.

6. Blasting the dirt beneath a rock (but not a rock formation) frees it for an eventual fall. You have just enough time to take three actions

QUEST

by Barry Diller

(for example: move under the rock, blast the next block of dirt, and then move on) before the rock falls. If your robot miner is under the rock when it falls, it will be destroyed. Rocks fall until they hit something, so be careful where you drop them.

7. Monsters can only move in open areas and tunnels; they cannot burrow. Monsters move erratically and unpredictably, but they cannot outrun your robot miner. Monsters can be killed by dropping rocks on them, but they will rematerialize elsewhere in the mountain. Contact with a monster will destroy the robot miner.

8. Monsters continue to move throughout the mountain regardless of whether or not they are visible on the screen. Beware of monsters approaching you from "off-screen."

9. The fuel ore (the strange-looking artifact located directly below the landing site and next to a cache of crystals) and the crystals can be captured by moving your robot miner onto them. An indicator appears on the screen whenever the ore is in the possession of the robot. Return to the ship with the ore to win the game.

Avoid unnecessarily releasing monsters from their lairs. If you must release some monsters in order to pass through a lair or to get some crystals, clear an escape route for your robot first. Study the monsters' movements carefully, and at an opportune moment, "open" the lair with your blaster. When the monsters leave the lair to chase your robot miner, follow your escape route and lose, trap, or kill the pursuers. Have your robot circle back to the lair as soon as it is safe to do so.

Monsters are most deadly when their victim is in their direct line-of-sight. Avoid moving your robot miner to positions directly horizontal or vertical to the monsters.

One last piece of advice: the rocks in the mountain are not randomly positioned. Clever use of the rocks is the key to completing your quest and winning the game.

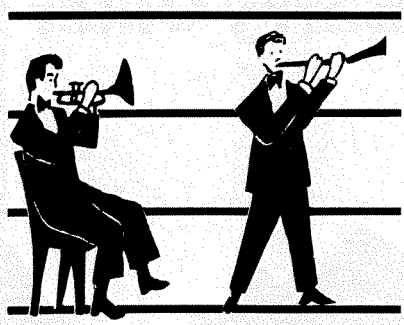
After loading *Cavern Quest,* you will see a list of options from which you may select. Type (1) to start a new game or (3) to load a previously saved game from disk. The games will start automatically.

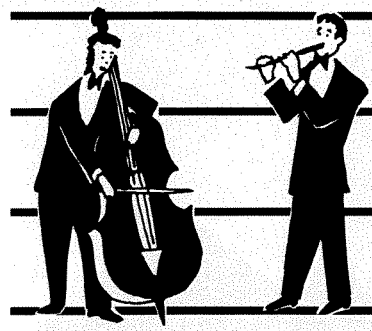
During the game, pressing (ENTER) and (CLEAR) simultaneously will return you to the list of options.

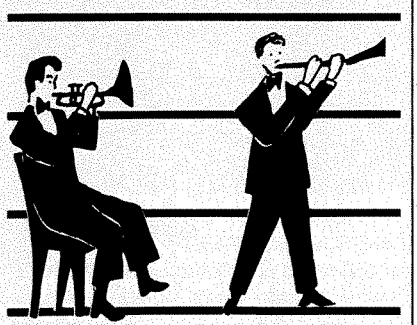
To save a game in progress, hit (2) at the list of options. When the game has been saved, you will be returned to the option list. Hit (3) to resume the game that you just saved. Data for a stored game is given the file name CAVE-DATA/TXT and occupies about 5 grans of disk space (single-density TRSDOS, Model I reference). Games can be saved to disk drive 0 only.

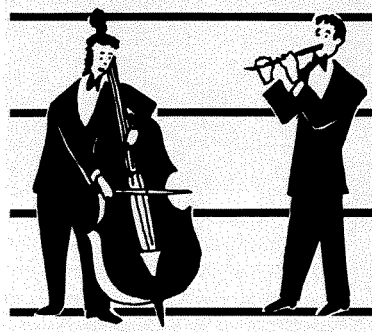
Hitting (4) at the option list has the same effect as hitting the computer's reset button. The computer will reload the DOS from disk.

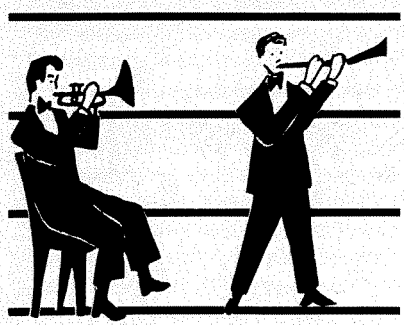
TUNEIN

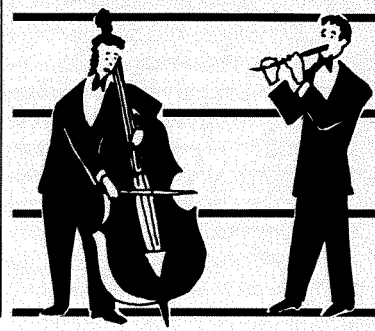










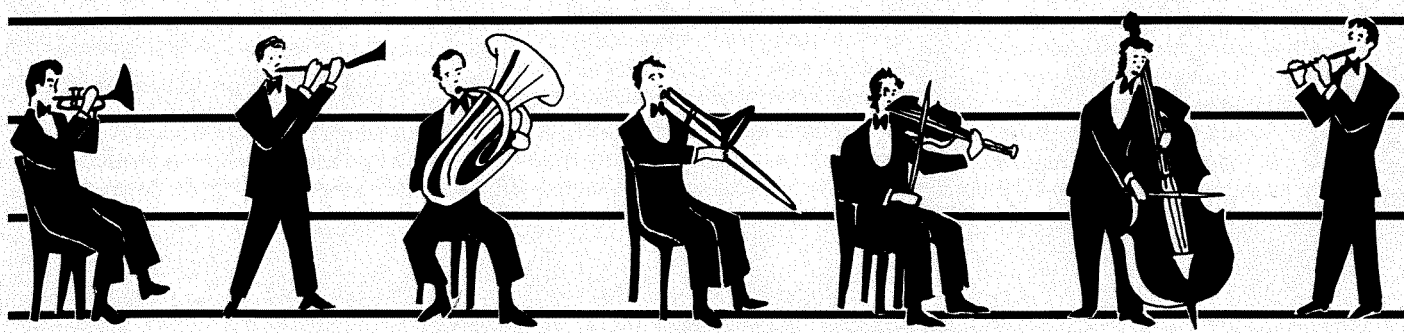


by William Morris and John Cope. TRS-80® translation by Ronny Ong.

***Tunein* is a music/memory game program for a 16K RAM TRS-80 Model I or III.**

This is a translation of the Atari® *Tunein* published in issue #31 of ***SoftSide***. Although the program logic of the two versions is the same, there are a number of changes necessitated by differences in hardware.

First, hook an amplifier to the AUX plug for sound. The game will begin by asking for the number of players. Simply press the appropriate number key (1-4). At the prompt which follows, you have three options. Pressing the period key restarts the game. Pressing "0" increases the number of notes you must repeat in order to complete the current level and score points. If you increase past the maximum, it will reset to one. Pressing "5" causes the computer to begin the current player's turn.



Each player has three "lives" per game. Just before a player's turn, the screen will display the number of lives he or she has remaining and the number of notes in his or her current level. This display will flash once for every remaining life.

After pressing the five key, one of eight graphic squares on the screen will flash as a corresponding musical note is played via the cassette port. Once the sequence stops, it's time for you to perform, using the number inside each of the squares. You should have realized by now that, except for typing "RUN" to start the program and the BREAK key to exit, if necessary, the game can be played using only the numeric keypad if your computer has one. Since the squares in *Tunein* are arranged with their numbers corresponding to the numeric keypad layout, keypad owners won't have to memorize the series by numbers in order to duplicate the pattern, and they may not have to look at the keys at all after a few games. If your TRS-80 is not so equipped, you can still enjoy *Tunein*, so don't rush out to the neighborhood Computer Center and buy one unless you intend to moonlight as an accountant. Just use the top row of numbers and the period on the regular keyboard.

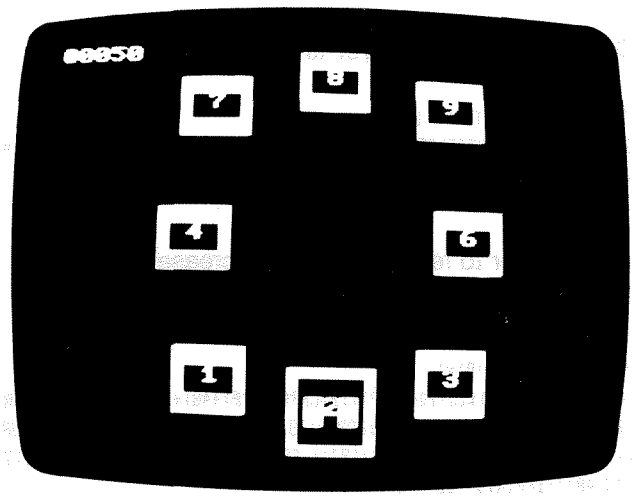
When you press the numbers to repeat a sequence, press them, don't tap quickly. (This advice applies mostly to the Model I, which has a slow clock speed.) When the program registers your response, the corresponding square will flash as one of two sounds is produced: the proper musical note for a correct answer or the dreaded "you blew it" buzzer for an incorrect answer.

A miss will cost you a "life." You will be out of the game when your lives are all gone. When all the players' lives are kaput, the game begins anew. As you get better, the program gets harder. TRS-80 *Tunein* allows you to mimic a 200-note sequence before short circuiting (if only anyone could).

Variables

Note: All variables beginning in A, B or C are defined by the program to be string variables.

A(): Graphic strings.
B: INKEY$ storage.
C: "Dummy" string to hold Machine Language sound routine.
D(): Players' scores.
FF: Flag that is set to one when the player increases the number of notes to be repeated.
I,J,K: Misc. loops and other uses.
J(): PRINT@ locations for the eight graphic blocks.
LF(): The number of lives left for each player.
LV(): The level of the game.
M(): Note values for the sound routine.
MM: The number of notes that have been repeated by the player during his current turn.
N(): Storage for the sequence to be repeated.
NO(): The maximum number of notes to be played at the current level of the program.
NN: The number of notes to be played at each stage during a player's turn.
P: Player counter.
S2: PRINT@ location for current player's score display.
W: Value typed by player when guessing a sequence.
Y,Z: Misc. loops and other uses.




```
SS SS SS SS SS SS SS SS SS SS SS
SS                             SS
SS         TRS80 BASIC         SS
SS          'TUNEIN'           SS
SS  AUTHORS: WILLIAN MORRIS    SS
SS       AND JOHN COPE         SS
SS     TRANSL: RONNY ONG       SS
SS     COPYRIGHT (C) 1982      SS
SS SOFTSIDE PUBLICATIONS, INC SS
SS                             SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
10 GOTO250
```

Input the number of players.

```
20 I=I+1+8*(I=8):B=INKEY$:IFB<"1"ORB>"4"THENPRINT@J(I),A(0);:PRI
NT@J(I),A(I);:K=USR(M(I)-4096):GOTO20ELSEPL=VAL(B):PRINT@414,CHR
$(195);:PRINT@476,CHR$(197);:PRINT@540,CHR$(197);:GOTO50
```

Subroutine to input one character, then rerun the program if "." is struck, or increase the number of notes if "0" is hit.

```
30 POKE16443,0:B=INKEY$:POKE16442,0:POKE16443,0:IFB="."THENRUNEL
SEIFB="0"IFPTHENNO(P)=NO(P)+1:FF=1
```

```
SEIFB="0"IFPTHENNO(P)=NO(P)+1:FF=1
40 RETURN
```

Initialize information about all players.

```
50 FORP=1TO4:LV(P)=1:D(P)=0:LF(P)=3:NO(P)=1:NEXT:P=0
```

Check to see if any players have lives remaining.

```
60 Y=0:FORZ=1TOPL:Y=Y+LF(Z):NEXT:IFY=0THEN230
```

Determine which player is next.

```
70 P=P+1:IFP>PLTHENP=1
80 IFLF(P)=0THEN70ELSEFORZ=1TOLV(P)*10:N(Z)=RND(8):NEXT:IFP=1S2=
0ELSEIFP=2S2=52ELSEIFP=3S2=896ELSES2=948
```

Allow player to restart program, increase the number of notes he or she is to attempt, or start turn.

```
90 IFNO(P)>10NO(P)=1
100 B=STR$(LV(P)*10-10+NO(P)):B=CHR$(LF(P)+48)+"-"+STRING$(4-LEN
(B),48)+RIGHT$(B,LEN(B)-1):FORI=1TO3:PRINT@S2,B;:K=USR(8224):FOR
K=1TO150:NEXT:PRINT@S2,CHR$(197);:FORK=0TO99:NEXT:NEXT:PRINT@412
,"START";:PRINT@478,"= 5";
110 GOSUB30:IFFFTHENFF=0:GOTO90ELSENN=0:IFB<>"5"THEN110ELSEPRINT
@412,CHR$(197);:PRINT@478,CHR$(195);
```

Display current score, and check if the player has completed the required number of notes. If not, then present note pattern.

```
120 GOSUB180:NN=NN+LV(P):FORZ=1TO4:GOSUB30:NEXT:IFNN>LV(P)*NO(P)
THEN200ELSEFORZ=1TONN:PRINT@J(N(Z)),A(0);:PRINT@J(N(Z)),A(N(Z));
:K=USR(M(N(Z))):FORY=1TO50:NEXT:NEXT:MM=0
130 MM=MM+1:IFMM>NNTHEN120
```

Input player's guess of the note sequence.

```
140 B=INKEY$:IFB<"1"ORB>"9"GOSUB30:IFFFTHENFF=0:GOTO90ELSE140
150 W=VAL(MID$("654703812",VAL(B),1)):IFW=0THEN140ELSEPRINT@J(W)
,A(0);:PRINT@J(W),A(W);
160 IFW=N(MM)K=USR(M(W)):D(P)=D(P)+10*LV(P):GOSUB180:GOTO130ELSE
LF(P)=LF(P)-1:FORI=1TO3:K=USR(16634):NEXT:GOTO60
```

Subroutine to wrap score around if necessary, and to display score.

```
180 'IF999999<D(P)D(P)=0
190 B=STR$(D(P)):PRINT@S2,STRING$(6-LEN(B),48)RIGHT$(B,LEN(B)-1)
;:RETURN
```

This routine is reached when the player has completed the required number of notes. If an increase in level is needed, that is done here. In any case, the program allows the player to try a harder series of notes.

```
200 NO(P)=NO(P)+1:FORI=256TO511:K=USR(I):NEXT:IFNO(P)>10THENNO(P
)=1:LV(P)=LV(P)+1
210 IFLV(P)<19THEN60ELSECLS:PRINTCHR$(23):PRINT"CONGRATULATIONS,
 PLAYER"P"!":FORI=0TO999:NEXT:PRINT"YOU HAVE MANAGED TO OVERLOAD
":PRINTTAB(5)"THIS GAME. GOODBYE.":FORI=0TO999:NEXT:FORI=0TO1STE
P0:PRINTCHR$(28)CHR$(23);:K=USR(RND(32767))
220 POKE15359+RND(1024),RND(255):NEXT
```

When game is over, allow the game to be repeated.

```
230 PRINT@412,"AGAIN";:PRINT@480,"=";:PRINT@540,"ENTER";
240 IFINKEY$=CHR$(13)RUNELSEI=RND(8):K=RND(8):IFK=ITHEN240ELSEB=
A(I):A(I)=A(K):A(K)=B:PRINT@J(I),A(0);:PRINT@J(I),A(I);:I=USR(M(
I)-4096):PRINT@J(K),A(0);:PRINT@J(K),A(K);:K=USR(M(K)-4096):GOTO
240
```

Present title page, then initialize variables, create graphic strings, and the machine language sound routine.

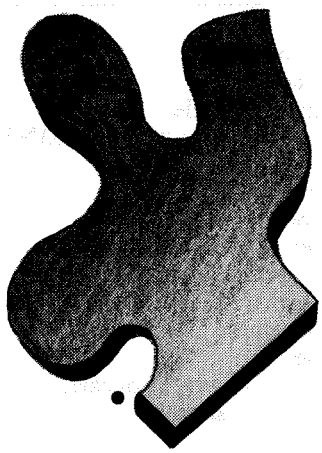
```
250 CLS:PRINT@346,CHR$(23);"TUNEIN";:PRINT@450,"(c) Wm. Morris &
 J. Cope 1982";:PRINT@576,"TRS-80 translation by Ronny Ong":CLEA
R500:DEFSTRA-C:DEFINTF-Z:DIMN(200),M(8),LV(4),NO(4),D(4),LF(4),J
(8),A(8):M(1)=8252:M(2)=8256:M(3)=8264:M(4)=8273:M(5)=8283
260 M(6)=8288:POKE16553,255:M(7)=8300:M(8)=8313:J(1)=28:J(2)=106
:J(3)=428:J(4)=746:J(5)=796:J(6)=718:J(7)=396:J(8)=78:C=CHR$(205
):FORI=1TO28:READK:C=C+CHR$(K):DATA127,10,62,1,14,0,30,8,69,47,2
30,3,179,211,255,13,40,4,16,246,24,242,37,32,241,201,0,0
270 NEXT:K=VARPTR(C)+1:FORI=1TO8:A(I)=CHR$(168)+STRING$(3,188)+C
HR$(148)+CHR$(26)+STRING$(5,24)+CHR$(170)+" "+MID$("89632147",I,
1)+" "+CHR$(149)+CHR$(26)+STRING$(5,24)+CHR$(138)+STRING$(3,143)
+CHR$(133):NEXT
272 IFPEEK(16809)=201THENPOKE16526,PEEK(K):POKE16527,PEEK(K+1):E
LSEDEFUSR=PEEK(K)+PEEK(K+1)*256+65536*(PEEK(K+1)>127):CMD"T"
280 A(0)=CHR$(151)+STRING$(3,131)+CHR$(171)+CHR$(26)+STRING$(5,2
4)+CHR$(149)+CHR$(191)+STRING$(2,25)+CHR$(191)+CHR$(170)+CHR$(26
)+STRING$(5,24)+CHR$(181)+STRING$(3,176)+CHR$(186)
290 TM=ABS(PEEK(293)=73):PRINT@1,CHR$(192):IFPEEK(15361)=192PRIN
TCHR$(21)
```

Initialize screen display.

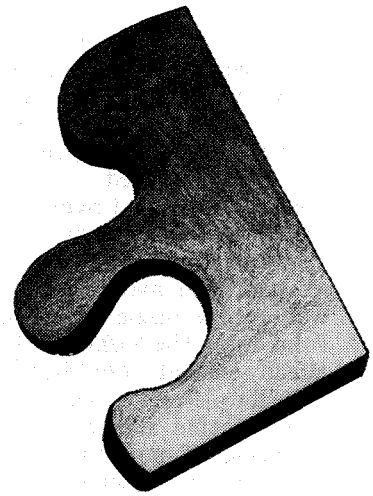
```
300 FORI=0TO999:NEXT:RANDOM:CLS:PRINT@414,CHR$(23);"HOW";:PRINT@
476,"MANY";:PRINT@540,"(1-4)":FORI=1TO8:PRINT@J(I),A(I);:NEXT:I=
0:GOTO20
```

TRS-80® SWAT TABLE FOR:
TUNEIN

LINES	SWAT CODE	LENGTH
10 - 100	XK	591
110 - 200	MU	557
210 - 250	SO	615
260 - 280	TF	582
290 - 300	MJ	147

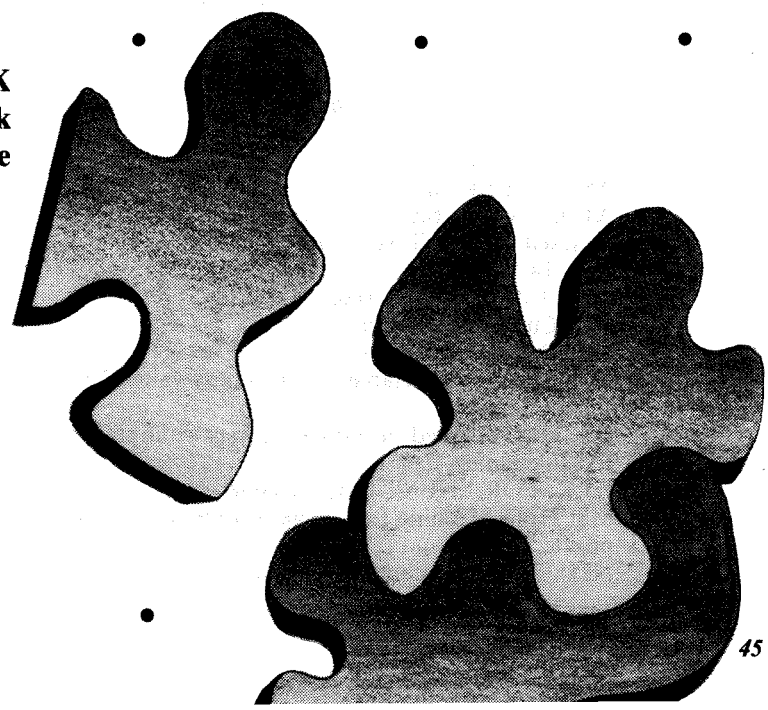


PUZZLE JUMBLE



by Gary Cage

Puzzle Jumble **is a graphics game for a 32K RAM TRS-80 Model I or Model III with disk drive. Two sample puzzles are included on issue 34 TRS-80** ***DV*****.**



Note: *Puzzle Jumble* is a "self-modifying" program. When run, it changes its program code. Once it is run, it may not be run again from memory — it must be loaded from disk again. When typing in the program, ***always*** save it to disk before running it. Failure to do so will make the program unusable. If you forget to save to disk, however, the program may usually be salvaged by retyping line 7, and then deleting line 15360 (which is not in the program listing, but will be created by the process of RUNning the program).

If you enjoy puzzles, you and your TRS-80 are likely to spend many hours with this program. First, you draw a picture (or load one of the two pictures included on the ***SoftSide DV***). It will then be scrambled, and you must try to reconstruct it in as few moves as possible.

When the program starts, a graphic box will be displayed with a flashing cursor inside. Lines may be drawn by moving the cursor with the arrow keys, and erased by holding down the shift key and moving the cursor over the area you wish to erase. Arrow keys may be combined for diagonal movement, i.e., holding the up and right arrows will draw an angular line up and to the right.

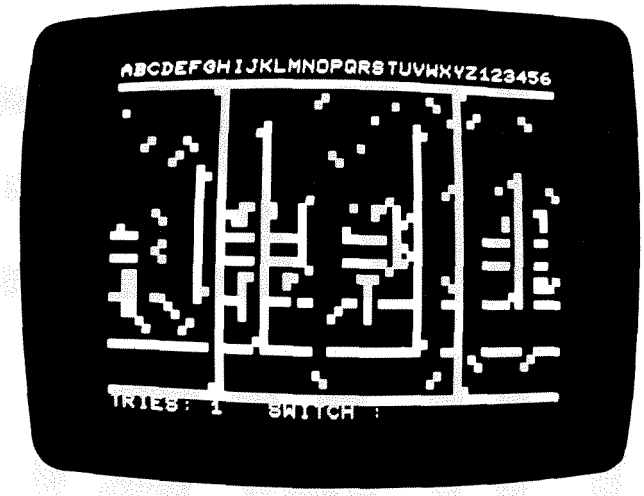
There are four special options which may be selected, each invoked by typing the proper letter. "S" will save the puzzle to disk, and "L" will load a puzzle from disk. (The ***SoftSide DV*** disk contains two puzzles already created, *SOFTSIDE* and *TRENCH*.) "C" will erase the picture in memory, and "P" will allow you to play the game, attempting to solve a scrambled puzzle.

During the game, the scrambled 32 column picture is displayed on the screen, along with a display of how many moves have been taken. On the top of the screen are the letters A through Z, and the digits 1 through 6. These are used to identify each column of the picture.

The object of the game is to unscramble the picture. This is done by specifying columns to be interchanged, until the puzzle has returned to its original appearance. While solving the puzzle, two characters gain special importance. "*" will end the game, returning to the picture editor, and "=" will display the completed puzzle. Hitting "=" again will return you to your scrambled puzzle.

Variables:

A: Used in keyboard scan.
A$: INKEY$ variable.
A1,A2: Sound routine variable.
B: Used in SHIFT key scan.
B1$: 14 blanks.
CK: -1 if puzzle is correct, 0 if not.
CX: Cursor x position.
CY: Cursor y position.
D,D1: Dummy variables used to pass values to subroutines.
DS,DX,DY: Used in blinking cursor routine.
F$: Filename.
I,J,K: FOR...NEXT loop variable.
M$: Hold Machine Language routine to display puzzle.
N2$: Translation author's name.
NM$: Author's name.
P$(0,0-31): Original puzzle.
P$(1,0-31): Scrambled puzzle.
P1,P2: Positioning variables used in PRINT@ statements.
PZ: Number of puzzles completed.
R: Random number.
SM: Total number of tries for all puzzles.
SW: Switch — 0 = puzzle not in P$ array, 1 = in array.
T,T$: Temporary values.
TI$: Program title.
TR: Number of tries in current puzzle.
X,Y,Z: Sound routine variables.




```
SS SS SS SS SS SS SS SS SS SS SS
SS                           SS
SS        TRS80 BASIC        SS
SS       PUZZLE JUMBLE       SS
SS     AUTHOR: GARY CAGE     SS
SS  TRANSL: STEPHEN MILLIKEN SS
SS     COPYRIGHT (C) 1982    SS
SS SOFTSIDE PUBLICATIONS, INC SS
SS                           SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
5 CLS:PRINTCHR$(23);:PRINT@530,"PUZZLE JUMBLE";:PRINT@980,"LOADI
NG...";:PRINT@0,"";:GOTO60000
7 M$="MOVE ROUTINE GOES HERE!":RETURN
10 CLS:GOSUB1140:RANDOM
```

Main Routine

```
20 CLS:CX=65:CY=25:SW=0
30 P1=145:FORI=1TO12:PRINT@P1,CHR$(149);:PRINT@P1+31,CHR$(170);:
P1=P1+64:NEXT
40 PRINT@82,STRING$(30,131);:PRINT@914,STRING$(30,176);
50 PRINT@81,CHR$(151);:PRINT@112,CHR$(171);:PRINT@913,CHR$(181);
:PRINT@944,CHR$(186);
60 PRINT@13,"<S>AVE - <L>OAD - <P>LAY - <C>LEAR";
70 PRINT@976,"ARROWS DRAW ";CHR$(149);" <SHIFT> ARROWS ERASE";
80 P1=80:FORT=1TO14:PRINT@P1,MID$(TITLE$,T,1);:PRINT@P1+33,MID$(
```

continued on page 48

SOFTWAREHOUSE™ INC.
INTERNATIONAL

The Christmas Club that lets you have the Christmas spirit without haunting your budget in January.

APPLE

	Retail	Club
Wizardry	49.95	36.80
Castle Wolfenstein	29.95	22.05
Time Zone	99.95	73.60
Swashbuckler	34.95	25.75
Knight of Diamonds	34.95	25.95
Choplifter	34.95	25.75
Cannonball Blitz	34.95	25.75
Ultima	39.95	29.45
Escape from Rungistan	29.95	22.05
Pinball (A2-PB1)	29.95	23.90
Home Accountant	74.95	55.20
Tax Preparer '82	150.00	110.45
Master Type	39.95	29.45
Word Handler	250.00	153.40
Super Text 40/80	150.00	128.85
PFS	125.00	92.05
DB Master	229.00	168.60
VisiCalc 3.3	250.00	199.40
PFS Report	95.00	69.95
DOS Boss	24.00	17.70
Utility City	29.50	21.75
Zoom Grafix	39.95	29.45
Global PLE	64.95	47.85
Pascal Tutor	125.00	92.05
Compu-Read	29.95	22.05
English SAT No. 1	30.00	22.10
Super Disk Copy III	30.00	22.10
Kabul Spy	34.95	25.75
The Complete Graphics System	69.95	51.50
Disks (Memorex)		26.00

ATARI

Adventure No. 12-Golden Voyage	19.95	14.70
Night Rally	34.95	25.75
Pac Man	44.95	34.15
Centipede	44.95	34.15
Curse of Ra	19.95	14.70
Star Blazer	31.95	23.55
Compu-Math/Fractions	39.95	29.45
VisiCalc	250.00	199.40
Zork II	39.95	29.45
Alien Swarm	34.95	25.75
Intruder	34.95	25.75
Letter Perfect	149.95	110.40
Ultima II	59.95	44.15
Cyclod	29.95	22.05
Disk Manager	29.95	22.05
3-D Supergraphics	39.95	29.45

We carry software for Apple, TRS-80, Atari, IBM, Commodore, and VIC-20 computers. If you don't see what you want on our list, just write in the manufacturer and the program names. We'll do everything we can to get it for you RIGHT AWAY!

TRS-80

	Retail	Club
Eliminator	19.96	14.70
Lost Colony	29.95	22.05
Sea Dragon	24.95	18.40
Invaders From Space	20.95	15.45
Stone of Sisyphus	29.95	22.05
Missile Attack	20.95	15.45
Adventure No. 1, 2, 3	39.95	29.45
Color Games by Lance (COLOR)	19.95	14.70
Danger in Drindisti	19.95	14.70
Morloc's Tower	19.95	14.70
Galaxy Invasion	19.95	14.70
Galactic Trilogy	39.95	29.45
Scarfman (COLOR)	19.95	14.70
Softerm	49.95	36.80
Inventory Plus	29.95	22.05
Cash Register 80	29.95	22.05
Sargon II	34.95	25.75
Penetrator	24.95	18.40
Olympic Decathlon	29.95	22.05
Typing Tutor	19.95	14.70
Flight Simulator	33.50	26.75
Vaults of Cymarron	39.95	29.45
Death Planet	19.95	14.70
Maxi Stat	199.95	147.20
Maxi Manager	99.95	72.00
Upper Reaches of Apshai	19.95	14.70

SOFTWAREHOUSE INC. INTERNATIONAL
802 12th Ave.
Menomonie, WI 54751
Phone Orders Call
715-235-3126

YES I want to be a member of Software House.
☐ Membership alone $10.00
☐ Membership $10.00 Plus Software

Software Titles	Club Price	Membership	Shipping	Total
		10.00	2.50	
		—	—	
		—	—	

Name: _____

Address: _____

City, State: _____ Zip: _____

Software House Accepts:
☐ Visa or Master
☐ Cash or Check
MasterCard VISA
☐ COD (add $1.50 extra for shipping)

Card No. _____ Exp. Date _____

Signature _____

All items subject to availability. Prices subject to change. SN

```
TITLE$,T,1);:P1=P1+64:NEXT
90 PRINT@436,"IN PLAY MODE";:PRINT@564,"(*) CANCELS";:PRINT@692,
"(=) DISPLAYS";:PRINT@760,"CORRECT";:PRINT@824,"PUZZLE";
100 IFA$="*"THEND=0:P1=81:GOSUB610
110 PRINT@576,"PUZZLES";:PRINT@640,"TRIED:";PZ;:PRINT@768,"AVERA
GE";:PRINT@832,"SCORE:";
120 IFPZ=0THENPRINT" 0";ELSEPRINTSM/PZ;
130 A$="":A$=INKEY$
140 DX=CX:DY=CY:DS=POINT(DX,DY)
150 IFDS=-1THENRESET(DX,DY)ELSESET(DX,DY)
160 IFA$="S"ORA$="s"THENGOSUB410
170 IFA$="C"ORA$="c"THEN20
180 IFA$="L"ORA$="l"THENGOSUB520
190 IFA$="P"ORA$="p"THENGOSUB680:CLS::GOTO60
200 A=PEEK(14400):B=PEEK(14464)
210 IFAAND8THENCY=CY-1:SW=0
220 IFAAND16THENCY=CY+1:SW=0
230 IFAAND32THENCX=CX-1:SW=0
240 IFAAND64THENCX=CX+1:SW=0
250 IFCX<35THENCX=96
260 IFCX>96THENCX=35
270 IFCY<4THENCY=43
280 IFCY>43THENCY=4
290 IFDS=-1THENSET(DX,DY)ELSERESET(DX,DY)
300 IFB<>0THENRESET(CX,CY)ELSESET(CX,CY)
310 GOTO130
```

Subroutine to store picture in P$(0,0-31).

```
320 P1=81:FORI=0TO31
330 P$(0,I)="":P2=P1:FORJ=1TO14
340 D=PEEK(15360+P2)
350 IFD=32THEND=128
360 P$(0,I)=P$(0,I)+CHR$(D)
370 D=191-D+128:POKE15360+P2,D
380 P2=P2+64:NEXT
390 P1=P1+1:NEXT
400 SW=1:RETURN
```

Subroutine to save picture on disk.

```
410 PRINT@320,"** SAVING **";
420 IFSW=0THENGOSUB320
430 ONERRORGOTO1130
440 PRINT@256,"FILENAME";:PRINT@320,"(NO EXTENSION)";:PRINT@384,
"";:INPUTF$
450 OPEN"O",1,F$+"/PUZ"
460 FORI=0TO31
470 PRINT#1,CHR$(34);P$(0,I);CHR$(34);
480 NEXT
490 ONERRORGOTO0
500 PRINT@256,B1$;:PRINT@320,B1$;:PRINT@384,B1$;
510 CLOSE:P1=81:D=0:GOSUB610:RETURN
```

Subroutine to load picture from disk.

```
520 ONERRORGOTO1120
530 PRINT@256,"FILENAME";:PRINT@320,"(NO EXTENSION)";:PRINT@384,
"";:INPUTF$
540 OPEN"I",1,F$+"/PUZ"
550 ONERRORGOTO0
560 FORI=0TO31
570 INPUT#1,P$(0,I)
580 NEXT
590 PRINT@256,B1$;:PRINT@320,B1$;:PRINT@384,B1$;
600 CLOSE:D=0:D1=1:P1=81:GOSUB610:SW=1:RETURN
```

Subroutine to print a puzzle on screen.

```
610 FORI=0TO31
620 P2=P1:FORJ=1TO14
630 PRINT@P2,MID$(P$(D,I),J,1);
640 P2=P2+64:NEXT
650 P1=P1+D+D1
660 IF(D=1ANDD1=1)OR(D=0ANDD1=2)THEN:PRINT@0,"";:SOUNDP1*3+100,1
0
670 NEXT:RETURN
```

Subroutine in which player solves puzzle.

```
680 IFSW=0THENGOSUB320
690 CLS:PRINTCHR$(23);
700 PRINTH$;
710 FORI=0TO31:P$(1,I)=P$(0,I):NEXT
720 D=0:D1=2:P1=64:GOSUB610
730 PRINT@980,"MIXING PUZZLE.";
760 FORI=0TO31
770 FORJ=1TO3
780 PRINT@0,"";:SOUNDRND(I*J+100)+20,5
790 T$=P$(1,I):R=RND(32)-1:P$(1,I)=P$(1,R):P$(1,R)=T$
800 NEXTJ,I
810 TR=0
820 PRINT@966,CHR$(28);:J=USR(0):CLS:PRINTCHR$(23);H$;:D=1:D1=1:
P1=64:GOSUB610
830 PRINT@960,"TRIES:";TR;
840 PRINT@982,"SWITCH : ";:GOSUB1070:IFA$="*"THENRETURNELSEI=D
850 IFA$="="THENPRINTCHR$(28);:J=USR(0):PRINTCHR$(23);:GOSUB1060
:PRINTCHR$(28);:J=USR(0):PRINTCHR$(23);:GOTO840
860 PRINT@0,"";:SOUND200-I*3,10:SOUND200-I*3,10:SOUND150-I*3,10
870 PRINT@1004,"WITH : ";:GOSUB1070:J=D
880 PRINT@0,"";:SOUND200-J*3,10:SOUND200-J*3,10:SOUND150-J*3,10
890 PRINT@982,STRING$(20,32);
900 T$=P$(1,I):P$(1,I)=P$(1,J):P$(1,J)=T$
910 P1=I*2+64:FORK=1TO14:PRINT@P1,MID$(P$(1,I),K,1);:P1=P1+64:NE
XT
920 P1=J*2+64:FORK=1TO14:PRINT@P1,MID$(P$(1,J),K,1);:P1=P1+64:NE
XT
930 CK=-1
940 FORI=0TO31
950 IFP$(0,I)<>P$(1,I)THENCK=0:I=31
960 NEXT
970 TR=TR+1:IFCKTHEN990
980 GOTO830
```

Tell player he has solved the puzzle.

```
990 FORI=1TO8
1000 PRINT@980,"****** CORRECT ******";
1010 PRINT@0,"";:SOUND50,50:SOUND100,50:SOUND50,50
1020 PRINT@980,STRING$(21,32);
1030 PRINT@0,"";:SOUND100,5:SOUND250,10:SOUND100,5
1040 NEXT
1050 PRINT@980,"*** HIT ANY KEY ***";:GOSUB1060:PRINT@980,STRING
$(20,32);:PZ=PZ+1:SM=SM+TR:A$="*":RETURN
```

INKEY$ subroutine.

```
1060 A$=INKEY$:IFA$=""THEN1060ELSERETURN
```

Convert A-Z to 0-25, 1-6 to 26-31.

```
1070 GOSUB1060
1080 IFA$="*"ORA$="="THENRETURN
1090 D=ASC(A$):IFD>48ANDD<55THEND=D+42ELSEIFD>96ANDD<123THEND=D-
32
1100 D=D-65:IFD<0ORD>31THEN1070ELSEPRINTA$;
1110 RETURN
```

Error trap on input.

```
1120 IFERR/2+1=54THENCLOSE:PRINT@448,"FILE NOT FOUND";:FORT=1TO2
000:NEXT:PRINT@448,B1$;:PRINT@384,B1$;:RESUME530ELSEONERRORGOTO0
```

Error trap on output.

```
1130 CLOSE:PRINT@448,"*ERROR:";ERR/2+1;"*";:FORT=1TO2000:NEXT:PR
INT@384,B1$;:PRINT@448,B1$;:RESUME440
```

Title Display.

```
1140 DIMP$(1,31):B1$=STRING$(14,32):H$="ABCDEFGHIJKLMNOPQRSTUVWX
YZ123456"
1150 NM$="BY G. CAGE":TR$="TRANSLATION BY":N2$=" S. MILLIKEN  "
1160 TI$="JUMBLE PUZZLE"
1170 PRINTCHR$(23);:PRINT@16,TI$;
1180 FORI=1TO14
1190 PRINT@(770+I*2),TI$;
1200 PRINT@(798-I*2),TI$;
1210 IFI=14THENTI$="PUZZLE JUMBLE":PRINT@(64*I-112),TI$;
1220 IFI>2THENPRINT@(64*I-112),TI$;
1230 PRINT@0,"";:SOUNDI*2+50,50
1240 NEXT
1250 FORI=1TO12
1260 PRINT@960,;SOUND(13-I)*2+50,50
1270 NEXT
1280 PRINT@720,STRING$(14,"-");
1290 PRINT@848,STRING$(14,"-")
1300 FORT=1TO200:NEXT
1310 FORI=1TOLEN(NM$)
1320 PRINT@(808-I*2),MID$(NM$,1,I);
1330 GOSUB1510
1340 NEXT
1350 FORT=1TO500:NEXT
1360 FORI=1TOLEN(TR$)
1370 PRINT@(812-I*2),MID$(TR$,1,I);
1380 GOSUB1510
1390 NEXT
1400 FORT=1TO500:NEXT
1410 FORI=1TOLEN(N2$)
1420 PRINT@(812-I*2),MID$(N2$,1,I);
1430 GOSUB1510
1440 NEXT
1450 FORT=1TO500:NEXT
1460 FORI=1TO14
1470 PRINT@960,;GOSUB1510
1480 NEXT
1490 FORT=1TO500:NEXT
1500 RETURN
1510 PRINT@0,"";:FORK=1TO5:SOUNDRND(100)+50,5:NEXT:RETURN
```

Sound routine from 11/81 SoftSide. Lines 60000, 60130 and 60140 have been slightly changed.

```
60000 Z=0:FORX=1TO158:READY:Z=Z+Y:NEXT:IFZ<>15220THENCLS:PRINT"D
ATA BASE ERROR IN LINES 60060-60160, CHECK LISTING.":PRINT:LIST6
0060-60160ELSEY=86:X=255:POKE-1,0:IFPEEK(-1)<>0THENX=192:POKE-16
385,0:IFPEEK(-16385)<>0THENX=127
60010 POKE16562,X:POKE16561,Y:CLEAR50:A1=PEEK(16561)+2:A2=PEEK(1
6562):A=A1+A2*256:Z=A-1:FORX=1TO158:Z=Z+1:Z=Z+65536*(Z<32767)
60020 READY:IFY<0THENY=A1+ABS(Y):POKEZ,Y+256*(Y>255):Z=Z+1:POKEZ
,A2-(Y>255):NEXTELSEPOKEZ,Y:NEXT
60030 IFPEEK(16396)=201POKE16526,A1:POKE16527,A2ELSECMD"T":DEFUS
R=A1+(A2+256*(A2>127))*256:POKE14308,0
60040 IFPEEK(16807)+PEEK(16808)*256<>A+24THENPRINTCHR$(28);:A=US
R(0):PRINTCHR$(23);
60050 SOUND11,11:GOTO60170
60060 DATA58,166,65,50,-164,42,167,65,34,-165,62,195,50
60070 DATA166,65,33,-24,34,167,65,201,245,123,254,2,40,4,254
60080 DATA16,32,79,229,213,42,230,64,126,183,32,4,35,35,35,35
60090 DATA215,6,5,17,-156,26,190,32,104,19,35,16,248,43,215
60100 DATA43,34,230,64,241,241,241,241,197,213,215,205,55,35
60110 DATA229,205,127,10,42,33,65,34,-167,225,215,43,34,230,64
60120 DATA35,205,55,35,43,229,205,127,10,42,33,65,58,-167,60
60130 DATA183,87,24,4,24,48,24,44,66,62,9,211,255,16,252,66,62
60140 DATA10,211,255,16,252,58,64,56,230,4,32,7,124,181,40,3,43
60150 DATA24,228,175,50,154,64,225,209,193,215,195,30,29,83,79
60160 DATA85,78,68,209,225,241
```

Routine to poke screen save subroutine into M$.

```
60170 A1=PEEK(16561):A2=PEEK(16562):A2=A2-4:A1=A1-1:IFA1<0THENA2
=A2-1:A1=255:POKE16561,A1:POKE16562,A2ELSEPOKE16561,A1:POKE16562
,A2
60180 CLEAR5000:GOSUB7
60181 READD:IFD<>-999THEN60181
60185 A1=PEEK(16561):A2=PEEK(16562):A1=A1+1:IFA1>255THENA1=0:A2=
A2+1
60200 I=VARPTR(M$):J=PEEK(I+1)+256*PEEK(I+2)
60205 FORK=JTOJ+6:READD:POKEK,D:NEXT
60210 POKEK,A1:K=K+1:POKEK,A2
60215 FORK=J+9TOJ+22:READD:POKEK,D:NEXT
60220 IFPEEK(16396)=201THENPOKE16526,PEEK(I+1):POKE16527,PEEK(I+
2)ELSEDEFUSR0=PEEK(I+1)+256*PEEK(I+2)
60230 GOTO10
60240 DATA-999,1,0,4,33,0,60,17,126,245,26,119,241,18,35,19,11,1
20,177,32,243,201
```

TRS-80® SWAT TABLE FOR: PUZZLE JUMBLE (TRANSLATION)

LINES	SWAT CODE	LENGTH
5 - 90	TI	583
100 - 210	JI	348
220 - 330	DG	250
340 - 450	LL	261
460 - 570	GG	246
580 - 690	FN	251
700 - 830	GX	297
840 - 950	BZ	489
960 - 1070	YH	315
1080 - 1190	AQ	473
1200 - 1310	FQ	250
1320 - 1430	HU	188
1440 - 60030	PH	546
60040 - 60130	TT	525
60140 - 60220	JX	513
60230 - 60240	QF	85

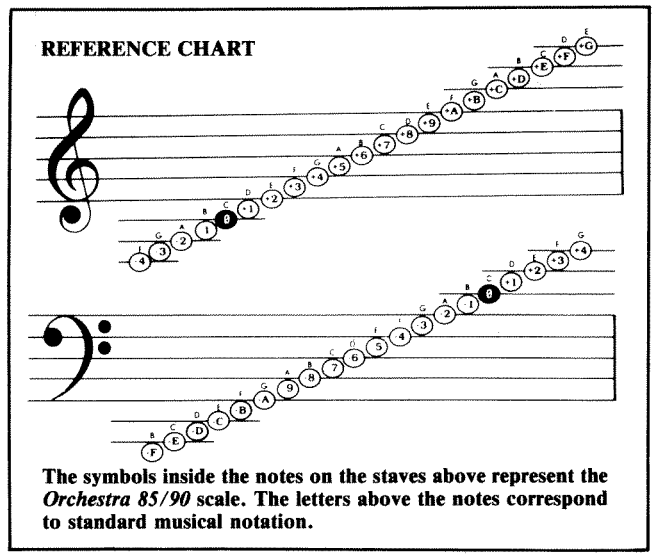
Orchestra 85 Composer's Edition & Orchestra 90

Reviewed by Robb Murray

Editor's Note: The following review assumes some previous knowledge of the Orchestra 80 and 85 music synthesizers. For further information concerning these products, see the reviews in SoftSide, October and November, 1981.

from Software Affair, Ltd., 858 Rubis Drive, Sunnyvale, CA 94087. Suggested retail prices: *Orchestra 85 Composer's Edition* and Orchestra 90 — $99.95. Upgrade from *Orchestra 85* to *Orchestra 85 Composer's Edition* — $20. Upgrade from *Orchestra 80* to *Orchestra 85 Composer's Edition* — $69.95.

Software Affair, Ltd., the manufacturers of *Orchestra 80*, released two new packages for the TRS-80® during the past year: the *Orchestra 85 Composer's Edition* (for the Model I), and *Orchestra 90* (for the Model III) which also incorporates "composer's" features. The two items are directly comparable, since they both bring the following features, for the first time, to the particular TRS-80 models on which they run: both also introduce the ORCHUTIL Utility Program, which is used to transfer and reformat files, replacing the more limited ORCHCOPY Program of the original *Orchestra 80*. Both systems perform quality stereo music, in up to four simultaneous parts (without modification). With a speed-up attachment, five parts can be played. Percussive sounds are an option that both products offer. The exact equivalence of the *Composer's Orchestra 85* and *Orchestra 90* can be seen by comparing their instruction manuals. (The *Orchestra 90* manual, incidentally, was arranged to keep all the material related to musical coding together — a good change from *Orchestra 80* and *85.)*



The symbols inside the notes on the staves above represent the *Orchestra 85/90* scale. The letters above the notes correspond to standard musical notation.

Time-Saving "Composer's" Features.

Of all the new developments, the most exciting are three new "composer's" (advanced editing) commands. My personal delight with the "composer's" features was instant; they're wonderful! I only wish they had always been available.

These commands are used primarily during the "proof-listening" of music. Two of the features, in particular, will cut hours out of your coding time. They allow you to simply play until you hear a problem, stop to correct it, and then go on from that point — all without the need to insert breakpoints that you'd only have to erase later.

To illustrate, imagine you are listening to some music you've coded and you hear an error. You interrupt playing by pressing "0". With *Orchestra 80*, in order to correct the mistake, you had to locate your musical error in the file by either (1) using a score; (2) string-searching your *Orchestra 80* file; or (3) counting passing measures as the music played. Now, none of these steps are absolutely necessary; all you do is hit "@" and, in the words of the manual, you will instantaneously be shown "Where It's At" (i.e., where the error is). The cursor will jump to where you stopped playing, (To be exact, it will go to the measure, part, or repeat boundary nearest the stopping point.) usually landing just after the error.

"Both systems perform quality stereo music, in up to four simultaneous parts (without modification). With a speed-up attachment, five parts can be played."

At this point, imagine that you go into the edit mode and fix the mistake you just found. Formerly, hearing the piece from that point on was not immediately possible; you had to either set a part-boundary at that point, or play the piece from the top of the file. Now, you need only position the cursor where you want to begin listening, get out of EDIT, and enter "!". The piece will compile and play right from the cursor. "!" Indeed!

The purpose of the third "composer's" feature may seem a little puzzling. When you stop playing and enter "?", the com-

"It is a good idea to let someone with an ear for music listen to the pieces, with a score in hand if possible, before releasing a piece to the public."

mand gives a display at the top of the screen of "the variables (registers and voice transpositions) associated with each voice at the location of the cursor." You might be a little inclined to ask "So what?", until you realize that this command can help you understand how someone else has set a particular voice to produce a unique musical sound. Such information might help in fathoming, for example, good percussive effects. You, too, can analyze the hoofbeats in Bryan Eggers' *Camptown Races* or the laser fire in Roy Niederhoffer's *Star Wars*. Command will help you along the way.

Bulletin Board Aesthetics

Scanning the telecommunication bulletin boards reveals that many users are coding *Orchestra 80*-type music for others to hear. Among the better boards for music are:

Roy Niederhoffer's board: (516) 482-8491
George Somer's board: (201) 842-7644
All Systems Go ("ASG") board: (305) 877-2829
(The "ASG" board, at last report, contained over 250 selections!)

Another board, recommended by Bryan Eggers, President of Software Affair: (303) 690-4566.

Having listened to a large number of such samples, I have some suggestions I think are worth mentioning:

1. Proof-Listen To What You Publish. There is no substitute for proof-listening — not just proof-reading — musical code. When completely random notes pop out of a coded selection, you sometimes suspect that the code has passed someone's eye-test, but not the test of an educated ear. If the code out there on the boards is indeed being written by people with "tin ears" or limited musical background, it is certainly a tribute to their coding accuracy that so few inappropriate notes are sounding. However, for anyone in doubt, it is a good idea to let someone with an ear for music listen to the pieces, with a score in hand if possible, before releasing a piece to the public.

2. Code With Stereo In Mind. *Orchestra 80* files will play on *Orchestra 85*. (Although the reverse is not always true.) *Orchestra 80* users, however, hear their music from a single point source, whereas *Orchestra 85* users hear their music from two points. The choice of voice numbers for an *80* file will therefore determine the kind of stereo sound that many *85* users will hear. If you have an *80* (mono) piece to put on a bulletin board, remember that, on *85*, Voices 1 and 2 default to Channel A, and Voices 3 and 4 to Channel B. If you code a two-part piece using voices 1 and 2, both will sound from the same speaker on *85*; using Voices 1 and 3 will, however, bring the voices out of opposing speakers. Basically, whenever voices within a selection seem to "take turns," or answer each other, you should code so these voices come from separate speakers. This greatly adds to the charm of the music. (It is true that *85* users have the "Z" variable available for switching voice from speaker to speaker, but they may misinterpret your intentions.)

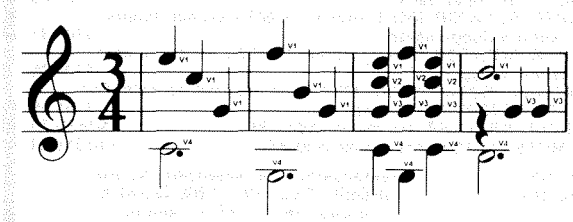
Conversely, *85* files can often be coded simply enough for *80* to play. Before releasing an *85* file to a bulletin board, check whether tone register definition, percussion, or the "Z" parameter are unnecessarily present. If you can eliminate these, *80* can play the file, too, and you'll be assured of more listeners.

3. Choose Memorable File Names. When titling pieces, remember that you can help the listener find your selection again, among the dozens of others out there, if you choose meaningful and memorable directory names, words that are distinct and appropriate. For ideas, sample the names others are choosing.

Unique naming can be especially tricky with classical transcriptions. The user may be tempted to pick very general titles that quickly come to mind, such as "Concerto," "Mozart," or "Piano." Be careful. There are many three-part inventions by Bach, for example, and neither the label "Three" nor "Bach" seems meaningful.

4. Space Pieces Apart. It is a good practice always to begin and end pieces with rests. Did you ever call up a piece and hear it begin playing even before your disk drive stopped spinning? Inserting a few rests at the top of your files will eliminate this problem. Then, too, a

MUSIC SAMPLE



The above music sample, reprinted from the *Orchestra 85/90* manual, illustrates the voicing capabilities of the synthesizers.

file beginning with notes, as opposed to rests, will often generate an opening buzz or snap, as the sound system crackles to life, masking the beginning notes of the piece. This is another good reason to buffer the top of a musical file with rests, so that any crackling will not interfere with the music itself. Rests are good at file ends, too, to avoid destroying the mood of a piece entirely when more musical numbers are queued up to be called automatically. You don't want the disk drive to click into motion to find another piece just when your moving rendition of *A Time For Us* has finished its last note!

More Products — Now And Later

Since my review of *Orchestra 85* in ***SoftSide***, November, 1981, Software Affair has made steady strides forward in its product line. One of their recent new items is the *Piano* (retail price, $39.95). It is used in conjunction with either *Orchestra 85* or *90* and requires 100% speedup (to at least 3.5 Megaherz). *Piano* attacks notes percussively, as does a piano, but allows them to die away at a faster or slower rate, as the user chooses. The results can be joined sounds, like those of an organ, or disjoined ones, like those of a banjo. The *Piano* also allows individual notes to be emphasized or muted. Larry Alexander, a user who experiments extensively with external filtering systems (such as the *Polychorus* model of the Electro-Harmonics System), is now using *Piano* to good effect. Another product, *Fanfare* (retail price, $24.95, on disk, for Models I and III) is a music synthesizer that produces one to four separate musical parts for use with other BASIC programs. It requires no special hardware for its synthesis and puts out its signals directly through the cassette port.

Although the *Orchestra 80*-type products are still more powerful and versatile than anything yet coded has shown to full advantage, creative musical coding is going on all the time. Software Affair has released a "Greatest Hits" diskette ($20, or $10 each for the two-volume cassette version). Of recent note are the many excellent novelty arrangements by Larry Alexander of Tyler, Texas, such as his *Hoedown Medley* (virtuoso fast passages), *In My Merry Oldsmobile* (complete with honking horn), and *Tea for Two* (with the old soft shoe).

Still The Best

As the *Orchestra 80* family develops further, users still wait for an *Orchestra 80/85/90* newsletter, as well as additional documentation, such as a table showing the register settings that would give a wide variety of instrumental and percussive sounds. A more contoured system for controlling tempo and dynamic changes, and the ability to fine-tune the system to accord with another musical instrument are definitely needed.

All things considered, however, the *Orchestra 80* line has three great strong points: its popularity (which has resulted in hundreds of available music files on the bulletin boards), its performance as a musical product, and its price.

The new *Orchestra 80* "Composer's" commands are most helpful. For the Model III owner, *Orchestra 90*, with its many amenities, is clearly the product of choice, and the premier system in its class.

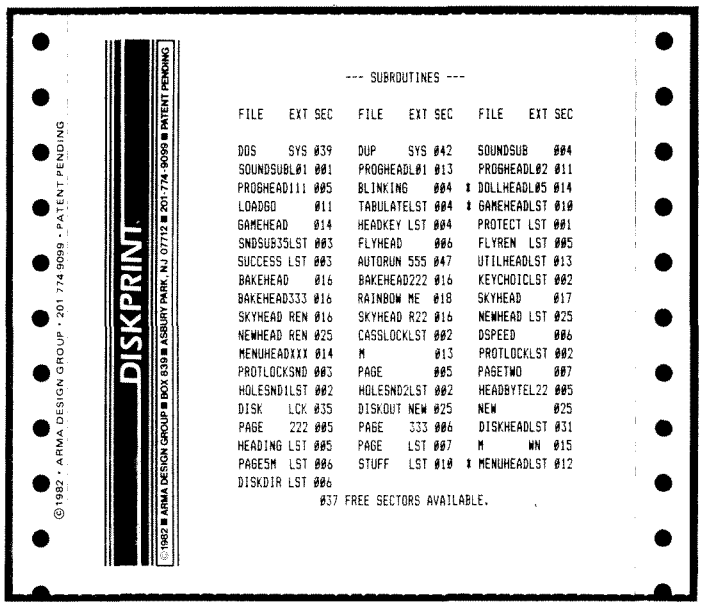
The Arma Design Group Proudly Announces

Diskprint®

If you're using a computer with one or more disk drives, have we got a product for you!

DISKPRINT is beyond a doubt the most useful new package to become available since the computer revolution began.

DISKPRINT will forever free you from having to search for or through that looseleaf binder with all your disk directories listed (if you have one!), or having to spend valuable minutes or hours scanning through each disk directory on-screen looking for that one particular program. DISKPRINT does for you what your computer was supposed to do for you, it eliminates the drudgery of paper shuffling and wasted time.



HERE'S HOW DISKPRINT WORKS...

Simply insert DISKPRINT into one of your disk drives and power up the system. DISKPRINT is self-booting and will load and run automatically. Now load our DISKPRINT custom forms into your printer and switch it on. Insert any disk in your library into your drive and answer the two prompts (which drive and disk title). If you do not choose to title your disk, DISKPRINT will default to its own. Press (RETURN) and DISKPRINT will print out a listing of every program on the disk along with extensions and sectors used as well as the number of free sectors available. This whole process will take about 20 or 30 seconds of your valuable time, so you can run through a one-hundred disk library in about 45 minutes or so. After your form has been printed, advance the next form to the print bar, insert another disc and go again. When you have run through your library, simply remove the tractor strips from the forms and insert them in the jackets with the disks. ***Every disk in your library will have a printed directory in the jacket with it at all times!***

DISKPRINT is delivered with our program on disk, 50 custom DISKPRINT forms and complete instructions. Refill packs of 50 forms each are available at your local dealer or directly from us by mail. (An order form is enclosed).

DISKPRINT is currently availble for:
• ATARI 400/800 & ATARI 825 or EPSON MX Series Printers • IBM-PC & IBM or EPSON MX Series Printers • APPLE II/APPLE II+ & EPSON MX Series Printers • TRS 80 Models I/III & TRS Series Printers • CMB 4032/8032 & CMB/EPSON MX Series Printers

DISKPRINT is now being converted to run on many more makes and models of computers. Call or write for further information.

COMPLETE DISKPRINT KIT FOR ALL MACHINES $24.95
Refill Packs of 50 Custom Forms $8.95
Please add $1.50 per order for P&H

ARMA DESIGN GROUP

"We make the computer revolution a little less revolting." ©

ARMA DESIGN GROUP • P.O. BOX 839 • DEPT SS1182 • ASBURY PARK, N.J. 07712 • 201-774-9099

NAME _____

ADDRESS _____

CITY STATE ZIP _____

TELEPHONE_____**COMPUTER SYSTEM:**_____

☐ **MASTERCARD** ☐ **VISA** ☐ **MONEY ORDER** ☐ **CHECK** • **INTERBANK# (M.C.)** _____

CREDIT CARD # _____

The Rotberg Synthesizer

by Ed Rotberg

The *Rotberg Synthesizer* is a Machine Language music playback program for an Atari® computer with one disk drive and 32K RAM. It is provided as this issue's DV enhancement for the Atari.

Editor's Note: The program and music files are recorded on the reverse side of the disk. To run, remove the BASIC cartridge, place the disk in Drive 1 and boot the system. The program will run with the BASIC cartridge in place, but you must then type "DOS" to run the program after you see the "READY" prompt on the screen. This side of the disk is copy-protected and contains its own operating system. Do not try to copy this side of the DV, or to access the files from Atari DOS.

When the program starts, a menu will be displayed. Follow the screen prompts to make your selection and play a song. Five songs are provided with the synthesizer: ***New Country, Prelude and Fugue in C Minor, Bolero*** *(unfinished),* ***Pokey Percussion,*** *and* ***Disco Dirge.***

The following is an edited reprint of an article from ***Antic*** *magazine, reprinted by permission of Antic Publishing Co., 297 Missouri Street, San Francisco, CA 94107.*

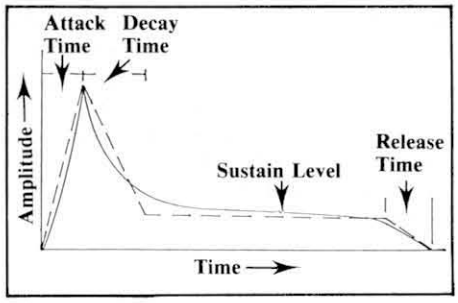
Reprint: ANTIC™ The Atari® Resource, 297 Missouri St, San Francisco, CA, 94107

Never heard of *The Rotberg Synthesizer,* right? Well, this is a program that has been passed around on dark street corners, and only talked about in veiled whispers. (Have you ever seen a veiled whisper — or heard one?) It does a pretty good job of shaping POKEY sounds into something approximating a musical instrument.

I have always had mixed feelings about disclosing the inner workings of the *Synthesizer* for a number of reasons. First of all, the whole thing really started as a gag when I was working as a programmer in Atari's Coin-op division. It was first designed as a means to create "pre-programmed, electronic disco" for one of the programmers, a disco freak, who was getting married in a short while. Thus, the birth of both the *Synthesizer,* and *Disco Dirge*, written especially for the impending bachelor party by Dan Pliskin, another ex-Atarian.

I don't feel there is much to be gained by copying someone's program and running it. While techniques can be learned this way, creativity is badly hampered. I would much rather discuss approaches to solutions, rather than the solutions themselves, thereby letting others exercise their creativity in the implementation. Therefore, I will discuss the synthesizer program in a general fashion, and lay out the specific data structures used by the program, rather than give a listing.

The single most important reason that *The Rotberg Synthesizer* sounds so different and, in my obviously tainted opinion, significantly better than the vast majority of music written for the Atari, is its ability to apply "envelopes" to the frequency, and perhaps more importantly, to the amplitude of each note. "Envelope" is a term used to describe the temporal variation of some aspect of a sound. In this case, the aspects varied are frequency and amplitude. "Attack, Decay, Sustain, and Release" (ADSR) has become the de facto standard method of specifying an amplitude envelope, as most of the electronic performance synthesizers use this method. In order to give you a better idea of what these values mean, a graph is given below, indicating these terms with respect to a harpsichord-like amplitude envelope.



I will present one approach to implementing envelopes in a music generating program, specifically, *The Rotberg Synthesizer.*

Throughout this article, I will refer to various registers in the POKEY chip, and certain functions of the POKEY. I will in no way describe the capabilities of the chip; it is assumed that you already have, or will obtain, this knowledge. Also, the techniques described here are, for the most part, useless in BASIC due to execution time, nor is any consideration given to interfacing with BASIC. I will not be discussing any music/sound editing programs, only the means of generating the music and sounds.

There are basically two major classes of sound generation used on the Atari: static and dynamic. The first consists of nothing more than storing a few values to the various POKEY registers, and sitting back and listening. The capabilities of this approach quickly become exhausted. More useful, and far more interesting to the programmer, are the dynamic sounds — those in which the values stored to the POKEY are constantly changed during the sound's duration. The three approaches to dynamic sound generation are:

1) Algorithmic: A short routine is used to calculate the values to be stored. The possibilities are limited only by the imagination of the programmer, time of development of the routines, and their execution speed.

2) Table driven: A short program keeps an index to a lookup table to determine what values are to be stored into POKEY during that time interval. New sounds can be generated very quickly by slopping some new values into the tables until you find a sound you like.

3) Interpretive: A small interpreter program will read instructions and data from a command stream, causing the sounds to be generated by a few, preset rules. A major advantage of this method is that it keeps the data tables short, as compared to a purely table-driven approach, while allowing the programmer to add new "rules" and their corresponding instructions to the interpreter, as the need arises. It also allows for fairly quick sound development.

First, let's go over just what the *Synthesizer* is capable of doing. It has the ability to produce sound on all four channels of the POKEY, simultaneously. The basic unit of sound is called a note, since this program was intended to be primarily a music synthesizer. However, it is capable of generating a wide variety of sounds. The frequency of the note is specified by eight bits which define either a pointer into a table of frequencies, or the actual frequency itself. (This is an implementation decision, and each method has its merits and drawbacks.) If the actual frequency is stored, the note must also specify the "noise content or distortion" value to be stored in the control register along with the sustain volume for each channel. Each note can specify a 4 bit value for its sustain volume and can have a duration, specified by 16 bits (2 bytes). This duration is relative to the current tempo. The tempo is specified by an 8 bit value, which is used as a delay loop counter. The tempo can only be changed relative to its current value by a 2's complement add of any 8 bit value. Note that in versions of the *Synthesizer* that run during the vertical blanking interval, such as the Atari POP Demo program, the tempo feature is not implemented, as the timing interval is fixed at 60 hertz. Each channel can specify its own current envelope table, controlling the attack/decay of either amplitude, frequency, or both. Attack and decay are not specified as rates or times, but rather as a table of digitized amplitudes during the attack/decay period. This period can cover a span from a few milliseconds to a few seconds. The values specified by the envelope are always relative (2's complement additive) to the specified frequency/sustain volume of the note. Care must be

taken not to wrap either of these values, unless, of course, that is the intended result. At the present time, "Release" is not implemented. The *Synthesizer* has the ability to repeat a section of music up to 100 (hex) times. These repeats may be nested without any restriction except that the total number of repeats in a piece of music must not exceed 100 (hex). The *Synthesizer* can also play phrases. I have chosen not to implement the four separately tracking stacks necessary to allow for nesting of phrases, although this is certainly simple enough to do. Each phrase must specify its own return. In addition, any channel's instruction stream can cause AUDCTL to be changed on the fly. That's about it. The rest is left up to the cleverness of the programmer in writing the instructions to the driver. This driver is only about 33 (hex) bytes long...and could easily be shorter!

In its current form, *The Rotberg Synthesizer* supports seven instructions:

1) Repeat
2) Set/change envelope
3) Set/change AUDCTL register
4) Play Phrase
5) Return from Phrase
6) Change Tempo
7) Play 1 note

The *Synthesizer* processes four sets of these instructions simultaneously, one for each channel in POKEY. Each instruction stream is made up entirely of these instructions, in addition to a STOP directive that is only valid when encountered in channel 1's instruction stream.

The data structure format for each instruction follows, where each cell represents one byte; all value/ranges are given in hexadecimal.

REPEAT: op-code = FF

FF
nn
ll
hh
ii

FF = REPEAT op-code
nn = repeat count (0 = 100, 1 = NOP, count indicates number of times section is to be played)
ll = low byte of address of 1st instruction of section
hh = hi byte of address
ii = index into RAM table for this section's repeat counter

This instruction has the effect of conditionally repeating a section of the instruction stream a specified number of times. Because each REPEAT instruction has its own loop counter in a RAM table 100 (hex) bytes long, any amount of nesting of these repeat instructions is allowed, as long as the total number of repeats in any composition is 100 or fewer. Each repeat can play its section up to 100 times. This instruction appears at the end of the section to be repeated, and refers to the first instruction of that section in its operand field.

SET ENVELOPE: op-code = FE

FE
ll
hh

FE = set envelope op-code
ll = low byte of address of envelope table
hh = hi byte of address

This instruction sets the pointer to the current envelope table for that channel. A "set ENVELOPE" instruction must precede the first note instruction on any channel. Envelopes may be changed at any time, and as often as necessary.

CHANGE AUDCTL: op-code = FD

FD
cc

FD = change AUDCTL op-code
cc = new audctl value

This instruction is used to change AUDCTL on the fly. This represents powerful, dynamic control of the POKEY. It may be used from any channel, but in practice, it is best only altered from one channel within a piece, as AUDCTL can affect ALL channels.

CALL PHRASE: op-code = FC

FC
ll
hh

FC = CALL PHRASE op-code
ll = low byte of address of 1st instruction of phrase
hh = hi byte of address

This instruction will transfer control to a phrase which can be "called" any number of times. In the current implementation, there is no nesting of phrase calls, (i.e., only one level of calling a phrase). Phrases themselves may therefore use any instructions other than "CALL PHRASE", and must terminate with a RETURN instruction. Note that, while possible, it is dangerous to have two channels use the same phrase, especially if that phrase contains REPEAT instructions.

RETURN FROM PHRASE: op-code = FB

FB

FB = RETURN op-code.

This instruction is used to return from a phrase.

CHANGE TEMPO

FA
tt

FA = CHANGE TEMPO op-code
tt = 2's complement delta change to TEMPO

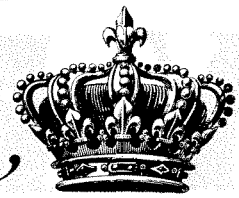
This instruction is used to change the current tempo by a 2's complement delta value. This instruction can appear in any channel, and obviously affects all channels.

NOTE: op-code = FA

ca
ff
dd
ee

c = control nibble (upper nibble of volume)
a = sustain volume
ff = sustain frequency or pointer to freq. table
dd = low byte of 16 bit duration
ee = hi byte of 16 bit duration. Duration is relative to tempo. For convenience, a value of 100 (hex) is usually used to represent a whole note. This means that for long durations, the high byte (ee) of the duration represents a measure count in 4/4 time.

Royal Software

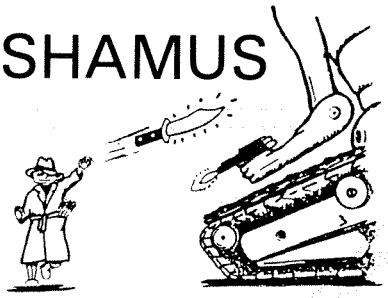


WE DEAL EXCLUSIVELY IN PRODUCTS
FOR THE ATARI (THE BEST) COMPUTER

YOUR MARKETPLACE FOR:

* ATARI

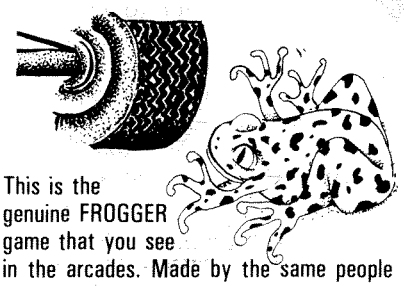
SHAMUS



32 Rooms with four levels of INTENSE arcade action with this game. Every room bristling with DANGER.

16K Disk or Tape $31.50

FROGGER



This is the genuine FROGGER game that you see in the arcades. Made by the same people that made Jawbreaker (One of the top ten sellers.)

32K Disk 16K Tape $31.40

Just like the arcade game that has been so popular. This is one of our BEST sellers for all ages.

$35.10 24K Disk **$39.90** 16K Cart.

HARDWARE

Prices listed with (*) are the CASH discount price, charge prices are higher.

800 Computer 16K	$648
800 Computer 48K	$757
400 Computer (used)	$199
810 Disk Drive	$448
850 Interface	$178
410 Recorder	$ 78
16K Ram module	$ 69
32K Ram module	$109
Percom Disk Drive	$648
48K Ram module for 400	$197
WICO JOYSTICK	$ 26
Trac Ball controller	$ 63
800/400 Dust cover	$ 9
10 Blank Disks	$ 28
Mark II Modem	$ 99
Alien Group Synthesizer	$153
Epson MX-80FT/Plus	$588
Amdek Color 1 monitor	$387
12' Joystick Extension Cord	$ 10
Computer Paper 1000 8½x11	$ 17

Trade in your 400 for a NEW 800 Computer Call for details.

SOFTWARE

ZAXXON
Now you acn get that powerful 3-D game for the Atari. Made by one of the BEST game makers Datasoft Order early to get yours D $35.10

EASTERN FRONT
The MOST popular of all the APX programs and truly one of the best war games for the Atari. D,T $27.90

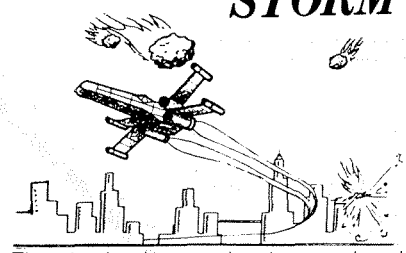
SAMMY THE SEA SERPENT
A must for the little computerists ages 4-7. The kiddies LOVE this one and "Sammy" too. Tape $15.90

BAJA BUGGY
If you have been waiting for a good Car racing game for the Atari then this is it! Disk or Tape $29.90

BUG ATTACK
You must defend your garden from hungry bugs. But, look out these bugs shoot back! Disk/Tape $26.90

FREE
ATARI PRODUCTS CATALOG
with any order or
send $1.00 refundable with order.

METEOR STORM



The city is about to be destroyed and ONLY you can save its people from destruction. FAST, arcade type action game.

16K Disk or Tape **$29.95**

O'RILEY'S MINE

16K Disk or Tape $31.50

A SUPER graphic game in which you are the miner and must watch out for mine flooding and monsters of the deep.

CROSSFIRE

The aliens have landed and it's your job to save the city. (If you can!) TOP RATE GAME. ARCADE QUALITY. HI-RES Graphics & Sound. 16K TAPE 32K DISK $26.90

VISA master charge

TO ORDER CALL (503) 683-5361

HOW TO ORDER: Send check or money order or phone your order using your charge card. Ask about our 48 hour delivery service. Shipping on programs is $2.00 per order in USA or $3.90 for the 48 hour AIR service. Call for Hardware shipping costs. Prices subject to price change without notice.

Store Hours
8 am – 6 pm
Mon. – Sat.

Royal Software
2160 West 11th Ave., Eugene, Oregon 97402

POKEY PLAY

***Pokey Player* is a music editing/playing utility for the Atari 400/800 with 32K and Atari BASIC.**

Of the three special chips in the Atari computers, the POKEY (POrt and KEYboard controller) is the most overlooked and underused. The POKEY is a rather versatile chip, used by Atari not only in their 400/800 computers, but in their arcade games as well, such as *Missile Command*. Yet, with few exceptions, the only sounds one ever hears from this chip are explosions and special effects for outer space, shoot-em-up type games. Isn't it about time somebody put POKEY through its paces?

Presenting...*Pokey Player*

Enter *Pokey Player*, a music playing utility for Atari 400/800 computers. This music player is designed to allow full access to the wide variety of sounds generated by the POKEY chip. Using *Pokey Player*, it is possible to produce music of excellent quality.

Pokey Player Features

Three simultaneous voices are supported by *Pokey Player*. The primary voice has an eight octave range (C0 to C8); the two secondary voices each have a five octave range (C1 to F6). Middle C is C4.

For a nicer sound, an automatic decay feature has been implemented. This causes the volume to fade towards the end of a note. The automatic decay can be defeated in order to tie notes together. A wide range of tempo selections is available, and the tempo can be changed at any time.

A sequence of notes can be repeated up to 255 times, or forever. Repeats cannot be nested, but each of the three voices can repeat independently.

Music data is stored in a compact form for minimal memory usage. In most cases, just a little more than one byte is used for each note.

Other advanced features will be described in a later issue of ***SoftSide.***

A Three Step Process

To develop music for *Pokey Player*, three programs are provided. The EDITOR is used to easily create music files. The COMPILER converts a music file from the EDITOR into a file of DATA statements and provides a listing of the music part. The DATA statements can then be merged with the PLAYER program. Running PLAYER, after it has been merged with DATA statements for three different parts, produces real music from your Atari computer.

The Editor

This program takes a few moments to initialize, but it is certainly worth the wait. First, a joystick must be plugged into the first controller jack. Pitches, rests, ties, and durations are all selected using the joystick.The general procedure is: push the stick up and down to move from one section to the next; push the stick left or right while in a particular section to actually select a certain option. Other options that require use of the keyboard will be explained later.

After the program has completed initialization, the main section is highlighted in a bright white, while the other sections are displayed in a dark gray. The main section consists of a grand staff (treble and bass staves), and twelve piano keys. As the joystick is moved left or right, a note displayed on the grand staff moves up and down, an arrow points to the proper key on the piano keyboard, and a little bell sounds. The pitch of the bell corresponds to the position of the note on the grand staff and the indicated piano key. A full eight octaves are available just by moving the joystick. The note and octave number are also displayed below the section. The display of the current note in these several forms should facilitate note entry.

If a rest is desired instead of a note, the joystick must be pushed up once to highlight the section where "RST" is displayed. The main section will be dark. By pushing the stick left or right, the selection will alternate between the current note and the rest. This is reflected in the line below the main section where the current note and octave are displayed.

Once the note value has been selected, be it a regular note or a rest, a length must be specified. Push the joystick down twice to highlight the area below the main section. Various

ER: EDITOR & COMPILER

by Craig Chamberlain and Harry Bratt

durations are displayed, from thirty-second note to whole note. All of these notes, except the thirty-second note, can be dotted. A dot signifies that a duration of one half of the note's normal duration should be added to the normal duration. Thus, the total duration of a dotted note is one and one half times the duration of the same note without the dot. For special cases, a third of an eighth note triplet is available at some tempo selections. Three of these special durations are equivalent in duration to one quarter note. The joystick is moved left or right to select one of them.

Sometimes notes are connected using a tie or slur. Whereas notes are normally played separately, tied or slurred notes are played together without any break in tone. A tie could be thought of as a method of extending the length of a note. Slurred notes create a smooth, legato effect. To choose this option, push the stick down once to highlight the section marked "TIE". Push the stick left or right to alternate between selecting a tie/slur or no tie/slur. It is not necessary, or possible, to use this option on a rest.

Having set all of these parameters, it is time to create a note by pressing the red trigger button on the joystick. When the button is pressed, an image of the selected note will appear in a little box at the bottom of the screen and then move to the left. Every time the button is pressed, the main section will again be highlighted so the next note can be chosen. For all succeeding notes, only those parameters which are different from the previous note need be changed. For example, if a music part requires a series of half notes, the half note duration will have to be set only once until the pitch is changed. The tie/slur must be selected each time it is needed.

After selecting several notes, it may be necessary to change one. To do this, the joystick must be pushed down until the bottom section, with the box, is highlighted. Moving the stick left or right will cause the notes to move left or right through the box, so that any note can be changed. Two key commands are used to quickly move to either end of the music part — the B key to move to the beginning, and the E key to move to the end.

To delete a single note, move the joystick so that the note to be deleted shows in the box, then press the CTRL and DELETE combination. The note will disappear, and any notes to the right will move to the left to fill the gap. To totally erase all notes, press the SHIFT and CLEAR key combination, followed by the Y key in response to the confirmation prompt.

Inserting a note is accomplished by pressing the CTRL and INSERT key combination. An opening will be created between the note in the box and the note to the *left* of the box. A note can be inserted at any place in the music. The function will work slowly, however, if inserting is done near the beginning of a long piece. As notes are added, the free memory counter displayed above the piano keys is decremented. When there is no more memory, no more notes can be added or inserted. Any attempt to do so sounds a warning buzzer.

Music files can be saved and loaded for editing purposes. To save or load a file, press the respective key, S or L, followed by the D or C key, as prompted, for disk or cassette. When the disk is chosen, a filename must also be specified. Any SIO errors will be reported. Files too large for memory cannot be loaded.

There are three special command options currently available in the EDITOR. These special options are displayed, along with the normal notes, in the section at the bottom of the screen.

The first special option allows a sequence of notes to be repeated. Press the R key, followed by the RETURN key for confirmation, to indicate the start of a repeat. A character consisting of two vertical lines and two dots will be shown. The number of times the sequence will be repeated must then be entered, with valid numbers ranging from one to 255. Entering a value of zero will cause the sequence to repeat forever. The end of the repeat sequence is marked by again pressing the R key. This time, the character will be two dots, followed by two lines. Repeats can be used any number of times, but they cannot be nested.

The tempo can be changed by pressing the T key. Fourteen different tempo choices are available. Press the number keys 0 to 9, and the letter keys A to D, to display each tempo. The metronome value will appear, along with notices of any duration values not available with that tempo. Sometimes, the thirty-second note or the eighth note triplet is not available. To select a tempo, press the RETURN key when your choice is displayed. It is best to use tempo changes only on the first voice, and only at a point in the music when all three voices will be starting a new note.

The last special option is used to indicate the end of a part. This is known as the "stop" or "halt" command, and is abbreviated HLT. This command must be followed by a note, usually a rest.

Despite its many functions, the EDITOR is very easy to use. It will streamline creation of music for *Pokey Player.*

The Compiler

The notes in a music file from the EDITOR are stored in a format convenient for editing. Unfortunately, this tends to nearly double the memory requirements for a musical part. The COMPILER is used to convert the music file into a more compact form to be written to a file as a bunch of

DATA statements. This file is then merged with the PLAYER program by use of the ENTER command in Atari BASIC.

The program first requests the music source file, the one used by the EDITOR. A full device specification must be given, including the D: and C:, and the filename if the disk is being used. The COMPILER then asks if the part is for a primary or secondary voice. Remember that secondary voices have a range of slightly more than five octaves (from C1 to F6), and an error will be reported if the voice is secondary and the note data goes beyond this range.

Just as a device specification was given for the input file, so must one be designated for the output file of DATA statements. The starting line number of the statements, and the line number increment, must also be specified. The practice used thus far has been to start the first voice at line 3100, the second at 3200, and the third at 3300. The primary voice **must** assume the third position. Numberings is always done by two. The first number in the first DATA line tells how many bytes are to be READ for the part.

While compiling, a status report is also produced. This report describes various important characteristics of the output file, and is usually sent to either the screen (E:) or the printer (P:). Each line tells the byte count, running frame count, byte value in decimal and hexadecimal, and other information relative to the particular byte. If the byte is a note, the note octave, name, and duration is given. The plus and minus signs are used to indicate sharps and flats. A letter "D" will appear in a separate column if a note is dotted. Dotted notes are generated by tying a note of half the indicated duration to a note of the full indicated duration. Tied notes are marked by a letter "T" in a special column. Special option commands have names associated with them, as listed below.

PPHEAD beginning of a repeat
PPTAIL end of a repeat
PPTEMP tempo change
PPSTOP halt command

Extra, unlabeled bytes are required for some special commands. These will be explained in a later article, as will the +/- signs, followed by numbers, that occasionally occur in the octave column.

The COMPILER changes source files into a more convenient form, and also produces a handy report useful for documenting or editing a music part.

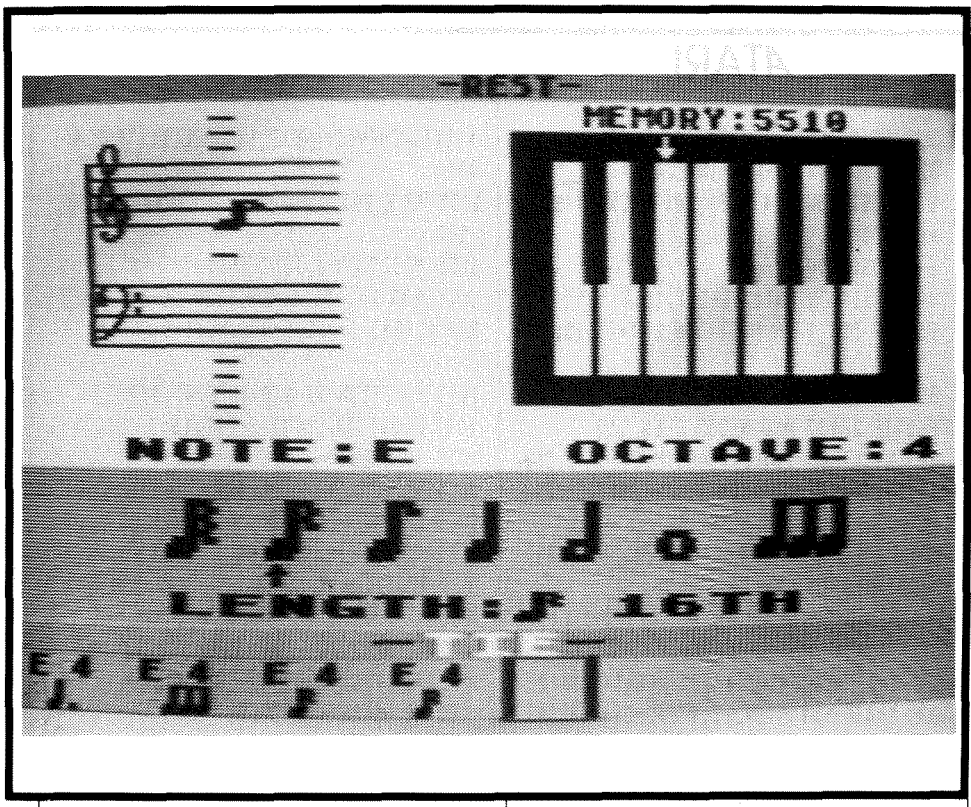
The Player

Once three different voices have been compiled into DATA statements, it is time to add them to the *Player*. First type the Atari BASIC command NEW, LOAD the *Player*, then ENTER each of the three DATA files. The *Player* will then be set to RUN. After initializing for a few moments, you will hear the music, and the program will END. The *Player*, complete with DATA statements, can then be SAVED for later listening pleasure.

Two demonstration pieces have been provided. *Happy Birthday,* and the March from *Scipio* by Handel can be found immediately following the main program listing for *Player*. Each is listed separately as a data file running from lines 3100 on up. To use any data file, first LOAD the *Player* program into memory, then ENTER in the data file. Be sure the data file is in the LIST format. For example: After typing in the *Scipio* piece by Handel, type LIST "D:SCIPIO" (or LIST"C:" for cassette). Then type LOAD "D:PLAYER" (or CLOAD for cassette) to load the *Player* program. Now type ENTER"D:SCIPIO" (or ENTER "C:" for cassette) to merge the music data into the *Player* program. A simple RUN command will now start the music. You can use this method for both music files.

This concludes the description of the process used to create and play music. *Pokey Player* was developed with technical assistance from Lee Actor. The authors are most anxious to hear comments regarding the programs, and to learn of any difficulties or bugs. Any music produced with *Pokey Player* is certainly welcome. Anyone wishing to use *Pokey Player* in a commercial application, however, must first acquire permission. Please send all correspondence to ***SoftSide*** magazine.

Despite some serious limitations, *Pokey Player* is flexible and memory conservative. The demonstration pieces are just a peek at what is possible with this program. Be watching future issues of ***SoftSide*** for PART TWO which reveals all the other special options and surprises built into *Pokey Player.*




```
SS SS SS SS SS SS SS SS SS SS SS
SS                             SS
SS         Atari  BASIC        SS
SS           "Editor"          SS
SS     Author:  Harry Bratt     SS
SS      Copyright (c) 1982      SS
SS SoftSide Publications, Inc  SS
SS                             SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
10 A=PEEK(106)-16:POKE 106,A:GRAPHICS
0:GOSUB 4900
12 OPEN #2,4,0,"K:"
15 DIM NN$(28),NTE$(63),D$(14),F$(12),
SL$(18),TIE$(6),INS$(3),TEM$(45)
17 NMAX=INT((FRE(0)-256)/2):DIM MSC$(N
MAX*2):POSITION 30,3:? NMAX
20 OCT=4:LNT=6:N=48:ON=1:LT=3:OO=67
25 TEM$="56 60 64 69 75 82 90 10011312
8150180225300---";TEM$(43)=CHR$(14):TE
M$(44)=CHR$(14):TEM$(45)=CHR$(14)
30 POKE 82,0
35 T$="C "
40 NN$="B C C#D E$E F F#G A$A B$B C "
50 SL$="SAVING TOLOAD FROM"
60 TIE$="tieTIE"
70 NTE$="% 32ND   & 16TH   ' EIGHTH (
QUARTER) HALF   * WHOLE  +,TRIPLET"
75 INS$=CHR$(157):INS$(2)=INS$:INS$(3)
=INS$
80 U=USR(ADR(M$),85)
```

```
100 REM MAIN ROUTINE
105 U=USR(ADR(U$),48)
107 OSEC=2:SEC=OSEC:PSEC=6
110 POSITION 30,3:? NMAX-NN;" "
111 S=STICK(0):IF  NOT STRIG(0) THEN 7
00
112 IF PEEK(764)<>255 THEN 310
115 IF S=15 THEN 111
120 IF S<13 THEN ON SEC+1 GOTO 110,410
,200,110,600,450,800
130 SEC=SEC+(S=13)-(S=14):SEC=SEC+6*(S
EC=0)-6*(SEC=7):IF SEC=3 THEN 130
140 POKE 1568+OSEC,PSEC
145 PSEC=PEEK(SEC+1568):POKE 1568+SEC,
10
150 OSEC=SEC:U=USR(ADR(U$),24)
155 FOR I=1 TO 45:NEXT I
160 GOTO 110
200 REM NOTE CHANGER
205 RST=0:SKIP=0
207 POSITION 22+NTE+(NTE>4),4:? "_"
210 NE=(S<8)-(S>8):NTE=NTE+NE:OCT=OCT-
(NTE<0)+(NTE>11):NTE=NTE+12*(NTE<0)-12
*(NTE>11)
215 N=12-NTE-12*(NTE=12):N=N+(7-OCT)*1
2:IF N>96 OR N<0 THEN S=16-S:GOTO 210
220 U=USR(ADR(U$),N):GOSUB 230:GOTO 11
0
230 POSITION 7,14:N2=NTE*2+3:T$=NN$(N2
,N2+1)
240 ? T$;" "
245 Q=ASC(T$(1,1)):CHANGE=(Q<>OO):OO=Q
250 POSITION 18,14:? OCT
255 POSITION 22+NTE+(NTE>4),4:? CHR$(2
```

```
7);CHR$(157)
260 IF SKIP THEN 270
262 IF  NOT CHANGE THEN MOVE=0:GOTO 27
0
265 MOVE=2+4*((N=48 AND S<8) OR (N=47
AND S>8) OR (N=49 AND S>8) OR (N=46 AN
D S<8))
267 MOVE=256*(S<8)-NE*MOVE
270 U=USR(ADR(M$),MOVE)
280 RETURN
300 REM PROCESS KEYPRESS
302 REM 83,125,76,254,255,66
310 GET #2,A
312 IF A=254 THEN 1110
314 IF A=255 THEN 1210
316 IF A=76 THEN 505
318 IF A=125 THEN 980
320 IF A=82 THEN 1310
322 IF A=72 THEN 1410
324 IF A=84 THEN 1510
328 IF INS OR NN=0 THEN 110
330 IF A=83 THEN 505
332 IF A=66 THEN 1260
334 IF A=69 THEN 1035
390 GOTO 110
400 REM REST
410 RST=1-RST:POSITION 7,14
415 FOR I=1 TO 20:NEXT I
417 U=USR(ADR(U$),N)
420 IF RST THEN ? "RST":FOR I=1 TO 20:
NEXT I:GOTO 110
430 MOVE=0:SKIP=1:GOSUB 230:GOTO 110
440 REM TIE
450 TIE=1-TIE:POSITION 28,17
455 U=USR(ADR(U$),N)
460 ? TIE$(TIE*3+1,TIE*3+3)
470 FOR I=1 TO 40:NEXT I:GOTO 110
500 REM SAVE & LOAD
505 POKE 1568+OSEC,PSEC:POKE 1568,10
510 POSITION 5,0:? SL$(9*(A=76)+1,9*(A
=76)+9);
515 ? " DISK OR CASSETTE? ";:GET #2,I:
? CHR$(I+128)
520 IF I=67 THEN D$="C:":POSITION 11,1
:? "PRESS RETURN";:GOTO 540
530 IF I<>68 THEN GOSUB 580:GOTO 110
535 POSITION 5,1:? "NAME OF FILE";:D$=
"D:":INPUT F$:D$(3)=F$
540 CIO$(10,10)=CHR$(64):U=USR(ADR(CIO
$)):CIO$(10,10)=CHR$(192)
542 IF A=76 THEN 1010
545 NHI=INT(NN/256):NLO=NN-NHI*256:TRA
P 1060:OPEN #1,8,0,D$
550 PUT #1,NLO:PUT #1,NHI
560 FOR I=1 TO NN*2:PUT #1,ASC(MSC$(I,
I)):NEXT I
```

```
570 CLOSE #1:TRAP 40000:POKE 205,0
575 F=USR(ADR(CIO$)):GOSUB 580:GOTO 11
0
580 POKE 1568+OSEC,10:POKE 1568,6
585 POSITION 0,0:? ,,,;"      ":? ,,,;
"   "
590 RETURN
600 REM LENGTH CHANGER
610 POSITION LNT*2+6-DOT,16:? " "
620 IF DOT THEN POSITION LNT+23,15:? "
 "
630 LNT=LNT+(S<8)-(S>8):LNT=LNT+13*(LN
T=-1)-13*(LNT=13):IF LNT=1 THEN 630
631 DOT=(LNT/2>INT(LNT/2))
632 IF DOT THEN POSITION LNT+23,15:? "
."
635 POSITION LNT*2+6-DOT,16:? CHR$(27)
;CHR$(28)
640 POSITION 10,17
645 LT=INT(LNT/2)
650 ? NTE$(LT*9+1,LT*9+9)
665 IF  NOT DOT THEN 680
670 POSITION 11,17:? "."
680 IF N<128 THEN U=USR(ADR(U$),N)
690 GOTO 110
700 REM ENTER NOTE
710 IF CN=NMAX*2+1 THEN 790
712 U=USR(ADR(U$),N)
715 INS=INS-(INS>0)
720 POSITION 20,18
723 IF RST THEN ? "RST":MSC$(CN,CN)=CH
R$(255):TIE=0:GOTO 730
725 ? T$;OCT:MSC$(CN,CN)=CHR$(N)
730 POSITION 21,19:? NTE$(LT*9+1,LT*9+
2);CHR$(30);CHR$(DOT*15+31);
732 MSC$(CN+1,CN+1)=CHR$(LNT+128*TIE)
735 IF TIE THEN ? CHR$(92);"/":GOTO 74
0
737 ? "  "
740 TIE=0:POSITION 28,17:? TIE$(1,3)
750 CN=CN+2:NN=NN+(CN=NN*2+3)
755 POSITION 0,18
760 S=7:GOSUB 995
770 POKE 1568+OSEC,PSEC:POKE 1570,10
780 TN=CN+6:POSITION 35,18:IF INS<4 TH
EN GOSUB 900
785 GOTO 107
790 SOUND 3,52,12,15:FOR I=1 TO 10:NEX
T I:SOUND 3,0,0,0:GOTO 110
800 REM NOTE EDITOR
805 IF CN=1 AND S>8 THEN 110
807 IF CN=NN*2+1 AND S<8 THEN 110
808 IF INS THEN 110
810 CN=CN+2*(S<8)-2*(S>8)
815 IF S>8 THEN POSITION 35,18:? "   "
820 POSITION 0,18:GOSUB 995
822 IF CN=NN*2+1 THEN 110
825 POSITION 22+NTE+(NTE>4),4:? "_"
830 POSITION 35*(S<8),18:TN=CN-8*(S>8)
+6*(S<8)
840 GOSUB 900
845 POSITION LNT*2+6-DOT,16:? " "
850 IF DOT THEN POSITION LNT+23,15:? "
 "
860 I=ASC(MSC$(CN,CN))
862 IF I>127 AND N<255 THEN 110
865 N=I:TRAP 110:LNT=ASC(MSC$(CN+1,CN+
1)):TIE=(LNT>127):LNT=LNT-128*TIE:TRAP
 40000
870 IF N=255 THEN POSITION 7,14:? "RST
":RST=1:GOTO 631
875 RST=0:POSITION 28,17:? TIE$(TIE*3+
1,TIE*3+3)
880 I=96-N:OCT=INT(I/12):NTE=I-OCT*12
885 I=ASC(NN$(NTE*2+3,NTE*2+3))-67:I=7
*(I<0)+I+OCT*7:I=56-I
890 MOVE=I*2+25:MOVE=MOVE+8*(MOVE>81)+
4*(MOVE=81):MOVE=MOVE-PEEK(1576):MOVE=
256*(MOVE<0)+MOVE
895 SKIP=1:GOSUB 230:GOTO 631
900 REM SUBR.
910 TRAP 948:I=ASC(MSC$(TN,TN)):TRAP 4
0000:IF I=255 THEN ? "RST";CHR$(29);CH
R$(30);CHR$(30);:GOTO 940
920 IF I>127 THEN ON I-127 GOSUB 1420,
1370,1370,1580:RETURN
930 I=96-I:J=INT(I/12):I=I-J*12
935 ? NN$(I*2+3,I*2+4);J;CHR$(29);CHR$
(30);CHR$(30);
940 I=ASC(MSC$(TN+1,TN+1)):T=(I>127):I
=I-T*128
945 J=INT(I/2):K=(J<I/2):? NTE$(J*9+1,
J*9+2);:IF K THEN ? CHR$(30);".";
947 IF T THEN ? CHR$(92);"/"
948 RETURN
950 REM CLEAR NOTE MEM
960 MSC$="":NN=0:CN=1:R=0:INS=0
970 POSITION 0,18:? INS$:RETURN
980 POKE 1568+OSEC,PSEC:POKE 1568,10
985 POSITION 7,0:? "PRESS -Y- TO CLEAR
 MEMORY":GET #2,A:IF A=89 THEN GOSUB 9
60
990 GOSUB 580:GOTO 110
995 FOR I=1 TO 5:? CHR$(255-(S<8));CHR
$(29);CHR$(255-(S<8));CHR$(28);:NEXT I
:RETURN
1000 REM LOAD
1010 TRAP 1060:OPEN #1,4,0,D$:GOSUB 96
0
1015 GET #1,NLO:GET #1,NHI:I=NHI*256+N
LO:IF I>NMAX THEN POKE 195,2:GOTO 1060
1020 NN=I:FOR I=1 TO NN*2:GET #1,J:MSC
$(I,I)=CHR$(J):NEXT I:CLOSE #1:TRAP 40
000
1030 POKE 205,0:F=USR(ADR(CIO$)):GOSUB
 580
1035 POSITION 0,18:? INS$
1040 FOR B=0 TO 4:POSITION 20-B*5,18:T
N=NN*2-B*2-1:GOSUB 910:NEXT B:CN=NN*2-
1
1045 POSITION 22+NTE+(NTE>4),4:? "_"
1047 POSITION LNT*2+6-DOT,16:? " "
1050 GOTO 860
1055 REM TRAP I/O ERRORS
1060 TRAP 40000:CLOSE #1:POSITION 5,0:
? "ERROR-  ";PEEK(195),,
1070 POSITION 5,1:POKE 205,0:F=USR(ADR
(CIO$))
1075 ? "PRESS -Y- TO RETRY",,:GET #2,I
:IF I=89 THEN GOSUB 585:GOTO 500
1080 GOSUB 580:GOTO 110
1100 REM DELETE
1110 IF CN=NN*2+1 THEN 110
1115 INS=INS-(INS>0)
1120 POSITION 20,18:S=7:GOSUB 995
1125 POSITION 22+NTE+(NTE>4),4:? "_"
1130 TRAP 1140:MSC$(CN)=MSC$(CN+2)
1140 TRAP 40000:NN=NN-1:MSC$(NN*2+1)="
"
1145 POSITION 35,18:TN=CN+6:IF INS<4 T
HEN GOSUB 900
1150 IF CN=NN*2+1 OR INS THEN 110
1160 GOTO 845
1200 REM INSERT
1210 IF CN=NN*2+1 THEN 110
1212 IF NN=NMAX THEN 790
1215 INS=INS+1:POSITION 35,18:? "   "
1220 FOR I=NN*2+2 TO CN+2 STEP -1:MSC$
(I,I)=MSC$(I-2,I-2):NEXT I:MSC$(CN,CN)
=CHR$(0):MSC$(CN+1,CN+1)=CHR$(0)
1225 POSITION 20,18:S=11:GOSUB 995
1230 NN=NN+1:GOTO 110
1250 REM GO TO BEGINNING OF FILE
1260 POSITION 0,18:? INS$
1270 FOR B=0 TO 3:POSITION 20+B*5,18:T
N=B*2+1:GOSUB 910:NEXT B
1280 CN=1:GOTO 1045
1300 REM REPEAT
1310 POKE 1568+OSEC,PSEC:POKE 1568,10
1320 POSITION 9,0:? "PRESS RETURN FOR
";CHR$(60+2*R):GET #2,I:IF I<>155 THEN
 1385
1325 MSC$(CN,CN)=CHR$(129+R)
1327 INS=INS-(INS>0)
1330 IF R THEN 1350
1340 ? ,"HOW MANY TIMES";:TRAP 1385:IN
PUT I:TRAP 40000
1350 MSC$(CN+1,CN+1)=CHR$(I):POSITION
```

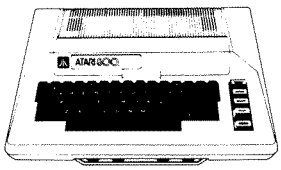
```
20,18:TN=CN:GOSUB 1370
1360 GOSUB 580:R=1-R:GOTO 750
1370 ? CHR$(31);CHR$(29);:I=ASC(MSC$(T
N,TN));? CHR$(60+2*(I=130));CHR$(28);C
HR$(30);CHR$(30);
1371 IF I=130 THEN ? "   ":RETURN
1380 ? ASC(MSC$(TN+1,TN+1));"  ":RETUR
N
1385 GOSUB 580:GOTO 110
1400 REM HALT
1410 POSITION 20,18:GOSUB 1420:MSC$(CN
,CN)=CHR$(128):MSC$(CN+1,CN+1)=CHR$(0)
1412 NN=NN+(CN=NN*2+1):S=7:POSITION 0,
18:GOSUB 995:INS=INS-(INS>0)
1415 CN=CN+2:GOTO 110
1420 ? "   ";CHR$(29);CHR$(30);CHR$(30
);CHR$(30);:? "HLT":RETURN
1500 REM TEMPO
1510 POKE 1568+OSEC,PSEC:POKE 1568,10
1520 POSITION 4,0:? "ENTER TEMPO NO. (
 HEX 0 thru D )"
1525 POSITION 7,1:? "NO.=  (=    LIMIT
S:NO";I=14
1530 GET #2,K:J=K-48-7*(K>57):IF J=100
 THEN 1550
1532 POSITION 29,1:? "    "
1535 I=J:IF I<0 OR I>13 THEN I=14:K=I
1540 POSITION 11,1:? CHR$(K):POSITION
15,1:? TEM$(I*3+1,I*3+3):IF I=14 THEN
1530
1543 IF I/2<>INT(I/2) THEN POSITION 29
,1:? "%"
1545 IF (I-1)/3<>INT((I-1)/3) THEN POS
ITION 31,1:? "+,"
1547 GOTO 1530
1550 IF I=14 THEN GOSUB 580:GOTO 110
1560 MSC$(CN,CN)=CHR$(131):MSC$(CN+1,C
N+1)=CHR$(I):TN=CN
1570 POSITION 20,18:GOSUB 1580:GOSUB 5
80:GOTO 1412
1580 ? "(=";CHR$(29);CHR$(30);CHR$(30)
;:I=ASC(MSC$(TN+1,TN+1)):? TEM$(I*3+1,
I*3+3);RETURN
4900 REM VBI
4902 POKE 559,0:POKE 1791,160:POKE 712
,148:POKE 711,10:RESTORE 9200
4905 DIM U$(58),T$(2),M$(67),CIO$(14)
4910 FOR I=1577 TO 1594:READ J:POKE I,
J:NEXT I
4915 REM VBI INIT
4917 U$=""
4920 FOR I=1 TO 15:READ J:U$(I,I)=CHR$
(J):NEXT I
4930 I=USR(ADR(U$))
4940 FOR I=1 TO 58:READ T$
4950 U$(I,I)=CHR$(16*(ASC(T$)-48-7*(AS
C(T$)>64))+(ASC(T$(2,2))-48-7*(ASC(T$(
2,2))>64))):NEXT I
4960 FOR I=1595 TO 1788:READ T$
4970 POKE I,(16*(ASC(T$)-48-7*(ASC(T$)
>64))+(ASC(T$(2,2))-48-7*(ASC(T$(2,2))
>64))):NEXT I
5000 REM INITIALIZATION SUBR.
5010 POKE 709,0:POKE 708,0:POKE 752,1
5015 RESTORE 9000
5020 DL=PEEK(560)+256*PEEK(561)
5030 POKE DL+2,160:FOR I=6 TO 28:READ
J:POKE DL+I,J:NEXT I
5040 FOR I=1536 TO 1567:READ J:POKE I,
J:NEXT I
5045 POKE 512,0:POKE 513,6
5050 FOR I=1568 TO 1576:READ J:POKE I,
J:NEXT I
5055 POKE 203,32:POKE 204,6:POKE 205,0
5099 REM DRAW SCREEN
5102 POSITION 17,2:? "-REST-"
5105 POSITION 10,5:? "POKEY PLAYER EDI
TOR"
5108 POSITION 8,8:? "by Harry Bratt  8
-28-82"
5112 POSITION 14,12:? "please wait"
5113 ?
5114 ? "NOTE:C   OCTAVE:4"
5115 POSITION 23,15:? "% & ' ( ) * +,"
5116 POSITION 18,16:? CHR$(27);CHR$(28
)
5117 ? " LENGTH:( QUARTER        -tie-
"
5120 FOR I=1 TO 14:READ J:CIO$(I,I)=CH
R$(J):NEXT I
5130 POKE 205,0:U=USR(ADR(CIO$))
5190 REM P.M.
5200 POKE 559,62
5205 POKE 54279,A
5210 PM=A*256
5220 FOR I=0 TO 1023:POKE PM+I,PEEK(57
344+I):NEXT I:POKE 756,A
5225 FOR I=3 TO 12:FOR J=0 TO 7:READ K
:POKE PM+I*8+J,K:NEXT J:NEXT I
5230 FOR I=65 TO 78:FOR J=0 TO 7:READ
K:POKE PM+I*8+J,K:NEXT J:NEXT I
5240 FOR I=1 TO 8:READ C:FOR J=0 TO 7:
READ K:POKE PM+C*8+J,K:NEXT J:NEXT I
5250 POKE 756,A
5300 PM=PM+1024:POKE 1152,PM/256
5305 POKE 53248,80
5310 FOR I=0 TO 512:POKE PM+I,0:NEXT I
5315 POKE 53249,126:POKE 705,66:POKE 5
3257,1
5320 POKE PM+453,255:POKE PM+470,255:F
OR I=454 TO 469:POKE PM+I,129:NEXT I
5400 REM MOVEM
```

```
5405 RESTORE 9400
5410 FOR I=1 TO 67:READ T$
5420 M$(I,I)=CHR$(16*(ASC(T$)-48-7*(AS
C(T$)>64))+(ASC(T$(2,2))-48-7*(ASC(T$(
2,2))>64))):NEXT I
5430 M$(48,48)=CHR$(A)
5900 POSITION 2,3
5910 ? "      ";CHR$(14);"
  MEMORY:"
5912 ? " ";CHR$(1);CHR$(2);"   ";CHR$(
95)
5913 REM SEE EXPLANATION OF ATARI
     LINE LISTINGS FOR 5914 TO 5945
5914 ? CHR$(25);CHR$(3);CHR$(4);"-----
---",:? :REM 8 UNDERLINES
5916 ? CHR$(25);CHR$(5);CHR$(6);"-----
---":REM 8 UNDERLINES
5918 ? CHR$(25);CHR$(7);CHR$(8);"mmmm
mm":REM 8 CONTROL M'S
5920 ? CHR$(25);"     ";CHR$(13),,:?
5922 ? CHR$(25);CHR$(9);CHR$(10);"----
----":REM 8 UNDERLINES
5924 ? CHR$(25);CHR$(11);CHR$(12);"---
-----":REM 8 UNDERLINES
5926 ? CHR$(16);"mmmm_mmm":REM CONTR
OL M'S & UNDERLINE
5928 ? "      _",,:? :REM UNDERLINE
5930 ? "      _":REM UNDERLINE
5935 POKE 82,20:POSITION 20,3:?
5937 ? "__";CHR$(27);CHR$(157);"____
      "
_______
5940 FOR I=1 TO 4:? "_v___!_____v__
":NEXT I:REM SEE EXPLANATION
5945 FOR I=1 TO 3:? "_v!!!!!!v__
":NEXT I:REM SEE EXPLANATION
5950 ? "                "
5999 POKE 53277,3:RETURN
9000 REM DL
9010 DATA 130,130,2,2,2,2,2,2,2,2,2,2,
2,6,128,130,7,2,6,144,134,2,130
9015 REM DLI
9020 DATA 72,152,72,164,205,177,203,14
1,10,212,141,26,208,141,24,208,230,205
,165,205,201,8,144,4,169,0,133
9030 DATA 205,104,168,104,64
9035 REM DLI COLOURS
9040 DATA 6,246,10,246,6,246,6,148,0
9050 DATA 104,173,11,212,201,0,208,249
,169,192,141,14,212,96
9100 DATA 108,254,108,254,108,0,0,0,19
2,192,240,216,240,0,0,0,30,27,30,27,30
,123,248,240
9102 DATA 30,27,30,27,24,120,248,240,3
0,27,24,24,24,120,248,240,24,24,24,24,
24,120,248,240
9104 DATA 12,12,12,12,12,124,204,248,0
,0,0,0,120,204,204,120,31,24,24,24,24,
123,255,247
9106 DATA 254,198,198,198,198,222,254,
188
9114 DATA 0,0,0,7,12,24,24,24,0,0,0,12
8,192,192,192,192,255,13,15,6,255,14,2
7,27
9116 DATA 255,128,0,0,255,0,0,0,255,49
,55,55,255,54,24,12,255,224,176,152,25
5,216,216,240
9118 DATA 255,0,24,25,15,0,0,0,255,96,
96,192,0,0,0,0,255,48,96,112,255,112,0
,0
9120 DATA 255,96,51,51,255,48,51,99,25
5,0,1,3,255,28,112,0,255,192,128,0,255
,0,0,0
9122 DATA 255,0,0,0,0,0,0,0,0,0,0,0,25
5,0,0,0
9124 DATA 80,3,0,0,0,0,0,0,0
9126 DATA 89,3,3,3,3,3,3,3,3
9128 DATA 60,192,96,48,28,7,0,0,0
9130 DATA 63,255,0,0,0,255,0,0,0
9132 DATA 14,0,0,0,0,0,0,192,192,15,3,
6,12,56,224,0,0,0
9134 DATA 28,216,216,219,216,216,219,2
16,216
9136 DATA 30,27,27,219,27,27,219,27,27
9200 REM VBI
9202 DATA 173,255,6,170,41,15,240
9210 DATA 7,202,142,255,6,142,3
9220 DATA 210,76,98,228
9230 REM VBI INIT
9240 DATA 104,240,3,76,116,228,169
9250 DATA 7,162,6,160,41,76,92,228
9260 REM VBI USR
9270 DATA 68,C9,01,F0,03,4C,74,E4
9275 DATA A9,03,8D,32,02,8D,0F,D2
9280 DATA A9,00,8D,01,D2,A9,50,8D
9285 DATA 08,D2,AD,FF,06,29,F0,8D
9290 DATA FF,06,68,68,AA,BD,3B,06
9295 DATA 8D,00,D2,BD,9C,06,8D,02
9297 DATA D2,AD,FF,06,09,0F,8D,FF
9298 DATA 06,60
9300 REM NOTE DATA
9305 REM LO BYTE
9310 DATA D1,DF,ED,FB,09,1E,2C,41
9312 DATA 4F,64,79,95,A5,BD,D9,F5
9314 DATA 11,3B,57,81,9D,C7,F1,29
9316 DATA 4B,79,B1,E9,21,75,AD,01
9318 DATA 39,8D,E1,51,A5,15,85,F5
9320 DATA 65,F1,61,09,95,3D,E5,8D
9322 DATA 51,15,F5,D5,D1,CD,01,FD
9324 DATA 31,65,D1,3D,A5,2A,EA,AA
9326 DATA A2,9A,02,FA,C2,CA,A2,7A
9328 DATA 4B,54,D4,54,44,34,04,F4
9330 DATA C4,94,44,F4,96,A8,A8,A8
```

continued on page 66

A.S.C., INC
Authorized Service Center

"Call us for discount prices, availability, or advice. Remember, service is our middle name." Joe Guzzo, President, A.S.C. Inc.



ATARI 800

800 16K $654.95
800 32K $745.95
800 48K $815.95

ATARI 400

400–16K $264.95
400-32K $364.95
400-48K $430.95

410 Recorder $76.00
810 Disk Drive $444.95

Kits

481 Entertainer $83.00
482 Educator $129.95
483 Programmer $56.95
484 Communicator . . . $339.95

Software

We carry the full line of Atari hardware and software products backed by our Authorized Service Department.

Please call for special prices on all Atari software and APX software.

TEXAS INSTRUMENTS

All models . . . . . Call for prices

MONITORS

Amdex 12-in. B&W . . . $129.00
Amdex 12-in. Green . . $139.00
Amdex 13-in. Color . . . $349.00
BMC--BM 12A $114.95
NEC 12-in. B&W $169.95
NEC 12-in. Color $339.00
TI 10 . . . . . $344.95
Corvus . . . . . Call for prices
Zenith Z19 12-in. $115.00

DISKS

Elephant . . . . . 10/$21.95
Elephant DD . . . . . 10/$24.95
Dysan 5¼-10/16 . . . . . 10/$38
Verbatim 8-in.SD--SS . . . 10/$32
Verbatim . . . . . 100/$245
Scotch 3M . . . . . 10/$25.95

DISK DRIVES

Corvus . . . . . Call
Percom . . . . . Call
Paradynamics . . . . . Call

VIC-20

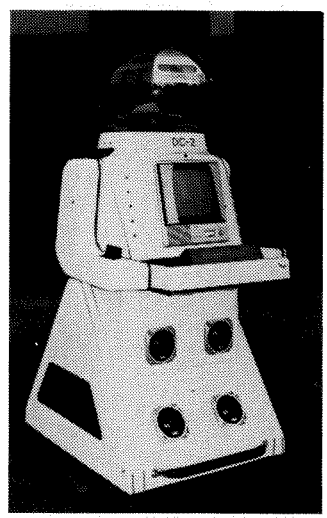
VIC-20 Personal Computer $175

GAMES/ PROGRAMS

Unlimited Availability . . . . . Call

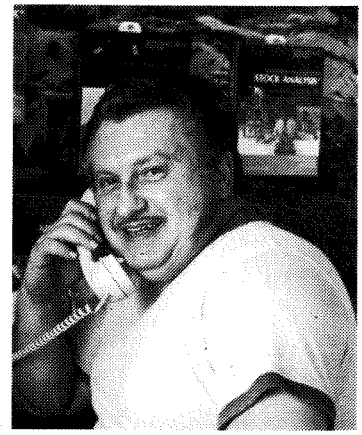
COMPUTER DESKS

Send for catalog and photos.



DC-2 ROBOT

Operates on any home computer.
Stands 4 feet high.
Call for details and prices.



TERMINALS

Televideo $569.00
910 . . . . . $569.00
912C $694.95
920C $744.95
950 . . . . . $934.95

Call for Computers

MODEMS

Lexicon $149.00
U.D.S. 1030ALP $169.00
Racal Vadic VA 3212 . . $749.00
Prentice Star $179.95

PRINTERS

Brother HR-1 Parallel . . $829.95
Brother HR-1 Serial . . . $880.00
Smith-Corona TP-1 Parallel $665
Smith-Corona TP-1 Serial . $665
C. Itoh Starwriter $1,350
Diablo 630 . . . . . $1,999.95
Epson printers . . . . . Call
Centronics printers . . . . . Call
All name brands . . . . . Call

NEW•NEW

Coleco-vision, Mattel Intellivision, and Atari VCS game machines and game cartridges. Call us for information on the latest, improved models and best prices.

Limited space prevents a full listing of the lines we carry. If you don't see it listed, we can get it for you, for less.

TO ORDER: Phone orders accepted via Mastercard, Visa, or bank wire transfer. Visa and MC service charge of 2%. Mail orders may send charge card number (include expiration date), cashiers check, money order, or personal check (allow 10 business days for personal or company checks to clear). Please add 3% ($5.00 minimum) for UPS shipping, handling, and insurance. APO and FPO include 5% ($7.00 minimum) for postage. California residents add 6% sales tax. Please include phone number on all orders. FOREIGN ORDERS include 1% handling. Credit cards not accepted on foreign orders. All equipment is in factory cartons with manufacturer warranty. Opened products not returnable. Restocking fee for returned merchandise. Equipment subject to price change and availability. Retail prices differ from mail order prices. WE SHIP THE SAME DAY ON MOST ORDERS.

Call Collect (714) 284-5615 7 Days/Week

Monday through Saturday 8am–8pm
Sundays and Holidays 12n–6pm

After 11/6/82 , Call (619) 284-5615

A.S.C. Inc., 7436 University Ave.
La Mesa, CA 92041

```
9332 DATA 88,68,08,E8,88,28,88,E8
9334 DATA 48
9340 REM HI BYTE
9350 DATA 00,00,00,00,01,01,01,01
9352 DATA 01,01,01,01,01,01,01,01
9354 DATA 02,02,02,02,02,02,02,03
9356 DATA 03,03,03,03,04,04,04,05
9358 DATA 05,05,05,06,06,07,07,07
9360 DATA 08,08,09,0A,0A,0B,0B,0C
9362 DATA 0D,0E,0E,0F,10,11,13,13
9364 DATA 15,16,17,19,1A,1C,1D,1F
9366 DATA 21,23,26,27,2A,2C,2F,32
9368 DATA 35,38,3B,3F,43,47,4C,4F
9370 DATA 54,59,5F,64,6A,70,77,7E
9372 DATA 86,8E,98,9F,A9,B3,BE,C9,D5
9400 REM MOVEM
9410 DATA A5,80,8D,FD,06,A5,81,8D
9412 DATA FE,06,A9,00,85,80,AD,80
9414 DATA 04,85,81,A9,00,A2,08,AC
9416 DATA 28,06,91,80,C8,CA,D0,FA
9418 DATA 68,F0,15,68,68,18,6D,28
9420 DATA 06,8D,28,06,A8,BD,38,06
9422 DATA 91,80,C8,E8,E0,08,D0,F5
9424 DATA AD,FD,06,85,80,AD,FE,06
9426 DATA 85,81,60
```

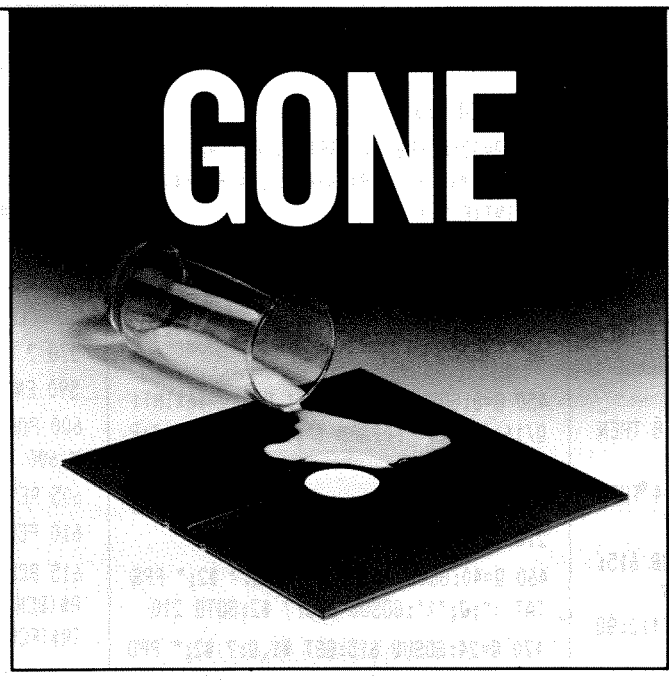
ATARI® SWAT TABLE FOR: EDITOR (Modified Parameters: NU=2, B=500)

LINES	SWAT CODE	LENGTH	LINES	SWAT CODE	LENGTH
10 - 20	NQ	337	1220 - 1260	VQ	260
25 - 50	JH	226	1270 - 1320	SF	256
60 - 100	RN	178	1325 - 1350	ZU	176
105 - 112	DA	159	1360 - 1385	CM	234
115 - 145	PV	262	1400 - 1420	PV	280
150 - 205	IV	114	1500 - 1530	VD	233
207 - 220	CB	302	1532 - 1545	SQ	311
230 - 255	IJ	202	1547 - 1580	AV	276
260 - 270	GY	206	4900 - 4915	CO	194
280 - 312	QT	93	4917 - 4950	TF	234
314 - 322	SJ	115	4960 - 5015	DF	254
324 - 334	BN	117	5020 - 5050	BM	209
390 - 417	IW	102	5055 - 5108	UI	200
420 - 455	UU	157	5112 - 5116	GW	153
460 - 510	CL	216	5117 - 5200	FJ	147
515 - 540	TD	307	5205 - 5230	FN	236
542 - 570	BH	219	5240 - 5310	ZO	191
575 - 600	EC	150	5315 - 5410	QJ	189
610 - 632	QT	237	5420 - 5912	EL	284
635 - 665	QG	158	5913 - 5920	BN	307
670 - 710	QP	118	5922 - 5930	NN	242
712 - 725	CU	139	5935 - 5950	OH	250
730 - 740	AD	236	5999 - 9020	QS	208
750 - 780	FX	183	9030 - 9100	JQ	213
785 - 807	ST	211	9102 - 9116	AQ	367
808 - 822	PI	169	9118 - 9126	LJ	241
825 - 850	AW	205	9128 - 9136	FP	182
860 - 875	CW	278	9200 - 9230	EA	98
880 - 895	KQ	306	9240 - 9275	ZJ	131
900 - 935	QT	304	9280 - 9297	QA	145
940 - 950	QA	217	9298 - 9312	DN	97
960 - 990	IH	230	9314 - 9322	QF	145
995 - 1020	AV	325	9324 - 9332	TZ	145
1030 - 1047	LJ	267	9334 - 9354	KH	108
1050 - 1075	HL	226	9356 - 9364	MB	145
1080 - 1120	GM	132	9366 - 9400	ZZ	130
1125 - 1150	BU	219	9410 - 9418	HS	145
1160 - 1215	QY	118	9420 - 9426	TA	101

```
   SS SS SS SS SS SS SS SS SS SS SS
   SS                           SS
   SS        Atari  BASIC       SS
   SS         "Compiler"        SS
   SS Author:  Craig Chamberlain SS
   SS     Copyright (c) 1982    SS
   SS SoftSide Publications, Inc SS
   SS                           SS
   SS SS SS SS SS SS SS SS SS SS SS
```

```
100 DIM D(7),M(13),R$(120),F1$(16),F2$
(16),N$(8),P$(24),D$(7):IF PEEK(87) TH
EN GRAPHICS 0
101 P$="C C+D E-E F F+G G+A B-B ":D$="
TSEQHW3":FOR K=1 TO 7:READ P:D(K)=P:NE
XT K
102 FOR K=0 TO 13:READ P:M(K)=P:NEXT K
110 TRAP 110:? CHR$(125):? "POKEY PLAY
ER COMPILER":? "by Craig Chamberlain
8/24/82":?
115 ? "This program accepts a music":?
 "source file from the"
120 ? "POKEY PLAYER EDITOR and convert
s it":? "to DATA statements to be merg
ed"
125 ? "with the POKEY PLAYER program."
:?
130 ? "Please enter the device":? "spe
cification from which the music"
135 ? "source file will be read.":? :G
OSUB 600:F1$=R$:?
140 ? "Is this for a primary or":? "se
condary part? (P/S) ";:POKE 764,255
145 P=PEEK(764):IF P=10 THEN ? "PRIMAR
Y":V=0:GOTO 160
150 IF P<>62 THEN 145
155 ? "SECONDARY":V=12
160 ? :? "Please enter the device":? "
specification for the output file."
165 ? :GOSUB 600:F2$=R$:TRAP 170:? :?
:? :POSITION 2,21
170 ? "STARTING LINE NUMBER";:INPUT LN
:IF LN>32767 OR LN<>INT(ABS(LN)) THEN
170
175 ? "    LINE NUMBER STEP";:INPUT LN
S:IF LNS=0 OR LNS<>INT(ABS(LNS)) THEN
175
180 ? :? "While compiling a source fil
e,":? "POKEY PLAYER COMPILER generates
"
185 ? "status information.  Please":?
"enter the device specification for"
190 ? "the device to which this":? "in
formation should be sent.":? :? :? :PO
SITION 2,21
195 GOSUB 600:TRAP 570:OPEN #1,4,0,F1$
```

continued on page 68



ATARI 400/800

This disk and program
was destroyed by a simple spill of milk.

NOW YOU CAN PREVENT A TOTAL LOSS

The Replica 1+1 Backup and Utilities Systems Package makes it possible for the user to build a complete reserve library to replace a damaged or faulty disk when necessary.

PROGRAM FEATURES

DISK BACKUP: Copies the sections of the disk which contain data, and also permits read and write without verify, speeding up the backup process considerably. Copies 'bad sectors' with ease. UTILITIES PACKAGE: Scans any ATARI disk, providing the user with information as to which sectors contain data, which are empty and which are 'bad'. Includes a sector editor which allows the user to see and modify data within a sector. Disassembles a sector into machine language op codes. Searches for a series of bytes or a string within a section of a disk. Includes a custom format scan and patch analysis enabling the user to backup any presently available* ATARI computer disk. And much more. . . .

To receive the Replica 1+1 package and documentation fill out the coupon below and send $50 to:

1

NUDMEHI SOFTWARE
495 Old York Road Suite 425
Jenkintown, Pa. 19046

Visa and Mastercard Phone Orders: 215-635-2722

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

☐ MY CHECK FOR $50 IS ENCLOSED
☐ BILL BY VISA # _____
EXPIRATION DATE _____
☐ BILL MY MASTERCARD # _____
EXPIRATION DATE _____

_____
CARDHOLDER'S SIGNATURE

*As of 8/15/82. Subsequent updates will become available at a nominal fee.
Please Note: It may be necessary to adjust your drive speed when backing up certain disks.

THE REPLICA 1+1 IS NOT INTENDED FOR THE ILLEGAL DUPLICATION OF COPYRIGHTED MATERIALS

```
:GET #1,P:GET #1,KMAX:KMAX=P+256*KMAX
200 OPEN #2,8,0,R$:DIM S$(FRE(0)-100)
205 ? #2;"POKEY PLAYER COMPILATION REP
ORT";? #2:OLDP=48:K=0
210 IF K=KMAX THEN 500
212 GET #1,PNEW:K=K+1:IF PNEW>96 THEN
350
215 IF V AND (PNEW<12 OR PNEW>84) THEN
 ? #2:? #2;"ERROR: NOTE OUT OF RANGE":
GOTO 590
220 DIFF=ABS(PNEW-OLDP):IF DIFF<8 THEN
 250
225 OCT=INT((DIFF+4)/12):IF OCT>4 THEN
 OCT=4
230 Q=127+OCT+4*(PNEW<OLDP):GOSUB 615:
? #2;CHR$(44+SGN(PNEW-OLDP));OCT
235 OLDP=OLDP+SGN(PNEW-OLDP)*OCT*12:GO
TO 220
250 NS=(DIFF+8*(PNEW<=OLDP))*8:OLDP=PN
EW
255 GET #1,P:TS=0:IF P>127 THEN P=P-12
8:TS=128
260 DOT=(P/2<>INT(P/2)):DUR=(P-DOT)/2:
IF DOT=0 THEN 300
265 IF DUR=0 THEN ? #2:? #2;"ERROR: DO
TTED THIRTY SECOND NOTE":GOTO 590
270 Q=128+NS+DUR:GOSUB 610:? #2:NS=64
300 DUR=DUR+1:Q=TS+NS+DUR:GOSUB 610
305 PNEW=96-PNEW:N=INT(PNEW/12):? #2;"
 ";N;" ";P$(2*(PNEW-12*N)+1,2*(PNEW-12
*N)+2);" ";
310 ? #2;D$(DUR,DUR);" ";CHR$(32+36*DO
T);" ";CHR$(32+52*(TS<>0)):GOTO 210
350 IF PNEW<>255 THEN 400
355 GET #1,P:IF P>127 THEN ? #2:? #2;"
ERROR: TIED REST":GOTO 590
360 DOT=(P/2<>INT(P/2)):DUR=(P-DOT)/2:
IF DOT=0 THEN 375
365 IF DUR=0 THEN ? #2:? #2;"ERROR: DO
TTED THIRTY SECOND REST":GOTO 590
370 Q=DUR:GOSUB 610:? #2
375 DUR=DUR+1:Q=DUR:GOSUB 610:? #2;"
 R  ";D$(DUR,DUR);" ";CHR$(32+36*DOT):
GOTO 210
400 IF PNEW>127 AND PNEW<135 THEN ON P
NEW-127 GOTO 410,420,430,440,450,460,4
70
405 ? #2:? #2;"ERROR: ILLEGAL COMMAND"
:GOTO 590
410 Q=120:GOSUB 615:? #2;" PPSTOP":GET
 #1,P:GOTO 210
420 Q=16:GOSUB 615:GET #1,Q:? #2;" PPH
EAD (";Q;")":GOSUB 615:? #2
425 Q=64:GOSUB 615:? #2:Q=OLDP-V:GOSUB
 615:? #2:GOTO 210
430 Q=0:GOSUB 615:? #2;" PPTAIL":GET #
1,P:GOTO 210
440 Q=32:GOSUB 615:GET #1,P:? #2;" PPT
EMP (";M(P);")":P=16-P:D(1)=(P/2)*((P/
2)=INT(P/2))
442 FOR J=2 TO 6:D(J)=P:P=P*2:NEXT J:D
(7)=D(4)/3*((D(4)/3)=INT(D(4)/3))
444 FOR J=1 TO 7:Q=D(J)*(D(J)<256):GOS
UB 615:? #2:NEXT J:GOTO 210
450 Q=8:GOSUB 615:GET #1,Q:P=Q-16*INT(
Q/16):? #2;" PPPARM (";INT(Q/16);",";P
;")"
455 GOSUB 615:? #2:Q=P+1:GOSUB 615:? #
2:GOTO 210
460 Q=40:GOSUB 615:GET #1,Q:? #2;" PPS
TAT (";Q;")":GOSUB 615:? #2:GOTO 210
470 Q=24:GOSUB 615:GET #1,Q:? #2;" PPO
FST (";Q;")":GOSUB 615:? #2:GOTO 210
500 CLOSE #1:? #2:? #2;"TOTALS: ";BCNT
;" BYTES, ";FCNT;" FRAMES":CLOSE #2:?
510 OPEN #1,8,0,F2$:K=0:R$=STR$(LN):R$
(LEN(R$)+1)=" DATA ":R$(LEN(R$)+1)=STR
$(BCNT):GOTO 530
520 R$=STR$(LN):R$(LEN(R$)+1)=" DATA "
:R$(LEN(R$)+1)=STR$(ASC(S$(K))):IF K=B
CNT THEN 540
530 K=K+1:R$(LEN(R$)+1)=",":R$(LEN(R$)
+1)=STR$(ASC(S$(K))):IF K<>BCNT AND LE
N(R$)<110 THEN 530
540 ? #1;R$:LN=LN+LNS:IF LN>32767 THEN
 ? "LINE NUMBER OUT OF RANGE":GOTO 590
550 IF K<>BCNT THEN K=K+1:GOTO 520
560 CLOSE #1:GOTO 590
570 ? "I/O ERROR"
590 END
600 POKE 764,255:INPUT R$:IF R$="" THE
N 600
605 RETURN
610 FCNT=FCNT+D(DUR)
615 BCNT=BCNT+1:S$(BCNT)=CHR$(Q):N$=ST
R$(BCNT+100000):? #2;N$(2,6);" ";:N$=S
TR$(FCNT+1000000)
620 ? #2;N$(2,7);" ";:N$=STR$(Q+1000):
? #2;N$(2,4);" ";
625 N=Q:M=INT(N/16):? #2;CHR$(48+M+7*(
M>9));:M=N-16*M:? #2;CHR$(48+M+7*(M>9)
);" ";
630 RETURN
900 DATA 5,10,20,40,80,160,1
905 DATA 56,60,64,69,75,81,90,100,112,
128,150,180,225,300
```

ATARI® SWAT TABLE FOR: COMPILER (Modified Parameters: NU = 2, B = 500)

LINES	SWAT CODE	LENGTH	LINES	SWAT CODE	LENGTH
100 - 101	BR	201	350 - 355	CU	102
102 - 110	UG	129	360 - 365	FJ	152
115 - 120	TE	140	370 - 375	NT	125
125 - 130	IB	111	400 - 405	NB	153
135 - 140	PW	133	410 - 420	EE	157
145 - 150	SL	87	425 - 430	LQ	144
155 - 160	AR	102	440 - 442	CF	263
165 - 170	PU	122	444 - 450	JZ	193
175 - 180	JI	140	455 - 460	EI	167
185 - 190	WE	165	470 - 500	KU	184
195 - 200	FI	150	510 - 520	KY	195
205 - 210	UK	100	530 - 540	MH	170
212 - 215	JX	138	550 - 560	LZ	57
220 - 225	FY	88	570 - 590	KE	23
230 - 235	XI	108	600 - 605	XS	46
250 - 255	LE	105	610 - 615	II	112
260 - 265	SW	152	620 - 625	GH	226
270 - 300	YJ	89	630 - 900	DR	31
305 - 310	MA	214	905 - 905	CP	54

```
SS SS SS SS SS SS SS SS SS SS SS
SS                           SS
SS        Atari  BASIC       SS
SS          "Player"         SS
SS Author:  Craig Chamberlain SS
SS     Copyright (c) 1982    SS
SS SoftSide Publications, Inc SS
SS                           SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
1000 IF PEEK(87) THEN GRAPHICS 0
1010 ? CHR$(125):? "POKEY PLAYER I":?
"by Craig Chamberlain  8/29/82":?
1020 ? "Tuning instruments...":?
1040 POKE 1536,1:POKE 1537,0
1050 DIM PP$(442):FOR K=1 TO 442:READ
P:PP$(K)=CHR$(P):NEXT K
1070 P=ADR(PP$):K=INT(P/256):POKE 1538
,P-256*K:POKE 1539,K
1080 P=P+91:K=INT(P/256):POKE 1540,P-2
56*K:POKE 1541,K
1090 P=P+199:K=INT(P/256):POKE 1542,P-
256*K:POKE 1543,K
1100 DIM F1$(194):FOR K=1 TO 194:READ
P:F1$(K)=CHR$(P):NEXT K
1110 P=ADR(F1$):K=INT(P/256):POKE 1544
,P-256*K:POKE 1548,K
1200 DIM F2$(73):FOR K=1 TO 73:READ P:
F2$(K)=CHR$(P):NEXT K
1210 P=ADR(F2$):K=INT(P/256):POKE 1545
,P-256*K:POKE 1549,K
1500 FOR K=1564 TO 1646:READ P:POKE K,
P:NEXT K
1600 READ P:DIM V1$(P):FOR K=1 TO P:RE
AD P:V1$(K)=CHR$(P):NEXT K
1610 P=ADR(V1$):K=INT(P/256):POKE 1552
,P-256*K:POKE 1558,P-256*K
1620 POKE 1555,K:POKE 1561,K
1700 READ P:DIM V2$(P):FOR K=1 TO P:RE
AD P:V2$(K)=CHR$(P):NEXT K
1710 P=ADR(V2$):K=INT(P/256):POKE 1553
,P-256*K:POKE 1559,P-256*K
1720 POKE 1556,K:POKE 1562,K
1800 READ P:DIM V3$(P):FOR K=1 TO P:RE
AD P:V3$(K)=CHR$(P):NEXT K
1810 P=ADR(V3$):K=INT(P/256):POKE 1554
,P-256*K:POKE 1560,P-256*K
1820 POKE 1557,K:POKE 1563,K
1940 ? "Now playing.":?
1950 P=USR(ADR(PP$))
1960 ? "FINIS"
1990 END
2000 DATA 104,173,0,6,240,84,165,20,19
7,20,240,252,169,3,141,50,2,141,15,210
,173,59,6,141,0,210,173,63,6,141,1
2002 DATA 210,173,60,6,141,2,210,173,6
4,6,141,3,210,173,61,6,141,4,210,173,6
5,6,141,7,210,173,62,6,141,6,210,173
2004 DATA 66,6,141,5,210,173,67,6,141,
8,210,162,0,32,94,6,232,32,94,6,232,32
,94,6,24,144,167,96,189,16,6,133,203
2006 DATA 189,19,6,133,204,188,28,6,18
5,8,6,133,205,185,12,6,133,206,222,31,
6,240,20,189,40,6,208,11,189,31,6,221
2008 DATA 46,6,176,3,222,63,6,96,32,97
,6,32,100,6,133,207,41,248,201,128,208
,18,165,207,41,7,168,185,68,6,24,125
2010 DATA 34,6,157,34,6,24,144,227,165
,207,41,7,240,218,168,136,185,52,6,157
,31,6,165,207,74,74,74,41,15,208,10
2012 DATA 157,63,6,169,1,157,40,6,208,
78,168,136,185,76,6,24,125,34,6,157,34
,6,168,189,43,6,157,63,6,177,205,24
2014 DATA 125,37,6,157,59,6,224,2,208,
13,152,24,105,97,168,177,205,141,62,6,
24,144,26,189,43,6,41,240,201,160,208
2016 DATA 17,189,34,6,201,50,144,10,18
9,43,6,41,15,9,192,157,63,6,165,207,41
,128,157,40,6,165,203,157,16,6,165
2018 DATA 204,157,19,6,96,160,0,165,20
7,41,120,208,21,189,49,6,240,5,222,49,
6,240,10,189,22,6,133,203,189,25,6
2020 DATA 133,204,96,201,8,208,13,32,1
00,6,157,43,6,32,100,6,157,46,6,96,201
,16,208,17,32,100,6,157,49,6,165,203
2022 DATA 157,22,6,165,204,157,25,6,96
,201,24,208,7,32,100,6,157,37,6,96,201
,32,208,22,160,6,177,203,153,52,6,136
2024 DATA 16,248,169,7,24,101,203,133,
203,144,2,230,204,96,201,40,208,7,32,1
00,6,141,1,6,96,201,48,208,7,32,100
2026 DATA 6,141,67,6,96,201,56,208,7,3
2,100,6,157,28,6,96,201,64,208,7,32,10
0,6,157,34,6,96,140,0,6,96
2050 DATA 209,223,237,251,9,30,44,65,7
9,100,121,149,165,189,217,245,17,59,87
,129,157,199,241,41,75,121,177,233
2052 DATA 33,117,173,1,57,141,225,81,1
65,21,133,245,101,241,97,9,149,61,229,
141,81,21,245,213,209,205,1,253,49
2054 DATA 101,209,61,165,42,234,170,16
2,154,2,250,98,202,162,122,75,84,212,8
4,68,52,4,244,196,148,68,244,150,168
2056 DATA 168,168,136,104,8,232,136,40
,136,232,72,0,0,0,0,1,1,1,1,1,1,1,1,1,
1,1,1,2,2,2,2,2,2,2,3,3,3,3,3,4,4,4
2058 DATA 5,5,5,5,6,6,7,7,7,8,8,9,10,1
0,11,11,12,13,14,14,15,16,17,19,19,21,
22,23,25,26,28,29,31,33,35,38,39,42
2060 DATA 44,47,50,53,56,59,63,67,71,7
6,79,84,89,95,100,106,112,119,126,134,
142,152,159,169,179,190,201,213
2100 DATA 1,3,6,9,12,15,18,22,23,25,26
,28,29,31,33,35,37,40,42,45,47,50,53,5
7,60,64,68,72,76,81,85,91,96,102,108
2102 DATA 114,121,128,136,144,153,162,
173,182,193,204,217,230,243,255,110,11
6,122,131,137,146,155,167,173,185,197
2104 DATA 206,221,233,245,87,91,97,102
,108,115,121,130
2150 DATA 1,1,0,1,1,1,36,36,48,0,0,0,0
,0,0,168,168,168,9,9,9,0,0,0,5,10,20,4
0,80,160,1,0,0,0,0,0,0,0,0
2152 DATA 40,12,24,36,48,244,232,220,2
08,1,2,3,4,5,6,7,0,255,254,253,252,251
,250,249,108,2,6,108,4,6,108,6,6,160
2154 DATA 0,177,203,230,203,208,2,230,
204,96
```

ATARI® SWAT TABLE FOR: PLAYER

(Modified Parameters: NU = 2, B = 500)

LINES	SWAT CODE	LENGTH
1000 - 1010	SC	98
1020 - 1040	SW	71
1050 - 1070	HX	119
1080 - 1090	IU	136
1100 - 1110	SV	119
1200 - 1210	YA	119
1500 - 1600	BG	85
1610 - 1620	BC	100
1700 - 1710	OD	121
1720 - 1800	BU	75
1810 - 1820	XM	100
1940 - 1950	PC	38
1960 - 1990	JK	19
2000 - 2002	IQ	214
2004 - 2006	SI	217
2008 - 2010	PW	215
2012 - 2014	TD	216
2016 - 2018	JX	212
2020 - 2022	SN	217
2024 - 2026	UQ	205
2050 - 2052	UG	212
2054 - 2056	UB	215
2058 - 2060	PG	210
2100 - 2102	GU	217
2104 - 2150	WR	148
2152 - 2154	CD	148

```
SS SS SS SS SS SS SS SS SS SS SS
SS        Atari  BASIC         SS
SS      "Happy Birthday"       SS
SS Author:  Craig Chamberlain  SS
SS      Copyright (c) 1982     SS
SS SoftSide Publications, Inc  SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
3100 DATA 67,32,1,2,4,9,18,36,72,16,26
,5,0,5,8,134,7,134,6,12,5,4,68,4,68,4,
68,68,4,68,6,68,5,4,68,5,68,68,68
3102 DATA 68,4,6,68,5,4,68,5,67,67,67,
67,68,68,4,6,68,5,68,4,68,4,67,67,68,6
7,67,68,4
3200 DATA 61,16,26,5,0,5,24,2,84,68,84
,4,5,20,4,108,4,12,4,6,52,68,84,5,68,2
0,100,92,4,20,6,44,84,20,132,68,4,5
3202 DATA 28,4,36,4,12,4,5,20,4,132,36
,68,12,4,5,36,4,84,4,20,4,6,120,4
3300 DATA 81,6,128,213,213,149,237,14,
165,213,149,253,22,173,132,197,157,165
,141,149,132,165,141,165,213,22,5
3302 DATA 132,44,68,84,4,5,20,4,108,4,
12,4,6,52,68,84,5,68,20,100,92,4,20,6,
44,84,20,132,68,4,5,28,4,36,4,12,4,5
3304 DATA 20,4,132,36,68,12,4,5,36,4,8
4,4,20,4,6,120,4
```

ATARI® SWAT TABLE FOR:
HAPPY BIRTHDAY DATA
(Modified Parameters: NU = 2, B = 500)

LINES	SWAT CODE	LENGTH
3100 - 3102	ZS	186
3200 - 3202	CU	174
3300 - 3302	LI	213
3304 - 3304	XR	50

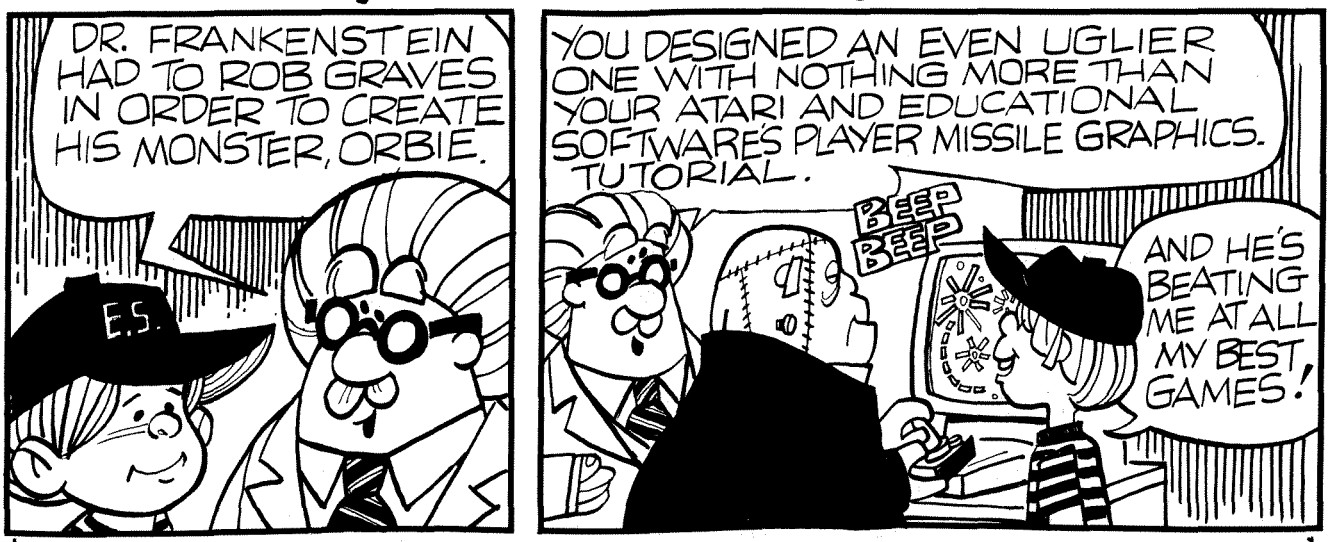
```
SS SS SS SS SS SS SS SS SS SS SS
SS        Atari  BASIC         SS
SS          "Scipio"           SS
SS       Music by: Handel      SS
SS  Arranged by: Harry Bratt   SS
SS      Copyright (c) 1982     SS
SS SoftSide Publications, Inc  SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
3100 DATA 192,32,0,7,14,28,56,112,0,21
0,195,82,85,93,236,69,128,212,210,195,
202,210,195,210,148,146,195,138,148
3102 DATA 69,236,210,195,210,204,140,1
48,149,21,138,195,146,146,195,234,140,
146,195,18,196,69,132,228,148,148,139
3104 DATA 203,212,189,21,140,203,139,1
48,148,252,69,252,170,195,146,210,195,
210,148,148,212,69,212,204,139,147
3106 DATA 212,146,195,18,196,69,20,138
,195,146,210,195,74,12,20,93,13,108,20
,20,12,148,69,108,210,195,210,76,12
3108 DATA 20,21,21,138,195,146,146,195
,234,140,146,195,18,196,69,194,195,82,
85,93,236,69,128,212,210,195,202,210
3110 DATA 195,210,148,146,195,138,148,
69,236,210,195,210,204,140,148,149,21,
138,195,146,32,0,8,16,32,64,128,0
3112 DATA 146,195,234,32,0,9,18,36,72,
144,0,140,146,195,18,32,0,10,20,40,80,
160,0,196,69,6
3200 DATA 139,4,85,85,212,69,172,204,1
40,204,210,195,146,140,69,196,202,195,
210,210,195,202,212,128,228,76,69,12
3202 DATA 36,84,92,60,140,69,132,92,69
,69,60,69,12,204,4,5,140,204,84,132,15
6,188,148,84,196,212,172,76,196,109
3204 DATA 196,60,140,204,84,28,69,69,7
7,68,12,77,12,76,140,69,68,202,195,210
,210,195,74,84,128,100,76,69,12,202
3206 DATA 195,2,154,195,2,220,138,195,
2,188,69,4,132,37,85,212,69,172,204,14
0,204,210,195,146,140,69,140,210,195
3208 DATA 210,210,195,202,212,128,228,
76,197,12,202,195,2,154,195,2,220,138,
195,2,60,5,6
3300 DATA 134,4,128,85,13,148,69,20,10
8,84,84,76,84,44,60,132,28,60,132,68,2
0,20,13,20,108,60,108,84,128,68,108
3302 DATA 44,60,133,196,140,148,148,14
8,141,20,108,60,196,140,148,148,212,84
,164,236,140,204,156,236,212,84,204
3304 DATA 156,164,212,128,196,236,212,
84,36,234,195,138,202,195,82,20,12,21,
21,132,20,12,20,20,124,44,60,132,28
3306 DATA 60,132,68,20,20,13,20,108,60
,60,84,68,108,44,60,4,132,69,13,148,69
,20,108,84,84,76,84,44,60,132,28,60
3308 DATA 132,68,20,20,13,20,108,60,60
,84,68,108,44,60,120,6
```

ATARI® SWAT TABLE FOR:
SCIPIO DATA
(Modified Parameters: NU = 2, B = 500)

LINES	SWAT CODE	LENGTH
3100 - 3102	FB	216
3104 - 3106	YB	213
3108 - 3110	ZP	213
3112 - 3200	PN	194
3202 - 3204	KB	214
3206 - 3208	PV	192
3300 - 3302	OZ	214
3304 - 3306	NP	214
3308 - 3308	KB	56

The Adventures of PROFESSOR VON CHIP & ORBIE



TRICKY TUTORIALS (tm)

There are many things that the ATARI computers can do either better, or easier than other small computers. The following series of programs is designed for anyone who is at least familiar with BASIC programming. What each tutorial offers is similar to an extensive magazine article with all discussion in as simple language as possible, plus you get MANY examples already typed in and running. The instruction manuals range from 10 to 50 pages, and some tutorials fill up a complete tape or disk. There is little overlap in what is taught, so anyone wanting to know all they can should buy them all (my banker thanks you). ATARI buys these from us to use in training their own people! Rave reviews have been published in ANTIC, ANALOG, CREATIVE COMPUTING, and even INFOWORLD. You trust INFOWORLD, don't you?

TT #1: DISPLAY LISTS—This program teaches you how to alter the program in the ATARI that controls the format of the screen. Normally, when you say "Graphics 8", the machine responds with a large Graphics 8 area at the top of the screen and a small text area at the bottom. Now, you will be able to mix various Graphics modes on the screen at the same time. The program does all of the difficult things (like counting scan lines). You will quickly be able to use the subroutines included in your own programs.
16K Tape or 24K Disk. **$19.95**

TT #2: HORIZONTAL/VERTICAL SCROLLING—The information you put on the screen, either GRAPHICS or TEXT, can be moved up, down, sideways, or diagonally. We provide the basic methods and leave the rest up to your skill and imagination. Includes 18 examples to get you started, with several using a small machine language subroutine for smoothness.
16K Tape or 24K Disk. **$19.95**

TT #3: PAGE FLIPPING—Now you don't have to redraw the screen every time you change the picture or text. You will learn how to have the computer draw the next screen you want to see while you are still looking at the previous screen, then flip to it instantly. You won't see it being drawn, so a complicated picture can seem to just appear. Depending on your memory size and which graphics or text modes you are using, you can instantly look at up to 50 pages. The basic method takes only 9 lines and the usefulness is infinite.
16K Tape or 24K Disk. **$19.95**

TT #4: BASICS OF ANIMATION—This program shows you how to animate simple shapes (with some sound) using the PRINT and PLOT commands, and it also has a nice little PLAYER/MISSILE GRAPHICS game you can learn from. The P/M example is explained and will get you started on this complicated subject (more fully explained in TT #5). This would be an excellent way to start making your programs come alive on the screen with movement! Recommended for beginning users.
16K Tape or 24K Disk. **$19.95**

TT #5: PLAYER/MISSILE GRAPHICS—Learn to write your own games and other animated applications! The tutorial begins with many small examples that compliment the 50 page manual, then gradually builds up to a complete game where everything you need to know is fully explained. Also included are two machine language utilities that you can use to animate Players with from BASIC. Next we include two of the best editors currently available; one for editing playfield shapes (backgrounds); and one to edit your players, and all in glorious Technicolor!! Everything except the two editors run in 16K Tape or 32K Disk. **$29.95**

TT #6: SOUND AND MUSIC—Unless you have spent many years experimenting with the four voice channels, you will learn a lot from this one! Learn to play standard notes, chords, and whole songs using some simple "tricks". One of the nicest parts are the examples of special sound effects that you can refer to whenever you need a sound for a program or to impress a friend. This program will be of interest to all ages and levels of experience!
16K Tape or 24K Disk. **$19.95**

SPECIAL DISCOUNT

Order the first six tutorials in a 3-ring binder for $99.95, a $30.00 savings!

TT #7: DOS UTILITIES—We at Educational Software have been shocked by some of the prices others are charging to offer you small utilities to help in the use of your Disk Drive. We now offer you all of the following plus explanation as to how each was written, and how to use them: A UNIQUE MENU PROGRAM, AN AUTORUN.SYS BUILDER, DISK INSPECTOR (LOOK AT SECTORS), DISK JACKET PRINTER, AUTOMATIC FORMATTER, RECORD SAVE AND LOAD UTILITY.
32K Disk Only. **$29.95**

MASTER MEMORY MAP (tm)

This book is the most valuable source of information for your ATARI you can buy. It starts out by explaining how to PEEK and POKE values into memory, so that even new computer owners can use many of these "Tricks". Then you are given 32 pages of the memory locations that are the most useful, along with hints on how to use many of the locations. Finally, it includes hints on problems you may be having with the computer and discusses the new Graphics modes 9 to 11. Even ATARI buys this book from us! **$6.95**

CONTACT YOUR LOCAL DEALER or ORDER BELOW

We have other fine programs for ATARI computers.
Write for a catalog.
Send us your programs to sell too!

USER SUBMITTED PROGRAMS

MINI-DATABASE/DIALER—stores and edits up to 8 lines of information such as names & addresses, phone numbers, messages, inventories, or anything you want. It has the usual sort, search, and print options, but it also has an unusual feature: If your file includes phone numbers and your phone company allows touch-tone phone signals, the program will DIAL THE PHONE NUMBER FOR YOU! 16K Tape or 24K Disk. **$24.95**

THE GRAPHICS MACHINE—allows the ATARI to act like more expensive graphics computers using simple commands like line, box, circle, polygon, fill, and savescreen to get a high resolution picture you can save on disk in only five seconds! Many more features! 48K Disk Only. **$19.95**

BOB'S BUSINESS—14 small business type programs accessed from a common menu. 16K Tape or 32K disk. **$14.95**

MINI WORDPROCESSOR—A simple text editor to write, save, and print several pages at a time. 32K Tape or Disk. **$19.95**

KID'S #1—Includes a MATH QUIZ, a children's TREASURE HUNT, and a DIALOGUE program. 16K Tape or 24K Disk. 3 for . . . **$14.95**

KID'S #2—SPELLING BEE, WORD SCRAMBLE, and TOUCH. 16K Tape or 24K Disk. 3 Educational games for . . . **$14.95**

PLAYER PIANO—Turns your keyboard into a mini-piano and more. 24K Tape or 32K Disk. **$14.95**

DOG DAZE—Two cute little doggies race for the fire hydrants, shoot their bones, and just have a lot of fun! A fast action program for all ages. 8K Tape or 16K Disk, in machine language. **$16.95**

GRAPHIC SYMBOL LABELS—for your keyboard to remind you of the built-in Graphics symbols. 2 complete sets for . . . **$2.95**

OUR NEWEST PROGRAMS

*** OUR BEST GAME ***

SPACE GAMES—Our family is being attacked by ALIENS, and only you can save us. A comic book manual will guide you through three games that test your ability in space skills. Includes ALIENS, SURVIVE, and ROBOT ATTACK, and is for all ages. The first two games require 16K for Tape. The last game and all Disk users need 32K. **$24.95**

MATHS FOR FUN—Another ENGLISH import teaching basic math skills. Very colorful and enjoyable to use. For ages 5 to 16. 16K Tape or 24K Disk. **$19.95**

TT #10: SOUND EFFECTS—From laser blasts to ringing phones, this tutorial will show you how to make unique sound effects in all of your programs! 16K Tape or 24K Disk. **$19.95**

MARATHON—This is a unique math quiz for one or two players. You are in a race to move your runner across the screen first! There are four levels of play with five modes of operation for each. The game uses joysticks for all input, so play is easy for young children. This wonderful learning tool is imported from ENGLAND for your learning pleasure. Your kids will never even notice they are playing an EDUCATIONAL program. 16K Tape or 24K Disk. **$19.95**

TT #8: CHARACTER GRAPHICS—Character Graphics is the best way to animate your ATARI! Make letters look like space monsters, gunfighters, or a myriad of other shapes. Use our editor to create these multicolor shapes and then we'll show you how to move them around the screen. This tutorial even shows how our Space Games were written!
16K Tape or 24K Disk. **$19.95**

To order COD, VISA or MasterCard call — **(408) 476-4901**

By mail include $3.00 postage (USA only) or $1.50 Memory Map only—
California residents add 6.5% TAX.—Specify Tape or disk.

Educational Software inc.

4565 Cherryvale Avenue
Dept. SS
Soquel, Ca. 95073
(408) 476-4901

EXPLORING THE ATARI® FRONTIER

ANTIC and the Display List: Part III

by Alan J. Zett

Editor's Note: This is the third part of a series. To avoid confusion, we advise that you read the first two installments (*SoftSide*, May, 1982 and Issue #31) before reading this one. A solid background in BASIC, as well as some knowledge of assembly language, are also recommended.

The Beat Goes On

In the last two installments, we took our first giant step into the realm of Atari graphics. The way was made clear for us to explore the staggering potential and versatility that goes unnoticed by the BASIC programmer. We learned about the basics of the Display List and tried them out with a sample title display. We have barely scratched the surface, however. As we dig deeper and deeper into the Atari, we find a veritable gold mine of capabilities just waiting to be refined and used by intrepid programmers. This month, we are going to conclude the discussion of mixing display modes with a difficult example which will utilize all the tricks you may ever need. Take lots of notes, and forge ahead.

Figure 1: Table Of Important Memory Locations.

DECIMAL	HEX	NAME	DESCRIPTION
00087	$0057	DINDEX	Current GRAPHICS mode of screen.
00088	$0058	SAVMSC	LSB of top of screen memory.
00089	$0059	SAVMSC	MSB of top of screen memory.
00560	$0230	SDLSTL	LSB of Display List in memory.
00561	$0231	SDLSTH	MSB of Display List in memory.
00656	$0290	TXTROW	Cursor row in split screen text window.
00657	$0291	TXTCOL	LSB of cursor column in split screen text window.
00658	$0292	TXTCOL	MSB of cursor column in split screen text window.

Beating The Odds

The true heart of Atari BASIC is *not* in the BASIC cartridge. The Operating System (OS) is the true software workhorse of the Atari computer. It contains all of the routines to set up and use the video display and just about every other function performed by the system. Atari BASIC is nothing more than a set of systematic calls into the OS. The OS is actually a set of instructions for the microprocessor to follow. All work done by the Atari must sooner or later pass through the OS. By design, the Atari's 6502 Central Processing Unit (CPU) requires the use of the first five pages (a page is 256 bytes) of computer memory for storage. As long as OS is in the computer, important information pertaining to every aspect of the computer is constantly being updated in the lower five pages of memory. Since we are in BASIC, we can take a look at these locations to find out what any part of the computer is do-

ing. In fact, we can also modify these locations from BASIC and fool the OS into changing its operation. Figure 1 contains a list of all the locations we will need to handle a modified Display List. Each of these locations will be explained in more detail a little later.

The Longest Day

Take a look at Listing 1. Type it in and RUN it. For a short program, there are many complicated techniques being used. Although the display isn't very dramatic, the methods used are the tools with which masterpieces are made. At the top, there are six lines of text followed by 64 lines of GRAPHICS 7 and two more lines of text. If you look at Figure 2, you can see our custom display. The first three bytes are Blank Mode Line instructions to bring the top of the video display down 24 scan lines. Then there is a GRAPHICS 0 character mode line with the Load Memory Scan (LMS) option. Bytes five and six are the location of screen memory in LSB/MSB order. Following that are five more lines of GRAPHICS 0, 64 lines of GRAPHICS 7, and two more bytes of GRAPHICS 0. At location byte number 78, there is a Jump instruction with the Wait Vertical Blank (WVB) option and two more bytes pointing to the start of the Display List. Pretty standard stuff. Why did I say it was difficult? Not because it has several mixed modes, but because of their type and positions!

Memory Recall

Last time I said that if a mode line is positioned where its Y coordinate is greater than what is allowed for a standard Display List of that type, certain problems arise. Also, when mixing graphics with text, a few normal procedures must be followed. The best way to cover all the reasons is to dissect the BASIC program step-by-step in conjunction with the Display List listing and the important memory locations. We'll start at the obvious place — line 10.

Anatomy Of A Program

Line 10: When making a custom Display List, we follow certain guidelines. One is to always use the graphics mode that requires the most memory as a base. We use more GRAPHICS 7 than any other, so the first statement in line 10 will be GRAPHICS 23 which is GRAPHICS 7 plus 16. This sets up a standard Display List of full screen (no text window) GRAPHICS 7. Now all we have to do is modify the existing Display List. DL will contain the calculated memory location of the Display List which is found within memory locations 560 and 561. We add four to point us to a convenient position in the Display List. (This is a matter of personal preference.) The first visible mode line will be at DL-1 and all following it will be from DL + 2 on up. When we POKE DL-1,66 we are setting up a GRAPHICS 0 mode line with the LMS option. DL and DL + 1 contain the location of screen memory in LSB/MSB order. The FOR/NEXT loop then POKEs in 5 more lines of GRAPHICS 0 at DL + 2 to DL + 6. The POKE to 709 simply sets a color register. It works faster than a SETCOLOR statement and takes fewer bytes of memory. (All numbers in a program require 6 bytes regardless of the number of digits.)

Line 20: POKEing DL + 71 and DL + 72 with 2 sets the two bottom lines of GRAPHICS 0. Since we haven't used all of the bytes of the original Display List, we have to add our own ending. POKE DL + 73, 65 sets the Jump mode with WVB option and PEEKs to 560 and 561 locate the start of the Display List. POKEs to 710 and 712 also set color registers.

Line 30: The modifications to the Display List are completed. All that remains is to demonstrate how to use it. Location 87 still contains a 7 from when we first built the Display List with a GRAPHICS 23 statement in line 10. As long as location 87 contains the graphics mode in which we currently wish to PLOT, and the Y coordinates are all less than or equal to the normal screen limit when used from BASIC, all we have to do is skip over the non-graphic lines at the begining. Since there are six lines of text at the top, numbered from zero to five, the GRAPHICS 7 lines start at 6 and continue for 64 lines. All the FOR/NEXT loop does is PLOT graphics in four colors onto the GRAPHICS 7 screen. The remainder of the line exists to fix a problem associated with mixed text and graphics. If we were to go back to the text line and print something, the byte at the last location of the cursor in the graphics lines would be set to the background color. This puts an annoying blue bar on our graphics display. To get around that, we first set the color to 32 (a space character). A little known feature is that you can PLOT text in the text modes. If you set the COLOR to the ASCII character you wish to use and then PLOT X,Y, the character will appear at that location. When we set the COLOR to a space and PLOT 0,0 we will, in effect, position and PLOT on the graphics screen. But since the first line is a text line, a space is printed instead. Memory location 87 contains the current screen graphics mode which was set at 7 when the GRAPHICS 23 command was executed. By POKEing it with 0, we fool the OS into thinking we are now in GRAPHICS 0. It is now safe to simply PRINT on the top lines of text. The problem that causes the discoloration on the screen has been eliminated because OS thinks that the background color is an ASCII space which is invisible to the text display. Now comes the tricky part.

Listing 1: Sample program listing

```
10 GRAPHICS 23:DL=PEEK(560)+PEEK(561)*
256+4:POKE DL-1,66:FOR X=2 TO 6:POKE D
L+X,2:NEXT X:POKE 709,2
20 POKE DL+71,2:POKE DL+72,2:POKE DL+7
3,65:POKE DL+74,PEEK(560):POKE DL+75,P
EEK(561):POKE 710,10:POKE 712,148
30 FOR Y=6 TO 69:COLOR INT((Y+5)/5):PL
OT 0,Y:DRAWTO 159,Y:NEXT Y:COLOR 32:PL
OT 0,0:POKE 87,0
40 ZERO1=PEEK(88)+PEEK(89)*256
50 SEVEN1=ZERO1+(6*40)
60 ZERO2=ZERO1+(70*40)
70 POKE 752,1
100 X=ZERO2:GOSUB 1000
110 POSITION 2,0:PRINT "HELLO";
200 X=SEVEN1:GOSUB 1000:POKE 87,7
210 COLOR 3:PLOT 79,0:DRAWTO 79,63
300 X=ZERO1:GOSUB 1000:POKE 87,0
310 POSITION 2,0:PRINT "HELLO";
400 END
1000 POKE 89,X/256:PLOT 0,0
1010 POKE 88,X-(256*PEEK(89))
1020 RETURN
```

Lines 40-60: If we POKE 87,0 to make it think we are in GRAPHICS 0, we find that the X and Y locations of the bottom text lines start at 0,70. In GRAPHICS 0, POSITION 0,70 would generate a cursor out of range error. (The Y coordinate maximum for GRAPHICS 0 is 23.) To get around this, we have to fool the OS into thinking that the bottom two lines of text are actually the first two lines of the

Figure 2: Modified Display List Generated By Listing 1.

DL BYTE NO.	BYTE VALUE	MODE TYPE	DL BYTE NO.	BYTE VALUE	MODE TYPE
1	112	Blank	15	13	GRAPHICS 7
2	112	Blank	.	.	.
3	112	Blank	.	.	.
4	66	GRAPHICS 0 w/LMS option	.	.	.
5	nn	LSB of screen memory	72	13	GRAPHICS 7
6	nn	MSB of screen memory	73	13	GRAPHICS 7
7	2	GRAPHICS 0	74	13	GRAPHICS 7
8	2	GRAPHICS 0	75	13	GRAPHICS 7
9	2	GRAPHICS 0	76	2	GRAPHICS 0
10	2	GRAPHICS 0	77	2	GRAPHICS 0
11	2	GRAPHICS 0	78	65	Jump w/WVB option
12	13	GRAPHICS 7	79	nn	LSB of Display List
13	13	GRAPHICS 7	80	nn	MSB of Display List
14	13	GRAPHICS 7			

screen. Lines 40-60 calculate the actual top left memory location for each of the three mode line divisions. They are calculated by adding the start of screen memory in locations 88 and 89 to the number of mode lines down to the start of the division multiplied by the number of bytes required for each mode line. In this case, every mode line requires 40 bytes, both text and graphics. ZERO1 is set to the top of the screen. (We'll need this for later.) SEVEN1 is set to the top of the GRAPHICS 7 segment and ZERO2 is set to the top of the second set of GRAPHICS 0 text lines.

Line 70: Disables the text cursor block.

Line 100: Call a routine to set the top of screen memory to the top of the second GRAPHICS 0 text lines. POKEing 87,0 tell OS to function as in GRAPHICS 0.

Line 110: POSITION to new top of screen and PRINT hello. Remember, OS now thinks that the two lines at the bottom are the FIRST two lines in a screen of 24. DO NOT PRINT anything past Y location 1 or OS will overwrite the screen.

Line 200: Set top of screen to top of GRAPHICS 7 window. POKE 87,7 to make OS work in GRAPHICS 7.

Line 210: Select a COLOR and PLOT.

Line 300: Restore top of screen memory to its original position.

Line 310-400: PRINT hello and stop while still in GRAPHICS 0 mode.

Line 1000: POKE the MSB of the new top of screen memory into the pointer at location 89. Put the cursor at the top left corner.

Line 1010-1020: POKE the LSB of the new top of screen mem in to pointer at location 88 and RETURN.

Time Out!

Whew! That certainly was a lot of material! Note that we didn't really have to set a new top of screen for the line we drew in lines 200-210. This is merely a different technique for doing the same thing we did in line 30. All you need do, if you find yourself trying to write to a line beyond its normal Y coordinate range, is to use the routines presented here to set the top of screen. Note that these POKEs must be done each time you wish to write there. Also note that in our case, the number of screen bytes per mode line is always 40. If you must mix modes of different byte lengths, remember to use the technique described in the last column and adjust your memory position calculations accordingly (e.g. lines 40-60 in our demo). There is much you can do now by just experimenting. Since the program ENDs in GRAPHICS 0, you can type or do anything to the screen you like. An interesting effect can be achieved by playing with the cursor editing keys. Experiment! Modify the program however you please. If you come up with anything interesting, let us know so we can share it with all of our readers. Next time, we will get down to the nitty-gritty world of Display List Interrupts. BASIC programmers take note: this area requires an understanding of 6502 Machine Language (albeit only the basics). Now is a good time to buy that book you've been eyeing at your local computer store! That's it for this time. I'm anxious to hear from *SoftSide* readers who have comments or advice they would like to share.

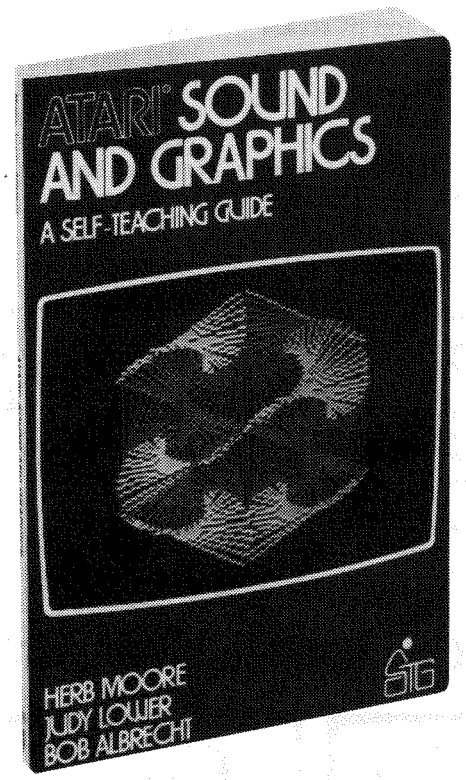
ATARI® Sound and Graphics

Reviewed by Sheldon Leemon

by Herb Moore, Judy Lower, and Bob Albrecht. (John Wiley & Sons, Inc., 234 pages, paperback) Suggested retail price: $9.95.

In this second book on ATARI® BASIC in the *Wiley Self-Teaching Guide* series, the authors have taken into account the large impact that color graphics and sound have had on the market for home microcomputers. With these capabilities, a computer is no longer just a machine to help figure your income taxes, or facilitate boring math drills for the kids, but can be a super video-game, music synthesizer, artist's tool, and much more. These possibilities have led many consumers who formerly shied away from computers to take the plunge and purchase one for their home. These people represent a new kind of computer owner — they want to be able to **do** something with the machine without first obtaining a Master's degree in computer science. Graphics and sound present an ideal opportunity to such people because, from their first programming session, they can give the computer commands whose results they can see and hear immediately. It's no wonder that many educators are excited about languages with "Turtle" graphics as a means of involving young children in the programming experience. It's also no wonder that the first thing many new computer owners want to know is how to control the colors and sounds that the machine can produce.

The first *Wiley Self-Teaching Guide* took the traditional approach of formally introducing the "core" BASIC keywords and concepts such as conditional structures and loops first, and then briefly looking at graphics and sound "extensions." This book, however, reverses those priorities to meet the needs of the new computer user. First, the SOUND statement is examined, and the user is taught to type commands to produce a wide variety of sounds. Next, the user is led through simple PLOTing of colored dots, then drawing lines, and whole pictures. Eventually, the traditional programming concepts begin to ease their way into the picture, answering questions about ways to make the computer do even fancier sound and graphics. Do you want to produce a whole range of sounds without typing each SOUND command in, one at a time? Would you like to pause between each note? The book shows you how to use a FOR-NEXT loop to accomplish these tasks, and shows you how to nest loops to combine these functions. For picking out the notes of a song, the READ, DATA combination is explained. Some of the ways that BASIC concepts are introduced are quite inventive, allowing almost the entire range of BASIC keywords to be covered, while still remaining within the original framework of exploring graphics and sound. The whole time that the authors are showing us how to use arrays to hold note values, and conditional structures to check that INPUT data is within permissible parameters, they are also exploring such concepts as the effect of attack and decay times on the shaping of musical notes.



For those not familiar with the *Wiley Self-Teaching* series, the approach that they use should be explained. The books are written in a friendly, non-technical style, and when jargon cannot be avoided, there is usually a cute cartoon explaining the term, and reinforcing the concept. For example, in the beginning, a cartoon character reminds you to hit the RETURN key after entering a statement. They assume no prior knowledge about computers whatsoever. The book takes you through the most basic concepts, such as the difference between entering a command directly, and typing it as part of a numbered program statement. Each topic is addressed as part of a hands-on session. It is assumed that as you are reading the guide, you will be typing in the examples, and observing the effect on the screen. The result is a guide that is extremely well-suited to the first-time user. Anyone should be able to sit down at the computer, and within a few minutes be typing and running simple programs. In this particular guide, the programs are of a nature that will be likely to stimulate further inquiry and experiment into the fields of graphics and sound.

The subject of ATARI® graphics and sound covers a lot of territory, and this guide should not be mistaken for a comprehensive treatment. However, for the beginner, it serves as an excellent introduction to the graphics and sound capabilities of the ATARI® computers, and through them, to BASIC programming in general.

WIN $5,000 Plus Royalties!*

For the best Talking Game

For the Atari® or Apple®II+

Using the VOICE BOX™

Now you can make your Atari® 400/800 or Apple® II games and other programs come alive with the VOICE BOX by the Alien Group — the first low-cost, smart speech synthesizer with unlimited vocabulary.

Add jokes to your programs. Insults. Compliments. Help messages. Stories. Alien voices. Animal roars. Have your computer talk to the fire department or police in emergencies. To kids. Or blind people. Teach touch typing with immediate spoken feedback. Or just about any other subject — the fun way. Or help a speech-impaired friend communicate . . . the possibilities are limitless.

The VOICE BOX plugs into your Atari's serial port. And talks directly through your TV set. Or into any Apple II slot. No power supply or special interfaces needed.

Just select from its simple screen menu. A dictionary with thousands of common words (on diskette or cassette) automatically translates your text into speech. It's that easy.

But don't let its friendliness fool you. The VOICE BOX has all 64 phonemes (basic sounds, like "ah") built in. So you can precisely create any word or sound you can imagine. And store it all on diskette or tape. Names or foreign language words, for example. Or wierd non-human languages.

Let me entertain you — The VOICE BOX is creative too. It will crack you and your friends up with non-stop random, grammatically correct sentences, using words you specify. It also has an amusing talking face with lip-sync animation — a real crowd-stopper. Best of all, you can call the VOICE BOX from any BASIC program and make your program really hum —literally!

Speech Synthesizer

Singing Apples? — Apple owners get all these capabilities too — as a plug-in card plus diskette. Or there's a deluxe version with the dictionary in ROM (no diskettes to bother with), speaker, and ability to "sing" (hey we're not making this up folks) in any key. (Both Apple versions require 32K or more. Applesoft and DOS 3.3).

Don't confuse the VOICE BOX with "dumb" speech synthesizers that can't learn new words. Or software-based ones with lower speech quality — and an annoying tendency to blank out the display when they talk. The VOICE BOX is a true breakthrough in speech synthesis. Small wonder thousands of Atari and Apple owners have already bought the VOICE BOX.

The VOICE BOX is available now at leading computer stores throughout the world. Or direct from the Alien Group, with 10-day money back guarantee if you're not completely satisfied.

VOICE BOX For Atari. $169.00
16K and 32K versions included
(Specify diskette or cassette).

VOICE BOX for Apple II.+ $139.00.
(Requires speaker.)

VOICE BOX for Apple II+. $215.00
(Includes dictionary in ROM and singing capability. Comes with speaker.)

Enclose check or money order, Visa or Mastercard #. Please include expiration date.

CHECK YOUR LOCAL COMPUTER STORE FOR TALKING "VOICE BOX" VERSIONS OF YOUR FAVORITE GAMES FROM LEADING GAME COMPANIES

*Win a $5,000 prize — plus royalties — for the best Atari 400/800 or Apple II+ game using the VOICE BOX. Deadline: May 30, 1983. Write for contest details.

Please mail to: The Alien Group, Department , 27 W. 23 St., N.Y., N.Y. 10010
Or call in order to (212) 741-1770

Advanced Music System

by Lee Actor. (from APX, The Atari Program Exchange) System requirements: Atari® 400/800 with 32K RAM, Atari BASIC language cartridge and Atari 810 or compatible disk drive. Retail price — $29.95.

Have you ever wanted to show your fellow computer enthusiasts that your computer can do more than play games? Have you ever wondered if your computer's sound generators could do more than make explosion noises? Have you ever had the desire to compose your own music, but didn't want to learn an instrument? Have you ever been interested in electronic music but didn't want to spend several dollars on hardware? Or, have you ever tried to use the Atari Music Composer cartridge, but given up in disgust due to its many limitations?

Well, no matter how you responded to the above questions, Lee Actor's *Advanced Music System* from the Atari Program Exchange is for you. The *AMS* is a music composing/editing/playing program that offers something for everyone. Both the program and the documentation are of superior quality, and the price is reasonable, making the *AMS* an excellent value for anybody that owns an Atari 400/800 with a disk drive. Purchasers of the *AMS* will not find themselves singing the blues.

A Brief Course In Electronic Music

To fully appreciate the capabilities of the *AMS*, it is necessary to have an understanding of how electronic music works.

When your ear hears a sound, it is actually detecting vibrations. The rate of vibration is called the frequency, and determines the pitch of a sound. In a musical instrument, a metal string, wood reed, stretched membrane, or air in a tube, is vibrating, and the player usually has a method for changing the frequency of vibration. Further analysis reveals that there are different kinds of vibrations. When viewed with an oscilloscope, vibrations have another characteristic called a waveform. Square, triangle and sawtooth are common waveforms.

Sound's other essential characteristic is volume. As a string is plucked or air is blown, the volume changes over a short period of time. This pattern of changing volume levels is called an envelope, and is usually divided into four parts called the attack, decay, sustain, and release. The attack describes how quickly a sound reaches maximum volume, whereas decay tells how the volume fades away.

The frequency, waveform, and envelope are all essential parts of a note, and a sequence of notes creates music. Electronic music is merely a method of producing these qualities of sound by electronic means. A device which does this is a called a synthesizer. Theoretically, it is possible for a synthesizer to imitate any musical instrument, or to produce sounds not available from conventional instruments.

The actual process is much more complicated than this brief description indicates. However, some individuals have made incredible accomplishments in the field of electronic music. The first significant commercial recording of electronic music was *Switched-On Bach* by Walter Carlos, now Wendy Carlos, who recently did the music for the Disney movie, *Tron*. Other notable names are Tomita (See ***SoftSide***, Nov., 1981.) and Synergy.

How Atari Computers Make Sounds

One of the special chips inside your Atari is called POKEY, the POrt and KEYboard controller. The POKEY chip also contains four countdown timers which can act as sound channels. Each channel can have its own frequency, noise type, and volume. A square wave is the only waveform available.

The noise type is sometimes incorrectly called the distortion, and there are six different types. Noise type ten is the most commonly used and produces a clean, even tone. Other noise types can be used for white noise and other special effects. Sixteen volume levels, including "no volume," are possible. Each channel continually produces its sound, so duration must be controlled by the user.

For more information, see chapter seven of *De Re Atari*, or the Hardware Manual, chapter two, part B, and chapter three, part F.

Advanced Music System Technical Overview

The *AMS* is capable of playing up to four independent voices simultaneously, each with a range of five and a half octaves. The program uses the familiar noise type ten for the upper three octaves, and noise type twelve for the lower ones. The *AMS* can play up to 2400 notes per second, so it is capable of playing some very intricate parts. Each note can also have one of fifteen different volume levels, or be a rest.

Using The Advanced Music System

The version of *AMS* currently available from APX is an auto-

Reviewed by Craig Chamberlain

booting Atari BASIC program on a protected disk. The program uses a menu, reproduced here for reference, to direct all of its main activities.

A. PLAY
B. ENTER/EDIT MUSIC
C. ERASE
D. PLAY MEASURE(S)
E. SAVE
F. LOAD
G. DISK DIRECTORY
H. TEMPO
I. FORMAT DISK
J. AUXILIARY PROGRAM
K. AUTO-PLAY

The ERASE option should be used first to clear any previous music parts. The ENTER/EDIT MUSIC option uses the keyboard to enter new notes and edit old ones. When this option is selected, one of the four parts must be specified, along with a measure number. Actual editing is done from the perspective of one measure. A note is entered by typing a letter key, A to G. Unless otherwise directed, the *AMS* will automatically insert appropriate sharps or flats according to the current KEY — one of the special editing aids.

As each note is entered, additional parameters can be included. These parameters include the octave, duration, whether the note is tied or abrupt, and the volume level.

Valid octave numbers range from 1 to 6, the higher numbers indicating higher octaves. The range of the *AMS* is from C1 to F6, with C4 being "middle C."

The common durations for notes are whole, half, quarter, eighth, sixteenth, and thirty-second. However, the *AMS* allows any imaginable note length to be specified. Sixty-fourth notes, triplets, septuplets, and dotted notes are all possible. If no duration is given, the most recent one is used. A special error checking feature, the METER option, keeps track of the current time signature (4/4, 6/8, etc.) and warns you when there is a discrepancy between the time signature and the total duration of the notes in a measure.

Normally, when a part is played, there is a very short rest between notes to separate them. For a smooth, legato effect, the notes can be connected ("tied" or "slurred") by using the tie parameter. Conversely, the dot parameter shortens the playing time of notes, producing an abrupt, staccato effect. Rests are automatically inserted to compensate for the shortened playing times.

Additional editing features allow the repetition of a sequence of notes or measures (a great time saver) and the ability to insert or delete notes and measures. While editing, it is possible to jump from one measure to any other, and any measure can be played immediately, making the *AMS* a very interactive music tool.

Overall, the author has done a fine job in creating a usable editor. Important information, such as the key, meter, total beats in the measure, and notes free, is clearly displayed at all times. (There are about 1400 notes available on a 32K machine, and 3400 on a 40K or 48K machine.) Errors are identified with messages. My only criticism of the editor is that it is too slow. This is understandable, since it is rather complex and is written in BASIC. However, I find that I sometimes want to enter the notes faster than it can accept them. Persons who type using the one finger method are not likely to find this a serious restriction.

It Plays The Piano

Without a doubt, the PLAY option is the best part of the program. The graphics display is very good and reveals that the author has a firm command over the Atari's sound and graphics power. The display consists of the black and white keys of a piano, with a total of five and a half octaves represented. As a note is played, the corresponding key is highlighted in one of four colors, showing the four voices. This is an extremely clear method of showing the relationship between notes typed into the computer, and the resulting sounds as they would be generated on a piano.

The PLAY option allows any combination of the four parts to be played, from just one voice to all four at the same time. Playing starts at the first measure. To play just a few measures somewhere within a whole piece of music, the PLAY MEASURE(S) option is available. Normally, tempo will have been previously set with the TEMPO option, but the paddles can be used to change the tempo while a piece is playing. Playing can be stopped at any time.

Still More Features

Disk oriented options include SAVE and LOAD, for storage and retrieval of music using DOS files, a FORMAT DISK command, and a handy DISK DIRECTORY function which also enables the user to lock, unlock, and delete files. Disk operations must always use drive number one.

The auxiliary program provides an alternate menu that lacks the editing option, but has recording and file merge functions. The advantage of this program is that it uses less memory, and with the MERGE function, allows the

generation of longer pieces of music than is possible with the main menu program. The recording functions are for use with a suitable tape recorder. Through recording techniques, it is possible to extend the octave range and mix as many as sixteen voices together.

The AUTO PLAY program is a great way to show your friends what your computer can do. As the name implies, this program automatically searches the disk and plays every music file it finds. It will continue doing this until you stop it. Six demonstration music files are provided on the disk.

The *AMS* lacks a program to print music files to a printer; this could be a handy feature for some people. It is also important to note that the music generated by the *AMS* cannot be merged with other programs.

The Other Half Of *The Advanced Music System*

All too often, documentation for a program is overlooked. Fortunately, however, the author's attention to detail is as evident in the *AMS* manual as it is in the program. The manual is one of the very few to be granted a rating of EXCELLENT by the Atari Program Exchange. It is very complete — in thirty pages it fully describes all of the program's options, includes a sample session, and a section of helpful hints for entering music.

Something For Everyone

The *AMS* should appeal to a wide range of people. Because it is so interactive, the program is great both for the composer, or for someone just learning about music. The program is also a good demonstration of sound and graphics applications.

In order to enter music and understand most of the user manual, a minimal knowledge of music terms and theory is required. With the help of a knowledgeable friend, however, a novice can learn the fundamentals and start entering music in just a few hours. For someone interested in learning music theory, may I suggest the book, *Independent Study In Music Theory* by Frederic Fay Swift. It is intended for younger students, but is complete, inexpensive, and can be read by anyone.

If you would simply like to have the computer play music, and do not have a need for the music entry/editing capabilities of the *AMS*, the program *JUKEBOX #1* (APX 20135, 32K, disk, Atari BASIC, $15.95) might be of interest. Written by the same author, it uses the PLAY option of the AMS, including the piano keyboard display and tempo control by paddles, and comes with several demonstration pieces for you to enjoy.

The Advanced Advanced Music System

Although truly advanced, the *AMS* is still not the last word. The author is currently working on a revised version that will, no doubt, be the ultimate music program for

continued on page 82

SOFTSIDE ORDERING INFORMATION

FORM OF PAYMENT

USA

VISA, MasterCard, certified checks, money orders and personal checks are accepted.

Canada/Mexico

The preferred method of payment is by VISA or MasterCard. A bank check is acceptable if it has been preprinted for payment in U.S. dollars. No personal or company checks accepted.

Other Foreign Orders

Payment must either be by a bank check drawn on a U.S. bank payable in U.S. dollars or by affiliated bank credit cards of VISA or MasterCard.

GUARANTEE

All software is guaranteed to load and run. If you experience difficulties with the product within 30 days, it may be returned for replacement. Send your properly protected tape or disk to the attention of the Customer Service Representative and include your name, address, and the reason it is being returned.

LIABILITY

All software is sold on an as-is basis. **SoftSide** assumes no liability for loss or damage caused or alleged to be caused directly or indirectly by products sold or exchanged by them or their distributors, including, but not limited to, any interruption in service, loss of business or anticipatory profits or consequential damages resulting from use or operation of such software.

PRICES

Prices are subject to change without notice. We are not responsible for typographical errors.

Unless otherwise noted in a published advertisement, the following prices are in effect as of this issue:

	USA/Canada APO/FPO	USA/Canada FIRST CLASS Mexico	Other Foreign
SoftSide Magazine (yr)	$30	$40	$62
SoftSide Magazine (6 mo.)	$15	$20	$31

	USA APO/FPO	Mexico Canada	Other Foreign
CV (year) & magazine	$75	$95	$125
(6 mo.)	$40	$50	$70
DV (year) & magazine	$125	$145	$175
(6 mo.)	$70	$80	$90

Adventure of the Month	Disks	Cassettes
3-Mo. Trial	$29	$19
6-Mo.	$49	$29
12-Mo.	$89	$49
24-Mo. Charter	$169	$89

BACK ISSUES

Minimum order for magazines only — 3 issues. There is no minimum order for magazine/media combinations.
Price includes shipping to the 48 states only. Alaska, Hawaii, Puerto Rico, APO/FPO, and ALL foreign orders — postage is additional.

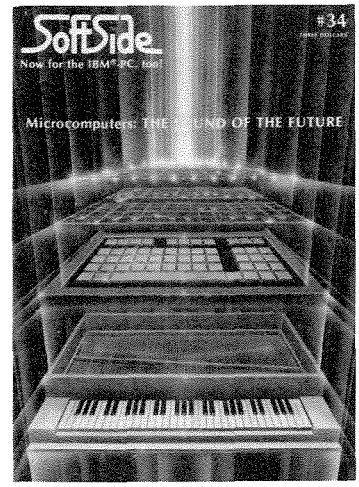
ALL Foreign orders and all magazine/media combination orders — Order directly from **SoftSide, 6 South St., Milford, NH 03055.**

Order This Month's Programs on Disk or Cassette! Get Additional Programs to Enjoy Without Typing!

SoftSide

ORDER THIS MONTH'S MEDIA

Having difficulty finding the time to input the programs from this issue of ***SoftSide***? Don't despair — the cassette and disk versions are still available! **And**, each DV (disk version) contains an additional program.



☐ **#34 DV $16.00** ☐ **#34 CV $8.00**

I own an ☐ Apple™ ☐ IBM®-PC **(Disk Only)**
☐ ATARI® ☐ TRS-80® Mod. I ☐ TRS-80® Mod. III

Name _____

Address _____

City/State _____ Zip _____

☐ **Check or Money Order** (Please enclose this card and your payment in an envelope.)

☐ MasterCard ☐ VISA

Name of Cardholder _____

MC# and Interbank#/VISA# _____

Exp. Date _____ Signature _____

Prices for USA only. For foreign orders, see page 80. Please include exact postage on all foreign orders.

Prices subject to change without notice. Apple™, IBM®, ATARI® and TRS-80® are registered trademarks of The Apple Computer Company, International Business Machines, Inc., Warner Communications, and The Tandy Corporation respectively.

Subscribe to Adventure of the Month

You never know where it might take you.

The Adventure is Waiting for You...

NEW MEMBERSHIPS NOW AVAILABLE

■ **Check the type of Membership you want:**

☐ Trial **Disk** Membership (3 mo.) $29
☐ Trial **Cassette** Membership (3 mo.) $19

Disks	**Cassettes**
☐ 6-Mo. Member $49	☐ 6-Mo. Member $29
☐ 12-Mo. Member $89	☐ 12-Mo. Member $49
☐ 24 mo. Charter $169	☐ 24 mo. Charter $89

Send: ☐ Apple™ ☐ ATARI® ☐ TRS-80®

■ **Super Disk Three Adventures — ($26 each):**

☐ Arabian, Alien, Treasure Island
☐ Black Hole, Windsloe Mansion, Klondike
☐ Jack the Ripper, Crime, Around the World
☐ James Brand, Witches' Brew, Titanic

NEW ☐ Arrow One, Robin Hood, The Mouse That Ate Chicago

Prices for USA only. For foreign orders, see page 80. Please include exact postage on all foreign orders.

■ **Individual adventures (please specify)** _____

_____

☐ **Cassette** — $7 each ☐ **Disk** — $10 each

Name _____

Address _____

City/State _____ Zip _____

Please include exact postage on all foreign orders.

☐ Check or Money Order (Please enclose this card and your payment in an envelope.)

☐ **MasterCard** ☐ **VISA**

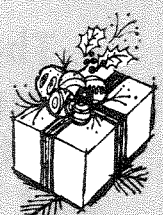
Name of Cardholder _____

MC# and Interbank#/VISA# _____

Exp. Date _____ Signature _____

Prices subject to change without notice. Apple™, IBM®, ATARI® and TRS-80® are registered trademarks of The Apple Computer Company, IBM, Inc., Warner Communications, and The Tandy Corporation respectively.

The Perfect Christmas Gift For



Sons ● Daughters
Dads ● Moms ●
Friends ● Grandparents
New Users

If they own an APPLE™, IBM®-PC, ATARI®, or TRS-80® You Can't Go Wrong With a Gift of

SoftSide™

Give *SoftSide* For Xmas!

If they own an Apple™, IBM®-PC, ATARI®, or TRS-80® there is no better gift than ***SoftSide*** for Christmas. The programs are entertaining, the reading relaxing and the education you'll be giving is priceless. Take the time to fill out this card and give the year round source of entertainment to a friend or relative — ***SoftSide.***

Initial Gift Subscription — $22.00/year

TO: Name _____

Address _____

City/State/Zip _____

FROM: _____

Send: ☐ Apple™ ☐ IBM®-PC ☐ ATARI® ☐ TRS-80®

☐ Check here if you would like us to send a card announcing your gift.

☐ Check or Money Order (Please enclose this card and your payment in an envelope.)

Prices for USA only. For foreign orders, see page 80. Please include exact postage on all foreign orders. Prices subject to change without notice, Apple™, IBM®-PC, ATARI® and TRS-80® are registered trademarks of The Apple Computer Company, IBM, Inc., Warner Communications and The Tandy Corporation respectively.

Additional Gift Subscriptions — $20.00/year

TO: Name _____

Address _____

City/State/Zip _____

FROM: _____

Send: ☐ Apple™ ☐ IBM®-PC ☐ ATARI® ☐ TRS-80®

☐ **MasterCard** ☐ **VISA**

MC# and Interbank#/VISA# _____

Name of Cardholder _____

Address _____

Exp. Date _____

Signature _____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 36 MILFORD, NEW HAMPSHIRE 03055

POSTAGE WILL BE PAID BY ADDRESSEE

SoftSide™

6 South Street
Milford, NH 03055

SS34

Order This Month's Programs on Disk or Cassette! Get Additional Programs to Enjoy Without Typing!

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 36 MILFORD, NEW HAMPSHIRE 03055

POSTAGE WILL BE PAID BY ADDRESSEE

Adventure of the Month
6 South St.
Milford, N.H. 03055

SS34

Subscribe to Adventure of the Month

You never know where it might take you.

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 36 MILFORD, NEW HAMPSHIRE 03055

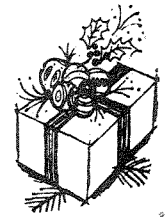
POSTAGE WILL BE PAID BY ADDRESSEE

SoftSide™

6 South Street
Milford, NH 03055

SS34

The Perfect Christmas Gift For



Sons ● Daughters
Dads ● Moms ●
Friends ● Grandparents
New Users

If they own an APPLE™, IBM®-PC, ATARI®, or TRS-80® You Can't Go Wrong With a Gift of

SoftSide™

The Adventure is Waiting for You...

How would you like to go back in time to 19th century London to match wits with Jack the Ripper? Out into space to brave the swirling vortex of a black hole? Into the depths of the ocean, or on a quest to rescue a beautiful princess from the clutches of evil monsters?

You never know where **SoftSide Magazine's Adventure of the Month Club** might take you. But you can be sure that each month you will experience new delights and new challenges as you receive an original adventure on tape or disk, ready to load into your computer.

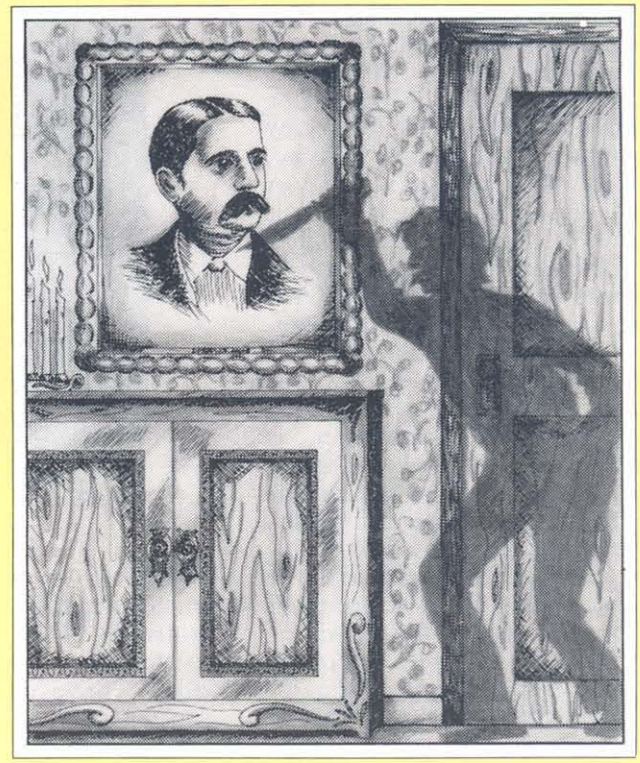
And now it's even easier for you to join **Adventure of the Month.** A Trial Membership (3 months 3 different Adventures), costs only $29 for Disks, $19 for Cassettes.

Or choose a Charter Membership, available on disk or tape, for 6 months, 12 months, or 24 months. Your choice of a 24 month Charter Membership will bring you a savings of almost half on the individual price of the Adventures you'll receive.

Super Disks, each containing three Adventures, are also available for $26 each.

For more detailed information on prices, and to enter your membership, see the handy postage-free bind-in card found facing page 80 of this issue.

Adventure #17



ADVENTURE OF THE MONTH
THE DEADLY GAME

Well, your uncle Henry has passed on, and now you're rich. But, first you have to attend the reading of the will. When you get there, you find that the will requires you and your five greedy cousins to spend a night in Henry's booby-trapped mansion and survive. You discover that one of the five is out to make sure that he is the only survivor. And the evening is just beginning...

Adventures Available:

June 1981
Arabian

July 1981
Alien

August 1981
Treasure Island

September 1981
Jack The Ripper

October 1981
Crime

November 1981
Around the World in Eighty Days

December 1981
Black Hole

January 1982
Windsloe Mansion

February 1982
Klondike

March 1982
James Brand

April 1982
Witches' Brew

May 1982
Titanic

June #13
Arrow One

July #14
Robin Hood

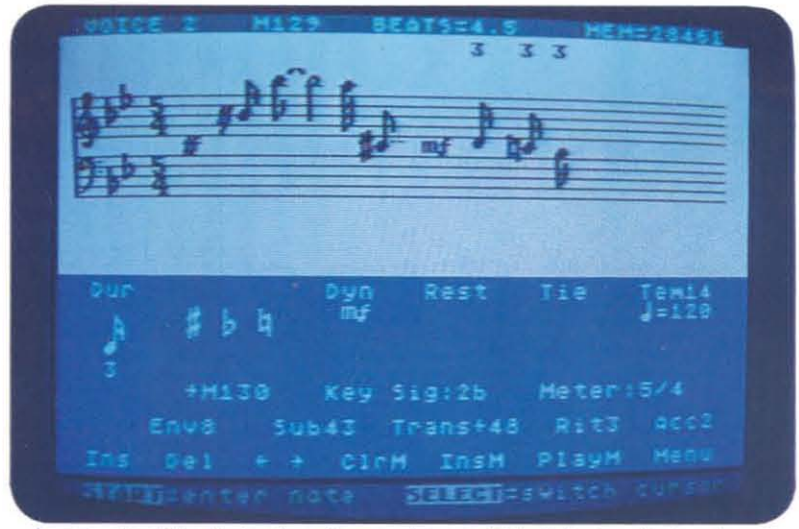
August #15
The Mouse That Ate Chicago

September #16
Menagerie

See bind in card for ordering your adventure **NOW**!

Advanced Music System *continued*

Atari 400/800 computers. The revision will allow the program to run on a 16K, cassette based, Atari 400. The new version will be written entirely in Machine Code. The keyboard editing will be replaced by graphics oriented joystick editing, complete with musical staves. A sample display of this new editor is shown here. There will also be selectable envelopes. (The only way to do envelopes on the current version is to tie together several notes of increasing or decreasing volume.) Another major enhancement will be nestable subroutines, which, along with other data compaction, is bound to reduce the size of music source files. Unlike the current version, the tempo will be changed by control of the music source, and different tone types should also be available. Other features are possible, but not yet definite. It will be many months, however, before the revision is completed, so persons interested in the *AMS* should get the current version now and start enjoying it.



An example of the display from the new *Advanced Music System*. (to be released in 1983.)

The Grand Finale

The *Advanced Music System* is a major step forward in Atari 400/800 software. The program is loaded with features, thoroughly debugged, idiot proofed, and properly documented. It is very interactive and fun to use. The graphics and sounds are most impressive. The many hours of development are evident, and it was time well spent. Finally, the program is priced right, making it a good software value. Needless to say, this reviewer is quite pleased with Lee Actor's *Advanced Music System*, and feels confident that users of the program will be pleased as well.

APPLE FUGUE

by William Morris and John Cope (with apologies to J. Bach)

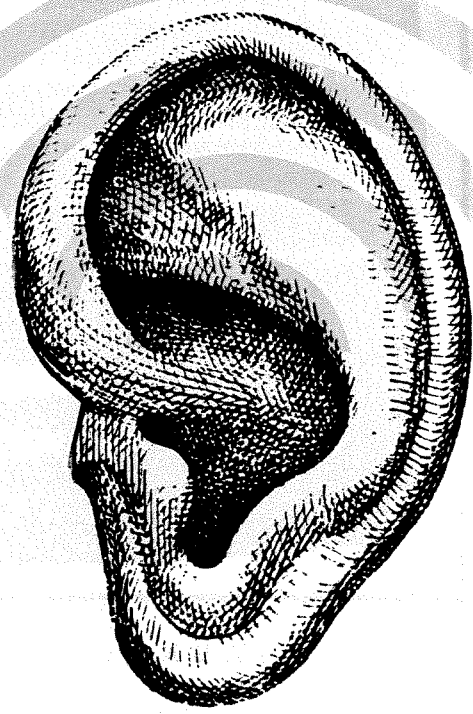
***Apple Fugue* requires a 16K Apple™ with Applesoft.**

Beauty, as a certain learned observer noted years ago, is in the eye of the beholder. The quality of sound, we would suggest, is quite properly within the province of the individual "earholder." Certainly, the Apple's ability to produce sound has been one of its enduring strengths. Consider what many of your favorite programs would be like without this capability. Silence may be golden, but it can also be extremely boring when you are sitting in front of your micro for more than five minutes! Of course, as we observed earlier, the type of sound produced can be of a highly subjective nature. But then, that is what this article is all about!

The Apple's sound is produced through its built-in speaker which occupies memory location 49200 (or -16336 if you are a purist). Whenever you access this memory location, the Apple's speaker is "toggled." If the cone was "out," it is popped in and vice versa. The result of this "toggling" is the production of a clearly audible tone. To confirm this, try the following little program:

```
10 FOR X = 1 TO 100: Z =
PEEK(49200) : NEXT X
```

Of course, the sound produced does not sound quite like a musical note. This led, several years ago, to the development of Machine Language routines to overcome this problem. While different in some respects, most of these routines followed the same general procedure, POKEing the pitch of the note to be played into one register, its duration into another location and, thereafter, calling the Machine Language sound routine. (This routine can be found in locations 927-949 in *Apple Fugue*). The fact that this method has persisted for so long is eloquent testimony to its value. The sound produced is crisp, clear, memory efficient, and, most importantly, subject to the programmer's control.



The standard method of producing musical notes on the Apple has some limitations. For example, the volume is always uniform with this technique, while the maximum duration of the note is limited to the largest number you can POKE into a single memory location (255). Trying to sustain long notes only produces a "choppy" effect, distracting from your original purpose. There is also the consideration that the sounds produced from one program to another have a certain sameness about them — "clone tones," if you will!

One possible approach in producing *different* tones on the Apple is to purchase any one of the many excellent sound boards currently available. These enable you to produce truly incredible results through an external sound system. Should you be obliged, however, to continue relying on the Apple's built-in speaker, don't despair. There is another way!

We stated earlier that sound is reproduced by means of "toggling" the built-in speaker. In his excellent publication for Machine Language programmers, *Apple Assembly Line*, Bob Sander-Cedarlof suggests another method for sound reproduction based on manipulating the Apple's internal machine cycles. By controlling the rate at which the speaker is toggled in a slightly different manner, it is possible to achieve startling effects. For example, you can control the volume of the sound produced. His technique also permits two tones to be played at one time and allows duration values far in excess of what was possible using the standard method for sound reproduction.

Apple Fugue represents an attempt to showcase each of the Apple's sound reproduction methods. The program first plays Bach's *Fugue* using the "standard method." This is followed by a demonstration of volume control technique while playing a single note (There is no volume control over the two tone reproduction.) and, finally, a replay of *Fugue* using the two tone techinque. This should allow you to make a proper evaluation of the strengths and weaknesses of each method.

One last observation: while the standard sound method is invoked with a "Call" statement, the other techniques are activated with the "ampersand" method. This technique of calling assembly language programs has been discussed at length in previous issues of ***SoftSide*** and other publications and requires no additional commentary. Should you employ these routines in your own programs, however, you should be aware of several key points.

In using the volume control routine, your calling procedure should be as follows:

100 &S,x,y,z

The variable x will represent the pitch of your note, while y represents the duration. This latter note can have a value of up to 65,535. The variable z denotes volume and can range from 0 to 15. We start the volume very low (1) in *Fugue* and increment it by steps of .1. This brings it up to maximum by the time it has completed playing the 142 notes. (The Machine Language routine treats 14.1 as 14 so there is no need for the INT command.)

The two tone method can be utilized in the following manner:

110 &T,x,y,z

The variable x represents the pitch, while y represents duration with the same limitations applying to the single note routine. The duration value has been set at twice the value of the single note routine. The variable z represents the pitch of the second note. In *Apple Fugue* we have chosen to have the second note one half the pitch value of the first note, creating a very interesting harmonic effect.

The Machine Language routine for the single tone and two tone techniques are located in memory locations 784-926.

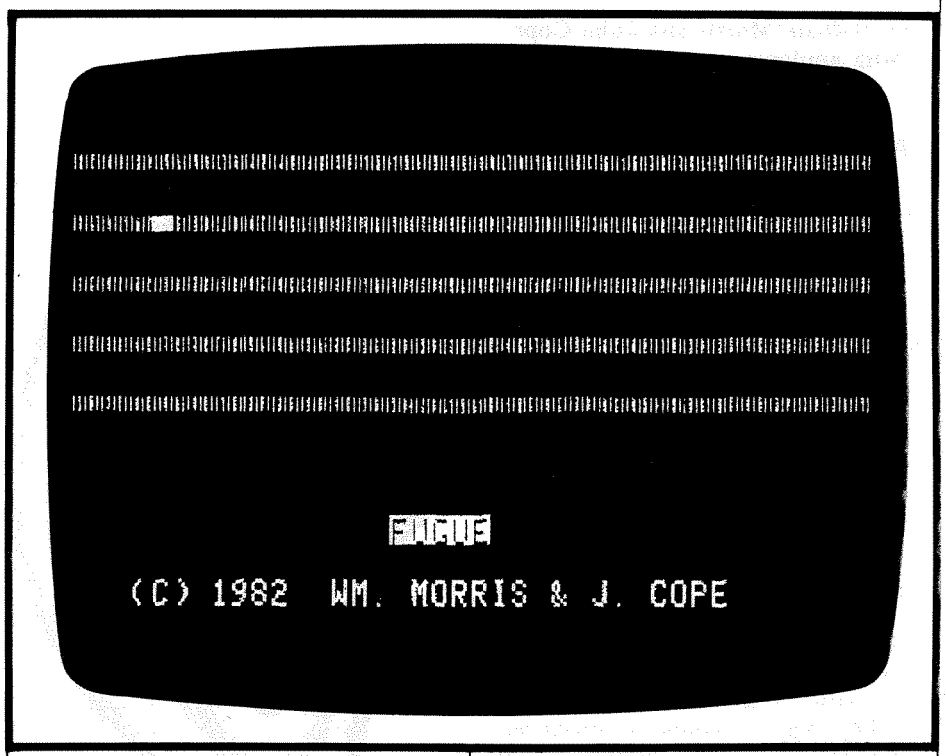
Variables

Array Variables

D(*): Duration of note to be played.
GG(*): Pitch of note to be played.
VP(*): Vertical position of note to be placed on the screen.

Other Variables

S: Passed by ampersand routine to Machine Language program to indicate a single note to be played.
T: Passed by ampersand routine to Machine Language program to indicate two notes are to be played at the same time.
V: Volume of note to be played.
XX: A flag used to determine which sound routine to play.
ZA: Horizontal position of note to be placed on the screen.
M, N, ZZ: Counters.




```
SS SS SS SS SS SS SS SS SS SS SS
SS                             SS
SS      APPLESOFT BASIC        SS
SS       'APPLE FUGUE'         SS
SS AUTHOR:W. MORRIS & J. COPE SS
SS     COPYRIGHT (C) 1982      SS
SS SOFTSIDE PUBLICATIONS, INC SS
SS                             SS
SS SS SS SS SS SS SS SS SS SS SS
```

Initialization.

```
10  GR : HOME
20  DIM GG(142),VP(108),D(142):V =
    1
```

Set up screen display.

```
30  COLOR= 4: FOR N = 16 TO 32 STEP
    4: HLIN 0,39 AT N: NEXT
40  INVERSE : HTAB 17: PRINT "FUG
    UE": NORMAL
50  VTAB 23: HTAB 4: PRINT "(C) 1
    982  WM. MORRIS & J. COPE"
```

Poke in sound routines.

```
60  FOR ZZ = 784 TO 949: READ N: POKE
    ZZ,N: NEXT
```

Set up "&" vector to point at routine located at 784.

continued on page 86

A feast of computing ideas.

If you work with a 6502/6809-based system, you're probably hungry for the facts and ideas that will help you understand the inner workings of your computer. You want to go beyond canned software—use your computer for more than games—learn the advanced programming techniques that enable you to get the most out of your 6502/6809 system.

MICRO, The 6502/6809 Journal, gives you page after page, month after month, of solid information to sink your teeth into. **MICRO** is the premier how-to magazine for serious users of the Apple, PET/CBM, OSI, Atari, AIM, SYM, KIM, and all 6809 based systems including the TRS-80 Color Computer. It's a resource journal internationally respected by professionals in business, industry, and education. Every issue of **MICRO** keeps you informed with up-to-the-minute data on new products and publications:

- **hardware catalog** with organized, concise description
- **software catalog** in an easy-to-use format
- **new publications** listed and annotated
- **reviews and evaluations** of significant products

And there's much more:

- **In-depth hardware tutorials** bring expert advice into your home or office.
- **Detailed discussions of programming languages** deepen and broaden your programming ability.
- **Complete program listings** enable you to increase your machine's capabilities.
- **Bibliography of 6502/6809 information** helps you to find pertinent articles in a timely manner.
- **Special monthly features** with in-depth treatment of one subject or system increase your knowledge of the field.
- **Balanced mix of machine-specific and general articles** for your everyday use as well as long-range reference needs.
- **Informative advertising** focused specifically on 6502/6809 machines keeps you abreast of latest developments.
- **Reader feedback** puts you in touch with other micro-computerists.

MICRO is the magazine you need to get the most from your own 6502/6809 system!

You'll love every byte.

To order, send your check or international money order (payable to MICRO) and the order form at left, to:

Subscription Fulfillment
MICRO, Dept. MI
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

Or, for your convenience, call our toll-free number:

1-800-345-8112
(In California, 800-772-3545, Ext. 564)

and charge your subscription to your MasterCard or VISA. (All orders must be prepaid in U.S. dollars or charged to your MasterCard or VISA.)

YES! I want to get more from my microcomputer. Please send me

___ year(s) of MICRO at $_____/year. (Outside U.S. and Canada, please indicate via ☐ surface or ☐ air mail.)

Name

Company

Street

City State Zip Code

☐ Check enclosed for $ _____
☐ Charge my credit card account
☐ VISA ☐ MasterCard

Signature

Card number

Expiration date

SUBSCRIPTION RATES (U.S. dollars) Yearly subscription (ISSN 027-9002) saves 20% off the single-issue price.

U.S. $24*

Canada $27

Europe $27 ($42 by air mail)

Mexico, Central America, Mideast, North and Central Africa $27 ($48 air)

South America, Far East, South Africa, Australasia $27 ($72 air)

* SPECIAL OFFER—U.S. ONLY: Save even more—30% off single-issue price: 2 years, $42

Dept. S S

```
70  POKE 1013,76: POKE 1014,16: POKE
    1015,3
```

Set up vertical screen position for each note value.

```
80 VP(33) = 10:VP(35) = 12:VP(37)
    = 12:VP(40) = 14:VP(42) = 1
    4:VP(45) = 16:VP(47) = 18:VP
    (50) = 18:VP(53) = 20:VP(57)
    = 20:VP(60) = 22:VP(64) = 2
    4:VP(68) = 24:VP(72) = 26:VP
    (76) = 26:VP(81) = 28
90 VP(85) = 28:VP(91) = 30:VP(96)
    = 32:VP(102) = 32:VP(108) =
    34
```

Read pitch and duration values into arrays D and GG.

```
100  FOR ZZ = 1 TO 142: READ N:GG
     (ZZ) = N: READ M:D(ZZ) = M
110  NEXT
```

Display current sound routine being used.

```
120  IF XX = 0 THEN  VTAB 23: CALL
      - 958: PRINT "    STANDARD
     APPLE MUSIC REPRODUCTION"
130  IF XX = 1 THEN  VTAB 23: CALL
      - 958: PRINT "   SINGLE TON
     E SOUND WITH VOLUME"
140  IF XX = 2 THEN  VTAB 23: CALL
      - 958: PRINT "   DUAL TONE
     SOUND REPRODUCTION"
```

Begin loop to play entire sequence of notes.

```
200 ZA = 0: FOR N = 1 TO 142
```

If note is a rest, delay, then go to the next note.

```
210  IF GG(N) = 0 THEN  FOR M = 1
      TO 400: NEXT : GOTO 290
```

Set horizontal position for note being played.

```
220 ZA = ZA + 2: IF ZA > 39 THEN
     ZA = 2
```

Standard Apple sound routine.

```
240  IF XX = 0 THEN  POKE 927,GG(
     N): POKE 928,D(N) * 40: CALL
     929: GOTO 270
```

Two-tone sound routine.

```
250  IF XX = 2 THEN  & TGG(N),200
     0 * D(N),GG(N) / 2: GOTO 270
```

Sound routine with volume.

```
260  IF XX = 1 THEN V = V + .1: &
     SGG(N),1000 * D(N),V
```

Clear note from display.

```
270  COLOR= 0:VP = VP(GG(N)): IF
     VP = 16 OR VP = 20 OR VP = 2
     4 OR VP = 28 OR VP = 32 THEN
      COLOR= 4
280  PLOT ZA,VP
```

Do next note. When loop ends, swap value for last note so value is in proper range for the Apple sound routine.

```
290  NEXT : IF XX = 0 THEN D(142)
      = 26
300  IF XX = 2 THEN D(142) = 6
```

Pause, then go to next demonstration.

```
310 XX = XX + 1: IF XX > 0 AND XX
     < 3 THEN  FOR M = 1 TO 3000
     : NEXT : GOTO 120
320 XX = 0:V = 1: VTAB 23: CALL -
     958: FOR N = 1 TO 2000: NEXT
     : GOTO 120
```

Machine language data.

```
1000  DATA  201,83,240,7,201,84,2
     40,55,76,201,222,32,132,3,13
     8,41,15,133,255,165,253,133,
     251,198,251,208,25,173,48,19
     2,165
1010  DATA  253,133,251,164,255,2
     34,234,136,16,251,173,48,192
     ,164,255,234,200,192,16,144,
     250,160,10,136,208,253,32,14
     3,3,144,217,96
1020  DATA  32,132,3,134,254,165,
     253,133,251,165,254,133,252,
     198,251,240,6,70,255,165,255
     ,16,7,173,48,192,165,253,133
     ,251,198,252,240,6,70,255,16
     5,255,16,7,173,48,192,165
1030  DATA  254,133,252,32,143,3,
     144,217,96,32,245,230,134,25
     3,32,190,222,76,70,231,165,8
     0,208,8,165,81,208,2,56,96,1
     98,81,198,80,24,96
1040  DATA  0,0,173,48,192,136,20
     8,5,206,160,3,240,9,202,208,
     245,174,159,3,76,161,3,96
```

Pitch and duration values.

```
1050  DATA  81,4,53,4,68,6,72,2,8
     1,2,68,2,72,2,81,2,85,2,72,2
     ,108,4
1060  DATA  81,2,108,2,72,2,108,2
     ,68,2,72,1,81,1,72,2,108,2,8
     1,2,108,1,81,1,72,2,108,1,72
     ,1,68,2,72,1,81,1,72,1,108,1
     ,53,1
1070  DATA  60,1,68,1,72,1,81,1,6
     8,1,72,1,81,1,85,1,72,1,81,1
     ,108,1,81,1,72,1,68,1,60,1,5
     3,1,47,1,45,1,47,1,53,1,45,1
     ,47,1,53,1,57,1,47,1,53,2
1080  DATA  72,2,53,2,47,2,45,1,4
     0,1,45,1,40,1,45,1,40,1,45,1
     ,40,1,35,1,40,1,35,1,33,1,35
     ,1,40,1,45,1,47,1,45,1,35,1,
     40,1,35,1,57,1,35,1,40,1,35,
     1,53,1,35,1,40,1,35,1,57,1,3
     5,1,40,1,35,1
1090  DATA  45,1,53,1,57,1,53,1,4
     0,1,53,1,57,1,53,1,35,1,53,1
     ,57,1,53,1,40,1,53,1,57,1,53
     ,1,72,2,45,2,81,2,47,2,91,2,
     72,2,53,2,45,2
1100  DATA  50,2,35,2,0,2,50,2,53
     ,2,40,2,0,2,53,2,60,1,68,1,6
     0,1,53,1,60,1,35,1,40,1,35,1
     ,68,1,40,1,42,1,40,1,72,4,45
     ,4,81,4,47,4,91,6
```

APPLE™ TABLE FOR: APPLE FUGUE

LINES	SWAT CODE	LENGTH
10 - 120	BK	515
130 - 290	UQ	413
300 - 1020	QS	508
1030 - 1070	CM	527
1080 - 1100	RB	436

What some have called "the best kept secret in the game software industry". Unleashed here are six new software games blending advanced graphics with precise elements of playability, challenge and intense fun for every gamer's tastes. Beyond these six new games are 26 more for you to consider . . . all we believe are the best software values for your money!

LEGIONNAIRE—Real-time simulation of tactical combat in Caesar's time. YOU as Caesar command up to ten legions. Finest **full-color graphics** plus playability way beyond the state of the art! Cassette . . . $35.00

V.C.—Faithfully recreates unconventional conflict in Viet Nam. YOU command chopper and artillery units, and face task of protecting civilian population where the enemy hides among the people. Cassette . . . $20.00
Diskette . . . $25.00

G.F.S. SORCERESS—Sci-Fi adventure game. YOU are Joe Justin trying to clear yourself of a false charge of mutiny and get back to the Galactic Federation Starship "Sorceress". Beautiful full-color manuals provide useful clues. Cassette . . . $30.00 Diskette . . . $35.00

ANDROMEDA CONQUEST—Vast scale space strategy game of galactic colonizing and conquest among unique star systems with strange life forms and alien technologies that provide exciting exploration and battle. Cassette . . . $18.00 Diskette . . . $23.00

MOON PATROL—Arcade Pak™ game of lunar invasion. Beats any quarter-gobbling game around! Four levels of increasing difficulty present new attackers to battle. Fast, furious and fun! Cassette . . . $25.00

TELENGARD—Dungeon adventure in a mysterious underworld with 50 levels of ever-more-complex mazes to explore. Real time fantasy and role-playing game. Using wits, magic and fast thinking, gamers fight monsters and reap valuable rewards. Cassette . . . $23.00
Diskette . . . $28.00

* Trademarks for Apple Computer, Warner Communications, Tandy Corp., International Business Machines and Commodore International Ltd.

Available at finer computer stores everywhere!

If your favorite dealer fails to have the games you want, call us toll free
1-800-638-9292

microcomputer games

® REGISTERED TRADEMARK OF MICROCOMPUTER GAMES, INC. DIVISION OF
The AVALON HILL Game Company

Avalon Hill MICROCOMPUTER GAMES are compatible with the following computer systems:
Apple II®
Atari 4/800®
TRS-80 Models I/III & Color®
IBM P.C.®
Commodore VIC-20, CBM PET & 2001®

For specific information, such as machine compatibility, memory requirements, cassette or diskette availability and price, call us toll-free 1-800-638-9292 and ask for Operator SS or write to:
Avalon Hill Microcomputer Games, Dept. SS
4517 Harford Road, Baltimore, MD 21214

A SPEECH SYNTHESIZER ON A DISK!

That's why Apples and Ataris are saying: "Talk Is Cheap"

IT'S CALLED THE SOFTWARE AUTOMATIC MOUTH, S.A.M. FOR SHORT

It's a high quality speech synthesizer created entirely in software. You use it as a software utility, load it into RAM, and then use your machine as usual, except now you can make your programs talk. It generates the speech sounds on demand, so there is no limit to what it can say.

When you hear S.A.M., you'll probably agree that it sounds better than all the hardware speech synthesizers for Apple or Atari computers. And, it has a truly remarkable price.

YOU CONTROL INFLECTION, PITCH AND SPEED

With its user-variable inflection, S.A.M. can accent words on the right syllable and emphasize the important words in a sentence.

You can also make S.A.M.'s speech higher or lower, and faster or slower, over a wide range of settings.

USE EASY PHONETIC INPUT OR PLAIN ENGLISH TEXT

S.A.M. understands a simple phonetic spelling system, not a mysterious alpha-numeric code. S.A.M. helps you learn phonetic spelling by showing you your mistakes, and the owner's manual gets you started with an English-to-phonetics dictionary of 1500 words. So it's easy to make S.A.M. produce exactly the sounds you want.

But suppose you want to type ordinary English, or you want your machine to read a word processor file aloud. The S.A.M. disk comes with RECITER, an English text-to-speech conversion program that lets S.A.M. speak from plain English text.

ADD SPEECH TO YOUR PROGRAMS WITH EASE

In a BASIC program, you add speech with just a couple of commands. In a machine language program, it's just as easy. S.A.M. comes with four demonstration programs to show off its distinctive features and help to write your own talking programs. Write adventure games with talking characters, educational programs that explain aloud, or utilities with spoken prompts – put your imagination to work.

S.A.M.

S.A.M. programmed by Mark Barton.

You can order S.A.M. directly from DON'T ASK. Add $2.00 for shipping and handling to your check or money order (or order C.O.D.)

S.A.M. for the Apple II/II+

Includes an 8-bit digital-to-analog converter and audio amplifier on a board. Only **$124.95**

Requires 48K, disk. (S.A.M. takes up to 9K; RECITER 6K.) You will also need a speaker.

S.A.M. for the Atari 400/800

S.A.M. talks through your television speaker. No additional hardware is required. Only **$59.95**

Requires 32K, disk. (S.A.M. takes up 9K; RECITER 6K.) Note: to produce the highest quality speech, S.A.M. automatically blanks the screen during vocal output; the display is preserved. S.A.M. can talk with the screen on, but the speech quality is reduced.

DON'T ASK COMPUTER SOFTWARE INC.

2265 Westwood Boulevard, Suite B-150
Los Angeles, California 90064
Telephone: (213) 397-8811

Hear S.A.M. at your favorite computer store today!
Dealer inquiries welcome.

APPLE™ DV

Auto Menu

by Fred J. Condo

Auto Menu is a disk utility for a 48K Apple™, Applesoft and a disk drive. It is included as the bonus program on this issue's Apple DV.

The impetus for creating this program was the frustrating trade-off between useful, descriptive names for the Apple's disk files and easy-to-type, short names. Wouldn't it be nice to have both descriptive names and easy typing? *Auto Menu* makes that possible.

Auto Menu turns your disks into menu-driven devices. This means that rather than having to type a file name, which may be long and easy to type incorrectly, you need only type one or two digits and a letter to access any file your Apple has created. Now any standard Apple disk can work something like ***SoftSide***'s DV, which has a menu program as its greeting. Unlike these programs, which are specific to their particular issue of ***SoftSide***, *Auto Menu* can catalog any disk.

A combination of a Machine Language routine and an Applesoft program does the trick. The Machine Language routine does the work of reading the disk's catalog without using the CATALOG command. The Applesoft program takes the information provided by the Machine Language routine, and presents it in an orderly fashion.

The first thing *Auto Menu* does is create a title page featuring your name. This distracts you while the Apple is frenetically creating its Machine Language routine and reading from the disk. The drive's red light comes on a few times, then the menu appears.

The menu consists of several lines, each consisting of a number, the type of the file in parentheses, and the file's name. At the end of the menu, all you need to do is type the number of a file or one of the standard options at the end of the menu. You can end the program at this point by hitting the RETURN key.

The standard options are REPEAT LIST, NORMAL CATALOG, and CATALOG ANOTHER DISK. The first one re-prints the menu, in case you've forgotten the files that have scrolled off the top of the screen. The second gives you a normal Apple catalog, and ends the program. The final standard option allows you to submit another disk to the program.

If, instead, you type the number of a file, one of three things can happen, depending on the file's type (the letter in parentheses). There are three types: BASIC (Applesoft or Integer), which appears as (A) or (I); binary (B); and text (T).

To choose an action, simply type the letter that appears between the angle brackets. You may run or load both BASIC and binary files; the only differences between them are the internal disk commands that must be issued to handle them. *Auto Menu* handles those details for you.

If you choose a text file, you may EXEC it, which causes its contents to substitute for the keyboard; or you may print its contents to the screen, after which you return to the menu.

Regardless of the file type, you may choose to say, "Never mind," which sends you back to the menu.

In order to personalize your copy of the program, do this:

Load the program
Type this:
30 NA$ = "your name"
Hit the RETURN key
Save the program

SABOTAGE

by Peter Adams

Apple™ translation by Peter J. Brown

***Sabotage* is an arcade-style game for an Apple™ with Applesoft and 16K RAM.**

You are alone, deep within the computer control center of *The Enemy*. Armed only with your laser pistol, you must wend your way past electrified walls and control boxes, and elude the searing energy of fearsome laser cannon. Your mission: Sabotage. If you get far enough into the control center, you will reach your objective: The Master Computer. Good luck.

You will begin in the lower left corner of each room. You are the orange spot. The red spots you see scattered throughout each room are control boxes. The blue lines are walls, and the green machine at the middle of the bottom of the screen is the laser cannon. The light blue box around the laser cannon is its force shield. All of these objects will be electrified, and contact between them and you will prove deadly.

You can move in any of four directions by pressing the appropriate key. Q will move you up; Z will move you down; the arrow keys will move you left and right. You won't need the REPT key, because your man keeps moving until you press another key. You may fire in the direction you are moving (or last moved, if stationary) by pressing the space bar. Only one of your missiles can be on the screen at one time. At any point in the game, the P key will pause the game; pressing any other key will restart it.

You cannot leave the room until you have destroyed six of the control boxes. Once you accomplish this, a door will materialize in the lower right corner. Each control box will earn you one point. The control boxes will regenerate in a short while, unless you disable the laser cannon. To do this, you must shoot the laser when its shields are down. It lowers its shields just prior to firing at you. You will earn five extra points for disabling the laser. If you destroy all the control boxes before you leave a room, you will receive yet another five-point bonus.

You can save the top ten scores on tape or disk. Simply press the appropriate key when the program prompts you. At the beginning of the game, you may load the previous high scores if you wish. WARNING: Although the program will check a disk for the presence of a high-score file, it is impossible to check a tape in this way. If you try to load high scores from a blank tape, the program will "hang" indefinitely, and you will have to press RESET and run the program again.

Variables

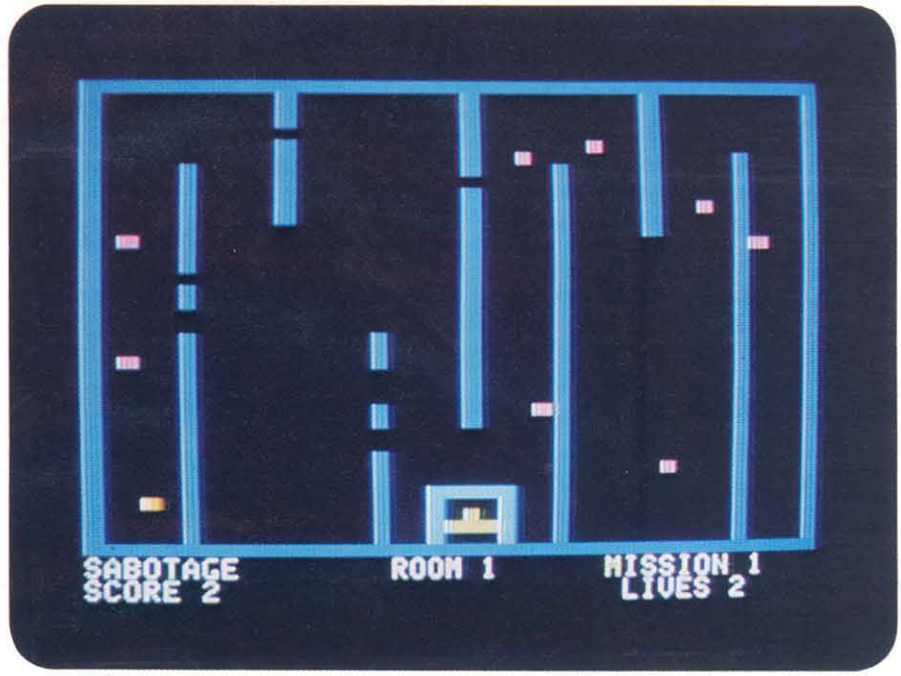
A, A$, T, U, V: Work variables.
A(10,4): High scores.
B: Room player is in.
BL: Player ranking in top 10.
CB: # of remaining control boxes.
CX, CY: Laser shot position.
D: # of lives left.
D$: CHR$(4).
DX, CY, H, I: X, Y change.
G: Difficulty.
HI: Lowest high score.
I$: Player's initials.
J, K: New player position.
KE: Keyboard input.
KK: Loop variable.
L: Is laser active?
M: Is the player firing?
M$: Messages.
MD: Mission length.
MX, MY: Missile position.
NL: Score needed for new life.
SA: Need to save scores?
SC: Player score.
T1: Time count for laser.
T2: Time spent in room.
T3: Time count for difficulty.
T4: Time laser is disabled.
TX, TY: Laser target position.
V1: Has laser been disabled?
V5: Time count for self destruct.
V8: Maximum difficulty.
V9: Mission number.
W0-W9, X0-X9, Y0-Y9, Z0-Z9,
PA: Constants.
X, Y: Player position.
X(10), Y(10): Control box positions.
XC, YC: Laser movement increments.
XM, YM: Missile movement increments.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                             SS
SS      APPLESOFT  BASIC       SS
SS         'SABOTAGE'          SS
SS    AUTHOR:  PETER ADAMS     SS
SS   TRANSL: PETER J. BROWN    SS
SS     COPYRIGHT (C) 1982      SS
SS SOFTSIDE PUBLICATIONS, INC SS
SS                             SS
SS SS SS SS SS SS SS SS SS SS SS
```

```
100  GOTO 590
```

Look at keyboard for movement commands.

```
110 KE =  PEEK (Z3): IF PA = KE THEN
    110
120  IF SC > NL THEN NL = NL + 10
```




```
    0:D = D + 1: POKE X1,Z0: POKE
    X2,X9: CALL X4: FOR V = Z0 TO
    Z8: NEXT : CALL X4: HTAB 36:
     PRINT D;
130  IF CB < 5 THEN  COLOR= Z0: VLIN
    W6,Y3 AT 39
140  IF KE = Z4 THEN H = Z0:I = Z
    2:DX = H:DY = I: GOTO 190
150  IF KE = Z5 THEN H = Z1:I = Z
    0:DX = H:DY = I: GOTO 190
160  IF KE = Z6 THEN H = Z0:I = Z
    1:DX = H:DY = I: GOTO 190
170  IF KE = Z7 THEN H = Z2:I = Z
    0:DX = H:DY = I: GOTO 190
180 H = Z0:I = Z0: GOTO 220
```

Move the player.

```
190 J = X + H:K = Y + I:T =  SCRN(
    J,K): COLOR= Z0: PLOT X,Y: COLOR=
    Z9: PLOT J,K:X = J:Y = K
200  IF T > Z0 THEN 1030
210  IF X = 39 THEN 540
```

Move the player's missile.

```
220  IF KE = Z8 AND M = Z0 THEN M
     = Z1:MX = X:MY = Y:XM = DX:
    YM = DY: POKE Y0,Z0
230  IF M = Z0 THEN 290
240  FOR T = Z0 TO Z1: POKE Y2,Z8
    : CALL Y1
250  COLOR= Z0: PLOT MX,MY:MX = M
    X + XM:MY = MY + YM
260  IF MX < Z1 OR MX > Y3 OR MY <
    Z1 OR MY > Y3 THEN M = Z0: GOTO
    110
270  COLOR= Z9: PLOT X,Y:J =  SCRN(
    MX,MY): IF J > Z0 THEN 470
280  COLOR= Y5: PLOT MX,MY: NEXT
```

Increment timer. If the maximum time is exceeded, then print an appropriate message.

```
290  IF T1 > Z0 THEN T1 = T1 + Z1
    : IF T1 > G + G THEN  GOSUB
    970:T1 = Z0
300  IF V5 > Z0 THEN V5 = V5 + Z2
    : VTAB 23: HTAB 28: PRINT V5
    ;" ";: VTAB 22: IF V5 = Z0 THEN
     HOME : INVERSE : PRINT "YOU
     HAVE FAILED TO ESCAPE...": NORMAL :
     FOR T = 1 TO 2000: NEXT : GOTO
     1070
310  IF T2 / Y6 +  RND (Z1) < Y7 OR
    L = Z0 OR T1 > Z0 THEN 400
320 T1 = Z1: COLOR= Z0: VLIN Y8,Y
    3 AT Y9: HLIN Y9,X0 AT Y8: VLIN
    Y8,Y3 AT X0: POKE X1,Z0: POKE
    X2,128: CALL X4
330 XC = Z0: IF X = X5 THEN YC =
    Z2: GOTO 350
340 T =  ATN ((Y - Y4) / (X - X5)
    ):XC =  COS (T):YC =  SIN (T
    ) + X6
350  IF X < X5 THEN XC =  - XC:YC
     = X6 - YC
360  IF XC = Z0 OR YC = Z0 THEN 3
    90
370  IF  ABS (XC) > ABS (YC) THEN
    YC = YC * Z1 /  ABS (XC):XC =
     SGN (XC): GOTO 390
380 XC = XC * Z1 /  ABS (YC):YC =
    SGN (YC)
```

```
390 TX = X5:TY = Y4:CX = TX + XC:
    CY = TY + YC
400  IF  RND (Z1) < X7 AND L = Z1
     THEN T =  INT ( RND (Z1) *
    X8) + Z1: COLOR= Z1: IF  SCRN(
    X(T),Y(T)) = Z0 THEN CB = CB
     + Z1: PLOT X(T),Y(T): POKE
    X1,X9: POKE X2,W0: CALL X4: IF
    CB > 4 THEN  COLOR= X3: VLIN
    W6,Y3 AT 39
410  IF L = Z1 THEN 450
420 T4 = T4 + Z1: IF T4 < G * W1 THEN
    450
430 L = Z1:T4 = Z0: COLOR= W2: HLIN
    W3,W4 AT W5: PLOT X5,W6:G =
    G + Z2: IF G < V8 THEN G = V
    8
440  COLOR= W9: VLIN Y8,Y3 AT Y9:
     HLIN Y9,X0 AT Y8: VLIN Y8,Y
    3 AT X0: POKE X1,Y3: POKE X2
    ,Z0: CALL X4
450 T2 = T2 + Z1:T3 = T3 + Z1: IF
    T3 > W7 THEN T3 = Z0:G = G +
    Z2: IF G < V8 THEN G = V8
460  GOTO 110
```

Handle collisions of the player's missile with objects.

```
470 M = Z0:T =  SCRN( MX,MY): IF
    T = Z1 THEN SC = SC + Z1:CB =
    CB + Z2: COLOR= Z0: PLOT MX,
    MY: HTAB 7: PRINT SC;: POKE
    X1,Z0: POKE X2,60: CALL X4: GOTO
    110
480  IF T = X3 THEN  COLOR= Z0: PLOT
    MX,MY: GOTO 110
490  IF T = W2 AND V1 = Z0 THEN L
     = Z0:V1 = Z1: COLOR= Z0:T1 =
    Z0: HLIN W3,W4 AT W5: PLOT X
    5,W6:SC = SC + 5: HTAB 7: PRINT
    SC;: GOTO 110
500  IF T < 15 THEN 110
510  FOR T = Z1 TO 5: COLOR= 15: FOR
    I = Z1 TO 3: HLIN Y4,Y3 AT I
    : NEXT : CALL Y1: COLOR= Z0:
     FOR I = Z1 TO 3: HLIN Y4,Y3
     AT I: NEXT : CALL Y1: NEXT

520  VTAB 23: HTAB 10: INVERSE : PRINT
    "MISSION ACCOMPLISED";: FOR
    T = Z1 TO Y6: NEXT : HTAB 10
    : PRINT "NOW YOU MUST ESCAPE
    !";
530  NORMAL : FOR T = Z1 TO Y6: NEXT
    :V5 = 260 - 10 * V9: HTAB 1:
     PRINT "TIME UNTIL SELF DIST
    RUCT = ";V5;: VTAB 22: POKE
    Y0,Z0:SC = SC + 30: HTAB 7: PRINT
    SC;: GOTO 110
```

Start new rooms and new missions.

```
540  POKE Y0,Z0: IF B < MD + X3 THEN
    570
550 MD = MD + Z1:V9 = V9 + Z1:V8 =
    V8 - 0.3:V5 = Z0:B = 0:Y7 =
    Y7 - X6: TEXT
560  HOME :SC = SC + 10: VTAB 3: PRINT
    "YOU HAVE ESCAPED...BUT THER
    E IS ANOTHER": PRINT : PRINT
    "COMPUTER TO DESTROY... GET
    READY.": FOR T = 1 TO 3000: NEXT
    : GOTO 580
570  IF CB = Z0 THEN SC = SC + 5:
     HTAB 7: PRINT SC;
580 B = B + Z1:T1 = Z0:T2 = Z0:X =
    3:Y = 35:T3 = Z0:T4 = Z0:L =
    Z1:V1 = Z0: GOTO 870
```

Program initialization.

```
590 PA = 208:X = 0:Y = 0:KE = 0:H
     = 0:I = 0:J = 0:K = 0:T = 0
    :Z0 = 0:Z1 = 1:Z2 =  - 1:Z3 =
     - 16384:Z4 = 209:Z5 = 149:Z
    6 = 218:Z7 = 136:Z8 = 160:Z9
     = 9
600 Y0 =  - 16368:Y1 = 768:Y2 = 7
    69:Y3 = 38:Y4 = 35:Y5 = 13:Y
    6 = 2000:Y7 = 0.97:Y8 = 34:Y
    9 = 18:X0 = 22:X1 = 783:X2 =
    785:X3 = 2:X4 = 782:X5 = 20:
    X6 = 0.05:X7 = 0.02:X8 = 10:
    X9 = 50
610 W0 = 250:W1 = 30:W2 = 12:W3 =
    19:W4 = 21:W5 = 37:W6 = 36:W
    7 = 300:W8 = 150:W9 = 6:MD =
    4:NL = 99:SP = 10000
620  DIM X(10),Y(10),A(10,4):D$ =
     CHR$ (4):M$ = "PRESS RETURN
     TO BEGIN "
630  FOR T = 768 TO 795: READ I: POKE
    T,I: NEXT
640  TEXT : HOME : VTAB 3: HTAB 1
    6: PRINT "SABOTAGE": PRINT :
     HTAB 13: PRINT "BY PETER AD
    AMS": PRINT : HTAB 4: PRINT
    "APPLE TRANSLATION BY PETER
    BROWN"
```

Check memory for the presence of high scores. Ask player whether to get high scores from disk or tape. Get and display high scores.

```
650  IF  PEEK (SP) = 254 THEN 690
660  FOR T = 8192 TO 8233: POKE T
    ,Z0: NEXT : PRINT : PRINT : PRINT
    "LOAD HIGH SCORES FROM:": PRINT
    : PRINT "1) TAPE  2) DISK  3
    ) NEITHER";
670 A =  PEEK (Z3):A = A - 176: IF
    A < 0 THEN 670
680  POKE Y0,Z0: IF A < 1 OR A >
    3 THEN 640
690 SA = 0: IF  PEEK (SP) = 254 THEN
     FOR T = 1 TO 10: FOR U = 1 TO
    3:A(T,U) =  PEEK (SP - 5 + T
     * 5 + U): NEXT :A(T,4) = 25
    6 *  PEEK (SP - 1 + T * 5) +
     PEEK (SP + T * 5): NEXT : GOTO
    800
700  PRINT
710  ON A GOTO 720,760,860
720  HOME : VTAB 3: INVERSE : PRINT
    "                WARNING
                    ";: NORMAL
730  PRINT "YOU MUST ALREADY HAVE
     SAVED HIGH SCORES ON YOUR T
    APE. HAVE YOU DONE SO? (Y/N)
    ";: GET A$: PRINT A$: IF A$ <
    > "Y" THEN 640
740  PRINT : PRINT : PRINT "START
     TAPE THEN PRESS ANY KEY";: GET
    A$: RECALL A: GOTO 790
750  ONERR  GOTO 1220: PRINT D$;"
    VERIFY HISAB": POKE 216,0
760  ONERR  GOTO 1220
770  PRINT D$;"VERIFY HISAB": POKE
    216,0
780  PRINT : PRINT D$;"OPEN HISAB
    ": PRINT D$;"READ HISAB": FOR
    T = 1 TO 10: FOR U = 1 TO 4:
     INPUT A(T,U): NEXT : NEXT :
     PRINT D$;"CLOSE HISAB"
790  POKE SP,254: FOR U = 1 TO 10
    : FOR V = 1 TO 3: POKE SP -
    5 + U * 5 + V,A(U,V): NEXT :
     POKE SP - 1 + U * 5, INT (A
    (U,4) / 256): POKE SP + U *
    5,A(U,4) - 256 *  INT (A(U,4
    ) / 256): NEXT
800  HOME : VTAB 3: HTAB 10: PRINT
    "SABOTAGE HALL OF FAME": VTAB
    5
810  FOR T = 1 TO 10: HTAB 15: FOR
    U = 1 TO 3: PRINT  CHR$ (A(T
    ,U));: NEXT : HTAB 25: PRINT
    A(T,4): NEXT :HI = A(10,4): FOR
```

continued on page 95

VISTA V1200

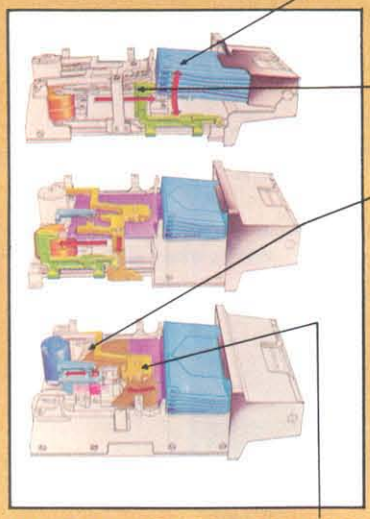
NEW 6MB HARD DISK ALTERNATIVE!

Mass Storage for your Apple II™ has always been a problem. On one hand, there were the exotic, expensive hard disks with no cost efficient means of backup. On the other hand, the Apple floppy drive lacked the speed and storage demanded by today's professionals.

Vista's V1200 offers both at an incredibly attractive price. The removable VistaPak cartridges offer 6 Megabytes of removable storage each and can be backed up like a floppy.

Now hard disk storage and speed can be yours with the added capability of interchangeable media. The V1200 eliminates the worries of head crashes, drive alignments, lost data, or backup with a new application of field-proven floppy technology.

VISTAPAK CARTRIDGE holds five 5¼" diskettes at 1.2 Mega-Bytes each. Selection of the diskette is made by the pak articulator to present the proper diskette to the picker arm.



DISKETTE PICKER grasps the proper diskette and loads it into the drive accurately. The drive senses improper loading, and reclamps the diskette automatically.

MICROSTEP SERVO CONTROL the drive automatically calibrates itself to each diskette individually, assuring interchangeability of media between drives, and making drive alignment a task of the past. Our special servo can move in increments as small as 1/100th of a track, and even track diskettes recorded off-center.

MICROPROCESSOR CONTROLLED SPINDLE speed ensures accurate recording and retrieval of your valuable data.

The Vista V1200 also incorporates features such as; automatic recalibration of the drive if left unused for 2½ minutes, automatic diskette storage should the drive door be opened, and power conservation logic to insure no more than one motor is operating simultaneously to keep temperature down, and reliability high.

The VistaPak cartridges hold 6MB of formatted data each. The removable cartridge allows you to keep duplicates of your valuable data as well as to keep separate paks for your accounting, wordprocessing, spreadsheet and other applications. No other storage device offers more in flexibility and capability.

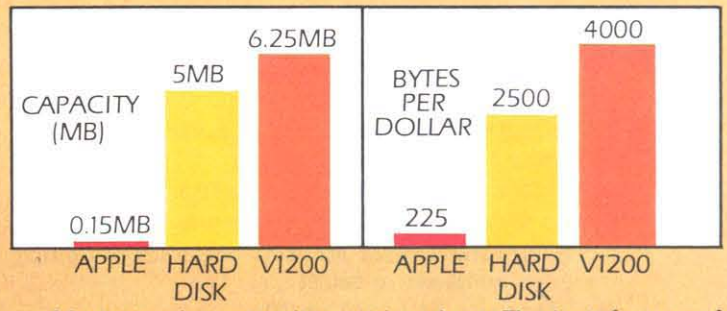
THE INNOVATIVE VISTAPAK CARTRIDGE CONCEPT

6MB PER CARTRIDGE

The Vista V1200 system uses an advanced DMA controller to insure data integrity, and to keep data moving at the maximum possible rate.

Included with every system is software for Apple DOS 3.3™, Apple PASCAL™, and MICROSOFT CP/M™. Also included is Quickcharge, our proprietary, high-speed enhancement to DOS 3.3, enabling a 5x speed increase.

OTHER STORAGE DEVICES VS. VISTA V1200



Maybe you're not quite convinced yet. The best feature of the V1200 is that it is priced significantly below the competition: $1549.00 (mfgr. sugg. list), and is available from your Local Dealer. See one today.

Contact Your Local Vista Dealer Today!

WESTERN REGIONAL DISTRIBUTOR:
Group 3 Electronics
(213) 973-7844 or (408) 732-1307
California only (800) 262-1556

Vista COMPUTER COMPANY, INC.
1317 E. Edinger / Santa Ana, CA 92705
(714) 953-0523 (800) 854-8017

Prices subject to change without notice.
™Apple Computer Co., Inc.

A Fast, Easy, Inexpensive Approach to Word Processing

NEW

PIE Writer:
Word Processing System

Hayden is thinking about your future. We know you're concerned with saving time and money . . . with increasing efficiency and productivity . . . and that is the reason for PIE Writer!

Now you can turn your Apple II into a complete word processing system. PIE Writer, a powerful editor and formatter, offers the versatility and machine efficiency of the larger systems—at a fraction of the cost!

Now all your documents and correspondence—reports, memos, direct mail pieces, even "personalized" form letters—can be composed, edited, formatted, and printed quickly and accurately. Simple keystroke commands let you alter characters, words, sentences, even entire paragraphs!

Learning to use PIE Writer is as easy as learning to type. There are many user-oriented features, designed for first-time users as well as serious programmers: an easy-to-understand, indexed instruction manual, a carefully designed reference card, an interactive tutorial with each diskette, a "help" screen listing key command descriptions for reference as text is edited, and an easel binder for at-a-glance aid.

If you're looking for high-quality at a low cost, PIE Writer is your best investment. Order yours today! 12009, Standard; 13409, Smarterm, Sup'R'Term, Videx

Available at your local computer store

Take a look at some of PIE Writer's powerful new features:

- Incremental Spacing–adds fractions of space between words for text justification.
- Fast page scrolling.
- Call command–allows access to machine language subroutines.
- Saves formatted output on disk file.
- Works with 40 or 80 column display boards.
- Word Tabbing–cursor stops at beginning of every word, in addition to set stops.
- Tab stops can be saved on file.
- Word Delete–deletes entire words at a time.
- Lower case capabilities.
- Prints **BOLD**.
- Centers.

All this and more . . . for only $149.95!

Upgrade your PIE 2.0: Send $75.00, check or money order, and original disk (at sender's risk) for upgraded version.

or order direct:

1-800-343-1218

in Massachusetts 617-937-0200

Hayden

Mail To: **Dept S, Box 600 • Hayden Software Co.**
600 Suffolk Street • Lowell, MA 01853

Please accept my order and send me the PIE Writer System for the following display board.

Please Check: ☐ 40 column (#12009) ☐ 80 column (#13409)

☐ My money order for $149.95 is enclosed; or
☐ Please charge my ☐ Visa ☐ MasterCard

Name _____

Address _____ Apt. _____

City _____

State _____ Zip _____

MasterCard/Visa# _____ Expires: _____

Signature _____

Residents of CA, MA, and NJ must add sales tax

ED-001 A501

```
    U = 1 TO 150: NEXT
820  IF BL = 0 THEN 840
830  VTAB BL + 4: FOR V = 1 TO 5:
     HTAB 1: CALL  - 868: FOR U =
     1 TO 400: NEXT : HTAB 15: FOR
     U = 1 TO 3: PRINT  CHR$ (A(B
     L,U));: NEXT : HTAB 25: PRINT
     A(BL,4);: FOR U = 1 TO 400: NEXT
     : NEXT :BL = 0: HTAB 1
840  VTAB 22: PRINT M$;: GET A$: IF
     A$ = "N" THEN 1150
```

Set up a room.

```
850  HOME
860 X = 3:Y = 35:L = 1:D = 3:V5 =
     0:V8 = 2:V9 = 1:B = 1:M = 0:
     T1 = 0:T2 = 0:T3 = 0:T4 = 0:
     SC = 0
870  GR : VTAB 21: HTAB 1: PRINT
     "SABOTAGE","ROOM ";B;"
     MISSION ";V9
880 G = 7 - B: PRINT "SCORE"; TAB(
     30);"LIVES";: HTAB 7: PRINT
     SC;: HTAB 36: PRINT D;:T2 =
     0:V1 = 0: IF B = MD + Z1 THEN
      COLOR= 15: FOR T = 35 TO 38
     : VLIN 1,3 AT T: NEXT
890  COLOR= 9: PLOT X,Y: COLOR= 2
     : HLIN 0,39 AT 0: VLIN 0,39 AT
     39: HLIN 0,39 AT 39: VLIN 0,
     39 AT 0
900  IF B = MD THEN  FOR T = 19 TO
     21: VLIN 0,33 AT T: NEXT
910  FOR T = 0 TO 30 STEP 10: VLIN
     0,20 * RND (1) + 12 AT T: VLIN
     25 - 20 * RND (1),39 AT T +
     5: NEXT : COLOR= 6: VLIN 34,
     38 AT 18: HLIN 18,22 AT 34: VLIN
     34,38 AT 22
920  COLOR= 1: FOR T = 1 TO 10
930 H =  INT ( RND (1) * 35) + 2:
     I =  INT ( RND (1) * 30) + 5
     : IF  SCRN( H,I) > 0 THEN 93
     0
940 X(T) = H:Y(T) = I: PLOT H,I: NEXT
     :CB = 10
950  COLOR= 12: HLIN 19,21 AT 37:
      PLOT 20,36
960  GOTO 110
```

Fire the laser.

```
970 T1 = Z0
980  POKE Y2,W8: CALL Y1: IF CX <
     Z1 OR CX > Y3 OR CY < Z1 OR
     CY > Y3 THEN  COLOR= Z0: PLOT
     TX,TY: COLOR= W9: VLIN Y8,Y3
      AT Y9: HLIN Y9,X0 AT Y8: VLIN
     Y8,Y3 AT X0: RETURN
990 T =  SCRN( CX,CY): IF T = 15 THEN
      COLOR= Z0: PLOT TX,TY: COLOR=
     W9: VLIN Y8,Y3 AT Y9: HLIN Y
     9,X0 AT Y8: VLIN Y8,Y3 AT X0
     : RETURN
1000  IF T = Z0 OR T = X3 OR T =
     Y5 THEN  COLOR= Z0: PLOT TX,
     TY: COLOR= Y5: PLOT CX,CY:TX
      = CX:TY = CY:CX = CX + XC:C
     Y = CY + YC: GOTO 980
1010  IF  SCRN( CX,CY) = Z1 THEN
      COLOR= Z0: PLOT CX,CY:CB =
     CB + Z2:SC = SC + Z1:CX = Z0
     : HTAB 7: PRINT SC;: POKE X1
     ,Z0: POKE X2,60: CALL X4: GOTO
     980
1020  IF  SCRN( CX,CY) = Z9 THEN
      POP : COLOR= Z0: PLOT TX,TY
     : COLOR= W9: VLIN Y8,Y3 AT Y
     9: HLIN Y9,X0 AT Y8: VLIN Y8
     ,Y3 AT X0
```

Handle the player's demise.

```
1030  POKE Y0,Z0:D = D + Z2: HTAB
     W6: PRINT D;: IF T = Z1 THEN
     CB = CB + Z2
1040  FOR T = Z0 TO 15: COLOR= T:
      PLOT X,Y: POKE Y2,15 * T: CALL
     Y1: NEXT : COLOR= Z0: PLOT X
     ,Y
1050  COLOR= X3: HLIN Z0,Y3 AT Z0
     : VLIN Z0,Y4 AT 39: HLIN Z0,
     39 AT 39: VLIN Z0,Y3 AT Z0
1060 X = 3:Y = 35: IF D > Z0 THEN
      COLOR= Z9: PLOT X,Y: GOTO 1
     10
```

Save new high scores.

```
1070  FOR T = 1 TO 1000: NEXT : TEXT
     : HOME : IF SC < HI THEN 114
     0
1080 BL = 1:SA = 1: VTAB 3: PRINT
     "YOU HAVE A NEW HIGH SCORE!"
     : VTAB 5: PRINT "PLEASE TYPE
      YOUR INITIALS": INPUT I$: IF
      LEN (I$) > 3 THEN 1080
1090  FOR BL = 1 TO 10: IF SC < A
     (BL,4) THEN  NEXT
1100  IF BL = 10 THEN 1120
1110  FOR U = 9 TO BL STEP Z2: FOR
     V = 1 TO 4:A(U + 1,V) = A(U,
     V): NEXT : NEXT
1120  IF  LEN (I$) < 3 THEN I$ =
     I$ + "-": GOTO 1120
1130  FOR U = 1 TO 3:A(BL,U) =  ASC
     ( MID$ (I$,U,1)): NEXT :A(BL
     ,4) = SC
1140 M$ = "PLAY AGAIN?": POKE Y0,
     Z0: GOTO 790
1150  IF SA = 0 THEN 1200
1160  HOME : VTAB 3: PRINT "SAVE
     HIGH SCORES TO:": VTAB 5: PRINT
     "1) TAPE  2) DISK  3) NEITHE
     R";: GET A$:A =  VAL (A$): IF
     A < 1 OR A > 3 THEN 1160
1170  ON A GOTO 1180,1190,1200
1180  HOME : VTAB 3: PRINT "START
      TAPE THEN PRESS ANY KEY";: GET
     A$: STORE A: GOTO 1200
1190  HOME : PRINT : PRINT D$"OPE
     N HISAB": PRINT D$"DELETE HI
     SAB": PRINT D$"OPEN HISAB": PRINT
     D$"WRITE HISAB": FOR T = 1 TO
     10: FOR U = 1 TO 4: PRINT A(
     T,U): NEXT : NEXT : PRINT D$
     "CLOSE HISAB"
1200  END
1210  DATA 162,0,173,48,192,232,2
     08,253,238,1,3,208,243,96,16
     2,0,160,128,136,208,253,173,
     48,192,202,208,245,96
```

Error-trap routine to keep player from trying to get high scores from non-existent disk file.

```
1220  POKE 216,0: INVERSE : PRINT
     : PRINT "WARNING:";: NORMAL
     : PRINT " YOU HAVE NO HIGH-S
     CORE FILE": FOR KK = 1 TO 12
     00: NEXT KK: GOTO 640
```

APPLE™ TABLE FOR: SABOTAGE

LINES	SWAT CODE	LENGTH
100 - 210	SF	367
220 - 330	OO	478
340 - 450	MI	514
460 - 550	RC	533
560 - 610	SU	540
620 - 720	PJ	521
730 - 810	DK	545
820 - 910	JR	525
920 - 1020	LY	513
1030 - 1140	ZX	488
1150 - 1220	MS	500

APPLE DISKOURSE

Part Three Of A Series

by Cary W. Bradley

In our last Diskourse, we looked at one of the many things that can be done by using the DOS RWTS subroutine to read specific disk sectors. The program we presented in that installment, *Disk Snooper*, makes extensive use of this idea; I hope you've found it a useful tool for seeing what DOS actually does while it's working. Although it's not absolutely essential, the *Disk Snooper* can be helpful in verifying what is being done in this month's column, as well as subsequent ones. Chances are, you'll get a much clearer picture of what's happening on your disks if you periodically run *Disk Snooper* to examine the sectors you've written to by means of DOS commands or RWTS calls.

WARNING!

This month's topic is not for the faint of heart; I'll warn you before we start that what we'll be attempting is risky. DO NOT, I repeat, DO NOT run the program presented here on a good disk before you have used *SWAT, Disk Snooper*, and any other utilities at your disposal to make sure that the entire program runs absolutely perfectly. I recommend that you use expendable copies of disks for testing purposes, as I did in developing the program. Neither ***SoftSide*** nor I will assume any responsibility for any disks you render useless through your attempts to use this technique.

The reason this warning is necessary is that we are going to be using the RWTS subroutine to write information on a disk without using the normal DOS commands. There is a lot of housekeeping information that DOS automatically takes care of when we use its commands. When we bypass the commands, we also bypass the safety provided by this automatic housekeeping. In doing so, it is not very hard to alter the data on our disk in such a way that DOS can't recognize it. But as long as what we write on the disk conforms EXACTLY to what DOS needs to see, we can do some neat things. If you're meticulously careful, you will find this issue's utility a handy one to have, and you'll learn something in the process.

Our topic this month is the DOS Catalog, which is maintained on track 17 (hex $11) of every normally formatted DOS disk. You can find a detailed summary of the format of catalog sectors in your DOS manual. Briefly, each of the sectors 1-15 of the catalog track is allocated for storage of catalog information on 7 disk files. That makes the total capacity of the catalog 105 files, but unless your files are very small, you will use up the space on the disk long before your catalog ever fills.

DOS selects the catalog sectors in descending order when files are created. That is, the first file in your catalog (When you INIT a disk, this will be your "Hello" program.) occupies the first position in sector 15. As new files are created, they are cataloged in successive positions, sequentially, until all 7 positions in sector 15 are filled. DOS then moves on to sector 14 for files 8 through 14, and so on, working its way toward sector 1. Sector 0 has a special purpose, which we'll investigate in our next article.

Each file in the catalog is given 35 bytes of the sector, in which are stored the file's name, type, sector count (mod 256) and the track and sector number where the file's track/sector list can be found. Details are in Appendix C of the DOS manual, so I won't repeat them here. Refer to the manual and examine some catalog sectors with the *Disk Snooper* to verify all of this for yourself.

An interesting thing happens when you DELETE a file. Only two bytes of the catalog entry are changed; the byte containing the track number of the t/s list is changed to 255 ($FF), and the number that was in that position is transferred to the end of the file name. Thus, when a file is DELETED, its name remains in the catalog until there is some reason for DOS to write over it. Normally, this is caused by the creation of a new file.

When a file is created through one of the DOS commands, DOS searches the catalog entries sequentially until it finds a position occupied by a DELETEd file, or the end of the list of active files, whichever comes first. Information for the new file is written there. You have probably noticed that DOS often puts new files at the end of the catalog, but that sometimes a new file shows up somewhere within the existing list of files. When this happens, DOS has found a DELETEd file in the catalog and has placed the information for your new file in that position.

This method is fine for DOS, because it can quickly find any file in the catalog by searching through the file names. But for you, it's not so easy. If you CATALOG a disk to see whether a particular file is there, you have to visually inspect the entire list, because the file you are looking for could be anywhere. This can be a hassle, especially when the disk contains a large number of files.

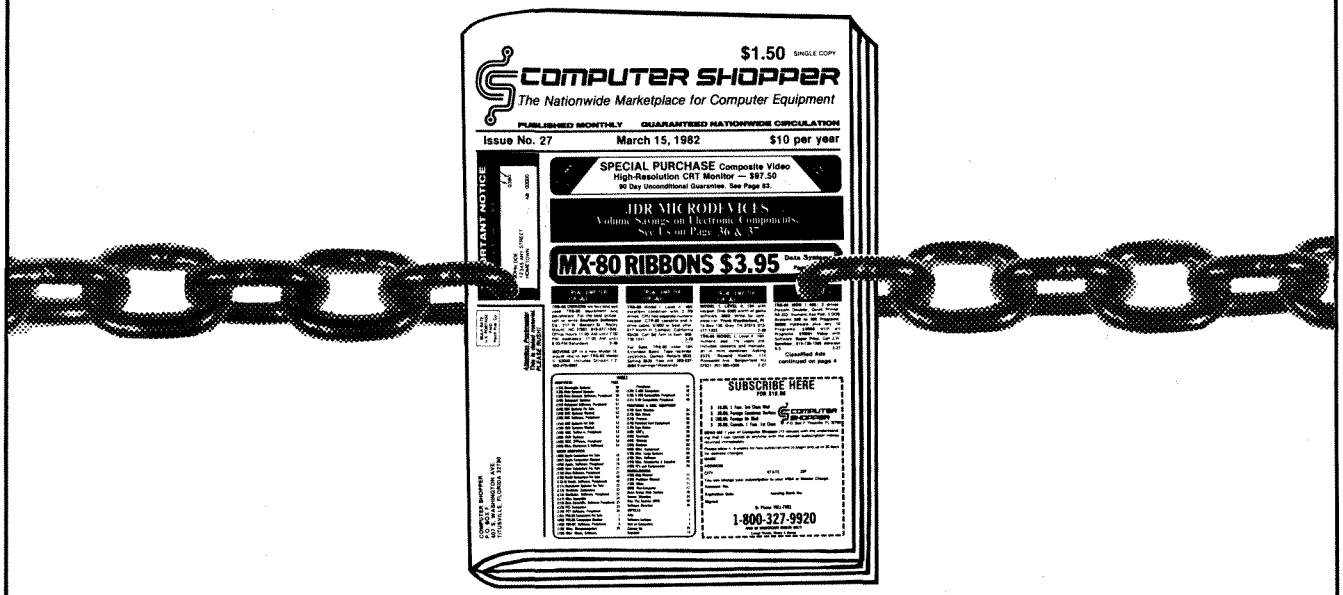
Wouldn't it be a lot easier if you could rearrange the catalog in alphabetical order? You can, if you know how to use RWTS (and if you're very careful how you go about it). This issue's utility does just that, and I affectionately call the program *ALPHIE*. Even if you never have an overwhelming desire to alphabetize a disk catalog, you should still get to know *ALPHIE*. He's short, sweet and instructive.

Here's how *ALPHIE* works: First, the entire 15 catalog sectors are read into 15 "pages" of the Apple's™ memory. These pages are then searched to determine how

continued on page 98

YOU'VE JUST FOUND

THE MISSING LINK!



Computer Shopper is *your link* to individuals who buy, sell and trade computer equipment and software among themselves nationwide. No other magazine fills this void in the marketplace chain.

Thousands of cost-conscious computer enthusiasts save by shopping in Computer Shopper every month through hundreds of classified ads. And new equipment advertisers offer some of the lowest prices in the nation.

Computer Shopper's unbiased articles make for some unique reading among magazines and there's a "help" column to answer difficult problems you may have with interfacing, etc.

For a *limited time* you can subscribe to Computer Shopper with a 6 month trial for $6 or 12 months for $10.

You risk nothing, because if you're not 100% satisfied after receiving the first issue, you may cancel and receive a full refund. No questions asked!

If you want faster service, call in your subscription and charge on your MasterCard or VISA account.

Our business office is open from 8 a.m. to 5 p.m. Monday through Friday. Call for charge card orders or send your check or money order TODAY while the offer is still good.

6 month trial, $6 or 12 months for $10

COMPUTER SHOPPER

P.O. Box F-704 • Titusville, FL 32780
305-269-3211

many active files are in the catalog, and what position each of them currently occupies. Then the file names are sorted alphabetically.

Finally, a new catalog is constructed, sector by sector, by moving all of the 35 bytes for each file's catalog entry into their proper position. These sectors are written over the old, unordered catalog on track 17 of the disk. As a side effect (advantage?), any DELETEd files in the catalog are "cleaned out;" they are not put into the new catalog. Because of this, any new files you add to an ALPHIEd disk will appear at the end of the catalog, unless you have DELETEd something in the meantime.

One of the applications in which *ALPHIE* is handy is program development. When I write a program, I go through many versions, all of which are SAVEd on the disk under a common file name with a version number suffix. When the disk fills up, I DELETE earlier versions, and, as a result, the latest version frequently gets entered somewhere in the middle of the catalog. Successive versions end up scattered throughout the catalog, and are often intermixed with unrelated files on the disk. *ALPHIE* puts all versions of the same work next to each other in the catalog.

If you use *ALPHIE* for this purpose, you should take note of the fact that he considers file names character by character when alphabetizing. He does not recognize version numbers as being different from any other characters in file names. The file PROGRAM.11, for example, would come before PROGRAM.2 in alphabetical order. If this presents a problem for you, use the name PROGRAM.02 instead of PROGRAM.2.

ALPHIE uses a simple exchange sort algorithm to do his job. Instead of actually exchanging the positions of the catalog entries in memory, however, an array of pointers is maintained. The array contains the starting memory address of each of the catalog entries. Only the pointers are exchanged during sorting.

The amount of time *ALPHIE* takes to sort your catalog depends on the number of files in the catalog, and how badly out of order they are. Don't be surprised if it takes several minutes to alphabetize a catalog. *ALPHIE* is slow, mainly because he is performing operations on memory addresses, rather than on variables — normally the way BASIC likes to work. All of those PEEKs take time. The sorting routine is really a BASIC emulation of a Machine Language procedure. If you're learning assembly language, it would be a good exercise to rewrite the sorting portion of this program using an assembler.

As an indication of how long sorting can take, I used a disk containing 52 files, in no particular order in the catalog. It took about two minutes to alphabetize them. By contrast, when I reran *ALPHIE* on the same disk, now with its catalog completely alphabetized, it took just over ten seconds to make two passes through the catalog; once to count the files and once to determine that no switching needed to be done.

In *ALPHIE*, I have used a technique included in all of our Diskourse programs thus far. The value of HIMEM (stored in memory locations 115-116) is saved at the very beginning of the program and restored before execution ends. This means that your Apple's usable memory size will be the same when you finish running *ALPHIE* as it was before you began.

This time, however, I have departed from my regular practice of writing the Diskourse utilities so that they'll run on the smallest possible memory configuration. Since I've already done it for you twice, I figured it's time you tried your hand at it if you have to make *ALPHIE* run on an Apple with less than 48K RAM. But, to show you what a nice guy I really am, I've tried to make it as easy as possible.

The memory organization used by *ALPHIE* is as follows: Sixteen memory pages (256 bytes each) are needed for RWTS input and output. The top 15 of these are used to store the 15 catalog sectors, and the one immediately below them is used to construct each new catalog sector before it is written to the disk. You'll need to know a little hexadecimal to make the conversion. Each of the sixteen pages starts at a memory address whose hex representation ends in 00. It is important that you determine your corresponding pages the same way.

As an example, I'll go through the procedure I used for the 48K version of the program. I know that booting DOS sets HIMEM at $9600 (DOS manual Appendix D). So, I stored the catalog in the 15 memory pages immediately preceding that; i.e., the pages which start at addresses $8700, $8800, ... , $9500. Since RWTS needs two bytes to store the address of the buffer, this is a convenience. One of these bytes is always a 0, so to specify the buffer address before reading or writing, I only need one POKE. All memory address calculations in the program are based on the first two hex digits of the lowest numbered page used for catalog storage. In this case, that is $87, and its decimal equivalent, which BASIC needs, is 135. In line 240 of *ALPHIE*, you must initialize the variable CB% to whatever that number turns out to be for your situation. *ALPHIE* automatically calculates which page is used as the output buffer; in my 48K program it is the page beginning at $8600. It will always be the page immediately below the lowest catalog page. Since $8600 is 34304 in decimal, I set HIMEM (line 210) to 34303. You will have to figure out what the corresponding number is for your Apple. Got all that? If you're running *ALPHIE* on a 48K Apple, you don't need to do any of it, but you should notice that there's lots of room for any modifications or enhancements you might want to add.

I wrote *ALPHIE* with the intention of making the conversion to a smaller machine easy. Only two lines need to be changed. But again, I'll warn you: Make ABSOLUTELY CERTAIN that your program does EXACTLY what it's supposed to before you use it on any disk that's of value.

Since I've made such a big deal out of the dangers involved in

writing to the disk without the benefit of the guardian angel who oversees the DOS commands, I'll also offer some suggestions as to how you might go about testing the program. First, make copies of some of your disks. These must be in regular DOS format; *ALPHIE* will not work on copy-protected disks, and it could cause problems if the DOS used to write on your disks has been altered in some way. These copies should be totally expendable; if there is an error in the *ALPHIE* program when you run it, you could easily make the disk catalog unreadable to both DOS and yourself.

Store your typed version of *ALPHIE* on another disk, and DO NOT run *ALPHIE* on that disk until the program has been completely tested and debugged. When you get a version ready to run, SAVE it, then put one of the expendable copies you made into the drive. Cross your fingers and run the program. CATALOG the ALPHIEd disk to see whether the result is what you expected. Once you get *ALPHIE* to the point that he makes your catalog look right, make sure he's not secretly doing something you didn't intend, by LOADing, LISTing or RUNning each of the program files, or somehow reading any text files, on the ALPHIEd disk. Assure that each file is complete. To be absolutely certain, add and delete some files, and retest all of the disk files to make sure everything is working. Re-ALPHIE the same disk a couple of times and go through the whole testing procedure again. You can't be too careful.

If you have two disk drives, you can make it a little easier on yourself. Refer to the RWTS section of the DOS manual, and change the appropriate byte in line 720 to make *ALPHIE* work on Drive 2 instead of Drive 1. You'll also want to change the message in line 300 to reflect this. Then you can keep the disk containing *ALPHIE* in Drive 1 and the disk being ALPHIEd in Drive 2. This should save you a lot of disk-switching.

Treated with proper respect, *ALPHIE* can make it easier for you to visually locate files in your disk catalogs. If you heed the warnings, and don't clobber any good disks while testing *ALPHIE*, you'll come away with not only a nifty little utility program, but also a better knowledge of one of the ways Apple DOS keeps track of what's on your disks. In our next article, we'll look at another of the file-management features of DOS, the Volume Table of Contents (VTOC). 'Til then, have fun with *ALPHIE*.

Variables

A(X): A defined function, which determines the starting memory address of file number X.
BF%: The memory address RWTS uses to determine the high byte of the current buffer starting address.
C1,C2: Numeric values of characters being compared in sorting.
CB%: Decimal equivalent of the high byte of the address at which the catalog starts in memory.
F1,F2: Pointers to the two files being compared in sorting.
FI%: Counter which tracks sequential file numbers.
I: Loop index.
IN$: Keyboard input.
J: Loop index.
MP: Main program loop index.
M$(*): Messages displayed during execution.
NF%: Number of active files on the disk.
NX: Used in constructing new catalog, to determine where the next file is to be placed.
OP%: Address of the byte which contains the code for the RWTS operation (1 = Read; 2 = Write).
PT(*): Array of pointers to addresses at which file data are stored in memory.
RP%: Relative position of characters being compared in sorting.
RW%: Address at which RWTS calling routine and data begin.
SC%: Address of memory location holding the sector number being read from or written to.
SW%: Flag to determine whether a switch (exchange) has been made in sorting.
T%: Temporary variable, miscellaneous uses.
VT%: Column number for VTAB.
WB: Starting address of the write (output) buffer.
X: Dummy variable for definition of function FN A(X).

```
   SS SS SS SS SS SS SS SS SS SS SS
   SS                           SS
   SS      APPLESOFT BASIC      SS
   SS          'ALPHIE'         SS
   SS  AUTHOR: CARY W. BRADLEY  SS
   SS     COPYRIGHT (C) 1982    SS
   SS SOFTSIDE PUBLICATIONS, INC SS
   SS                           SS
   SS SS SS SS SS SS SS SS SS SS SS
```

```
10  GOTO 200
```

Subroutine to determine number of files and alphabetize them.

```
20 NF% = 0: FOR I = 1 TO 105:PT(I
    ) = 0
30 T% =  PEEK ( FN A(I)): IF  NOT
    T% OR T% = 255 THEN 50
40 NF% = NF% + 1:PT(NF%) =  FN A(
    I)
50  NEXT :T% = NF%
60  PRINT "("NF%;" FILE";: IF NF%
     <  > 1 THEN  PRINT "S";
70  PRINT " ON THIS DISK)"
80  IF NF% < 2 THEN  VTAB VT%: PRINT
    M$(MP): PRINT : PRINT : POP
    : GOTO 360
90 FI% = 1:SW% = 0
100 F1 = PT(FI%):F2 = PT(FI% + 1)
    :RP% = 3
110 C1 =  PEEK (F1 + RP%):C2 =  PEEK
    (F2 + RP%)
120  IF C1 > C2 THEN PT(FI%) = F2
    :PT(FI% + 1) = F1:SW% = 1: GOTO
    150
130  IF C1 < C2 THEN 150
140 RP% = RP% + 1: IF RP% < 34 THEN
    110
150 FI% = FI% + 1: IF FI% < T% THEN
    100
160  IF SW% THEN T% = T% - 1: GOTO
    90
170  RETURN
```

Program initialization

```
200  POKE 235, PEEK (115): POKE 2
    36, PEEK (116)
210  HIMEM: 34303
220  DIM PT(105),M$(3)
230  DEF  FN A(X) = 256 * CB% + 3
    5 * X + 11 *  INT ((X - 1) /
    7) - 24
240 CB% = 135:RW% = 768:SC% = 782
    :BF% = 786:OP% = 789
250  FOR I = 0 TO 3: READ M$(I): NEXT
260 M$(3) = M$(3) +  CHR$ (13) +
    M$(0)
270  FOR I = RW% TO RW% + 29: READ
    T%: POKE I,T%: NEXT
280  GOSUB 600
```

Main program control

```
300  HOME : PRINT  CHR$ (7);"INSE
    RT DISK IN SLOT 6, DRIVE 1"
310  PRINT "AND PRESS ANY KEY (<E
    SC> TO ABORT)...": GOSUB 660
320  FOR MP = 1 TO 3:VT% = 2 * MP
     + 3 + (MP > 2)
330  VTAB VT%: FLASH : PRINT M$(M
    P): NORMAL
340  ON MP GOSUB 400,20,500
350  VTAB VT%: PRINT M$(MP): NEXT
    MP
360  VTAB 13: PRINT "FINISHED.": PRINT

    : PRINT "ANOTHER DISK (Y/N)?
    ";
370  POKE  - 16368,0: GET IN$: IF
    IN$ <  > "Y" THEN  GOSUB 700
    : END
380  GOTO 300
```

Subroutine to read disk catalog into memory.

```
400  POKE OP%,1
410  FOR I = 15 TO 1 STEP  - 1
420  POKE SC%,I: POKE BF%,CB% + 1
    5 - I
430  CALL RW%: NEXT I: RETURN
```

Subroutine to write new catalog.

```
500  POKE OP%,2:FI% = 1:T% = CB% -
    1: POKE BF%,T%:WB = T% * 256
510  FOR I = 15 TO 1 STEP  - 1:NX
     = WB + 11
520  FOR J = WB TO WB + 255: POKE
    J,0: NEXT J
530  IF (I - 1) THEN  POKE WB + 1
    ,17
540  POKE WB + 2,I - 1
550  IF FI% > NF% THEN 590
560  FOR J = 0 TO 34: POKE NX + J
    , PEEK (PT(FI%) + J): NEXT J
570 FI% = FI% + 1: IF  INT ((FI% -
    1) / 7) = (FI% - 1) / 7 THEN
    590
580 NX = NX + 35: GOTO 550
590  POKE SC%,I: CALL RW%: NEXT I
    : RETURN
```

Subroutine to print program title.

```
600  HOME : VTAB 2: HTAB 17: INVERSE
    : PRINT " ALPHIE ": NORMAL
610  VTAB 5: HTAB 13: PRINT "DOS
    3.3 CATALOG"
620  PRINT  TAB( 9)"ALPHABETIZATI
    ON PROGRAM"
630  VTAB 12: PRINT  TAB( 12)"BY
    CARY W. BRADLEY"
640  VTAB 19: PRINT  TAB( 12)"PRE
    SS <ESC> TO END": PRINT  TAB(
    19)" OR"
650  PRINT  TAB( 8)"ANY OTHER KEY
     TO CONTINUE"
660  POKE  - 16368,0
670 IN% =  PEEK ( - 16384): IF IN
    % < 128 THEN 670
680  IF IN% = 155 THEN  POKE  - 1
    6358,0: GOSUB 700: POP : END
690  RETURN
```

Subroutine to restore original HIMEM value.

```
700  POKE 115, PEEK (235): POKE 1
    16, PEEK (236): PRINT : RETURN
```

The data

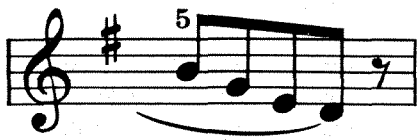
```
710  DATA DO NOT INTERRUPT.,READI
    NG CATALOG.,ALPHABETIZING.,W
    RITING NEW CATALOG
720  DATA 169,3,160,9,32,217,3,96
    ,0,1,96,1,0,17,0,26,3,0,0,0,
    0,0,0,0,96,1,0,1,239,216
```

APPLE™ TABLE FOR: APPLE™ DISKOURSE

LINES	SWAT CODE	LENGTH
10 - 120	JK	326
130 - 260	UY	285
270 - 400	HC	328
410 - 580	LE	292
590 - 700	XH	344
710 - 720	LZ	156

Apple Music: Two New Systems Soundchaser & Alpha Syntauri

Reviewed by Steve Birchall



***Soundchaser* (from Passport Designs, Inc., 785 Main Street, Half Moon Bay, CA 94019) requires an Apple II™, with monitor, one disk drive, game paddles, audio system (amplifier and speakers). Does not require Language Card. Retail price: $995.00.**

***Alpha Syntauri* (from Syntauri Corp., 3506 Waverley Street, Palo Alto, CA 94306) requires an Apple II with Language Card (for memory expansion board to 64K), monitor (color if possible), disk drive, game paddles, audio system (amplifier and speakers). Retail price is $1795 for the complete system. The Plus4 retails for $995.00.**

Turn your Apple into a live performance synthesizer. Use your computer as a recording studio to make multi-track recordings with full instrumentation. Define the sound of a real or imaginary instrument and store it on disk, ready to call up whenever you want it. Store complete performances on floppies. These are some of the capabilities of the new generation of music systems for microcomputers. The *Soundchaser* and the *Alpha Syntauri* use *Mountain Computer* boards to generate sounds, while adding their own keyboards and operating software. Best of all, you can start playing as soon as the disk is booted.

Passport *Soundchaser*

Essentially, the *Soundchaser* is simply an improved operating system, with a keyboard, for the *Mountain Computer Music System*. Mountain's human interface ranges from awkward to impossible, and the *Soundchaser* corrects many of its most uncomfortable aspects. Although most people will find that it does all they could ask, *Soundchaser* still lacks some important abilities for those seriously interested in computer music. The advanced levels of *Alpha Syntauri* offer some, but not all, of those sounds.

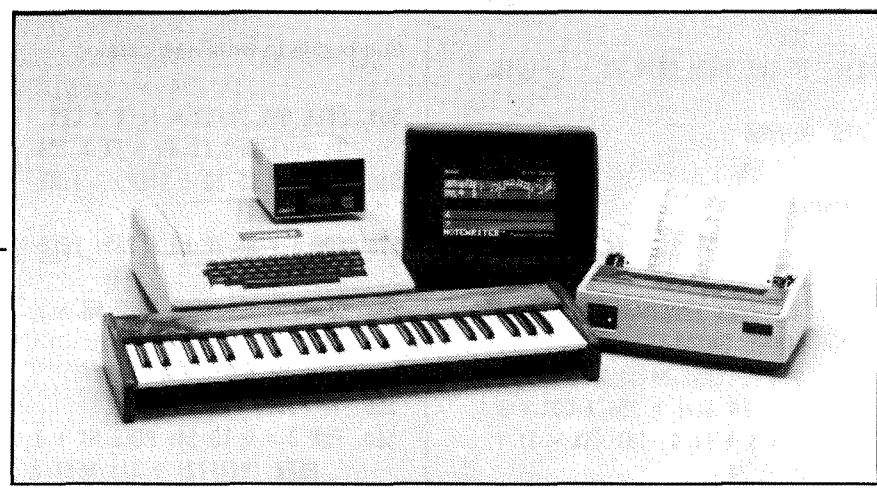
Conceptually, for each instrument definition you have two oscillators, each with 16 harmonics which you can adjust to obtain a wide range of timbres. Each oscillator has its own Envelope Shaper, which is independently controllable. Usually, one envelope and oscillator are for the main sound, while the other set are for the initial attack. A Low Frequency Oscillator provides pitch variation at infrasonic frequencies (vibrato). All the adjustments for these devices constitute an imaginary instrument, which Passport calls a preset. Up to ten presets are available at any time, and more groups of ten may be called from the disk.

Installation

Installing *Soundchaser* is fairly simple. You put the *Mountain Computer* cards in slots four and five of your Apple and the keyboard controller card in slot seven. The cable to the keyboard attaches to the card with a multi-plug like the one your disk drive uses. From the *Music System* boards come a double pair of audio cables with female phono plugs, and you simply connect a regular hi-fi patch cord (included) between these outputs and your stereo system. That's all you have to do. The *Soundchaser* is not bulky, but you need to have your Apple close enough to your stereo system to be able to hook it up and operate the controls on your amplifier. Also, you must have your Apple on a table big enough to accomodate the keyboard's rather short cable. You end up with a lot of equipment connected together, and you want to be within arm's reach of all of it. Needless to say, it's a satisfying array of gear when you first turn it all on and start playing music. If you have a good audio system, the *Soundchaser* will sound impressive.

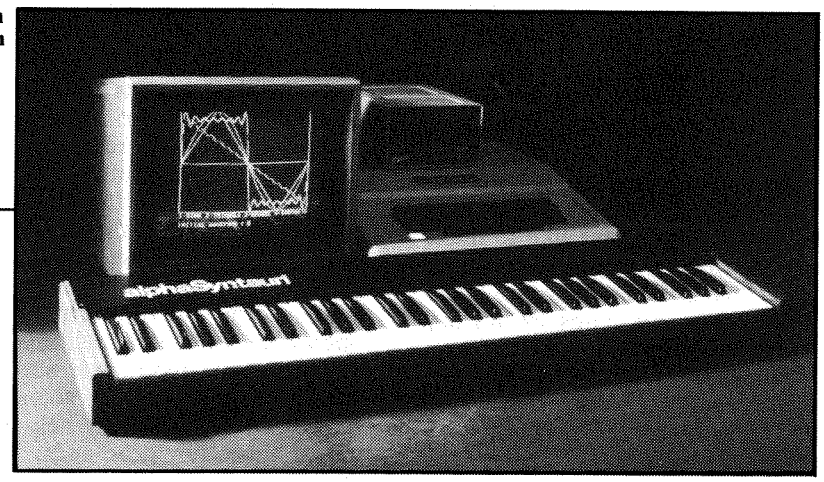
What's Included

1) Complete *Mountain Computer Music System* — two boards with lightpen, operating software, cable to connect to an audio system, manual



The *Soundchaser* music system from Passport Designs, Inc.

The Alpha Syntauri 5 music system from Syntauri Corporation



2) Keyboard (49 notes), connecting cable
3) Keyboard controller card
4) *Soundchaser* operating software
5) Manual

Options:

1) Music Tutor: a set of programs to teach music theory and ear training, with drills and quizzes. Price: $150

2) Notewriter: will print out single lines of music after you play it on the keyboard (on the Silentype printer or any printer compatible with the Grappler card). Also permits editing music for subsequent playback. Price: $99

Playing The *Soundchaser*

After booting the disk, a screen appears and you can begin playing immediately. This is a quantum leap over MC's operating system, which required considerable input before you could make music with it.

The large rectangular blank space outlined in the middle of the screen is the display area for graphs of the waveforms. If you want to see the waveform you are using, press (W) to enter this section. You then have a menu of choices: (C)reate, (E)dit, (D)isplay, and (S)mooth. Press (D) and you are asked to select one of the ten preset sounds, and which of the two waveforms you wish to see. Then watch as *Soundchaser* draws it for you.

To (C)reate a wave, press (C), then indicate which preset and waveform you want to alter. Type C to clear the bar graph and move the cursor with the right and left arrow keys to one of the 16 harmonics. Using the game paddle knob, raise or lower the height of the bar which indicates the strength of that harmonic. Pressing the paddle button sets the value and advances the cursor to the next harmonic, and Return computes the waveform so you can hear it.

The (S)mooth function, a simple 6dB/octave digital filter, has a variable cutoff frequency to eliminate unwanted high harmonics which cause harsh tonal qualities. It also prevents extreme high frequencies from recycling back as unwanted, distortion-producing, low frequencies (aliasing). Think of it as the Tone Control.

The (E)dit function lets you alter one harmonic value quickly. It also permits you to draw a waveform. Hold the game paddle's button down and rotate the knob as the cursor moves across the screen, tracing your waveform for you.

The (T)une control enables you to tune your instrument to others in your group and to move the range of the keyboard up or down. Default value is A = 440Hz, but the range is from 200Hz to 600Hz. Also, if you want to try other tuning systems besides Twelve Tone Equal Temperament, Passport has available the optional Just Tone and Mean Tone Tunings on a separate disk.

The two envelopes (one for each oscillator) can be adjusted by typing values into the table at the bottom of the main screen. These are ADSR (attack, decay, sustain, release) type and are independent of each other. The tricky thing is that you have to use hex values, 00 being the lowest (silence) and FF the highest (maximum loudness).

A low frequency oscillator provides vibrato effects by modulating the pitch of the audio oscillators at a very slow (infrasonic) rate.

Digital Recorder

The sequencer/four-track recorder is a very handy gadget. It works very much like a four-track tape recorder, but, rather than storing the actual musical signal, it stores all the instructions to the MC boards in digital form (either in RAM or on disk). The advantage of this approach is that the music is newly generated with each playback, so no signal deterioration takes place, as with analog tape recorders. The boards are capable of playing up to eight voices simultaneously, distributed among the four tracks of the sequencer.

The Soundchaser and the Alpha Syntauri use Mountain Computer boards to generate sounds, while adding their own keyboards and operating software. Best of all, you can start playing as soon as the disk is booted.

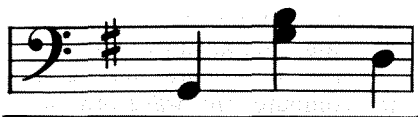
Used as a single voice sequencer, it can repeat the phrase continuously.

During playback you can change the presets any time you like and increase the tempo (with the left and right arrow keys on the Apple). Only four tempos, all faster, are possible, which is a distinct weakness of the system. Musicians like to make gradual changes in speed and have slower tempos available, as well. The four discrete jumps are not sufficient. Having the ten presets available at any given moment means you can alter the sound drastically or subtly, depending on how you have programmed the presets. However, in the four-track sequencer mode, the same group of ten applies to all four voices, which is somewhat limiting. To change, you must type the track number and the preset number, which although quick, can be confusing. Better human engineering would call for a letter designation for the track, and a number for the preset.

The four track recorder is easy to use. Press ESC to switch to the Sequencer screen. Type the number of the track on which you want to record and the letter R. (If you want to change the preset, do it now.) Hit the space bar and begin playing. Record a pause and one extra note, then the space bar to stop recording. For playback, type the number(s) of the track(s) you want to hear, followed by the letter P, hit the space bar, and listen. You can change from one preset to another anytime you like during playback. For playback on track one and recording on track two, type the sequence: 1P (preset number desired) 2R (preset) Space Bar. A number followed by the letter O turns that track off. What the system lacks here is a way to control relative volumes of the four tracks during playback — a mixer. Passport plans to introduce a 16-track recorder soon, for about $195, which presumably will take care of this problem, but users of the basic four tracker also need to have mixing available.

Although not available from the *Soundchaser* software, other waveforms, such as square, triangular and sawtooth, can be constructed and saved for future use. No white noise generator is provided, however, which is a serious omission. White noise is useful for making more realistic attacks, (since acoustical instruments have a strong noise component on attacks) and filtered noise is a frequently used sound in electronic music.

The manual is short, direct, to the point, and easy to follow. The program itself is written so that you cannot fall into traps and dead-ends, except when performing disk operations. For safety in live performance, when rebooting is not possible, a crash-prevention command, such as ESC ESC, would help to get out of those few dead-ends. Pressing a key on the musical keyboard will nearly always produce a sound. *Soundchaser* makes a wide range of pleasing tone colors, with low distortion and noise. You can play



"Soundchaser makes a wide range of pleasing tone colors, with low distortion and noise."

music with ease because the system doesn't get in your way.

Alpha Syntauri

This system is decidedly more complex than the *Soundchaser.* After passing through many revisions and updates, it has become more flexible, and capable of more refinements of sound qualities than earlier versions. No doubt those who started with the original *Syntauri* have been pleased with these developments. For a newcomer, though, it is rather bewildering.

Syntauri, like *Soundchaser,* is an improved operating system for the *Mountain Computer* cards. The Plus4 system is similar to *Soundchaser*, though without the sequencer/recorder. But hold on, they have three more levels of operating systems. The second level is called Alpha Plus, followed by Superplus and Metatrak. Each level adds more possibilities and is intended for a different purpose. Alpha Plus adds a large palette of sound modifications such as vibrato, pitch bend, timbre scan (which scans across all the presets for a glittering, tinkling sound quality), pitch sweep, oscillator offset (when very slightly out of tune, gives you a chorus effect; or when tuned in intervals like thirds or fourths gives you a sound like organ pipes), linear or exponential envelopes, alternative scale tunings, and an intervals per octave adjustment. Alpha Plus becomes your development system, creating new sounds and defining instrument presets during practice. Superplus is intended as the *Syntauri*'s performing software, enabling you to make quick changes, and split the keyboard (different instrument presets for different sections of the keyboard). It has an eight track recorder/sequencer, metronome (click track), and a master volume control. Finally, Metatrak brings you a 16-track recorder/sequencer with punch-in and punch-out capability, erase, fast forward, and mixdown facilities. Also, you can control each track independently for instrument preset and vibrato.

What's Included:

1) *Mountain Computer* boards only (none of the MC peripherals are included. However, you will never want to use them)
2) Keyboard controller card
3) Keyboard with 51 notes
4) Shielded keyboard cable (required, but costs $35 extra)
5) Two foot pedals
6) Set of disks with operating systems and examples
7) Manual
8) Price: $1795 for the complete *Alpha Syntauri System.* The Plus4 has a four octave keyboard, lacks the upper two levels of software and sells for $995.

Options:

1) Musicmaster, a theory teaching course, for $150.
2) Composer's Assistant, an eight or more voice polyphonic note

scoring program. Although not yet available, Syntauri claims this package will be capable of transcribing a piece previously recorded on the system, according to a metronomic setting. The price has not been announced.

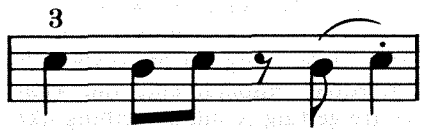
Installation is similar to the *Soundchaser*, but you can put the three cards in different slots, (with changeable default values) depending on what other peripherals you have connected.

Using *Alpha*

Reading through the manual reminds me of Phillip K. Dick's novels, especially *Ubik*. Just as you think you have everything figured out, a trapdoor swings open and you are confronted with new facts which wipe out everything you thought you knew. But, the rewards of sticking with it and learning each new level are considerable and musically worthwhile. *Alpha* requires quite a lot of homework, listening and practice, similar to learning a new word processing program. On the other hand, it's much easier to learn than playing the piano, so musicians should not be hindered by its complexity. However, computer beginners will have some difficulty dealing with the advanced levels of *Alpha Syntauri*, and should have some prior familiarity with typical Apple software (VisiCalc, word processors, Personal Filing System) and how to use it.

When I finally ascended to Superplus, I began to get excited about computer music systems. I began to draw interesting and useful sounds out of the instrument for the first time — sounds not quite so closely tied to keyboards, scales and traditional music. The system had enough ways to modify sounds that I began to feel some degree of freedom. The special effects modifications, which are the most enjoyable aspect of the entire system, enabled me to make wonderful bloops and squiggles and roars — sounds approaching what I was accustomed to getting from my analog synthesizers. The presets gave me a new freedom to jump from one sound to another instantly, which I never had with the analog equipment. Using the 16-track recorder I could build up complex textures of these sounds quickly and easily.

The manual is much longer than *Soundchaser*'s, but it has more ground to cover. Also, it's written in a nice, conversational, tutorial style, which makes learning to play it painless. At the ends of chapters, summaries of control commands are provided, which are useful when you are playing and forget how to ask for a modification. A separate card for quick reference would be a thoughtful addition. Despite its complexity, the system won't let you get trapped into dead-ends, and I never crashed. An escape route is always available.



"Alpha requires quite a lot of homework, listening and practice, similar to learning a new word processing program."

Summary

Both of these systems make many beautiful sounds, and, for a keyboard-oriented musician, offer an effective, flexible, immediately useable means for making music. The sounds are complex enough to benefit from a high performance audio system. When played through good equipment, no unpleasant distortion or noise occurs, which is an important consideration. In the Metatrack mode, *Alpha Syntauri* does have some faint warbles and post-decay sounds because of the complex computations taking place, but these are hardly objectionable. They add character to the instrument, much as chiff does to the organ, or the sound of fingers squeaking on a guitar string add to its authenticity.

I think *Soundchaser* or the *Alpha Plus4* will be the system of choice for many because they are simple to operate, and offer as much variety in sound quality as most people need. You can spend many happy hours playing *Soundchaser* or *Plus4*. However, for the more serious musician, the *Alpha Syntauri* 5 is the way to go. With it you can make more sounds, and have the capability of modifying them in more interesting ways, including special effects you create yourself. The tradeoff for all this is that the system, while not difficult, takes more time to learn, and costs more. With the lower levels for design and development of sounds, and the SuperPlus as a performing system, live performance becomes a much more manageable possibility.

The Metatrak system with its 16-track (eight simultaneous voices) recorder is exactly the right tool for making extended musical structures. The mixdown capability, which *Soundchaser*'s four-track recorder lacks, is a vitally important one. On the other hand, *Plus4, Syntauri*'s entry level system, lacks the sequencer entirely.

Since you can store the sounds of imagined instruments on disk and recall them, both of these systems expand their capabilities with use. The more inventive you are, the larger the array of sounds you have available. In this respect, the systems are like LOGO; they grow, and "learn" as you work with them. If you have a complex instrument definition you want to alter and improve, simply call it off the disk and continue, or use it as a preset in a performance — no need to start from scratch. The systems will evolve in different ways for everyone who plays them, depending on what the user asks them to do, making a personal, customized instrument for each musician.

Badly needed on both is a band-pass filter, or two or three (the smoothing filters provided on the *Soundchaser* do just that — they round off the rough edges of the digital oscillators). A filter with variable frequency and resonance ("Q") would be extremely useful, particularly if it could be connected

to an envelope generator, so you could make timbres which change with time. Also, such filters would permit you to use formants, (resonant peaks to simulate acoustical instruments) which would sound more realistic.

A fixed harmonic structure does not describe the sound of acoustical instruments adequately, since the tone quality changes from note to note. Because they have fixed resonance points (determined by the size and shape of their bodies) their harmonic structure is altered by those resonances. Play a scale on a trombone or clarinet and listen to the way the color changes from note to note. On the *Soundchaser* and *Alpha Syntauri*, the color is fixed and moves in parallel up and down the scale, which is artificial and unpleasing. Formants are not difficult to accomodate. If Texas Instruments can do it for Speak and Spell® and elevators can talk, we can reasonably ask for formant generators (a series of resonant filters on the output) on a music system for a computer.

Like analog synthesizers, digitals offer such an awesome variety of sounds that the biggest problem is limiting the range for a particular composition. The digital synthesizer's sounds are not too different from those of analog equipment. What present digital synthesis offers beyond analog is the ability to interconnect various pieces of equipment and change the settings on the controls, almost instantly, simply by typing a command. On analog equipment, you must plug in a bunch of wires to connect all the units, turn knobs, and flip switches, all of which takes time. The other solution, which current popular performing instruments use, is to limit the choices, but make them readily available by flipping a switch. On a digital synthesizer, the computer does it for you in a fraction of a second, taking all the nuisance out of live performance, and making more choices available. This is possible because the oscillators, envelope shapers, filters and other familiar components of analog synthesizers are only concepts contained in the digital software.

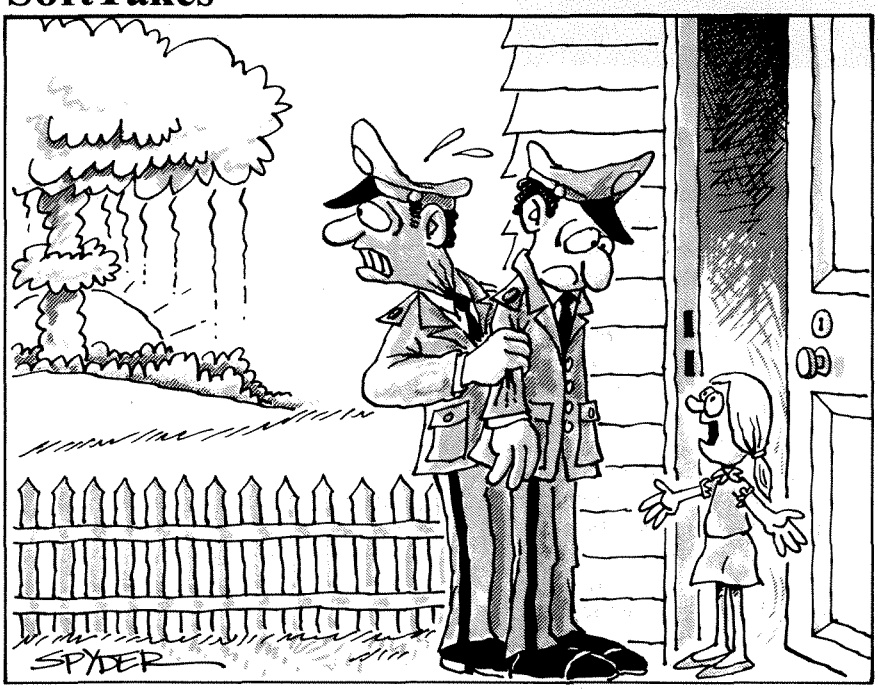
Unfortunately, manufacturers seem to be hung up on keyboards and equal temperament, and what we are getting is still something like a fancy electric piano. The unique and idiomatic capabilities of computer music are only beginning to be explored on micros. These second generation systems overcome the awkwardness and deficiencies of their predecessors, and offer some new possibilities. But we know computers are capable of generating more kinds of sounds, particularly sounds which are not possible to play through a keyboard, because this has been the nature of computer music for the past 30 years. In fact, we hear these sounds all the time in computer games. What we need in the third generation music system is a way to control those sounds easily and quickly. We need ways to shape timbres and musical structures beyond what a mere sequencer can do. That is the real potential of the computer for making music. But until then, *Soundchaser* and *Alpha Syntauri* are immensely enjoyable instruments to play. Anyone who knows a little about music and owns an Apple can have a lot of fun with them.

***Editor's Note: SoftSide** called Syntauri and Passport Designs just before press time to find out about any new product developments. Both told us of some very interesting packages.*

Syntauri is introducing a new version of their Metatrak operating system which will interface to the more popular drum machines, both digital and synthesized. It will operate from a metronomic setting in order to keep syncronization accurate. They're also upgrading the Alpha Plus operating system for both the four and five octave systems to include a feature called Quick Wave, allowing real time manipulation of wave forms during sound creation and playback via a bar graph representation on the monitor.

Passport Designs is also releasing a new operating system for the Soundchaser, called Turbo-Traks. It will allow the sequencing of 16 voice recordings with a variable number of oscillators and amplitude controls for each voice. It, like the Alpha Syntauri, will also interface to the more popular drum machines and allow accurate syncronization with tape recording devices. Turbo-Traks is scheduled to be available October 1, for $150. In the education field, Passport is developing packages for harmonic, rhythmic and keyboard training which they hope to make available in the first quarter of 1983.

SoftTakes



"SO I TAPPED INTO A MILITARY COMPUTER WITH MY ATARI. WHAT'S THE BIG DEAL?"

SOFTWARE MAGIC
P.O. Box 2184
Bramlea Postal Station
Bramlea, Ontario, CANADA L6T 3S4
(416) 451-9452

Maestro is a Machine Language program which turns your TRS-80® Model I or Model III into a music synthesizer. It has three playing modes, one of which produces synthesizer type sound. In this mode, the TRS-80 can produce two octaves of C to C ascending in semi-tones, as well as a number of pre-programmed scales and runs. A second mode has the same capabilities, but not in a synthesizer type sound. The final mode allows the player to glide up and down a full octave from each note selected.

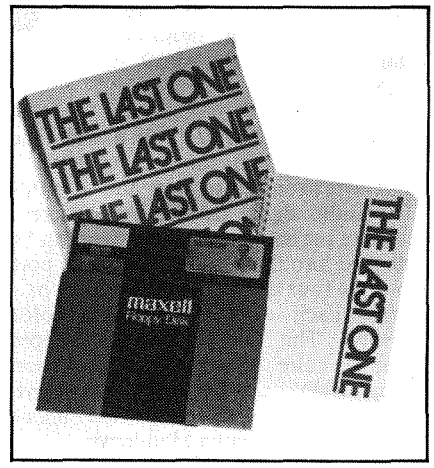
Maestro runs on any 16K, Level II, Model I or Model III TRS-80. It is available for $19.95 (check or money order) from Software Magic.

KROWN COMPUTING
Box 66763
Scotts Valley, CA 95066

The Last One is a program writer which produces programs in BASIC on a range of popular microcomputers, using your own rough program design, or "flowchart," as its starting point.

The Last One is used in conjunction with your usual resident BASIC. Yet, no knowledge of BASIC programming is required, since your input is performed using question and answer routines in plain English. The only BASIC listing you will ever see is the finished, up-and-running program. The programs, once created, are independent of *The Last One,* just as your own current programs are independent of you.

For more information, send for a free, descriptive brochure from Krown Computing.

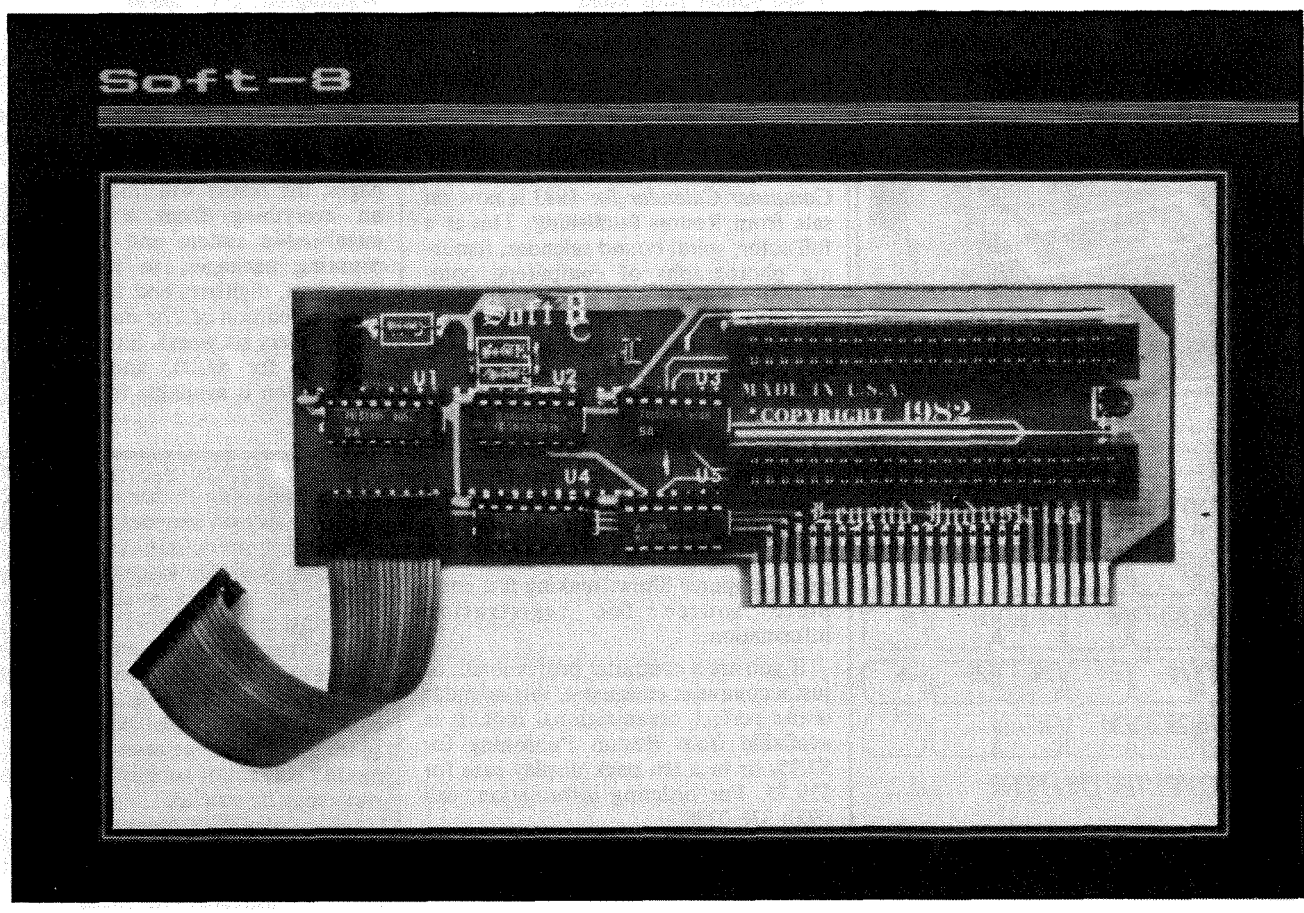


LEGEND INDUSTRIES LTD.
P.O. Box 112
Pontiac, MI 48056
(313) 674-0953

The new *SOFT 8* for the Apple II™ computer solves the problem of the restriction to only eight slots which the Apple has on the motherboard by providing an extra slot inexpensively. It plugs into slot 7 and thus provides slot 7 and 8. Switching between slots is software driven, and allows you to switch between one card and another with simple software commands. With *SOFT 8*, you can now put nine cards in your Apple and have them all software accessible.

The disk supplied with the *SOFT 8* will allow you to modify standard Apple DOS so that it will recognize the added slot.

For additional information and pricing, contact Legend Industries, Ltd.



PASSPORT DESIGNS, INC.
116 North Cabrillo Highway
Half Moon Bay, CA 94019
(415) 726-0280

KALEIDO-SOUND™ is a real time graphics program for the Apple II™ that synchronizes to any audio input. The audio signal from a home stereo, cassette player or a sound system can be plugged into the Apple II's cassette port to drive four different kaleidoscope patterns, each with a selectable color scheme. The full color kaleidoscopes change color, pattern, and location dynamically on a CRT monitor, television, or video screen as the music changes frequency and loudness.

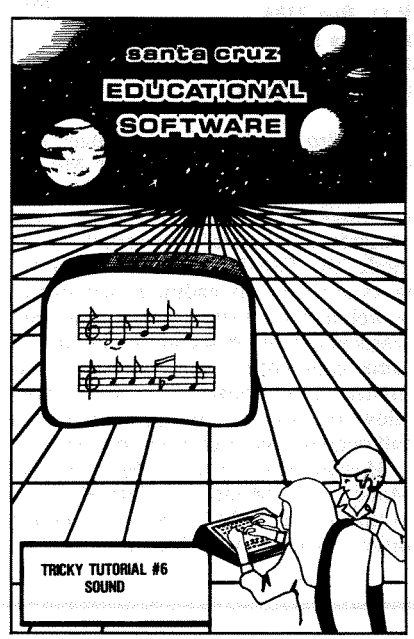
KALEIDO-SOUND™ has a suggested retail price of $39.95 (including connecting cable). It is available at computer stores nationwide or directly from Passport Designs, Inc.

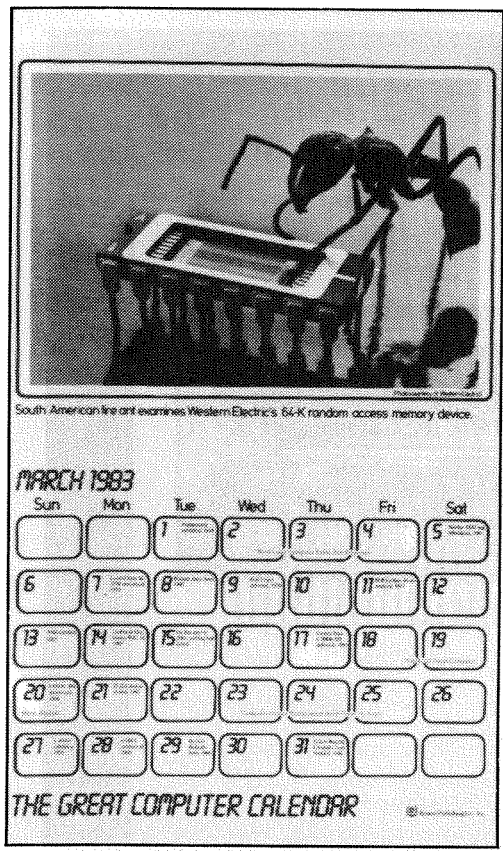
EDUCATIONAL SOFTWARE, INC.
4565 Cherryvale
Soquel, CA 95073
(408) 476-4901

The Sound Tutorial, number 6 in the Tricky Tutorials series, makes a complicated subject usable for the average Atari owner. It starts with the simple sound statement, but progresses to chords, complete songs and special effects. It also demonstrates the use of direct POKEs to the computer's built-in, four channel frequency controls. All of the material can be used by a beginner, but the more sophisticated user can also learn much from it.

The Sound Tutorial includes 32 pages of text, cartoons and programs. A program, Player Piano, can be added to the tutorial at no extra cost.

The Sound Tutorial requires an Atari 400/800 with 16K — tape, 24K — disk, and a BASIC cartridge. The retail price, for both cassette and disk versions, is $19.95.





RESTON PUBLISHING COMPANY, INC.
11480 Sunset Hills Road
Reston, VA 22090
(703) 437-8900

Essential for the office, ideal for the home, or the perfect gift for anyone who is hooked on computers, *The Great Computer Calendar for 1983* is now on sale from Reston Publishing. This is a full color, spiral bound calendar, featuring photographs of computers, components, people and places that computer people are talking about. It also includes an extra month, January, 1984, for advance planning convenience.

In the calendar's back pages, all of the important dates in the history of computers are fully explained. There will also be advance planning and registration information for the major computer conventions, such as the West Coast Computer Faire, and the Northeast Computer Show, making it a complete source for registration information.

If you are a computer professional, or just a computer enthusiast, this calendar is the perfect organizational tool. It is available from Reston Publishing for $7.95, or in a ten pack display case for $39.75. For ordering information, call (800) 336-0338.

STARWARE
1701 K Street, N.W.
Washington, D.C. 20006
(202) 466-7351

The *Buyer's Guide for the IBM® Personal Computer* describes hundreds of products for the PC, providing the most comprehensive source of product information for this computer. In its 28 pages, the reader will find information on everything from a health club membership system and 14 other accounting packages, to touch panels, light pens, digitizers and furniture.

A new edition of *The Buyer's Guide* is mailed every six weeks. Single copies are available for $1.00, and a one-year subscription is available for $8.00 from Starware.

SoftSide's New Products section is an effort to inform our readers of newly available microcomputer products. However, ***SoftSide*** assumes no responsibility for product quality, company reliability or data accuracy. The information printed is submitted by promoters and selected for publication on a first-come, editorial preference, and computer compatible basis. Due to space limitations, we reserve the right to modify content of submissions. Submissions may be sent to:

***SoftSide* Magazine**
New Products Editor
6 South Street
Milford, NH 03055

MARKET/SIDE

Advertiser's Index

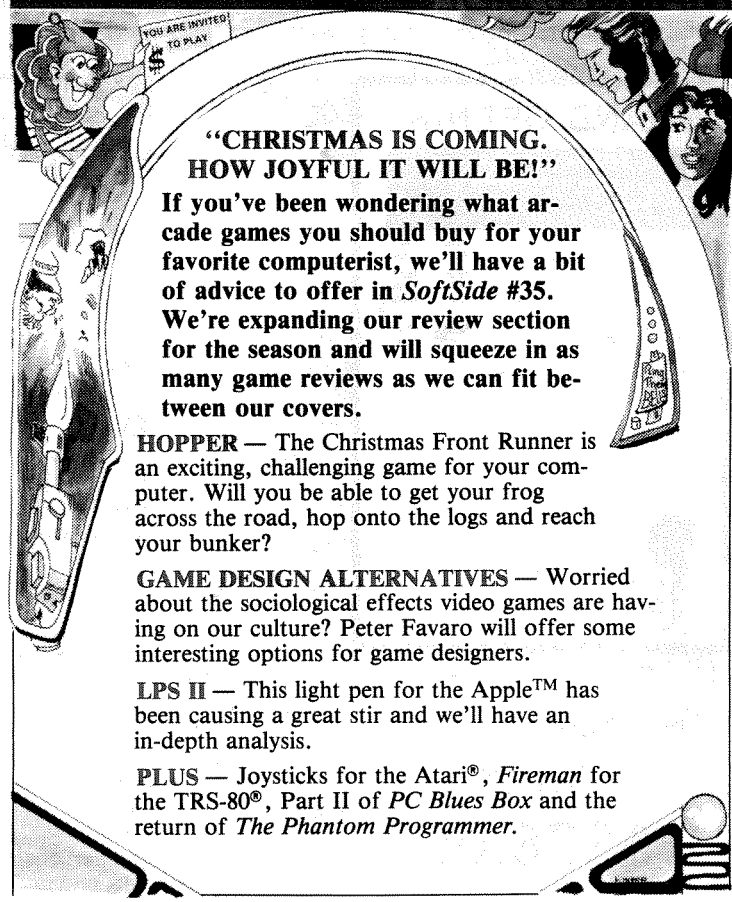
Advertising Representatives
Sue Rowland, Bob Mackintosh (603) 673-0585

National Representative
Christopher Smith, Inter-Marketing Associates
(603) 827-3976

Associate Publisher
Nancy Lapointe

Publisher
John G. Grow

Next time in SoftSide #35



"CHRISTMAS IS COMING. HOW JOYFUL IT WILL BE!"

If you've been wondering what arcade games you should buy for your favorite computerist, we'll have a bit of advice to offer in *SoftSide* #35. We're expanding our review section for the season and will squeeze in as many game reviews as we can fit between our covers.

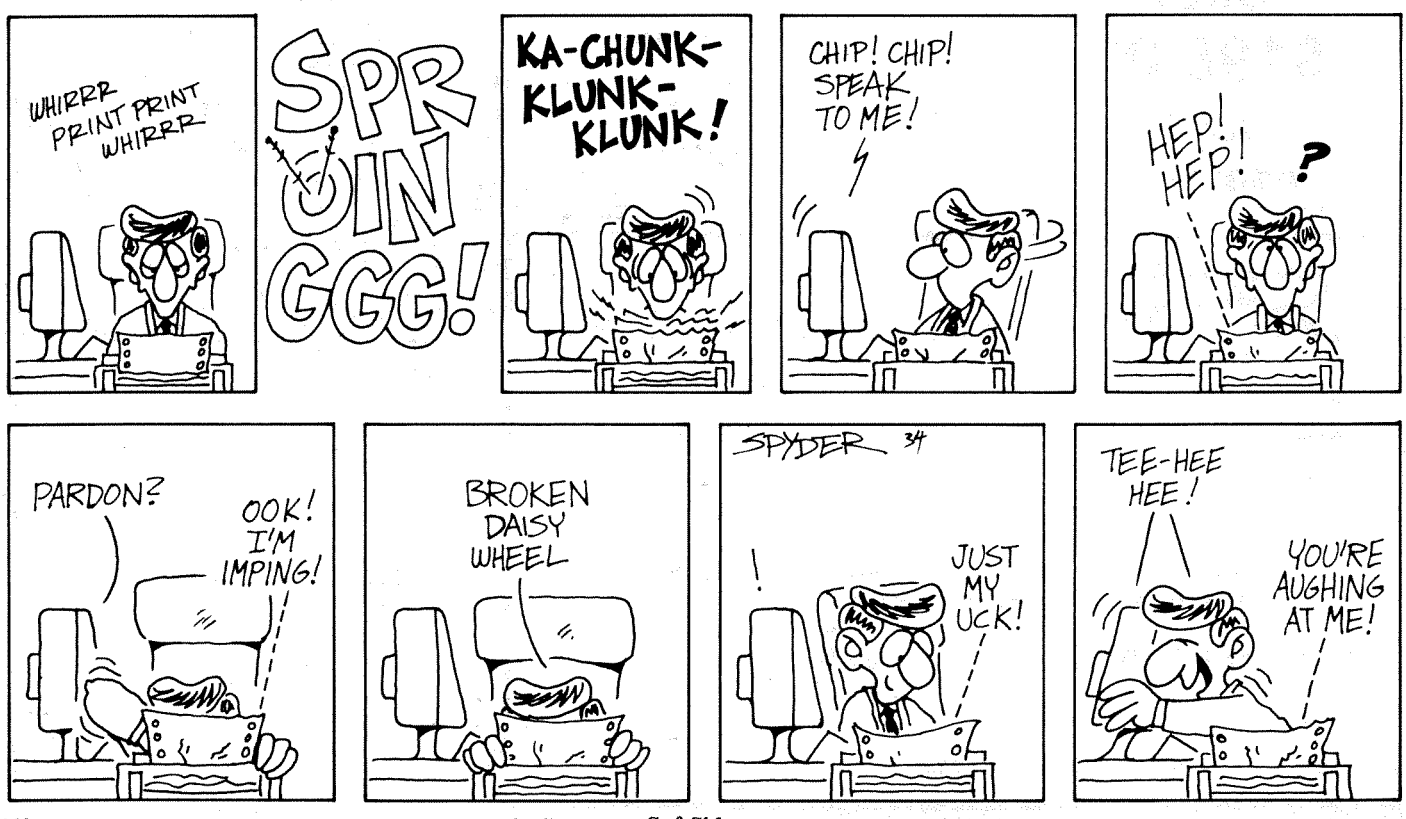
HOPPER — The Christmas Front Runner is an exciting, challenging game for your computer. Will you be able to get your frog across the road, hop onto the logs and reach your bunker?

GAME DESIGN ALTERNATIVES — Worried about the sociological effects video games are having on our culture? Peter Favaro will offer some interesting options for game designers.

LPS II — This light pen for the Apple™ has been causing a great stir and we'll have an in-depth analysis.

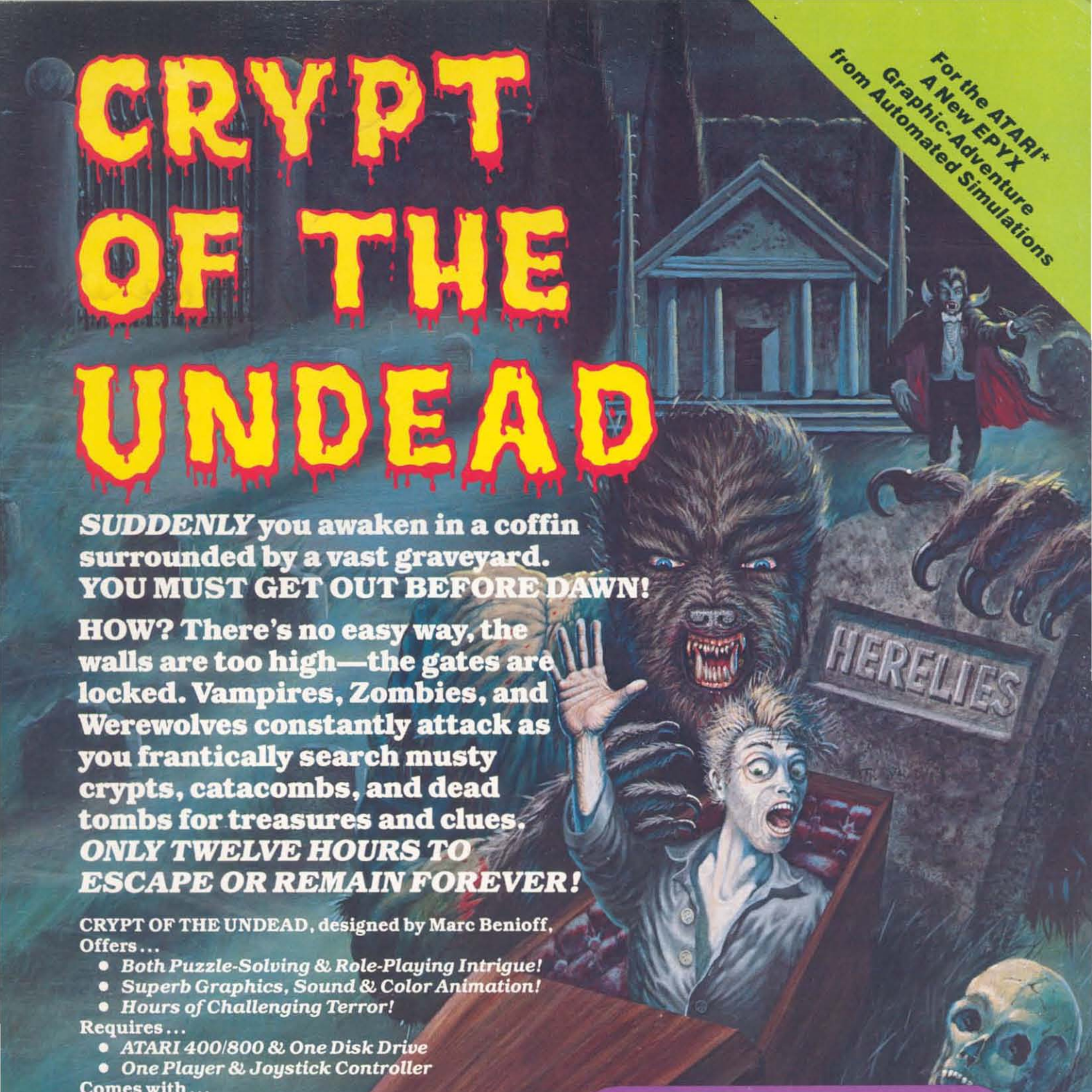
PLUS — Joysticks for the Atari®, *Fireman* for the TRS-80®, Part II of *PC Blues Box* and the return of *The Phantom Programmer.*

MACHINE HEAD BY SPYDER



CRYPT OF THE UNDEAD

For the ATARI* A New EPYX Graphic-Adventure from Automated Simulations



SUDDENLY you awaken in a coffin surrounded by a vast graveyard. YOU MUST GET OUT BEFORE DAWN!

HOW? There's no easy way, the walls are too high—the gates are locked. Vampires, Zombies, and Werewolves constantly attack as you frantically search musty crypts, catacombs, and dead tombs for treasures and clues. *ONLY TWELVE HOURS TO ESCAPE OR REMAIN FOREVER!*

CRYPT OF THE UNDEAD, designed by Marc Benioff, Offers...

- *Both Puzzle-Solving & Role-Playing Intrigue!*
- *Superb Graphics, Sound & Color Animation!*
- *Hours of Challenging Terror!*

Requires...

- *ATARI 400/800 & One Disk Drive*
- *One Player & Joystick Controller*

Comes with...

- *Game Program & Complete Instructions*
- *EPYX 30/FOREVER WARRANTY*

Now Available At Your Favorite Dealer... $29.95
For the name of your nearest EPYX dealer write:
"CRYPT OF THE UNDEAD"
EPYX/Automated Simulations, Inc.
P.O. Box 4247, Mountain View, CA 94040

EPYX
COMPUTER GAMES THINKERS PLAY

EPYX Temple of Apshai was the very first computer game ever to win the Hobby Industry award for excellence. EPYX pledges you that same excellence in every game you purchase from us ... the VERY BEST in entertainment!

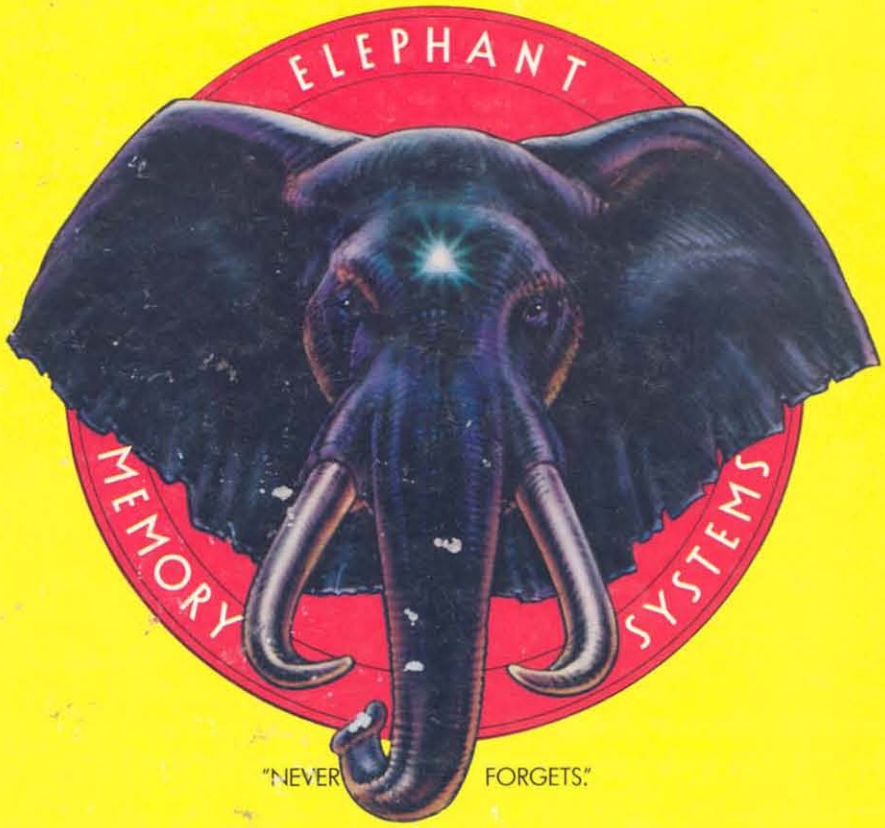
EPYX 30/FOREVER WARRANTY

*Our 30-day Unconditional Guarantee: If your EPYX Game has any defect whatsoever within 30 days of purchase, return it to us or your dealer and we will replace it free.

*Our Forever Warranty: If anything happens to your disk at any time after 30 days, for any reason, just send it back with $5.00 for shipping, and we will send you a replacement.

*ATARI 400/800 is a trademark of ATARI, INC.

REMEMBER:



MORE THAN JUST ANOTHER PRETTY FACE.

Says who? Says ANSI.

Specifically, subcommittee X3B8 of the American National Standards Institute (ANSI) says so. The fact is all Elephant™ floppies meet or exceed the specs required to meet or exceed all their standards.

But just who is "subcommittee X3B8" to issue such pronouncements?

They're a group of people representing a large, well-balanced cross section of disciplines—from academia, government agencies, and the computer industry. People from places like IBM, Hewlett-Packard, 3M, Lawrence Livermore Labs, The U.S. Department of Defense, Honeywell and The Association of Computer Programmers and Analysts. In short, it's a bunch of high-caliber nitpickers whose mission, it seems, in order to make better disks for consumers, is also to make life miserable for everyone in the disk-making business.

How? By gathering together periodically (often, one suspects, under the full moon) to concoct more and more rules to increase the quality of flexible disks. Their most recent rule book runs over 20 single-spaced pages—listing, and insisting upon—hundreds upon hundreds of standards a disk must meet in order to be blessed by ANSI. (And thereby be taken seriously by people who take disks seriously.)

In fact, if you'd like a copy of this formidable document, for free, just let us know and we'll send you one. Because once you know what it takes to make an Elephant for ANSI . . .

We think you'll want us to make some Elephants for you.

ELEPHANT.™ HEAVY DUTY DISKS.

For a free poster-size portrait of our powerful pachyderm, please write us.

Distributed Exclusively by **Leading Edge Products, Inc.**, 225 Turnpike Street, Canton, Massachusetts 02021
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.