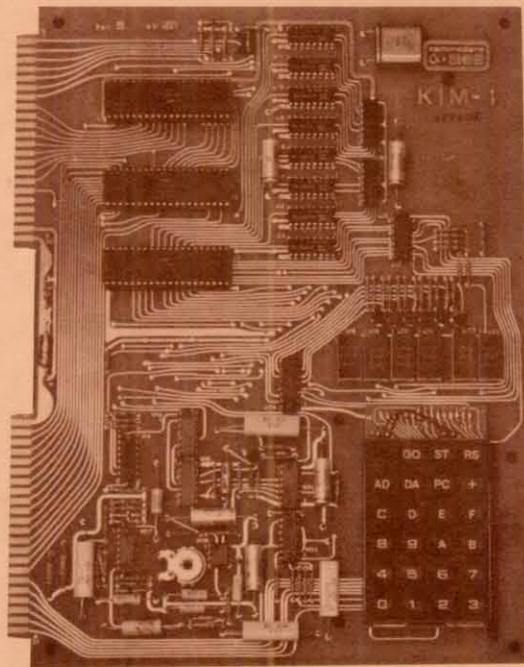


MICRO™

The Magazine of the APPLE, KIM, PET
and Other 6502 Systems



IN THE BEGINNING
WAS THE KIM-1

NO 11

April

1979

\$1.50



14052 EAST FIRESTONE BOULEVARD · SANTA FE SPRINGS, CALIFORNIA 90670

(213) 921-2111

(714) 739-0711

- BUSINESS
- EDUCATIONAL
- PERSONAL

16K APPLE II \$1195.00

- We are Apple Headquarters. Buy from us and let us help you get the most from your Apple.
- Free software with purchase of Apple. Choose up to \$100.00 worth of software from over 100 selections in our Apple software catalog.
- Our unique Apple II software catalog gives a short critique on each program so that you know what you are buying before you buy.
- We also have numerous selections of software free with any purchase.
- We offer service contracts for all equipment we sell.
- We provide demonstrations and orientation services for schools and businesses anywhere in Los Angeles or Orange Counties, California. Call for information regarding this free service.



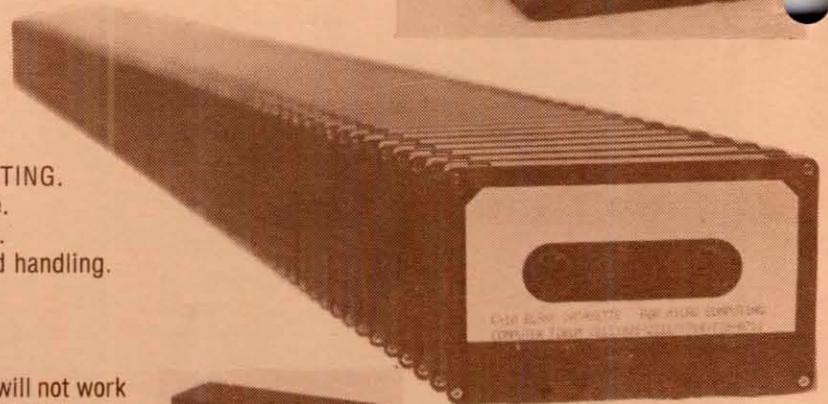
8K COMMODORE PET ... \$795.00

- FREE \$ 50.00 worth of software of your choice with each pet.
- Software Over 50 programs now available and still growing. Send for our software catalog.
- In Stock Off the shelf deliveries
- Service We have an on site service department.
- Users group now forming



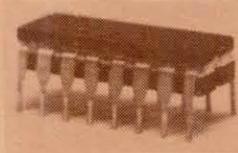
BLANK CASSETTES

- C-10 BLANK CASSETTES (W/O BOX) FOR MICRO COMPUTING.
- \$ 1.00 Ea. Other lengths will be available in the future.
 - \$ 7.50 for 10 Call or write for quotes on larger quantities.
 - \$32.50 for 50 Add 10% (minimum \$2.00) for shipping and handling.



16K RAM FOR APPLE II

- 4116 Chips. 8 per set makes 16K RAM. Just any 4116 chips will not work in the Apple II. 350NS or slower will not work properly. 250NS chips are adequate. 200NS chips are preferable.
- 200NS chips. Tested and guaranteed \$ 95.00 per set.
- Add \$ 2.00 for shipping and handling.



QUANT.	ITEM	PRICE	ABOVE ITEMS ARE NORMAL STOCK ITEMS. DUE TO CIRCUMSTANCES BEYOND OUR CONTROL, WE ARE SOMETIMES TEMPORARILY OUT OF STOCK. PLEASE CHECK APPROPRIATE BOX BELOW.
			CERTIFIED CHECK, MONEY ORDER, VISA OR MASTER CHARGE ORDERS SHIPPED SAME DAY. NO COD. ALLOW 2 WEEKS FOR PERSONAL CHECK TO CLEAR.
	CALIFORNIA RESIDENTS, ADD 6% SALES TAX		PLEASE BACK ORDER IF OUT OF STOCK <input type="checkbox"/> DO NOT BACK ORDER IF OUT OF STOCK <input type="checkbox"/>
	SHIPPING & HANDLING. ADD \$10.00 FOR COMPUTER SYSTEM.		IF USING CREDIT CARD, CHECK BOX AND ENTER CARD NUMBER BELOW.
	TOTAL		MASTER CHARGE <input type="checkbox"/> VISA <input type="checkbox"/>

MICRO™

APRIL 1979
ISSUE NUMBER ELEVEN

TABLE OF CONTENTS

In This Issue/MICRO Interrupts	3
An Apple II Program Edit Aid by Alan G. Hill	5
Lifesaver by J. Stelly	9
Corrected KIM Format Loader for SYM-1 by Nicholas J. Vrtis	12
A Close Look at the Superboard II by Bruce Hoyt	15
EKIM or MAXI-KIM by Andrew V. W. Sensicle	19
A Cassette Operating System for the Apple by Robert A. Stein, Jr.	21
ASK the Doctor - Part III by Robert M. Tripp	25
The MICRO Software Catalog: VII by Mike Rowe	29
SYM-1 6522-Based Timer by John Gieryic	31
The TVT-6: A User's Report by Edward Chalfin	34
6502 Bibliography - Part X by William R. Dial	35
The Ultimate PET Renumber by Don Rindsberg	37

MICRO™

STAFF

Editor/Publisher
Robert M. Tripp

Business Manager
Donna M. Tripp

Administrative Assistant
Susan K. Lacombe

Circulation Manager
Maggie Fisher

Distribution
Eileen M. Enos
Carol A. Stark

Micro-Systems Lab
James R. Witt

Gofer
Fred Davis

MICRO™ is published monthly by:

The COMPUTERIST®, Inc.
P.O. Box 3
So. Chelmsford, MA 01824

Controlled Circulation postage paid at:
Chelmsford, MA 01824

Publication Number: COTR 395770

Subscription in US: \$12.00/12 Issues

Entire contents copyright © 1979 by:
The COMPUTERIST®, Inc.

ADVERTISER'S INDEX

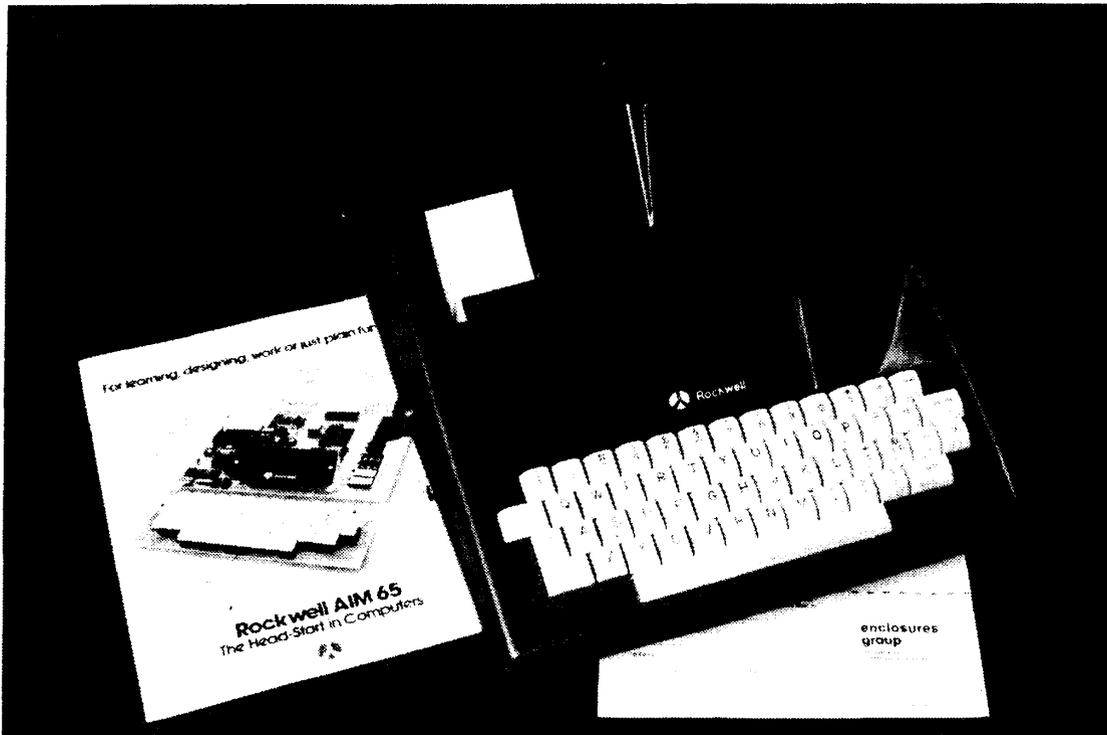
A B Computers	12	MICRO	45
Classified Ads	45	Optimal Technology, Inc.	26
Compas Microsystems	BC	P.S. Software House	47
Computer Forum	IFC	Plainsman Micro Systems	32
Computer Components	4	Programme International	48
The Computerist	8	Progressive Software	16
Connecticut Microcomputer	24	Seawell Marketing	28
Dr. Daley	27	Softside Software	IBC
Enclosures Group	2	Sybex	8
H. Geller Computer Systems	36	West Side Electronics	7
Hudson Digital Electronics	33		

Please address all correspondence, subscriptions and address changes to:

MICRO, P.O. Box 3, So. Chelmsford, MA 01824

MICRO

PERFECT AIM



ATTRACTIVE FUNCTIONAL PACKAGING FOR YOUR AIM-65 MICROCOMPUTER

- Professional Appearance
- Striking Grey and Black Color Combination
- Protects Vital Components

ENGINEERED SPECIFICALLY FOR THE ROCKWELL AIM-65

- All Switches Accessible
- Integral Reset Button Actuator
- Easy Paper Tape Replacement

EASILY ASSEMBLED

- Absolutely No Alteration of AIM-65 Required
- All Fasteners Provided
- Goes Together in Minutes

MADE OF HIGH IMPACT STRENGTH THERMOFORMED PLASTIC

- Kydex 100*
- Durable
- Molded-In Color
- Non-Conductive

AVAILABLE FROM STOCK

- Allow Three to Four Weeks for Processing and Delivery
- No COD's Please
- Dealer Inquiries Invited

TO ORDER: 1. Fill in this Coupon (Print or Type Please)
2. Attach Check or Money Order and Mail to:

NAME

STREET

CITY

STATE ZIP

Please Ship Prepaid SAE 1-1(s)
@ \$43.50 each
California Residents Please Pay
\$46.33 (Includes Sales Tax)

**enclosures
group**

753 bush street
san francisco, california 94108

*TM Rohm & Haas

Patent Applied For

IN THIS ISSUE ...

Andrew V. W. Sensicle increases the power of the basic KIM-1 with "EKIM OR MAXI-KIM", a small, page 17 monitor extension. This supports a PC "decrement" to compliment the normal "increment" function, "open up" and "close up" modes to move blocks of data to make room for adding code, and a "branch" calculator which simplifies determining the relative branch addresses.

Robert A. Stein, Jr. provides "A CASSETTE OPERATING SYSTEM FOR THE APPLE II" which makes it possible to maintain a library of programs which can be loaded by name from cassette. The article includes a cassette control circuit as well as the programs in assembler and BASIC to run the system.

Alan G. Hill presents "AN APPLE II PROGRAM EDIT AID" which helps the user locate all occurrences of any variable name, character string, or BASIC statement. The article includes a short assembler level program and a BASIC demo program.

J. Stelly makes it a lot easier to use the game of LIFE on your PET with his "LIFESAVER". This program supports creating a LIFE pattern, running LIFE at various rates, and saving and loading LIFE patterns on cassette.

Nicholas J. Vrtis helps overcome the SYM-1's KIM tape "2F" problem with a "CORRECTED KIM tape "2F" problem with a "CORRECTED KIM FORMAT LOADER FOR SYM-1". This program is carefully written with an interesting "frick" so that it does not itself contain a "2F" even though it must test for this troublesome character.

Bruce Hoyt comes through with a lot of good info on the OSI with "A CLOSE LOOK AT THE SUPERBOARD II". In addition to an overview, he presents a cassette save/hex memory dump program and a very useful table of memory usage.

Robert M. Tripp continues "ASK THE DOCTOR", a series on the AIM/SYM/KIM family of microcomputers, with a "Corrected AIM Sync Program", a "Patch for the AIM Disassembler", a "SYM Tape Evaluation", and "Comments on Synertek BASIC". Most of the info in this month's section has been provided by other ASK users.

"THE MICRO SOFTWARE CATALOG" continues with ten new entries.

John Gieryic has a tutorial article on a "SYM 6522-BASED TIMER" that gives insight into the workings of the 6522 VIA as well as the SYM.

Edward Chalfin has "THE TVT-6: A USER'S REPORT" which give his experiences and impressions of Don Lancaster's inexpensive method of getting a video signal out of a KIM-1.

William R. Dial continues to cover the expanding 6502 literature in his "6502 BIBLIOGRAPHY".

Don Rindsberg presents a major program in "THE ULTIMATE PET RENUMBER". This complete program can be used to rapidly renumber BASIC programs. The article also includes other useful info.

MICRO INTERRUPTS

The BEST of the PET GAZETTE has recently been published and should be of interest to all PET owners. It is available for \$9.95 from:

Microcomputer Resource Center, Inc.
1929 Northport Drive, Room 6
Madison, WI 53704

6502 COMPUTER GROUPS

The New England Apple Tree is now meeting on the third Wednesday of each month, 7 - 10:00 PM, at the cafeteria of the MITRE Corp. in Bedford, MA. You can contact, for further information:

Richard F. Sutor
166 Tremont Street
Newton, MA 02158

The Carolina Apple Core has been formed in the Research Triangle Area of North Carolina. The monthly meetings are on the third Tuesday of the month at different locations. Annual dues are \$5.00 and include a monthly newsletter. Contact

F. "Butch" Clayton, President
5212 Inglewood Lane
Raleigh, NC 27609
919/682-3756 or 596-8970

New York City now has an Apple users group: The Big Apple Users Group. Meetings are the second Tuesday of every month at the Computer Mart of Manhattan at 6:30 PM. For further info contact:

Neil Shapiro
34 Spencer Drive
Bethpage, NY 11714
516/579-4295 (home)
212/262-4808 (office)

The Apple Corps of San Diego is publishing an eight page newsletter. Unfortunately, by the time it reaches us, the information on the next meeting is too dated for us to print. The person to contact for information is:

Phillip A. Lemon
5485 Repecho Drive, L=108
San Diego, CA 92124
714/560-7962

***** ATTENTION ALL 6502 CLUBS *****

Now that MICRO is published monthly, we can get the word out on when and where you are meeting - if you get the word in to us. We need times and dates and places by the first of the preceding month - April 1 for the May issue and so forth. Also, please put us on your mailing list for any newsletter or other material you send out. We want to help your club prosper by giving it as much exposure as possible, but we need your input to make it happen.

*** On The Cover ***

With all of the new 6502 based microcomputers, it is easy to forget about the KIM-1 which was the first 6502 system. Many thousands have been sold, and after a period of production problems, the quality of the KIM-1 has been remarkably improved recently. Considering all of the articles we continue to receive about the KIM-1, it looks as though this system is here to stay for a long time.



SYSTEMS*

Apple II 16K RAM \$1195⁰⁰ • Commodore PET 8K RAM \$795⁰⁰ • Commodore KIM I \$159⁰⁰
• Microproducts Super KIM \$395⁰⁰

*Delivery on most systems is usually stock to 2 weeks. Call or write for specific information.

**16K RAM CHIP SET FOR APPLE II
ONLY (Tested & Burned In) \$95⁰⁰**

WORKSHOPS: Call for details.

- PET—3rd Saturday of the Month
- APPLE—4th Saturday of the Month

CLASSES: Apple Topics

We offer a series of classes on Apple II to acquaint owners with some of the unique features and capabilities of their system. Topics covered are Apple Sounds, Low Res. Graphics, Hi Res. Graphics, Disk Basics, and How to Use Your Reference Material. Sessions are held every Thursday Night at 7:00 p.m.

Reference Books For APPLE and PET Owners

- Programming the 6502 9.95
- PET User Manual (New from Commodore) 9.95
- First Book of KIM 8.95
- MOS Tech Programming Manual (6502) 12.00
- MOS Tech Hardware Manual 12.00

SOFTWARE

We now have a complete software catalog.

APPLE:

APPLE 21	9.95
Battlestar I	15.95
Income Tax	19.95
Super Star Wars	15.95
Appletalker*	\$15.95
Bomber*	9.95
Apple-Lis'ner*	19.95
Talking Calculator	12.95
Warlords	12.95
Escape	17.95
Tank War	12.95
Phasor Zap	15.00
Depth Charge	15.00
3-D Docking Mission	14.95
Microchess	19.95
Ron Graff's Educational Programs	(call for details)

ON DISK:

Editor / Assembler	19.95
3-D Animation	24.95
Talking Disk	19.95
General ledger	60.00
Check Book	35.00
Inventory System	125.00
Text Editor	50.00
Mailing List	30.00
Single Disc Copy	19.95
Memo Calendar	24.95
Electronic Index Card File*	19.95

*Programs by Bob Bishop

PET:

Pet Show	9.95
Galaxy Games	9.95
Space Talk & Space Fight	9.95
Space Trek	9.95
Finance	\$9.95
Microchess	19.95
Casino Pac (3 Games)	9.95
Off The Wall / Target Pong	9.95
Mortgage	14.95
Diet Planner / Biorythm	14.95
Basic BASIC	14.95

HARDWARE

APPLE II HARDWARE:

- **Modem for Apple II, ready to go**
Plug into telephone wall plug \$379.00
- **Upper & Lower Case Board**
Now you can display both upper and lower case characters on your video with the Apple II. Includes assembled circuit board and sample software \$49.95
- **Programmer Aide** \$50.00

PET HARDWARE

- **Beeper** \$24.95
- **Petunia**—for computer generated sounds \$29.95
- **Video Buffer**—to put your pet's pictures on a television set or monitor \$29.95

PRINTER SPECIALS FOR APPLE AND PET

- **TRENDCOM 100** with interface for Apple or PET \$405.00
- | | |
|---------------------------|-----------------------------------|
| 40 characters per second | Bidirectional look ahead printing |
| Microprocessor controlled | Low cost thermal paper |
| 96 character set | 4" x 80 ft. roll \$2.50 |
| High Reliability | Clear 5 x 7 characters |
| Quiet Operation | 8 bit parallel input |
| Sturdy metal/plastic case | No external power supply |
| | 40 characters per line |

Introducing

- **Anadex DP-8000** with tractor
8" paper width and Apple interface \$1050
- **Centronics 779-2 for Apple II**
With parallel interface \$1245.00

**See if you qualify for a CCI of OC P / F Card
and get great discounts on selected
purchases for your Apple and PET.**

WHY SHOULD YOU BUY FROM US?

Because we can help you solve your problems and answer your questions. We don't claim to know everything, but we try to help our customers to the full extent of our resources.

—Prices subject to change.—

COMPUTER COMPONENTS OF ORANGE COUNTY

6791 Westminster Ave., Westminster, CA 92683 714-898-8330

Hours: Tues-Fri 11:00 AM to 8:00 PM—Sat 10:00 AM to 6:00 PM (Closed Sun, Mon)

Master Charge, Visa, B of A are accepted. No COD. Allow 2 weeks for personal check to clear.
Add \$1.50 for handling and postage. For computer systems please add \$10.00 for shipping, handling and insurance. California residents add 6% Sales Tax.

AN APPLE II PROGRAM EDIT AID

Alan G. Hill
12092 Deerhorn Dr.
Cincinnati, OH 45240

When editing an Apple Integer Basic program, you often want to locate all occurrences of a variable name, character string, or BASIC statements. This is usually the case when you are changing a variable name, moving a subroutine, etc., and you want to be sure you have located all references. The BASIC Edit program presented here should aid your editing.

The BASIC program should be loaded into high memory and the program to be edited appended to it. The Edit program uses a machine language routine at hex 300 to 39F to search BASIC statements for the requested string and return the BASIC line number in memory locations 17 and 18. The routine is re-entered at 846 to find the line number of the next occurrence. This process is continued until no further occurrences can be found. The high order byte of the line number (location 18) is set to hex FF to indicate that the search is finished.

BASIC Edit Program

Note in line 32680 of the BASIC program that LIST LINE is an invalid BASIC statement. You will have to resort to a little chicanery to get the statement in. First code line 32680 as PRINT LINE. Then, enter the monitor and change the PRINT token (\$62) to a LIST token (\$74). This is easiest done if you code line 32680 first and then search for the token in high memory (\$3FFA when HIMEM is 16384).

After coding the BASIC program and the machine language routine, you will then need to append the program to be edited. Note that the program must have line numbers less than 32600. To append a program, you must first "hide" the Edit program. This is done by moving the HIMEM pointer (202) and (203) down below the Edit program. Then load the edited program and reset HIMEM: i.e.:

```
LOAD (EDIT PROGRAM)
POKE 76, PEEK (202)
POKE 77, PEEK (203)
LOAD (PROGRAM TO BE EDITED)
POKE 76,0 HIMEM MOD 256
POKE 77,64 HIMEM/256
```

FIND ROUTINE

A. G. HILL
MARCH 1979

```
HILO * $0003 HIMEM LO BYTE
HIHI * $0004 HIMEM HI BYTE
BSL * $0006 BASIC STATEMENT LO
BSH * $0007 BASIC STATEMENT HI
SEAL * $0008 STATEMENT ENDING ADDRESS LO
SEAH * $0009 STATEMENT ENDING ADDRESS HI
STRL * $000A STRING LO
LNL * $0011 LINE NUMBER LO
LNH * $0012 LINE NUMBER HI
```

You can then RUN 32600 the Edit program. Enter the character string or variable name to be searched when prompted by "FIND?". To search for a hex string (e.g. all occurrences of COLOR=), enter an @ character followed by the desired hex character pair (@66 for the COLOR= example)

EXAMPLES

To find all occurrences of:	Input
SCORE	SCORE
XYZ	XYZ
RETURN	@5B
DIM A	@4EC1
All references to 1000	@E803

The Edit program will end if the screen is full (> 18 lines). To continue the search for more occurrences, a RUN 32720 will return another page. Happy Editing!

Find Routine

Page Zero Memory Map

\$3-4	Address of search limit. Set to HIMEM by routine, but could be set lower to avoid searching Edit program.
\$6-7	Address of BASIC Token compared. Incremented until it exceeds Limit Address
\$8-9	Ending address - 1 of current statement being scanned
\$A-B	Address of string being searched. Set up by Edit program
\$ C	Length - 1 of string being searched. Set up by Edit program
\$11-12	Line number of statement containing the requested string. \$12 is set to \$FF if no more occurrences

```

0300          ORG      $0300

0300 A5 CA      START  LDA   $00CA  SET UP ADDRESS OF FIRST
0302 85 06          STA   BSL     BASIC STATEMENT IN
0304 A5 CB          LDA   $00CB  LOCS 6 AND 7
0306 85 07          STA   BSH
0308 A5 4C          LDA   $004C  SET UP TO STOP SEARCH
030A 85 03          STA   HILO   AT HIMEM.  COULD BE
030C A5 4D          LDA   $004D  CHANGED TO LIMIT SEARCH
030E 85 04          STA   HIHI   AT END OF PROGRAM BEING EDITED
0310 A0 00          LENGTH LDYIM $00  GET STATEMENT LENGTH
0312 B1 06          LDAIY BSL
0314 38            SEC
0315 E9 02          SBCIM $02   MINUS 2 TO POINT TO
0317 18            CLC        LAST TOKEN IN STATEMENT
0318 65 06          ADC   BSL
031A 85 08          STA   SEAL   SET UP STATEMENT ENDING
031C A5 07          LDA   BSH     ADDRESS IN 8 AND 9
031E 69 00          ADCIM $00   ADD IN CARRY IF ANY
0320 85 09          STA   SEAH
0322 A0 01          LDYIM $01   SAVE LINE NUMBER IN
0324 B1 06          LDAIY BSL   IN 11 AND 12
0326 85 11          STA   LNL
0328 C8            INY
0329 B1 06          LDAIY BSL
032B 85 12          STA   LNH
032D A2 00          LDXIM $00   ADJUST BSL TO POINT
032F A9 03          LDAIM $03   TO FIRST TOKEN
0331 20 64 03      JSR   INCPNT
0334 A0 00          LDYIM $00   COMPARE TOKEN TO
0336 B1 06          TTOKEN LDAIY BSL  FIRST CHARACTER IN
0338 D1 0A          CMPIY STRL  STRING
033A D0 03          BNE   NXTOKN  IF NOT EQUAL POINT TO NEXT
033C 20 7F 03      JSR   COMPAR  IF EQUAL COMPARE REMAINING CHARS
033F 20 70 03      NXTOKN JSR   INCTOK  POINT TO NEXT TOKEN
0342 90 F2          BCC   TTOKEN  CARRY CLEAR THEN LOOK AT NEXT
0344 A5 08          LDA   SEAL   AT END OF STATEMENT.
0346 C5 03          CMP   HILO   CHECK TO SEE IF AT END OF
0348 A5 09          LDA   SEAH   SEARCH LIMIT
034A E5 04          SBC   HIHI
034C B0 11          BCS   LIMIT  CARRY SET = LIMIT OF SEARCH
034E A5 08          LDA   SEAL   SET UP BSL AND BSH TO POINT
0350 85 06          STA   BSL   TO NEXT STATEMENT
0352 A5 09          LDA   SEAH
0354 85 07          STA   BSH
0356 A2 00          LDXIM $00   POINT TO LENGTH OF
0358 A9 02          LDAIM $02   STATEMENT BYTE
035A 20 64 03      JSR   INCPNT
035D D0 B1          BNE   LENGTH  ALWAYS BRANCH

035F A9 FF          LIMIT  LDAIM $FF  SET UP LARGE LINE NUMBER
0361 85 12          STA   LNH   TO INDICATE AT END OF SEARCH
0363 60            RTS        RETURN TO BASIC

```

0364 18	INCPNT	CLC		ROUTINE TO INCREMENT
0365 75 06		ADCX	BSL	POINTERS. ENTER WITH
0367 95 06		STAX	BSL	XREG = DISPLACEMENT
0369 B5 07		LDAX	BSH	FROM
036B 69 00		ADCIM	\$00	BSL, BSH
036D 95 07		STAX	BSH	ACC = INCREMENT AMOUNT
036F 60		RTS		
0370 A5 06	INCTOK	LDA	BSL	ROUTINE TO INCREMENT
0372 C5 08		CMP	SEAL	THE TOKEN ADDRESS BY 1
0374 A5 07		LDA	BSH	SET CARRY IF AT END
0376 E5 09		SBC	SEAH	OF STATEMENT
0378 E6 06		INC	BSL	
037A D0 02		BNE	REXIT	
037C E6 07		INC	BSH	
037E 60	REXIT	RTS		
037F A4 0C	COMPAR	LDY	\$000C	ROUTINE TO COMPARE
0381 B1 0A	COMPY	LDAIY	STRL	REMAINING CHARACTERS
0383 D1 06		CMPIY	BSL	(C) LENGTH OF CHARACTER
0385 F0 03		BEQ	COMPX	STRING -1
0387 A0 00		LDYIM	\$00	RESET YREG
0389 60		RTS		
038A 88	COMPX	DEY		
038B 10 F4		BPL	COMPY	FOUND A MATCH! POP STACK ADDRESS
038D 68		PLA		AND RETURN TO BASIC. LINE NUMBER
038E 68		PLA		IS ALREADY IN LNL AND LNH.
038F 60		RTS		

BASIC EDIT PROGRAM

```

32600 DIM A$(30)
32610 INPUT "FIND?",A$: CALL -936:
IF A$(1,1)='@' THEN 32630:
KK=LEN(A$): FOR I=1 TO KK:
POKE 911+I,ASC(A$(I,I)): NEXT I
32620 POKE 12,KK-1: GOTO 32650
32630 A$=A$(2,LEN(A$)): KK=LEN(A$):
FOR I=1 TO KK STEP 2:
I=ASC(A$(I,I))-176:
JJ=ASC(A$(I+1,I+1))-176
32640 IF J>9 THEN J=J-7:
IF JJ>9 THEN JJ=JJ-7:
POKE 912+I/2,J*16+JJ: NEXT I:
POKE 12,KK/2-1
32650 POKE 10,912MOD256: POKE 11,912/256
32660 CALL 768
32670 IF PEEK(18)>127 THEN 32730:
LINE=PEEK(17)+PEEK(18)*256
32680 LIST LINE
32690 IF PEEK(37)>18 THEN 32730
32700 CALL 846
32710 GOTO 32670
32720 CALL -936: GOTO 32700
32730 END

```

APPLE II® OWNERS:

West Side Electronics introduces the
APPLETIME™, a Real Time Clock for the Apple II

The Appletime (Model APT-1) is a single peripheral board which plugs directly into any I/O slot on the Apple II. Timing is done completely in hardware (ie. NOT an interrupt driven clock). Thus, the Appletime continues to operate even when the computer is turned off. Our exclusive Three Way Power System keeps the clock running via its own AC supply, the computer's, or battery backup in case of power failure. Other features include 12/24 Hour selection, AC or crystal timebase, 50/60 Hz, and BCD or ASCII data format. Fully assembled and tested, with instructions and

SPECIAL INTRODUCTORY PRICE

APT-1 Real Time Clock..... \$59.95

Proto-board for Apple II. Over 1300 holes on 0.1 inch centers for designing your own circuits.

APB-1 Prototyping Board..... \$16.95

★ ★ SUPER SPECIAL ★ ★

This month only, order the APT-1 and the APB-1 at a special combination price of..... **\$69.95**



WEST SIDE ELECTRONICS
P.O. Box 636
CHATSWORTH, CA. 91311



We pay all shipping in Continental U.S.A.
Others add 10%; California residents add 6% tax.

AIM PLUS[™]

ENCLOSURE
WITH BUILT IN
POWER SUPPLY



SPECIFICATIONS:
INPUT: 110/220 VAC 50/60 Hz
OUTPUT: +5V @ 5A
+24V @ 1A

GROUNDING THREE WIRE LINE CORD
ON/OFF SWITCH WITH PILOT LIGHT
Enclosure has room for the AIM and one
additional board: MEMORY PLUS or VIDEO PLUS

AIM PLUS: \$100⁰⁰ AIM and AIM PLUS: \$475⁰⁰

VIDEO PLUS[™] FOR AIM/SYM/KIM



UPPER/lower case ASCII
128 Additional User Programmable
Characters: GRAPHICS
SYMBOLS-FOREIGN CHARACTERS
Programmable Screen Format up to
80 CHARACTERS - 24 LINES

KEYBOARD and LIGHT PEN Interfaces
Up to 4K DISPLAY RAM
Provision for 2K EPROM
Provision to add 6502 for
STAND-ALONE SYSTEM

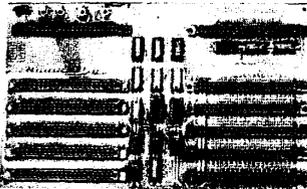
ASSEMBLED AND TESTED
WITH 2K DISPLAY RAM

VIDEO PLUS: \$245⁰⁰

MOTHER PLUS[™] FOR AIM/SYM/KIM

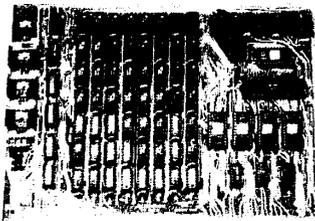
ADD UP TO FIVE ADDITIONAL BOARDS
AUDIO/TTY CONNECTIONS
POWER TERMINALS
APPLICATION CONNECTORS

FULLY BUFFERED
FULLY DECODED
KIM-4 Bus Structure



MOTHER PLUS: \$80⁰⁰ FULLY ASSEMBLED AND TESTED

MEMORY PLUS[™] FOR AIM/SYM/KIM



8K STATIC RAM LOW POWER
Sockets for 8K Eprom
6522 1/0 Port
ON BOARD REGULATORS
EPROM PROGRAMMER

MEMORY PLUS: \$245⁰⁰ FULLY ASSEMBLED AND TESTED

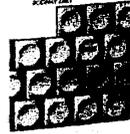


P.O. Box 3, So. Chelmsford, MA 01824
617/256-3649

SYBEX

LEADER IN
MICROCOMPUTER EDUCATION

AN INTRODUCTION
TO PERSONAL
AND BUSINESS
COMPUTING
RODNEY ZAKS



AN INTRODUCTION TO PERSONAL
AND BUSINESS COMPUTING
by Rodney Zaks

250 pp, ref C200 \$6.95

The basic introductory text
on microcomputers, with a
detailed evaluation of the fea-
tures and peripherals required
for specific applications. No
prior computer knowledge
required.

MICROPROCESSORS: from Chips
to Systems

by Rodney Zaks

420 pp, ref C201 \$9.95

An educational text, used
worldwide at universities and
in industry designed to teach
all the fundamentals of mi-
croprocessors, the assembly
of a system, and its use.



PROGRAMMING THE 6502

by Rodney Zaks

320 pp, ref C202 \$10.95

An introductory program-
ming text for the 6502. Does
not require any prior pro-
gramming knowledge. From
arithmetic to interrupt-driven
input-output techniques.

6502 APPLICATIONS BOOK

by Rodney Zaks

ref D302 \$12.95

Actual application programs
to interface the 6502 to
the real world, from LED to
motor, and analog-digital
conversion. Available Shortly

MICROPROCESSOR
INTERFACING
TECHNIQUES
RODNEY ZAKS



MICROPROCESSOR
INTERFACING TECHNIQUES

Austin Lesea and Rodney Zaks

416 pp, ref C207 \$11.95

All the basic interfacing
techniques, from keyboard to
floppy disk, including the
standard buses (S100 to
1EEE488).

MICROPROCESSOR LEXICON

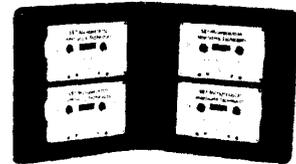
120 pp, ref X1 \$2.95

Dictionary and tables. All the
definitions of the micropro-
cessor world in a pocket
book format.

MICROPROGRAMMED APL
IMPLEMENTATION

330 pp, ref Z10 \$25.00

How to design an APL
interpreter.



SELF STUDY COURSES ON
CASSETTES

Ten courses to study at home
or in the car. The most time-
efficient way to learn. Includes
workbook and cassettes.

INTRODUCTORY \$29.95 ea

S1-INTRODUCTION TO MI-
CROPROCESSORS (2.5 hrs)

S2-PROGRAMMING MICRO-
PROCESSORS (2.5 hrs)

COMPREHENSIVE \$59.95ea

SB1-MICROPROCESSORS

(12 hrs)

SB2-MICROPROCESSOR

PROGRAMMING (10 hrs)

SPECIALIZED \$49.95

SB7-MICROPROCESSOR

INTERFACING (6 hrs)

TO ORDER

By phone: 415 848-8233, Visa, MC,
Amer Express

By mail: circle books on ad. Include
payment.

Shipping: add 65¢ per book (4th class)
or \$1.50 faster shipping (UPS).

Double for cassettes and overseas.

Tax: in California add tax.

FREE DETAILED CATALOGUE

SYBEX

2020 Milvia Street
Berkeley, CA 94704
Tel 415 848-8233 Telex 336 311

LIFESAVER
by J. Stelly
10918 Dunvegan Way
Houston, TX 77013

Is LIFE passing you by; does it progress so quickly than there is little time to enjoy it? Well, fear not—the LIFESAVER is here. Though time marches on, now you are in control. If you got "LIFE For Your PET" from Dr. Frank H. Covitz (**The Best of Micro**, p.65), LIFE moves along at a pretty good clip. LIFESAVER is a BASIC program that complements and provides some enhancements to Dr. Covitz machine language routines.

LIFESAVER provides a convenient grid for setting up cellular patterns, permits saving and loading of patterns on the built in cassette unit, and gives complete control of the time interval between generations. You may even single step through the LIFE sequences.

Commodore is supposedly mailing all owners of early model PET units the TIM monitor on cassette, so I will assume its availability in this discussion. It ain't the best monitor in the world, but it does allow you to load machine language programs directly from the cassette without any special loader routines. This does not exclude other methods the reader may have at his (or her) disposal if TIM is not available.

A single modification to Dr. Covitz program is required before it can be used with LIFESAVER. Location 191D (16) should be changed to read:

191D 60 RTS

When this change is made the program may be entered at 190A(16) e.g. SYS(6410). If the TIM monitor is used, simply do a hex dump of the machine language listing and save the program on tape using the instructions given in the manual.

Before loading LIFE (Dr. Covitz program) or LIFESAVER (by yours truly) from cassette, I recommend the following command be executed:

POKE 134,0:POKE 135,24

This lowers the BASIC boundary and prevents conflicts between the two programs. The regular BASIC limit can later be reinstated by POKE 135,32. It is also a good idea to load LIFE before LIFESAVER is loaded. This prevents the data pointer from getting initialized to the wrong location.

It may be possible to eliminate lines 3015 and 3035 from the BASIC listing, if you have a relatively late model PET. These lines are necessary for the older units that have a problem with writing file headers and cassette motor start/stop control. My unit was delivered in Sept. '78 and I was able to eliminate these lines.

Assuming that both LIFE and LIFESAVER have successfully been loaded, you may begin entering your favorite cell patterns. Please refer to Dialog 1 (human inputs are underlined) to see how this is done. After the grid is printed simply press the 'RETURN' key and enter your pattern anywhere in the grid area using the cursor keys and the dot (•) symbol above the Q key. After you've created the desired pattern press the 'HOME' key and the 'RETURN' key in

succession. This neat little trick returns control to the LIFESAVER routine without having to explicitly key in the command 'GOTO 1000'. After the PET has saved the pattern internally the user then has the options to save it on tape, have the computer generate LIFE patterns as described in Dr. Covitz article, or scrap it and input a new pattern.

The options are relisted after the execution of any LIFESAVER command. Examples on exercising the different options are given in the remaining dialogs.

LIFESAVER should relieve the user from the tedium of having to manually reenter a LIFE pattern every time it is desired to run it. It should also encourage the user to experiment with various LIFE forms, some of which are quite dazzling.

DIALOG 1

RUN

LIFE

PLEASE CHOOSE AN OPTION

1. CREATE A PATTERN
2. RUN LIFE GENERATOR
3. LOAD A PATTERN FROM CASSETTE
4. SAVE A PATTERN ON CASSETTE

OPTION NUMBER ? 1 (RETURN)

(SCREEN CLEARS, THEN ...)

GOTO 1000 ?

(At this point the user hits the RETURN key and proceeds to input a cell pattern.)

GOTO 1000 ?

READY					
		•	•		
		•	•	•	
		•	•	•	

(With the desired pattern on the CRT the user presses the HOME and RETURN keys to resume program execution.)

STORING CELL PATTERN

(After a slight delay the computer again responds with the option list.)

LIFE

.

.

(Option List)

.

.

OPTION NUMBER ? 2 (RETURN)

(Screen clears ...)

HOW MANY GENERATIONS ? 7

DEVELOPMENT RATE

0:SINGLE STEP VIA (G) KEY

1-99:INTERMEDIATE RATES

100:MAX (255 GENERATION LIMIT)

RATE ? 75

(The computer proceeds to display generations sequentially at the specified rate. The larger the numerical value of the rate the faster the generations are produced. A rate of 0 means that only one generation is produced at a time. The G key must be pressed to obtain subsequent generations.)

LISTING

```

1  REM LIFESAVER
2  REM BY JAMES W. STELLY
3  REM POKE 135,24 BEFORE USING
100 DIM A$(25)
110 PRINT "clrLIFE":PRINT
120 PRINT "PLEASE CHOOSE AN OPTION:":PRINT
130 PRINT "1. CREATE A PATTERN"
140 PRINT "2. RUN LIFE GENERATOR"
150 PRINT "3. LOAD A PATTERN FROM CASSETTE"
160 PRINT "4. SAVE PATTERN ON CASSETTE"
170 INPUT "OPTION NUMBER";N
180 ON N GOSUB 200,2000,4000,3000
190 GOTO 110

      CREATE GRID FOR PATTERN INPUT
200 PRINT "clr cd";
210 FOR I=1 TO 5
220 PRINT "┌-----┐"
230 PRINT "│"
240 PRINT "│"
250 PRINT "│:"
260 NEXT I
270 PRINT "└-----┘"
280 PRINT "│"
290 PRINT "│"
300 PRINT "└-----┘"
310 INPUT "home GOTO 1000";A$

```

LIFE

.

.

(Option list)

.

.

OPTION NUMBER ? 4

(Screen clears ...)

HOW MANY

PATTERN NAME ? CHESIRECAT (RETURN)

(Pattern is saved and the option list is printed.)

NOTE: In the following BASIC listing the lower case abbreviations stand for cursor control keys and have the following meaning:

```

clr = clear screen
home = home up
cd = cursor down
s = space key

```

STORE PATTERN

```
1000 PRINT "homeSTORING CELL PATTERN"  
1010 FOR I=1 TO 24:A$(I)="":NEXT I  
1020 FOR I=1 TO 24:FOR J=1 TO 39  
1030 IF PEEK(32767+J+(I*40))= 81 THEN A$(I)=A$(I)+"●":GOTO 1050  
1040 A$(I)=A$(I)+"-"  
1050 NEXT J:NEXT I  
1060 RETURN
```

ACCESS LIFE GENERATOR

```
2000 INPUT "clrHOW MANY GENERATIONS";G  
2010 PRINT "cdDEVELOPEMENT RATE:":PRINT  
2020 PRINT "0;SINGLE STEP VIA (G) KEY"  
2030 PRINT "1-99:INTERMEDIATE RATES"  
2040 PRINT "100:MAX (255 GENERATIONS LIMIT)"  
2050 INPUT "cdRATE";S  
2060 PRINT "clrGEN 0"  
2070 FOR I=1 TO 23:PRINT A$(I): NEXT I  
2075 PRINT A$(I);:FOR I=1 TO 2000:NEXT I  
2080 IF S=100 THEN POKE 6483,256-G:SYS(6410):GOTO 2140
```

INTERMEDIATE RATES

```
2100 POKE 6483,255:IF S=0 GOTO 2160  
2110 S=100-S:FOR I=1 TO G  
2120 SYS(6410):PRINT "homeGEN";I  
2130 FOR J=1 TO S*30:NEXT J:NEXT I  
2140 GET A$:IF A$<>"X" GOTO 2140  
2150 RETURN
```

SINGLE STEP

```
2160 G=1  
2170 SYS(6410):PRINT "homeGEN";G  
2180 GET A$: IF A$="X" THEN RETURN  
2190 IF A$="G" THEN G=G+1: GOTO 2170  
220 GOTO 2180
```

SAVE PATTERN

```
3000 INPUT "clrPATTERN NAME";A$  
3010 OPEN 1,1,1,A$  
3015 POKE 243,122:POKE 244,2  
3020 FOR I=1 TO 24  
3030 PRINT#1,A$(I)  
3035 POKE 59411,53  
3040 NEXT I  
3050 CLOSE 1:RETURN
```

LOAD PATTERN

```
4000 INPUT "clrPATTERN NAME";A$  
4010 OPEN 1,1,0,A$  
4020 FOR I=1 TO 24:INPUT#1,A$(I):NEXT I  
4030 CLOSE 1:RETURN
```

CORRECTED KIM FORMAT LOADER FOR SYM-1

Nicholas J. Vrtis
5863 Pinetree S.E.
Kentwood, MI 49508

My cassette is an old model GE, and it won't quite hack the high speed tape format of the SYM-1, so I have probably used the KIM format option more than most SYM owners. In the process, I have found a bug in the SYM monitor tape load routine. Synertek knows about the problem, but didn't have a nice fix when I called, so I worked up the attached program.

The problem with the monitor routines is that they will not load a slash (hex 2F) from a KIM format tape. The slash is used to indicate that the data is done, and the checksum follows. The monitor routines don't check for the slash until after the KIM characters have been read and combined. The error you get is a checksum error (ER CC).

Most of the code for this program has been copied from the SYM monitor routines, except these work. The basic logic change is that when a slash is read as a single KIM byte, it is treated as a non-hex

character. The non-hex routine checks for the slash instead of after every character. If it is a slash, it goes to the checksum check routine.

This routine is not as fancy as the monitor routines, but it sure beats re-keying a couple K bytes of program. It has turned out to be convenient to have this program available even for loading programs without the slash. By changing the branch after the compare for the slash to a branch back to LOADT7 it will ignore errors. Sometimes this will load a bad tape with only minor errors. Other times the program gets out of sync and loads garbage. It is worth the try for a tape you have spent a lot of time on.

One final comment about cassettes. If you have the remote control connected, putting a hex CC into location A00C will turn the cassette motor back on. It is easier than yanking the remote plug.

PET 8K Introductory Special
Includes software..... \$795.
and sound package

KIM-1 161.
SYM 238.

KL512 Power Supply with KIM or SYM
for KIM, SYM and extra memory \$30.00

KM8B Problem Solver Systems
8K Static RAM for KIM, SYM, AIM 149.00

KIM-4 Motherboard 95.00

Memory Plus - the Computerist
8K RAM, sockets for 8K EPROM,
EPROM Programmer, timers, and I/O 238.00

Enclosures from Enclosure Group

SYM \$36.00
KIM 23.00

North Star **Horizon 2** Double
Density \$1649.

CASSETTE TAPES

Premium quality, low noise in 5 screw housing with labels
ALL TAPES GUARANTEED

	10	50	100
C-10	5.95	26.00	48.00
C-30	7.00	31.00	57.00

Norelco hinged hard boxes 10/1.25
Soft poly plastic boxes 10/1.00

2716 EPROM (Intel + 5 Volt) *38.00
2114L 450ns 4K Static RAM 6.95
Programming a Microcomputer:
6502 Foster 8.95
6500 Programming Manual 6.50
6500 Hardware Manual 6.50
First Book of KIM 9.00
Microcomputer Programming:
6502 R. Zaks 9.00

New **PET 16K \$ 995**
PET 32K \$1195
special includes software and sound package

Write for list of 6502 and S-100 products

AB Computers

P.O. Box 104
Perkasie, PA 18944
(215) 257-8195

FIXED SYM-1 KIM FORMAT LOADER

NICHOLAS J. VRTIS
MARCH 1979

STRIPPED DOWN VERSIONS OF L1 COMMAND.
WILL LOAD A 2F WHICH CAUSES SYM-1 TROUBLE.
ONLY FOR KIM FORMAT TAPES.
ID SHOULD BE PUT INTO LOCATION 0000.

0080	CHAR	*	\$00FC	CHAR ASSEMBLY & DISASSEMBLY
0080	MODE	*	\$00FD	
0080	BUFADL	*	\$00FE	CURRENT CHAR INDIRECT ADDRESS
0080	BUFADH	*	\$00FF	

SYM-1 REFERENCES

0080	DDRIN	*	\$A002
0080	VIAACR	*	\$A00B
0080	LATCHL	*	\$A004

0080	ACCESS	*	\$8BA6	
0080	SLASH	*	\$8D3C	SLASH IN SYM MONITOR
0080	LOADTX	*	\$8D4F	
0080	NHERR	*	\$8D69	
0080	SYNC	*	\$8D82	
0080	START	*	\$8DB6	
0080	RDBYTX	*	\$8E28	
0080	PACKT	*	\$8E3E	
0080	RDCHT	*	\$8E61	
0080	CHKT	*	\$8E78	

0000		ORG	\$0000
------	--	-----	--------

0000	00	ID	=	\$00	RESERVED FOR PROGRAM ID
------	----	----	---	------	-------------------------

0001	20	A6	8B	LOADT	JSR	ACCESS	UN-PROTECT SYSTEM RAM
0004	A0	00			LDYIM	\$00	SET KIM MODE
0006	20	B6	8D		JSR	START	INITIALIZE
0009	AD	02	A0		LDA	DDRIN	
000C	29	BF			ANDIM	\$BF	BIT 6 = 0 INPUT IS PB6
000E	8D	02	A0		STA	DDRIN	
0011	A9	00			LDAIM	\$00	
0013	8D	0B	A0		STA	VIAACR	
0016	A9	AE			LDAIM	\$AE	SET UP CLOCK FOR GETTR (KIM)
0018	8D	04	A0		STA	LATCHL	STORE GETTR VALUE IN LO LATCH
001B	20	82	8D	LOADTA	JSR	SYNC	GET IN SYNC
001E	20	61	8E	LGADTB	JSR	RDCHT	
0021	C9	2A			CMPIM	'*	START OF DATA ?
0023	F0	06			BEQ	LOADTC	
0025	C9	16			CMPIM	\$16	NO - SYNC CHARACTER?
0027	D0	F2			BNE	LOADTA	IF NOT, RESTART SYNC SEARCH
0029	F0	F3			BEQ	LOADTB	IF YES, KEEP LOOKINT FOR THE *

002B A9 00 LOADTC LDAIM \$00 CLEAR "NOT IN SYNC BIT"
 002D 85 FD STA MODE

002F 20 28 8E JSR RDBYTX READ ID BYTE

CHANGE THE FOLLOWING IF ID LOCATION IS
 NOT HEX 0000

0032 C5 00 CMP ID COMPARE WITH REQUESTED ID
 0034 F0 02 BEQ LOADTD GO LOAD IF EQUAL
 0036 D0 E3 BNE LOADTA UNCONDITIONAL - RESTART SEARCH

0038 20 28 8E LOADTD JSR RDBYTX GET SAL FROM TAPE
 003B 20 78 8E JSR CHKT
 003E 85 FE STA BUFADL PUT IN BUF START LOW
 0040 20 28 8E JSR RDBYTX SAME FOR SAH
 0043 20 78 8E JSR CHKT
 0046 85 FF STA BUFADH

THE FOLLOWING JSR RDBYT IS THE ONLY
 INSTRUCTION THAT WOULD HAVE TO CHANGE
 TO RE-LOCATE THIS PROGRAM

0048 20 67 00 LOADTE JSR RDBYT GET A BYTE INPUT
 004B B0 0F BCS XNHERR BRANCH IF NON-HEX
 004D 20 78 8E JSR CHKT INCLUDE IN CHECKSUM
 0050 A0 00 LDYIM \$00 STORE BYTE
 0052 91 FE STAIY BUFADL
 0054 E6 FE INC BUFADL BUMP BUFFER ADDRESS
 0056 D0 F0 BNE LOADTE BRANCH IF NO CARRY
 0058 E6 FF INC BUFADH ELSE NEED TO UPDATE HIGH ORDER
 005A D0 EC BNE LOADTE UNCONDITIONAL

005C CD 3C 8D XNHERR CMP SLASH "/" IN SYM MONITOR
 005F D0 03 BNE YNHERR WAS IT REALLY AN ERROR
 0061 4C 4F 8D JMP LOADTX NOW LET HIM HANDLE CHECKSUM
 0064 4C 69 8D YNHERR JMP NHERR LET MONITOR DO THIS ALSO

0067 20 61 8E RDBYT JSR RDCHT READ ONE HALF
 006A CD 3C 8D CMP SLASH SEE IF A SLASH
 006D D0 02 BNE RDBYTA BRANCH IF NOT
 006F 38 SEC SET CARRY AS NON-HEX
 0070 60 RTS AND RETURN

0071 20 3E 8E RDBYTA JSR PACKT SEE IF GOOD CHARACTER
 0074 90 01 BCC RDBYTB BRANCH AROUND RETURN IF HEX
 0076 60 RTS

0077 AA RDBYTB TAX SAVE MSD
 0078 20 61 8E JSR RDCHT GET NEXT HALF CHARACTER
 007B 86 FC STX CHAR SAVE IT HERE
 007D 4C 3E 8E JMP PACKT CHECK FOR HEX & RETURN

A CLOSE LOOK AT THE SUPERBOARD II

Bruce Hoyt, Pastor
Sharon Associated Reformed Presbyterian Church
Route 1
Brighton, TN 38011

Late in December 1978 my dreams came true. Those dreams I had had in the mid 60's when I first learned how to program computers. I had dreamed of having my own desk-sized computer. That dream has come true to a degree I would not have thought possible then. The computer I now have is not desk-sized but is contained on one printed circuit board. Furthermore it is more powerful than the big monsters I worked on in the mid 60's. I don't want to bore you with a description of my continual amazement at a computer on a chip for such things are now old hat. Nor do I want to give just a general overview of the Superboard II manufactured by Ohio Scientific. For a general description you may check the March 1979 issue of **Popular Electronics**, p.76. I want to go somewhat deeper into evaluating and describing the Superboard II (Note: the Challenger IP also manufactured by Ohio Scientific is the same computer in a case with power supply).

HARDWARE

KEYBOARD:

The keyboard is mounted directly on the printed circuit board as can be seen in the advertisements. It is a polled keyboard which is polled by writing to a latch addressed at memory location DF00. This latch feeds the rows of the keyboard matrix. When a key is depressed the latch signal is fed through the key switch to a tri-state buffer and back onto the data buss. A read of address DF00 will pick up the signal from the column in which the key is depressed. This method of polling the keyboard makes the hardware very simple (and cheap) but it is effective. In my view a polled keyboard like the one on the Superboard II is better than a hardware implemented ASCII keyboard. Several nice features can be incorporated this way. For example every key has an automatic repeat feature. You have direct access to every key on the board for gaming purposes. Another keyboard can be put in parallel with the existing one. I plan to add a Hex keypad this way. OSI has provided a jack with several of the keyboard lines on it so that switch type joysticks may be connected for games.

For ordinary ASCII input from the keyboard the monitor includes a subroutine which returns the ASCII value of any key depressed. So for all practical purposes this arrangement works just like any other ASCII keyboard.

OSI has fed the signal from the keyboard through a resistor network and then out the game jack. This signal may be connected to a speaker to make sounds or music. The only reason I cannot give a further description of this feature is that OSI failed to include the resistors and I haven't yet gotten around to it.

VIDEO DISPLAY:

The video display is elegant and simple from a hardware point of view. The display on the screen is 32 by 32 but has no guard bands. My monitor displays about 27 by 30 screen size. The software supplied with the Superboard uses only 24 character lines since many who buy the Superboard may want to connect it to an ordinary TV through a video modulator. The video display is refreshed from a 1K memory located at D000-D3FF. Any byte written into this memory gets fed through a character generator and then sent to the screen. The character generator produces not only the full set of ASCII symbols but also more than 100 graphics symbols. It is complete enough to do just about anything you would want to on a 24 by 24 screen: Life, Tic-Tac-Toe, Pong, Racecar, Ship-tank-airplane warfare, etc.

You may wonder about the access to the refresh memory since both the CPU and the video display circuitry must use it. The video display memory is accessed through a multiplexer which is normally connected to the refresh circuitry. This multiplexer allows the CPU to access the memory whenever the CPU addresses any memory from D000 to D3FF. This causes a slight blink in the display on the TV monitor but the blink is almost unnoticeable. Even constantly writing to the display memory causes only a slight decrease in brightness and some flicker of the picture. But whoever writes constantly to the display memory anyway? There is no affect at all on the monitor when the CPU is accessing memory other than the video memory.

CASSETTE I/O:

The Superboard comes with a KC standard cassette interface built in. This operates at 300 baud. That is somewhat slow for loading long programs but the slowness is compensated for by the accuracy. I have yet to find a read error. The hardware for the interface uses a Motorola 6850 ACIA to generate serial data. I think that a small change in the clock used for this ACIA could speed up operation but I have not checked this out yet. This 6850 is located at F000F001 in the memory space.

The greatest difficulty with the cassette interface is that no provision has been made for motor control. It would have been simple to use the Request-to-Send output from the 6850 for this purpose. I plan to connect the Request-to-Send output to a small reed relay for this purpose.

COMPONENTS:

The board itself is high quality epoxy-glass. It is double sided, through the hole plated. The CPU is a 6502A and so has plenty of reserve. The RAM chips and other support are mostly low power variety. All have recent date codes. The character generator and the BASIC ROM's are masked programmed type but the monitor is an EPROM. I suppose you could reprogram the Monitor to suit some particular need you might have. The schematics are accurate and clear. They are very easy to follow since this computer is not really very complicated. The only complaint I would have is that various sections of the schematic are not labelled as to their function. But with a little study you can figure them out.

FUTURE EXPANSION:

An empty 40 pin DIP socket is provided for expansion. All the important control, address, and data lines are connected to this DIP socket. OSI makes a model 610 expansion board which connects to this DIP socket. The 610 expansion board comes with a timer, printer interface, and disk interface along with room for more memory. I personally plan to go from this DIP socket to a KIM type connector for interfacing but there are many possibilities for expansion including the S-100 bus or OSI's 48 pin bus.

SOFTWARE

MONITOR:

The monitor comes in an EPROM at the high end of memory and contains the interrupt vectors, the keyboard input routine, cassette I/O routines, and a memory access routine which allows you to view or change any memory location. With this capability it is very easy to load machine language programs by hand and then execute them or save them and later load them from tape. One deficiency is the lack of a cassette save routine in the monitor.

The monitor has a load routine but no save routine. I have written a save routine which incorporates a Hex memory dump. (See figure 1) This routine saves data in a format acceptable to the monitor load routine. I have located it at 0222 since this space is unused by the BASIC interpreter. The begin address and the end address of the code to be saved must be entered at 00F7 and 00F9 respectively. When you execute the save routine, be sure to turn on your recorder! The code will be saved on tape as well as displayed on the monitor screen. If you want to use this program as a memory dump just run it without turning on your cassette. Several important monitor routines as well as some Basic routines are listed in Table 1.

BASIC:

The BASIC in ROM is an 8K Microsoft product. It is called a 6 digit BASIC since only 6 digits of precision are displayed. Internally, however, all numbers are carried in floating point form with 23 bits of precision (actually the precision is 24 bits since a high order 1 bit is assumed). That amounts to 7½ digits of precision internally. Though this BASIC is very good and very fast it is still a BASIC interpreter and allowance must be made for that fact. I have a puzzle that I have programmed in both BASIC and machine language. The machine language program takes about 1½ hours to run to completion. The BASIC program would take over a month! Superboard is what OSI calls its "immediate mode." That means that any statement can be entered without a line number and it will be executed immediately. Since "?" can be used in place of "PRINT" it is possible to interrogate the computer for any piece of information you might want. For example ? A yields the value of the variable A in the memory. ? 45-20 yields 25. ? PEEK (255) yields the contents of memory location 255 in decimal. GOTO 40 sends BASIC to statement number 40 and begins execution at that point. This last feature is very useful in debugging. One could say that the immediate mode allows you to use the Superboard as a super-calculator and provides a built-in debugger. The BASIC alone is worth the price of the computer.

ASSEMBLER:

There is one available from OSI on tape but I haven't tried it out. I want to write my own and put it in an EPROM.

DOCUMENTATION:

A few words must be said about documentation. Frankly, it is not up to OSI's high quality in the hardware and software areas. The graphics manual is by far the best, providing pretty clear descriptions and giving good examples. The users manual leaves something to be desired in clarity. It is too brief and rather vague at points. I have had real trouble trying to use machine language since there is virtually no description of the machine instructions. I also had some trouble figuring out what pins to connect my cassette to since the diagram is not clearly labelled. The BASIC manual is very brief—admittedly so. OSI expects you to have on hand a BASIC reference manual if you are not thoroughly familiar with the workings of BASIC. One serious problem is an error in the BASIC manual relating to the USR function. It tells you to poke the starting address of the USR routine into locations 023E-023F but this does not work. In the graphics manual there is an example of the use of the USR function. In that example the starting address of the USR routine is poked into 000B-000C. This works. I do wish that manufacturers would supply complete documentation with their software including source code. OSI provides almost nothing in the way of description for either the monitor or BASIC. I have disassembled the monitor and figured it out but have not yet started on BASIC. If anyone has inside information on the inner workings of Superboard BASIC please let us know. Think of all those good routines in BASIC that we could use to memory saving advantage: conversion routines, arithmetic routines, text editor, scanner, etc. Though I have had to give a few negatives about the Superboard II I am well impressed with the quality of both hardware and software. If you are undecided as to what computer is the best buy for the money, I urge you to send your \$279 check to OSI and ask for a Superboard. I don't think there is anything as good for the price on the market.

PROGRESSIVE SOFTWARE

PRESENTS SOFTWARE AND HARDWARE FOR YOUR APPLE

SOFTWARE:

● **Hires Games** ●

- Missile—Anti—Missile By T. David Moteless
- Star Wars
- Rocket Pilon
- Saucer Invasion
- Space Maze By Robert J. Bishop

● **Other Program** ●

- Curve Fit by Dave Garson
- Sales Forecasting by Neil Lipson
- Morse Code
- Calendar by Ed Handley
- Polar Coordinate Plot(Hires) By David Moteles

Programs Require 16K rams and rom board

All Programs. \$9.95 EACH

HARDWARE

Neil Lipson's Original Light Pen Includes
5 Programs \$34.95

SEND Check or M.O. to P.O. Box 273, Ply. Mtg., PA 19462

Programs Accepted for Publicaton - Highest Royalty Paid

Postage and Handling—Add \$1.00 for first item them 50¢ for each add'l

PA Residents Add 6% Sales Tax

OSI CASSETTE SAVE/HEX MEMORY DUMP

BRUCE HOYT
MARCH 1979

TO USE, PLACE THE START ADDRESS OF CODE TO BE SAVED IN 00F7,00F8 AND THEN THE END ADDRESS IN 00F9,00FA. TURN ON THE TAPE RECORDER AND EXECUTE. NOTE: THIS PROGRAM WILL SAVE ITSELF ON TAPE.

```

0222                ORG    $0222

0222 A9 0D          START  LDAIM $0D    CARRIAGE RETURN
0224 20 2D BF      JSR    $BF2D    CRT
0227 20 7A FF      JSR    $FF7A    10 NULLS TO CASSETTE
022A A9 2E          LDAIM $2E    "." ADDRESS MODE
022C 20 75 02      JSR    CC
022F A5 F8          LDA    $00F8    FROM LOCATION (HIGH)
0231 20 63 02      JSR    AOUT
0234 A5 F7          LDA    $00F7    FROM LOCATION (LOW)
0236 20 63 02      JSR    AOUT
0239 A9 2F          LDAIM $2F    "/" DATA MODE
023B 20 75 02      JSR    CC

023E A2 00          LOOP   LDXIM $00
0240 A1 F7          LDAIX $00F7    GET BYTE
0242 20 63 02      JSR    AOUT    OUTPUT
0245 A9 0D          LDAIM $0D    CARRIAGE RETURN
0247 20 B1 FC      JSR    $FCB1    CASSETTE OUTPUT
024A A9 20          LDAIM $20    SPACE
024C 20 2D BF      JSR    $BF2D    CRT
024F E6 F7          INC    $00F7    INCREMENT FROM ADDRESS
0251 D0 02          BNE    BUMP
0253 E6 F8          INC    $00F8
0255 38            BUMP   SEC          CHECK IF DONE
0256 A5 F9          LDA    $00F9    TO
0258 E5 F7          SBCZ  $00F7    FROM
025A A5 FA          LDA    $00FA    TO + 1
025C E5 F8          SBCZ  $00F8    FROM + 1
025E 10 DE          BPL    LOOP
0260 4C 43 FE      JMP    $FE43    YES, RETURN TO MONITOR

0263 85 FC          AOUT   STA    $00FC    USE MONITOR DISPLAY
0265 20 AC FE      JSR    $FEAC    TO UNPACK
0268 AD CC DO      LDA    $DOCC    HI
026B 20 75 02      JSR    CC
026E AD CD DO      LDA    $DOCD    LO
0271 20 75 02      JSR    CC
0274 60            RTS

0275 20 B1 FC      CC     JSR    $FCB1    OUTPUT TO CASSETTE
0278 20 2D BF      JSR    $BF2D    AND CRT
027B 60            RTS

```

Figure 1

Page 0 Usage

0000	JMP to warm start in BASIC
00FB	cassette/keyboard flag for monitor
00FC	data temporary hold for monitor
00FE-00FF	address temporary hold for monitor

Page 1

0100-0140	stack
0130	NMI vector - NMI interrupt causes a jump to this point
01C0	IRQ vector

Page 2

0200	cursor position
0203	load flag
0205	save flag
0206	CRT simulator baud rate - varies from 0 = fast to FF = slow
0212	Control-C flag
0218	input vector = FFBA
021A	output vector = FF69
021C	Control C check vector = FF9B
021e	load vector = FF8B
0220	save vector = FF96
0222-02FA	unused

Page 3 and up to end of RAM is BASIC workspace

A000-BFFF	BASIC in ROM
D000-D3FF	Video refresh memory
DF00	Polled keyboard
F000-F001	Cassette port 6850
F800-FFFF	Monitor EPROM
FC00	Floppy bootstrap
FD00	Keyboard input routine
FE00	Monitor
FF00	BASIC I/O support

Useful Subroutine entry points

A274	warm start for BASIC
BD11	cold start for BASIC
BF2D	CRT simulator - prints char in A register
FD00	input char from keyboard, result in A
FCB1	output 1 byte from A to cassette
FE00	entry to monitor, clears screen, resets ACIA
FE0C	entry to monitor, bypasses stack initialization
FE43	entry to address mode of monitor
FE80	input ASCII char from cassette, result in A, 7 bit cleared
FE93	convert ASCII hex to binary, result in A, =80 if bad
FF69	BASIC output to cassette routine, outputs one char to cassette, displays on screen, outputs 10 nulls if carriage return character
FF00	Reset entry point
FF8B	Load flag routine
FF96	Save flag routine
FF9B	Control-C routine
FFBA	BASIC input routine

Table 1.

EKIM OR MAXI-KIM
Extended Keyboard Input Monitor

Andrew V.W. Sensicle
 155 Valois Bay Ave.
 Pointe Claire, Montreal
 Quebec, Canada H9R 4B8

Although KIM-1's ROM contains useful features like the tape and TTY input-output routines, when it comes to inputting data or coding via the key pad, KIM's resident monitor leaves much to be desired, for example the avoidance of repetitive pushing of the "t" between each entry or the ability to look back a few bytes without going into address mode. I would like to thank Jim Butterfield for his excellent BROWSE and BRANCH PROGRAMS which I put together in Page 1 and have used religiously since I got started in this game in mid '78.

However, these have their limitations and I have frequently found the need for a little more sophistication, not to mention the space they occupy in Page 1. Anyway the thing which irritated me most was the need to re-enter a long listing merely in order to open up a few spaces for additional instructions. The process of tidying up a finished program, entailing closing up unwanted spaces and the associated readdressing was also very time consuming.

I thus decided to try to write an extended monitor which would be compact enough to fit in Page 17 and yet provide the functions I needed. After much condensing and compressing I ended up with a program 6 bytes longer than the "legal" Page 17 RAM, but by stealing a little from KIM it fits nicely. KIM doesn't seem to mind. As long as you don't use the tape or TTY routines, he leaves you alone.

The NMI vector is loaded with the start address (1780) so that the ST key can be used to access the monitor at any open cell address. Before pressing ST or after exiting via RS the resident monitor is used as a normal in the AD mode. The ST key gives you 6 other modes of operation or functions.

1. **STAND BY MODE [ST]:** This starts the program which then sits looking at the open cell address and its contents, ie. nothing seems to happen. However, any HEX key is stored at the open cell address which each second key stroke increments the address.

2. **INCREMENT [t]:** Big deal! This works just like normal.

3. **DECREMENT [PC]:** This steps the address points backwards exactly the reverse of "t".

4. **OPEN UP MODE [AD]:** Each depression of this key causes one full page of bytes (FF) to be moved one place up starting at the open cell address.

5. **CLOSE UP MODE [DA]:** Each depression of this key causes one full page of bytes to be moved one place back to overwrite the open cell contents. Having made an "open up" or close up move of one or more steps you will, of course, have to fix up all affected addresses. This is not as onerous as it sounds if you use the sixth mode.

6. **BRANCH MODE [GO]:** When a branch instruction is encountered while entering a new program or fixing up an old one, all you need do is press "GO" followed by the actual destination address (low order only). The monitor will calculate the relative address, store it in the open cell and step on to the next cell all in the twinkling of an eye. The user is, as usual, responsible for ensuring that the branch does not exceed the normal half page range.

I hope that this little program will be as useful to others as it is and has been to me.

	ORG	\$1780		
	MODE	*	\$00FF	
	TEMPX	*	\$00FD	
	LAST	*	\$00F3	
	INL	*	\$00F8	
	POINTL	*	\$00FA	
	POINTH	*	\$00FB	
	SCAND	*	\$1F19	
	GETKEY	*	\$1F6A	
	UPDATE	*	\$1FBB	
	INCPT	*	\$1F63	
1780	D8	START	CLD	
1781	A2 01		LDXIM \$01	INITIATE MODE AND
1783	86 FF		STX MODE	COUNTER
1785	86 FD		STX TEMPX	

1787	20	19	1F	GETK	JSR	SCAND	LIGHT DISPLAY
178A	20	6A	1F		JSR	GETKEY	CHECK KEYS
178D	C5	F3			CMP	LAST	
178F	F0	F6			BEQ	GETK	
1791	85	F3			STA	LAST	NEW KEY
1793	C9	13			CMPIM	\$13	GO ?
1795	D0	02			BNE	SKIP	
1797	C6	FF			DEC	MODE	PUT IN BRANCH MODE
1799	C9	12		SKIP	CMPIM	\$12	+ ?
179B	F0	4A			BEQ	INCPNT	
179D	C9	14			CMPIM	\$14	PC ?
179F	F0	22			BEQ	DECPNT	
17A1	C9	11			CMPIM	\$11	DA ?
17A3	F0	11			BEQ	CLOSUP	
17A5	C9	10			CMPIM	\$10	AD ?
17A7	D0	26			BNE	INDATA	
17A9	A0	FF		OPENUP	LDYIM	\$FF	LOAD 255(10)
17AB	88			OPENX	DEY		
17AC	B1	FA			LDAIY	POINTL	LOAD AND STORE
17AE	C8				INY		ONE CELL HIGHER
17AF	91	FA			STAIY	POINTL	
17B1	88				DEY		
17B2	D0	F7			BNE	OPENX	NEXT
17B4	F0	CA			BEQ	START	
17B6	A0	01		CLOSUP	LDYIM	\$01	
17B8	B1	FA		CLOSX	LDAIY	POINTL	LOAD OPEN CELL
17BA	88				DEY		PLUS 1
17BB	91	FA			STAIY	POINTL	STORE IN OPEN CELL
17BD	C8				INY		THEN UP
17BE	C8				INY		UNTIL
17BF	D0	F7			BNE	CLOSX	
17C1	F0	BD			BEQ	START	CONE 255 (10)
17C3	C6	FA		DECPNT	DEC	POINTL	
17C5	A5	FA			LDA	POINTL	
17C7	C9	FF			CMPIM	\$FF	PAGE CHANGE?
17C9	D0	B5			BNE	START	NO
17CB	C6	FB			DEC	POINTH	YES, THEN DEC POINTH
17CD	10	B1			BPL	START	AS WELL
17CF	C9	10		INDATA	CMPIM	\$10	
17D1	B0	AD			BCS	START	FALSE START ACTUALLY NO KEY
17D3	20	BB	1F		JSR	UPDATE	ROL 4 BITS FROM A TO INL
17D6	A5	F8			LDA	INL	
17D8	91	FA			STAIY	POINTL	
17DA	C6	FD			DEC	TEMPX	
17DC	F0	A9			BEQ	GETK	ONE MORE KEY
17DE	A4	FF			LDY	MODE	IN BRANCH MODE?
17E0	DC	05			BNE	INCPNT	NO
17E2	18				CLC		
17E3	E5	FA			SBC	POINTL	CALC RELATIVE ADDRESS
17E5	91	FA			STAIY	POINTL	STORA IT IN OPEN CELL
17E7	20	63	1F	INCPNT	JSR	INCPT	NEW CELL
17EA	4C	80	17		JMP	START	RETURN

A CASSETTE OPERATING SYSTEM FOR THE APPLE II

Robert A. Stein, Jr.
2441 Rolling View Dr.
Dayton, OH 45431

Have you ever wished that as great as the Apple II computer system is that you were able load programs by name from a library cassette? Well, with this mini-sized cassette operating system you can stack many programs on one cassette and load the one you want by typing in its name. Great for showing off your system without juggling a dozen or so cassette tapes.

The **Cassette Operating System [CASSOS]** resides in memory at locations 02C0 to 03FF, where it won't get clobbered by BASIC programs or initialization. Add the optional cassette control circuit, or purchase one of the commercially available ones. (Candex Pacific, 693 Veterans BLVD, Redwood City, CA 94063) and you never need envy the PET for its loading technique again.

Operation

Load the 'CASSOS' tape, which you have created from the assembly listing, just like any other machine language program (2C0.3FFR), then initialize the BASIC pointers by depressing CTRL-B, return. To load a program depress CTRL-Y and RETURN. "PROG?" will be displayed, enter a 1-10 character program name. The cassette tape will be searched and the program loaded if found. "XXXXXXXXXX LOADED" will be output, where XXXXXXXXXXXX is the program now in memory. If the cassette control circuit (described later) is present the tape will also be stopped. A line of question marks (????????) are displayed if the requested program was not found. To write a program to the library cassette enter Yc (Ctrl-Y, "WRITE", and RETURN. Program will be saved under the name requested at PROG?. "XXXXXXXXXX OUT" will be displayed at completion and the recorder stopped. To end a cassette program file enter: Yc, "EOF", RETURN; a special record header will be written. Note that to conserve limited memory space the EOF routine utilizes the program write subroutine so the "XXXXXXXXXX OUT" message should be ignored.

The program is structured such that the last 63 locations of the input buffer is used for display messages, so if more than 191 characters are entered at one time the program will still function, but without messages. The listing as presented was for a 16K system, change location 0358 as follows for a different configuration:

2F-8K	6F-24K
3F-12K	8F-32K
4F-16K	9F-36K
5F-20K	CF-48K

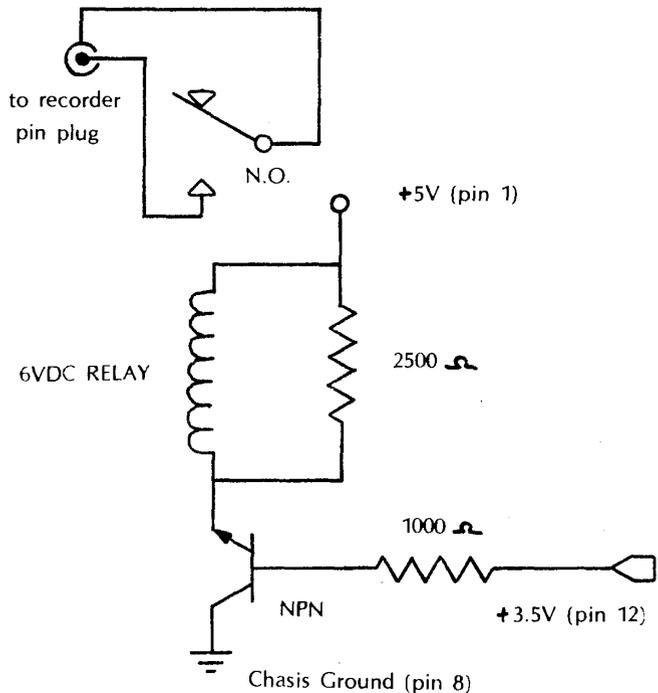
Program Design

The method by which CASSOS functions is to write a program header block consisting of header ID, program name, and start of the BASIC load. This is followed by the program data itself, utilizing the Apple monitor routines.

A Cassette On/Off Circuit

The following diagram describes a simple circuit for stopping and starting a cassette recorder which has a "remote" plug from the Apple II under program control. The theory involves activating or

deactivating the AN3 signal on the Apple game connector. A store to location C05F turns the recorder on and location C05E turns it off. The strobe triggers a transistor which in turn opens a relay and closes the connection to the remote plug, starting the recorder. If your recorder requires an open connection to start tape movement wire the relay normally closed instead of open. It is also possible to add a relay that would interrupt power to the recorder for control if you have no remote capability on your recorder.



Cassette Control Circuit

Parts List

All parts were purchased at a local Radio Shack
6VDC Relay (275-004)
NPN Transistor (2N3568 or equivalent)
1000 Ohm Resistor
2500 Ohm Resistor
Mini-Plug

All connections were made to a DIP Header which was modified by soldering a 16-pin IC to it so that the game paddles could still be used without modification when the cassette ON/Off circuit was in use. The common 6VDC relay was modified to be triggered by the game connector signals by wiring a 2500 ohm resistance (utilizing a series of resistors connected in series so that the sum is 2500 Ohms) in parallel with the relay coil. If your recorders rewind controls are disabled by the remote jack wire a switch to bypass the transistor between chasis ground and the relay, which will allow the rewind to operate when depressed. If all this is beyond your scope use the purchased control or simply stop and start the recorder manually.

0200-	A9 B3	LIA	##D3	0344-	CA	DEX	
0202-	8D B0 02	STA	\$02B0	0345-	D0 F8	BNE	\$033F
0205-	A9 B1	LIA	##B1	0347-	60	RTS	
0207-	20 67 03	JSR	\$0367	0348-	A0 B0	LDY	##B0
020A-	A9 FF	LIA	##FF	034A-	A2 00	LIX	##00
020C-	8D EB 02	STA	\$02BB	034C-	20 51 03	JSR	\$0351
020F-	A5 CA	LIA	\$CA	034F-	A0 BD	LDY	##BD
02D1-	8D BC 02	STA	\$02BC	0351-	A9 02	LIA	##02
02D4-	A5 CB	LIA	\$CB	0353-	D0 04	BNE	\$0359
02D6-	8D BD 02	STA	\$02BD	0355-	A0 FF	LDY	##FF
02D9-	20 CD FE	JSR	\$FECD	0357-	A9 3F	LIA	##3F
02DC-	A4 CA	LDY	\$CA	0359-	95 3D	STA	\$3D,X
02DE-	A5 CB	LIA	\$CE	035B-	94 3C	STY	\$3C,X
02E0-	20 60 03	JSR	\$0360	035D-	E8	INX	
02E3-	20 CD FE	JSR	\$FECD	035E-	E8	INX	
02E6-	A9 EB	LIA	##EB	035F-	60	RTS	
02E8-	20 7E 03	JSR	\$037E	0360-	A2 00	LIX	##00
				0362-	20 59 03	JSR	\$0359
				0365-	D0 EE	BNE	\$0355
02EB-	87 A0 CF D5 99			0367-	85 50	STA	\$50
02FB-	FF 87 A0 CC CF C1 C4 C5			0369-	A2 02	LIX	##02
02F8-	C4 FF D0 D2 CF C7 BF FF			036B-	A0 FA	LDY	##FA
				036D-	20 04 03	JSR	\$0304
				0370-	20 48 03	JSR	\$0348
0300-	A2 02	LIX	##02	0373-	A9 0A	LIA	##0A
0302-	D0 07	BNE	\$030B	0375-	A6 50	LIX	\$50
0304-	84 60	STY	\$60	0377-	20 20 03	JSR	\$0320
0306-	20 62 FD	JSR	\$FD62	037A-	8D 5F 00	STA	\$035F
0309-	A4 60	LDY	\$60	037D-	60	RTS	
030B-	8E 15 03	STX	\$0315	037E-	48	PHA	
030E-	8C 14 03	STY	\$0314	037F-	8D 5E 00	STA	\$035E
0311-	A0 00	LDY	##00	0382-	A2 02	LIX	##02
0313-	B9 FA 02	LIA	\$02FA,Y	0384-	A0 B1	LDY	##B1
0316-	C9 FF	CMP	##FF	0386-	20 04 03	JSR	\$0304
0318-	F0 2D	BEQ	\$0347	0389-	68	FLA	
031A-	20 ED FD	JSR	\$FDED	038A-	A8	TAY	
031D-	C8	INX		038B-	20 00 03	JSR	\$0300
031E-	D0 F3	BNE	\$0313	038E-	4C 03 E0	JMP	\$E003
0320-	48	PHA		0391-	A9 A3	LIA	##A3
0321-	A9 02	LIA	##02	0393-	20 67 03	JSR	\$0367
0323-	86 60	STX	\$60	0396-	20 48 03	JSR	\$0348
0325-	85 61	STA	\$61	0399-	20 FD FE	JSR	\$FEFD
0327-	A9 A0	LIA	##A0	039C-	AD B0 02	LIA	\$02B0
0329-	20 6C FD	JSR	\$FD6C	039F-	C9 D3	CMP	##D3
032C-	68	PLA		03A1-	D0 29	BNE	\$030C
032D-	AA	TAX		03A3-	AC BC 02	LDY	\$02BC
032E-	A0 00	LDY	##00	03A6-	AD BD 02	LIA	\$02BD
0330-	B9 00 02	LIA	\$0200,Y	03A9-	20 60 03	JSR	\$0360
0333-	C9 8D	CMP	##8D	03AC-	20 FD FE	JSR	\$FEFD
0335-	F0 08	BEQ	\$033F	03AF-	A2 00	LIX	##00
0337-	91 60	STA	(\$60),Y	03B1-	ED B1 02	LIA	\$02B1,X
0339-	C8	INX		03B4-	DD A3 02	CMP	\$02A3,X
033A-	CA	DEX		03B7-	D0 DD	BNE	\$0396
033B-	F0 0A	BEQ	\$0347	03B9-	E8	INX	
033D-	D0 F1	BNE	\$0330	03BA-	E0 0A	CPX	##0A
033F-	A9 A0	LIA	##A0	03BC-	D0 F3	BNE	\$03B1
0341-	91 60	STA	(\$60),Y	03BE-	AD BC 02	LIA	\$02BC
0343-	C8	INX		03C1-	85 CA	STA	\$CA

0303-	AD B0 02	LIA	\$02BD	03E3-	F0 10	BEQ	\$03F5
0306-	95 CB	STA	\$CB	03E5-	09 C5	CMF	#\$05
0309-	A9 F1	LIA	#\$F1	03E7-	D0 A8	BNE	\$0391
030A-	D0 B2	BNE	\$037E	03E9-	8D B0 02	STA	\$02B0
030C-	8D 5E C0	STA	\$005E	03EC-	20 48 03	JSR	\$0348
030F-	A2 20	LIX	##20	03EF-	8D 5F C0	STA	\$005F
03D1-	A9 BF	LIA	#\$BF	03F2-	4C CA 02	JMP	\$02CA
03D3-	20 ED FD	JSR	\$FDED	03F5-	4C C0 02	JMP	\$02C0
03D6-	CA	DEX		03F8-	4C DE 03	JMP	\$03DE
03D7-	D0 F8	BNE	\$03D1	03FB-	00	BRK	
03D9-	20 DD FB	JSR	\$FBDD	03FC-	00	BRK	
03DC-	F0 B3	BEQ	\$0391	03FD-	00	BRK	
03DE-	AD 01 02	LIA	\$0201	03FE-	00	BRK	
03E1-	09 D7	CMF	#\$D7	03FF-	00	BRK	

A Cassette Tape Catalog

Shown in exhibit is a short integer BASIC program which when loaded will list all the programs on a CASSOS format library tape. The CASSOS sub-routines are used so the software must be core resident. Just load the program, insert the library cassette into the cassette handler, and type RUN after starting the cassette player.

```

10 N=1: CALL -936: UTAB (10): DIM X$(1)
20 INPUT "INSERT LIBRARY TAPE AND DEPRESS 'RETURN'",X$
30 POKE -16289,0: CALL -936: GOSUB 300
40 PRINT "FILE # PROGRAM NAME BYTES"
50 PRINT "-----"
60 CALL 840: CALL -259
70 IF PEEK (688)=ASC("E") THEN 210
80 IF PEEK (688)#ASC("S") THEN 200
100 REM LOAD INTO NON-EXIST MEMORY (800-BFF)
110 POKE 60, PEEK (700): POKE 61,( PEEK (701)+128)
120 POKE 62,255: POKE 63,191: CALL -259
130 PRINT N,: POKE 789,2: POKE 788,177: CALL 785
140 L= PEEK (700)+ PEEK (701)*256
150 L=16384-L:N=N+1
160 PRINT "   ":L: GOTO 60
200 GOSUB 300: PRINT "NO EOF MARK"
210 POKE -16290,0: GOSUB 300
230 PRINT "***END OF FILE***"
240 CALL -155
300 FOR I=1 TO 30
305 L= PEEK (-16336)+ PEEK (-16336): NEXT I
310 CALL -1059: RETURN

```

```

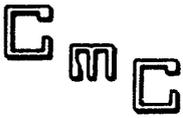
>RUN
INSERT LIBRARY TAPE AND DEPRESS 'RETURN'

```

```

FILE # PROGRAM NAME BYTES
-----
1DIRECORY 544
2BILLBOARD 238
3R.ROULETTE 530
4COLORBYROD 185
5HELLO 2830
6BOWLING 2119
7BOXING 2636
8TICTACTOE 3461
***END OF FILE***

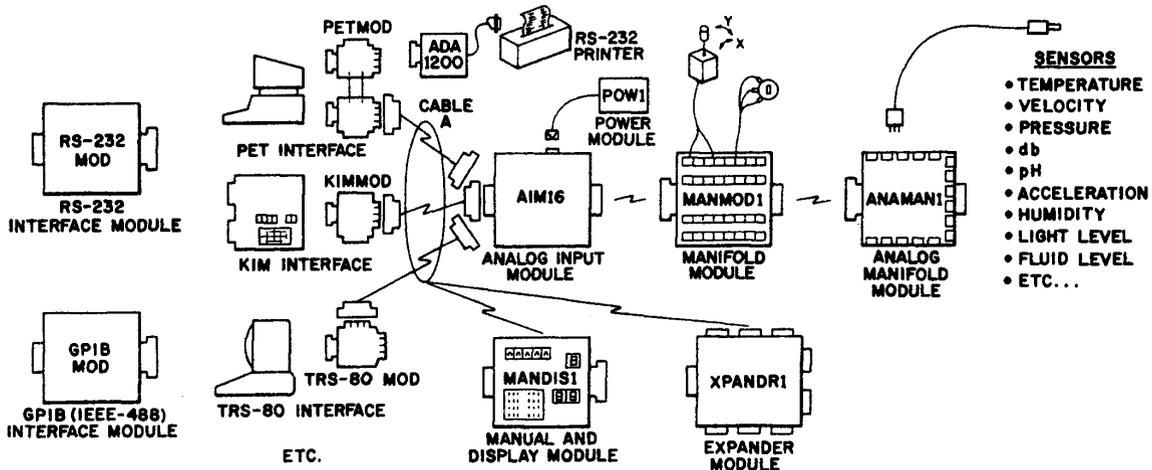
```



CONNECTICUT microCOMPUTER, Inc.

150 POCONO ROAD - BROOKFIELD, CONNECTICUT 06804

(203) 775-9659



DAM SYSTEMS by CmC
A complete system of modules to let your computer listen to the real world.

DAM SYSTEMS PRICE LIST

DAM SYSTEMS components

AIM161 - Analog Input Module 16 8-bit analog inputs - 100 microsecond conversion time - 3 state output - requires one 8-bit computer output port for control and one 8-bit computer input port for data.	\$179.00
AIM162 - Analog Input Module As above plus: greater accuracy - gold plated contacts - pilot light - switch selectable start, enable and ready polarities.	\$249.00
POW1 - Power Module Supplies power for one AIM16 module.	\$14.95
ICON - Input Connector For connecting analog inputs to the AIM16 - 20 pin card edge connector - solder eyelets.	\$9.95
OCON - Output Connector For connecting the AIM16 to a computer - 20 pin card edge connector - solder eyelets.	\$9.95
MANMOD1 - Manifold Module Use in place of ICON. Screw terminal barrier strips for connecting josticks, potentiometers, voltage sources, etc. Eliminates the need for soldering. Plugs into the AIM16.	\$59.95
ANAMAN1 - Analog Manifold Module Use in place of ICON. Connects DAM SYSTEMS SENSORS to the AIM16 without soldering - sensor cables just plug in. Plugs into the AIM16 or the MANMOD1.	TBA
SENSORS Sensors for temperature, pressure, flow, humidity, level, pH, motion, etc.	TBA
COMPUTER INTERFACES For the PET, KIM, TRS-80, etc. Use in place of OCON. Eliminates the need for soldering or special construction.	TBA
PETMOD - PET Interface Module Gives two IEEE ports, one user port and one DAM SYSTEMS interface port. Saves wear and tear on the PET's printed circuit board. Also called the PETAUR.	\$49.95

KIMMOD - KIM Interface Module Gives one application connector port and one DAM SYSTEMS interface port.	\$39.95
CABLE "A" - Interconnect Cables Connects computer interface to AIM16, MANDIS1, XPANDR1, etc.	TBA
CABLE A24 - Interconnect Cable 24 inch cable with interface connector on one end and an OCON equivalent on the other.	\$19.95
MANDIS1 - Manual and Display Module Connects between the AIM16 and the computer interface. Allows manual or computer control of the AIM16. Displays channel number and data.	TBA
GP1B MOD - GP1B (IEEE-488) Interface Allows the DAM SYSTEMS MODULES to be used with the GP1B bus instead of a computer's other I/O ports.	TBA
RS232 MOD - RS232 Interface Module Allows the DAM SYSTEMS MODULES to be used with an RS-232 port or terminal.	TBA
XPANDR1 - Expander Module Allows up to 128 8-bit analog inputs (8 AIM16 Modules) to be connected to one system.	TBA

DAM SYSTEMS sets

AIM161 Starter Set Includes one AIM161, one POW1, one ICON and one OCON.	\$189.00
AIM162 Starter Set Includes one AIM162, one POW1, one ICON and one OCON.	\$259.00
PETSET1a Includes one PETMOD, one CABLE A24, one AIM161, one POW1 and one MANMOD1.	\$295.00
KIMSET1a Includes one KIMMOD, one CABLE A24, one AIM161, one POW1 and one MANMOD1.	\$285.00

ASK THE DOCTOR - PART III BITS AND BYTES

Robert M. Tripp, Ph.D.
The COMPUTERIST, Inc.
P.O. Box 3
So. Chelmsford, MA 01824

The Doctor was busy this month and did not get a chance to write up the EPROM Programmer hardware as promised in the last issue. Look for it next time. A couple of people did submit some good info which is printed below. The Doctor encourages such input. Too much is happening with these new computers for anyone person to "know it all", so if you find out something interesting, please drop us a note and let us get the word out.

Corrected AIM SYNC Program

The early AIM User Manuals had a number of mistakes, as is to be expected the first batch. One of the more serious errors was in the listing for the SYN Write and SYN Read programs on page 9-11. The errors have been corrected in later versions of the manual, but for those of you who need the programs, here they are - corrected.

SYN Write Program:

```
0300 20 1D F2 JSR F21D
0303 20 4A F2 JSR F24A
0306 4C 03 03 JMP 0303
```

SYN Read Program:

```
0310 A2 00 LDX #00
0312 A9 CE LDA #CE
0314 20 7B EF JSR EF7B
0317 20 EA ED JSR EDEA
031A A2 00 LDX #00
031C A9 D9 LDA #D9
031E 20 7B EF JSR EF7B
0321 20 29 EE JSR EE29
0324 C9 16 CMP #16
0326 F0 F9 BEQ 0321
0328 D0 E6 BNE 0310
```

Patch for the AIM-DISASSEMBLER

It soon becomes obvious, that the disassembler is extremely paper consuming, because no single-stepping is provided. The following program will save you money and time!

Set F1 (010C) to 'JMP 03D9' and F2 (010F) to 'JMP 03CB'. After loading the desired program address (*), hitting F1 will disable just this line on the display. To advance, press the space-bar. If you want to modify, use 'I' and the program jumps to the **Instruction Mnemonic Entry**. The current address will not be changed. 'ESC' brings you back to the AIM-Monitor. With 'F1', the next address will be disassembled. 'F2', however, will subtract the last used op-code length from the current address and then disassemble the last entry! It is even possible to disassemble further "backwards", just keep switching from

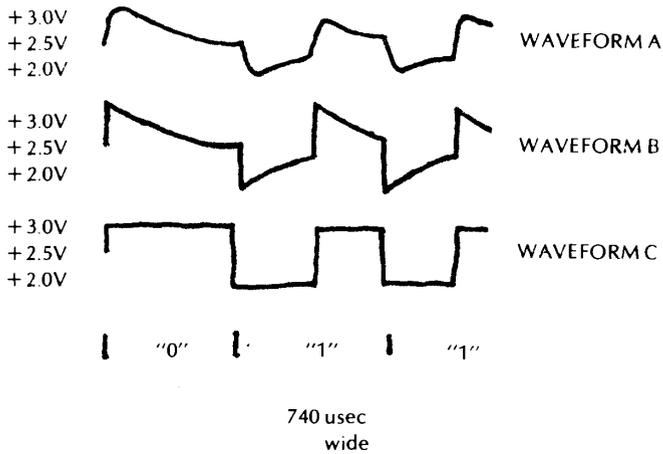
'ESC' to 'F2'. Of course, a change in the op-code length will bring up some unexpected results, but very soon you'll catch a proper op-code again!

```
03CB AD 25 A4 LDA A425
03CE 18 CLC
03CF E5 EA SBC EA
03D1 8D 25 A4 STA A425
03D4 B0 03 BCS 03D9
03D6 CE 26 A4 DEC A426
03D9 20 24 EA JSR EA24
03DC 20 6C F4 JSR F46C
03DF 20 07 E9 JSR E907
03E2 20 3C E9 JSR E93C
03E5 C9 49 CMP #49
03E7 D0 03 BNE 03EC
03E9 4C 9E FB JMP FB9E
03EC C9 20 CMP #20
03EE D0 F2 BNE 03E2
03F0 AD 25 A4 LDA A425
03F3 38 SEC
03F4 65 EA ADC EA
03F6 8D 25 A4 STA A425
03F9 90 DE BCC 03D9
03FB EE 26 A4 INC A426
03FE 90 D9 BCC 03D9
```

Submitted by
Gebhard Brinkmann
Koblenzer Str. 1.
D-5401 Kaltengers
West Germany

SYM Tape Evaluation

As a result of our telephone conversation on Monday, I decided to look for any possible hardware problems in the SYM Cassette Interface. Some results are shown below. Whether these are related to your cassette problems is unknown. In checking my Sony TC-62, I found an unexpected very slow acting AVC (increases gain very slowly, decreases rapidly). This could cause problems in a level sensitive system as the gain slowly increases during the recording process to a quite large degree.



All waveforms taken at PIN 3 of the LM311 (U26) with a sync tape generation program running (hi-speed). Audio OUT (HI) is connected directly to Audio In (A-P to A-L).

WAVEFORM A is the normal condition as received (VIM 80650912 E/C0003)

WAVEFORM B is with C14 (.0047uF) removed

WAVEFORM C is with C14 removed and C16(.01uF) paralled with 1uF

CONCLUSION: C16 is much too small and could easily cause the system to become marginal in the presence of noise and normal level variations. C14 has no apparent real value and seems to unnecessarily increase transition time uncertainty. The small value of C16 and the presence of C14 together simulate the waveform degradation of a very limited bandwidth recorder. Their effect augment rather than compensate for the deficiencies of a recorder. Surprisingly, it appears that it would be a recorder with poor low, rather than high, frequency response which would be most likely to have problems with C16 is maintained at its original .01 microfarad value.

Submitted by
Don Lloyd
101 Western Ave., Apt. 76
Cambridge, Ma. 02139

FLASH !!!

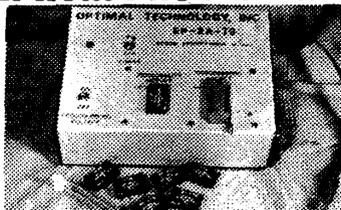
Synertek has finally solved the sensitivity problem which has been of concern to users of the tape cassette, according to a spokesman from Synertek Systems. I have sent them a pair of 2716 EPROMs to be programmed with the new monitor. If these are returned in time, I will make a full report in next month's issue.

Comments on Synertek BASIC (8K) V1.1

- 1) 2 ROM's, U21, U22, C000-DFFF, (J) (0) (CR to start BASIC)
- 2) **Commands** - CLEAR, LIST, NULL, RUNN, NEW CONT, LOAD "A", SAVE "A"
- 3) **Statements** - DATA, DEF, DIM, END FOR, GOTO, GOSUM, IF.. GOTO, IF.. THEN, INPUT, LET, NEXT, ON.. GOSUM, POKE, PEEK, PRINT, READ, REM, RESTORE, RETURN, STOP, WAIT.
- 4) **Functions** - ABS(X), INT(X), RND(X), SGN(X), SQR(X), TAB(I), USR(I), USR(I,J,...Z), EXP(X), FRE(X), LOG(X), POS(I), SPC(I) SIN(X), COS(X), TAN(X), ATN(X) all must be loaded separately - App Note 53-SSC not quite available.
- 5) **Strings** - DIM A\$, LET A\$, INPUT X\$, READ X\$, PRINT X\$
- 6) **String Functions** - ASC(X\$), CHR\$(I), FRE(X\$), LEFT\$(X\$,I) LEN(X\$), MID\$(X\$,I), MID\$(X\$,I,I), RIGHT\$(X\$,I), STR\$(X), VAL(X\$)
- 7) **Operators** =, +, exponentiation, *, =, (not equal), , , (LTE), (GTE), NOT, AND, OR
- 8) **Uses Memory** from 0200 HEX up until ROM or no memory, unless restricted at start up.
- 9) **Weaknesses** - Only editing is delete line, delete last character (RUB-OUT), no ROM TRIG, no program merging capability.
- 10) **Strengths** - Good array features (but no MAT functions), 9 digit accuracy floating points
4 byte floating point numbers
7 bits + 1 bit sign exponent
1 bit sign + 24 bit binary value (MSbit = 1 always) & "000F" = 15 decimal
hex string conversion to decimal
USR (I,J,...Z) Machine language subroutine multiple parameters on stack result (A,Y)
Speed is comparable to OSI Kilobaud Oct '77 ratings (1MHz)
Overall subjective by infrequent BASIC user: 7.5/10 seems appropriate to overall product.

Submitted by
Don Lloyd
101 Western Ave., Apt. 76
Cambridge, Ma. 02139

EPROM PROGRAMMER



Software available for F-8, 6800, 8080, 8085, Z-80, 6502, KIM-1, 1802.

The EP-2A-79 will program the 2704, 2708, TMS 2708, 2758, 2716, TMS 2516, TMS 2716, TMS 2532, and 2732. PROM type is selected by a personality module which plugs into the front of the programmer. Power requirements are 115 VAC, 50/60 HZ at 15 watts. It is supplied with a 36-inch ribbon cable (14 pin plus) for connecting to microcomputer. Requires 1 1/2 I/O ports.

Assembled and tested \$145, Plus \$15-25 for each personality module. Specify software.

OPTIMAL TECHNOLOGY, INC.
Blue Wood 127, Earlysville, Va. 22936
Phone 804-973-5482

DR. DALEY'S SOFTWARE FOR THE PET

DR. DALEY's software continues to expand offerings. Listed below are our most popular programs. No PET owner should be without these. Dealers, you should stock them as well.

PET TREK 3	Like STARTREK, but has several UNIQUE features. For example, the unpredictable EXPERIMENTAL RAY, who knows what it will do	\$ 7.95
BACKGAMMON	It's you vs the PET with an exciting game of BACKGAMMON	\$ 7.95
MASTER MIND	Plays two simultaneous games, one where you guess PET's secret code, and another where PET guesses yours	\$ 7.95
RENUMBER	Will renumber your BASIC programs, including all jump statements. For a 6K source code requires less than 5 seconds	\$12.95
PILOT	A BASIC coded PILOT interpreter. A second high level language for the PET. Simple to use, even a ten year old can learn to use PILOT quickly. With sample PILOT programs and documentation	\$12.95
CHECKBOOK	Will balance your checkbook and save totals in 16 categories on tape. Will produce end of month and year to date summaries. Categories can easily be changed to suit your own purposes.....	\$12.95
MAIL LIST	Keeps a mailing list and will sort the list into sub groups using up to three search parameters	\$12.95

All of our programs are available on tape or for the Compu-Think disk. We charge \$5.00 for the disk and shipping, but you can subtract \$1.00 for each program which we place on the disk. Order 5 programs and you get the disk free!

MAIL LIST	The above program has been modified for disk files. Will be placed on a disk by itself which you can then use for your mailing list	\$19.95
FLASH!	We have just acquired the rights to distribute a linking loader for BASIC programs! This will allow you to link exclusively numbered BASIC subroutines in memory. No serious programmer should be without this useful programming tool.....	\$12.95

An ideal companion to the linking loader will be our library of useful subroutines which can be linked into your own program. Currently over 25 useful routines are included. These range from plotting utilities to a beautiful display of rolling dice. Write or call for a list or order the set for only.....

\$49.95

* * *

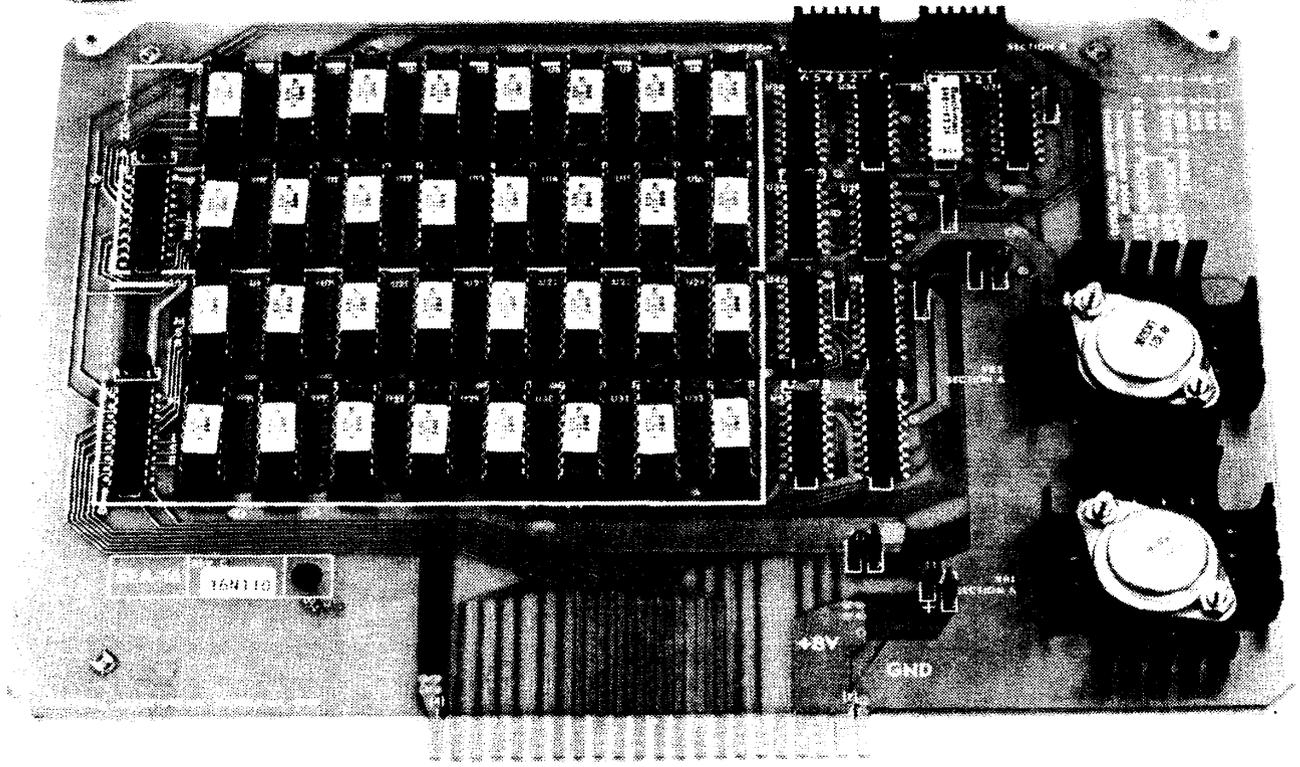
Remember that we GUARANTEE that your order will be shipped within four business days from receipt or you will receive a coupon for a discount on a future purchase.

Charge your order to
MC/VISA



DR. DALEY, 425 Grove Avenue, Berrien Springs, Michigan 49103

Phone (616) 471-5514 Sun. to Thurs. noon to 9 p.m. eastern time



The best memory board around.

Here's why

- Low power 2114 Static RAM's
- Fully buffered
- High quality IC sockets
- All switches accessible from top of board
- Top grade glass fiber PCB, with gold plated contact area.
- Dual regulators

- Two independent 8Kx8 memory banks
- One supply only, 7-9V unregulated

16K Static RAM \$325.00
Assembled only.

Designed specifically for the: KIM-1, SYM-1, AIM-65.

Specifications: Access time 450 nS max.
 Power consumption 1.35 amp. typ.

Also available: Buffered Mother Board, EPROM Programmer, CVT Power Supply. Software: Standard Forth Compiler, Sea-65 Editor/Assembler.

Order from local dealer or directly from:

SEAWELL MARKETING INC., 315 N.W. 85th, Seattle, WA 98117 • (206) 782-9480

Available in Europe. Write for dealer list.

Check or money order enclosed

Send more information _____

Charge my Master Charge or VISA

Name _____

Address _____

Acct. # _____ Expir. Date _____

City _____ State _____ Zip _____

THE MICRO SOFTWARE CATALOG: VII

Mike Rowe
P.O. Box 3
S. Chelmsford, MA 01824

Name: Slow-Scan Television Package

System: Apple II

Memory: 16K (min)

Language: Machine Language

Hardware: Standard Apple II

Description: This software system allows the Apple II to send and receive U.S. amateur standard slow-scan T.V. pictures (120 line-15 Hz) via any ham radio SSB transceiver. A real-time display of the received picture in high-resolution graphics is accomplished with a sophisticated image processing algorithm. Low-resolution images for transmission are prepared with a large-character display editor as well as a drawing editor. All modulation and demodulation of the audio FM subcarrier is performed by the software — replacing hundreds of dollars of hardware required by other SSTV systems. Comes on cassette with 8 mins. of test pictures.

Copies sold: about 100

Price: \$20.

Includes: Cassette tape and 5 pages of documentation.

Author: Chris H. Galfo — WB4JMD

Available from:

C.H. Galfo
602 Orange St
Charlottesville, VA 22901

Name: S-C Assembler II (disk version)

System: Apple II with at least one disk

Memory: 24K or more

Hardware: Apple II, Disk II, optional printer

Description: Disk version of the popular S-C Assembler for the Apple II. Combines a text editor and an assembler in one memory resident package of 3072 bytes (1000-1BFF). Carefully integrated with the Apple II ROM-resident routines, and with Apple DOS. Editor includes full screen-editing, BASIC-like line number editing, tab stops, and renumbering. LOAD and SAVE commands for storage of source programs on disk files or cassette. JOIN command for appending two source programs from cassette. Standard Apple II syntax for opcodes and address modes. Labels (up to 6 characters), arithmetic expressions, comments in a liberated line format. English language error messages (not coded numbers). DOS and Apple Monitor commands directly available within the assembler. Speed and suspension control over listing and assembly. Includes printer driver for Practical Automation printer, with instructions for modification to any other printer. (Cassette version is still available: it has fixed line format and labels up to four characters.)

Copies: Over 200 of cassette version, over 25 of disk version.

Price: \$35 for disk version, \$25 for cassette version (Texas residents add 5% sales tax)

Includes: 32-page reference manual, disk with assembler, Master. Create, RAWDOS, and two sample source programs.

Author: Bob Sander-Cederlof

Available from:

S-C SOFTWARE
P.O. Box 5537
Richardson, TX 75080

Name: PRO-CAL-I

System: Commodore PET

Memory: 8K

Language: Microsoft BASIC

Hardware: PET

Description: PRO-CAL-I is a reverse polish programmable scientific calculator program ideally suited to scientific and educational applications. It combines the best features of the PET with those of hand-held calculators such as the HP 97 and the TI "Programmer". It supports single key execution of more than 50 functions and implements calculations in binary, octal, decimal, and hexadecimal number systems. The program displays 10 memory registers, 5 stack registers, and a record of the 14 most current operations.

Copies: 40

Price: \$26.00 for software on cassette and an operating manual.

Author: Robert M. Munoz

Available from:

APPLICATIONS RESEARCH CO.
13460 Robleda Rd.
Los Altos Hills, CA 94022

Name: FINANCIAL ANALYSIS: A Tutorial

System: APPLE II and PET

Memory: 16K

Language: Basic

Hardware: APPLE II with cassette recorder, or a PET (8K)

Description: An interactive learning cassette with chapters on Risk, Short-term and Intermediate-term Financing, Financial Statements, and Key Business Ratios. The user is then put into the position of having to use these concepts by playing the Meany Manufacturing Business Game.

Copies: Hundreds available

Price: Sugg. Retail: \$16.50

Includes: Tape cassette and informative booklet

Author: Brian Beninger

Available from:

Local APPLE or PET dealers of:
SPEAKEASY SOFTWARE LTD.
P.O. Box 1220
Kemptville, Ont., K0G 1J0

Name: STAT III

System: Commodore PET

Memory: 8K

Language: BASIC

Hardware: Standard PET

Description: STAT III accepts a set of numbers and calculates the following: mean, median, mode, highest number in the data, lowest number in the data, range, variance, standard deviation, average deviation, and sample standard deviation. STAT III can display a bar graph of the users data on the CRT. In addition the user may correct errors in his inputted data before processing.

Copies: Just released

Price: \$7.95

Includes: Cassette, source listing (program is self documenting)

Author: Michael J. McCann

Available from:

THE PET PAPER
P.O. Box 43
Audubon, PA 19407

Name: Apple Pi 'Life'
System: Apple II
Memory: 4K
Language: BASIC and assembly
Hardware: Apple II with 2 operable game paddles with switches.
Description: Apple Pi 'Life' allows variable grid sizes from 8X8 up to 40X40 in increments of 1. Paddle 1 is only read when the switch is depressed. Speed is controlled by paddle 0 and can be varied from 550 gpm to 2000 gpm for an 8X8 grid. For a 40X40 grid, speed can be varied from 25 gpm to 140 gpm. The speaker is toggled each time a cell is processed, except at minimum or maximum speed, to give the sounds of 'Life'. The bottom of the grid wraps around to top of grid, and vice-versa. The right of the grid wraps around to left of grid, and vice-versa. There are three tables of pre-defined objects which can be setup on the grid by number and x,y location. A description of the object table structure is given in the documentation. Keyboard controls are: P-pause until next 'P', Z-zero grid and setup objects, O-setup objects on grid, N-new colors, and E-exit program. Any two distinct colors may be used for live and dead cells.

Copies: New, just released.

Price: \$12.00. Texas residents add sales tax.

Includes: Programs, object tables on cassette, documentation.

Order Info: Checks only.

Author: Harry L. Pruetz

Available from:

Microspan Software
2213A Lanier Drive
Austin, TX 78758

Name: Amateur Radio Communications Package
System: Apple II
Memory: 8K (min)
Language: Machine Language and Integer BASIC
Hardware: Apple II and user provided interface
Description: This software package allows the Apple II to communicate in any of three codes: Morse, Baudot, or ASCII, with a minimum amount of external hardware required. Some features include: Variable size text buffer and live keyboard allow preparing text for transmission while receiving or transmitting; 3 field screen display — each field scrolling separately; user defined stored messages are referenced by a keyboard and can be inserted anywhere in the text; automatic 72 character line formatting with word wrap-around; continuously variable code speeds; adaptive Morse receive and lots more! All I/O uses the on-board (game) I/O connector.

Copies sold: over 100

Price: \$18.

Includes: Cassette tape and documentation with sample interface.

Author: Chris H. Galfo - WB4JMD

Available from:

C.H. Galfo
602 Orange St.
Charlottesville, VA 22901

This Catalog is a FREE feature of MICRO. Your entry must be typed, must conform to the standard format, and Applications/Utilities will be given preference over Games.

Name: TRANSACTIONAL ANALYSIS: An Introduction
System: APPLE II and PET
Memory: 16K
Language: Basic
Hardware: APPLE II with cassette recorder, or a PET (8K)
Description: An introduction to T.A. - a system for understanding human behaviour. Chapters include: You As A Person, Stroking, Transactions, Are You Listening?, the Balancing Game. This interactive learning cassette will help you gain better understanding of why you get along with some people and not with others and may give you a better understanding of yourself!
Copies: Hundreds available
Price: Sugg. Retail: \$16.50
Includes: Tape cassette and informative booklet
Author: Joy Karp
Available from:

Local APPLE or PET dealers or:
SPEAKEASY SOFTWARE LTD.
P.O. Box 1220
Kemptonville, Ont., K0G 1J0, Canada

Name: DOS TEXT EDITOR
System: APPLE II
Memory: Cassetts-16K, Applesoft Rom-24K, DOS-32K
Language: Applesoft II
Description: EDIT is a program designed to facilitate changes to disk and cassette text files. The program has 24 commands to manipulate files. Included are: INSERT, DELETE, CHANGE, SEARCH, ADD, LIST, TEXT, DISPLAY, PACK, MODE, TAB, CLEAR, APPEND, SAVE, CONCAT, and STRING CHANGE. Commands that operate on blocks of data such as Range DELETE, LIST, SEARCH, and STRING replace are also provided. EDIT may also be used to create Disk files.

Copies: Just released

Price: \$16.95 (Add \$5 if desired on diskette)

Specify if Applesoft ROM

Includes: Program cassette or diskette, Complete documentation, and users manual.

Author: Robert Stein

Available From:

Services Unique, Inc.
2441 Rolling View Dr.
Dayton, Ohio 45431

Name: REAL-I
System: Commodore PET
Memory: 8K
Language: Microsoft BASIC
Hardware: PET
Description: REAL-I is a real estate investment analysis program which models an investment by computing the cash flow, tax advantage, inflation hedge, internal rate of return, and other quantities as they change over the years under the effects of inflation. It specializes the calculations to the tax position of the investor and helps him to judge the relative merits of various real estate investments opportunities.

Copies: Just released

Price: \$29.00 for software on cassette and an operating manual.

Author: Robert M. Munoz

Available from:

APPLICATIONS RESEARCH CO.
13460 Robleda Rd.
Los Altos Hills, CA 94022

SYM-1 6522-BASED TIMER

John Gieryic
2041 138 Avenue, NW
Andover, MN 55303

Your SYM-1 comes with a number of timers capable of a wide range of timing intervals. Unfortunately the SYM REFERENCE MANUAL does not provide information which can easily be digested by a novice. I'd like to attempt a more down to earth description of timer 1 on the **Versatile Interface Adapter 6522** for those of us who aren't hardware inclined. This timer is capable of very accurate time delays in the range of fractions of a second. It has an interrupt associated with it plus the ability to generate evenly spaced interrupts.

Setting Up The Interrupts

The first step in programming this timer is to place an address in the **Interrupt Request Vector [IRQ]** located at address A67E and A67F. A67E contains the low byte of the address and A67F contains the high byte. This address in the **IRQ** is the location you will be "jerked to" when the timer times down and generates an interrupt. Your code will be as follows:

Location	Code
200	20 86 8B JSR ACCESS disable memory write protect
203	A9 00 LDA #00 interrupt address
205	8D 7E A6 STA A67E Low byte
208	A9 03 LDA #03
20A	8D 7F A6 STA A67F High byte

Our next step is to set two locations so the hardware can "see" the interrupt and tell us where it is coming from. These two locations are the **Interrupt Flag Register [IFR]** at location A00D and the **Interrupt Enable Register [IER]** at location A00E. The **IER** controls interrupts from 7 different sources on the **6522**. We will only be interested in bit 6. This is the one for our timer T1. We must set this bit to a logic 1. This tells the **6522** we will accept interrupts from timer T1. The code follows:

Location	Code
20D	A9 CO LDA #CO
20F	8D OE AO STA AOOE

"Hey, wait a minute! Where did that 'C' come from? I thought you said we were only going to set bit 6?"

Yes, I did. We must supply the **6522** with a bit more information (no pun intended). We must tell it we are going to **SET** one of the **IER** bits. This is done by setting bit 7 to a logic 1, hence our CO. Note bits 0 thru 5 are a zero. This tells the **6522** we don't want to change the condition of any of the other bits in the **IER** when we do our store. From this you should be able to see how we **CLEAR** any one of the **IER** bits. You guessed it. Bit 7 will be a logic zero and the **IER** bit(s) to be cleared will be a logic 1.

The **Interrupt Flag Register [IFR]** tells the user which interrupt has occurred (when we get one). This information can be used by the interrupt routine to "see" which element on the **6522** gave us the interrupt. We want to initialize (clear) our flag bit for timer T1 (bit 6). I don't want to disturb any of the other bits. Note clearing a bit in the **IFR** is **not** the same as in the **IER**.

Location	Code
212	AD OD AO LDA AOOD
215	29 BF AND #BF
217	8D OD AO STA AOOD

When we do get an interrupt from any of the enabled **6522** devices (bit = 1 in the **IER**) then bit 7 in the **IFR** and the corresponding bit in the **IFR** will both be set to a logic 1. We can determine if this interrupt came from the **6522** by just looking at bit 7 of the **IFR** (ASL followed by a test of the C bit). If bit 7 is a logic zero then the interrupt came from some other place. This will save some time when we are trying to find out where this interrupt originated. You should log this bit 7 information in the back of your mind since I won't use it here.

Setting Up The Timer

One more step before starting our timer. I'm going to set our timer to the free running mode. This means it will count down, give an interrupt and then immediately begin counting down again. I won't need to worry about instruction cycle times within any timing loops. I know I will get repeated interrupts at the exact interval requested. Setting the **Auxiliary Control Register [ACR]** bit 7 to a logic 1 establishes the free running mode.

Location	Code
21A	A9 CO LDA #CO
21C	8D OB AO STA AOOB

Now we have the four mechanical steps finished...setting up the **IRQ, IFR, IER** and **ACR**. Setting the time delay is next. The T1 timer has two latches (high and low order) and two counters (high and low order). This results in a 16 bit counter. The low order latch is loaded first. In this example I will set up for a delay of .05 seconds. This corresponds to a count of C350 (one count for each microsecond).

Location	Code
21F	A9 50 LDA #50 load low order latch
221	8D 06 AO STA A006

Method 2

Now we will load the high order latch with the value C3. This instruction will do more than load the high order latch. It will also write the high order latch into the high order counter as well as write the low order latch into the low order counter. This one instruction will transfer all 16 bits from the latches to the counter at the same instant. Without this hardware assist we would be unable to load the counter accurately since the counter begins to count down immediately after being loaded.

Location	Code	
224	A9 C3	LDA #C3 load high order latch
226	8D 05 AO	STA A005

The timer is now running and will generate an interrupt .05 seconds (C350) later. This corresponds to 50,000 clock cycles. If you were programming a clock your remaining code at location 229 would now initialize your hours, minutes and seconds counters, initialize the display buffer and then go into a tight loop calling SCAND in order to illuminate the LED's.

Servicing The Interrupt

Our interrupt routine at location 300 is now executed when we receive the interrupt. The first thing we must do is SAVE the processor status and registers. This is done so we can restore these items when we are finished with our interrupt processing and jump back into SCAND from where we were "jerked out."

Location	Code	
300	08	PHP save processor status on stack
301	48	PHA save accumulator on stack
302	8A	TXA transfer X to A
303	48	PHA save X register on stack
304	98	TYA transfer Y to A
305	48	PHA save Y register on stack

If you were programming a clock you would now increment a counter. If the counter equalled twenty then reset it and increment the time in the display buffer by one second.

Now the interrupt is "serviced." In order to clear the way for the next interrupt, the T1 interrupt flag must be reset otherwise the next interrupt will be blocked. This clearing can be done in either of two ways. Method 1 will write into the high order latch. This write uses a different address for the store instruction than the write used to initialize the timer counter. In doing this the T1 interrupt flag will be reset but it will not disturb the current value in the counter. Remember this is a free running counter in our example and automatically resets itself when the interrupt occurred. By this point in time it has already counted down from its original value of C350 toward zero (and the next interrupt). Method 2 will read the low order counter. Either method will reset the T1 interrupt flag.

Method 1

Code	
A9 C3	LDA #C3
8D 07 AO	STA A007

Code

AD 04 AO LDA A004

Now the processor status and registers can be restored and a return executed to the location in SCAND at which the interrupt occurred. Remember you **must** restore the registers in the exact reverse order used at the entrance to the interrupt routine. This is a major point.

Code

68	PLA	pull accumulator from stack
A8	TAY	transfer to Y index
68	PLA	pull accumulator from stack
AA	TAX	transfer to X index
68	PLA	pull accumulator from stack
28	PLP	pull processor status from stack
40	RTI	Return from Interrupt

That's the end of the lesson for today. In a future article I will use the information presented here to develop an operating system for your SYM-1.

KIM™ BUS EXPANSION!

AIM™, VIM™, (SYM)™, KIM™ OWNERS
(and any other KIM™ bus users) buy the
best 8K board available anywhere:

GRAND OPENING SPECIAL!

HDE 8K RAM-\$169! 3 for \$465!

Industrial/commercial grade quality: 100 hour high temp burn-in; low power; KIM bus compatible pin for pin; super quality & reliability at below S-100 prices (COMMERCIALY rated S-100 boards cost 25-75% more). When you expand your system, expand with the bus optimized for 8 bit CPU's, the Commodore/Mos Technology 22/44 pin KIM bus, now supported by Synertek, MTU, Rockwell International, Problem Solver Systems, HDE, the Computerist, RNB, and others!

KIM-1 computer \$179.00; KIM-4 Motherboard \$119; power supply for KIM-1 alone—\$45; enclosure for KIM-1 alone \$29; HDE prototype board with regulator, heatsink, switch address & decoding logic included \$49.50; book "The First Book of KIM" \$9.95; book "Programming a Microcomputer: 6502" \$8.95; SPECIAL PACKAGE DEAL: KIM-1, power supply, BOTH books listed above, ALL for \$209!

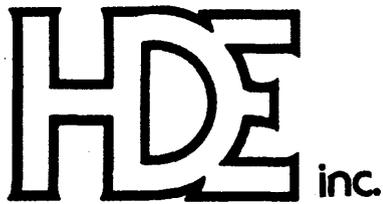
HDE FILE ORIENTED DISK SYSTEM (FODS) FOR KIM BUS COMPUTERS Make your KIM (or relative) the best 6502 development system available at any price. Expand with HDE's full size floppy system with FODS/Editor/Assembler. 2 pass assembler, powerful editor compatible with ARESO files KIM bus interface card: fast 6502 controller handles data transfer at maximum IBM single density speed for excellent reliability; power supply for 4 drives: patches to Johnson Computer/Microsoft BASIC. 45 day delivery. Single drive—\$1995 dual drive \$2750.

Shipping extra unless order prepaid with cashier's check ALL items assembled, tested, guaranteed at least 90 days.

PLAINSMAN MICRO SYSTEMS (div. 5C Corporation)
P.O. Box 1712, Auburn, AL 36830: (205)745-7735
3803 Pepperell Parkway, Opelika
[1-800-633-8724] Continental U.S. except AL.
Dealers for OSI, COMMODORE, COMPUCOLOR.

VISA

ALTOS



BOX 120
ALLAMUCHY, N.J. 07820
201-362-6574

HUDSON DIGITAL ELECTRONICS INC.

JUST THINK OF IT!

YOUR KIM-1 — No longer limited to those long cassette saves and loads.

YOUR KIM-1 — Backed up by 8K static ram so conservatively designed, well manufactured and thoroughly tested, HDE includes a no-nonsense, unconditional, 6 months parts and labor warranty. (Excluding misuse).

YOUR KIM-1 — Transformed into one of the most powerful 6502 development systems available today.

HDE, INC. supports the KIM-1 with 8" and 5" single and dual drive disk systems, prototyping cards, card racks, desk top cabinets, motherboards, and the finest memory board available, anywhere, at any price.

AND THIS IS JUST FOR STARTERS . . .

Consider: A fast, 2 pass assembler; a complete line oriented editor; a comprehensive text output processing system; an efficient dynamic debugging tool; and, a memory diagnostic package so thorough it's the basis of our memory warranty.

Plus, after the sale support that you've known you deserve, but just couldn't seem to get — until now.

And, HDE products are KIM-1, KIM-4 compatible. All boards include an oversized 5 volt regulator and address selection switches, in a state-of-the-art 4.5" X 6.5" format, designed, manufactured, and tested for commercial/industrial application.

HDE products — built to be used with confidence.

AVAILABLE FROM THESE FINE DEALERS:

JOHNSON COMPUTER
Box 523
Medina, Ohio 44256
216-725-4560

PLAINSMAN MICROSYSTEMS
Box 1712
Auburn, Ala. 36830
800-633-8724

ARESCO
P.O. Box 43
Audubon, Pa. 19407
215-631-9052

THE TVT-6: A USER'S REPORT

Edward Chalfin
447 Hendrix Street
Philadelphia, PA 19116

As a computer hobbyist who wanted a video interface for his KIM-1, it took me a long time to decide which video board of the many available to choose. The main factor which influenced my decision was the prohibitive cost of most video interfaces, typically around \$150 to \$200. Being on a college student's budget, these prices were way out of my range. I was beginning to lose all hope when, at the PCC '77 in Atlantic City, I discovered the TVT-6, a video board for the KIM-1, for only \$35!

After mulling it over for a few days (\$35 is still a lot of money!) and weighing the board's strengths and weaknesses, I decided to order a TVT-6 kit from PAIA Electronics. I received my kit in about 2 weeks and built it in one night. The construction was fairly easy, although it may have been slightly more difficult for someone with no construction experience. This was due mainly to the fact that the instructions consisted of a reprinted construction article from a magazine. The board worked the first time I hooked it up except for a capacitor which I found out later, after a call to PAIA, had to be a slightly lower value. The effect of the bad capacitor was to widen the characters on the screen to the point that they interfered with each other. I must mention that the Technical Services Dept. at PAIA was very helpful and sent the correct value capacitor to me within a few days.

I must say that Don Lancaster's TVT-6 design is truly amazing but, due to the fact that it uses the 6502 for its timing, it has a few drawbacks. In the remainder of this article I will try to describe the TVT-6's strengths as well as its weaknesses as I see them, having used this video board for over a year.

The main weakness one encounters using the TVT-6 is the fact that the display disappears whenever the display program is not running. When using an interrupt, such as an ASCII keyboard, the problem is not as apparent as when implementing another routine which is relatively long in execution time. Also, if the display program is repeatedly called as a subroutine from another program, a noticeable jitter in the display results, which can be very annoying. A good example of this would be trying to play a 'pong' type game on the screen.

Another minor annoyance is that the display program must be loaded from tape every time the computer is powered up before the display will work. Also, for KIM owners without expansion memory, pages 2 and 3 of the KIM RAM cannot be used for program space without garbage showing on the screen.

One drawback is the fact that memory locations \$8000 through \$DFFF are used by the TVT-6 scan PROM and cannot be used for expansion memory.

Most of the above weaknesses may seem pretty important at first, but in fact they are not, and there are ways of getting around them.

Since the screen flickers when called repeatedly by another program, such as a disassembler listing, I simply have the disassembler fill the screen before calling the display routine, thus displaying whole pages at a time.

Though locations \$8000 through \$DFFF cannot be used for expansion memory, this still leaves space for 36K of expansion, which I feel is more than enough for the average KIM owner.

Now that I have outlined the weaknesses of the TVT-6 and some ways of getting around them, I will describe its strengths, which, I feel, far outweigh the disadvantages.

The most outstanding advantage of the TVT-6 is its \$35 price tag. As far as I know, no other video board even comes close to that price range. Add to that its lower power consumption, (I have used the KIM +5 volt supply to power the KIM and the TVT-6), its small size, variable display size, and software cursor control, and you have what I believe is one of the biggest hardware buys around. Above all, I think that the TVT-6 display is one of the cleanest and sharpest that I have seen!

In conclusion, I must admit that if you can afford to lay out \$200 for a video interface, then one of the more expensive boards may be what you need. However, if you're looking for a video board that works great and won't empty your wallet, the TVT-6 is definitely for you!

Editor's Note: One important disadvantage which is not mentioned above is that of using the TVT-6 with existing software. If you are planning to write all of your own code, this is no problem. But if, more typically, you are going to use software which was written by others without a TVT-6, you may encounter serious problems. Some packages may be very difficult to interface to the TVT-6; others may be impossible due to timing considerations. The TVT-6 is a remarkable illustration of what can be done with a 6502 and may be a total solution for some applications. But, it may not be adequate in other applications. For a lot more information on the TVT-6, you should consult (buy?) **The Cheap Video Cookbook** by Don Lancaster, 1978. This book is published by Howard W. Sams & Co., Inc., 4300 West 62nd St., Indianapolis, IN 46288 and retails at your local computer shop for \$5.95. This is a good tutorial on videos whether or not you intend to buy/build the TVT-6, but is not for the complete novice.

For a catalog/price list on their line of cheap video pc boards, kits, etc., contact:

PAIA Electronics
1020 West Wilshire Blvd.
Box 14539
Oklahoma City, OK 73114
405/842-5480

6502 BIBLIOGRAPHY

PART X

William R. Dial
438 Roslyn Avenue
Akron, OH 44320

429. Recreational Computing 7 No 4 Iss 37 (Jan./Feb., 1979)

Carpenter, Church "APPLE II Easy I/O Sensing and Control". I/O control using the Apple II game connector.
Wells, Arthur Jr. "Some New Uses for Apple II". Debugging PONG, use of Modem, etc.
Shanis, Daniel "Breaking Trail in Alaska with Apple II". A project using 32K Apple II computers with diskettes in 9 remote village schools.
Swenson, Carl "Building a HI-RES SHAPE TABLE for the APPLE II". Here's a way to create your favorite shapes.
Saal, Harry "SPOT". Tips for the PET Owners. A machine language tape with two BASIC programs is available from Commodore. Also a manual on communication with the outside world. A PET SERVICE KIT from Commodore includes schematic diagrams and parts lists, a diagnostic jumper connector with diagnostic tapes, etc. Information on the "lost cursor fix". Head Alignment.

430. MICRO No 9 (Feb., 1979)

Reich, Dr. L.S. "Long Distance Interstate Telephone Rates". An Applesoft II program for phone rates which can be modified for PET or OSI computers.
Bullard, GARY J. "The Sieve of Eratosthenes". A prime numbers BASIC program for the PET.
Hertzfeld, Andy "Exploring the Apple II DOS". Useful information for disk users.
DeJong, Marvin L. "6502 Interfacing for Beginners: An ASCII Keyboard Import Port". Shows a system for the KIM with both polled or interrupt methods of service the device.
Tater, Gary L. "Two Short TIM Programs". One program provides a method for communication with TIM at 1200 BAUD or higher. Another offers a TIM Operating System Menu.
Tripp, Robert M. PhD "Ask the Doctor-Part 1". A comparison of the KIM, SYM and AIM microcomputers.
Watson, Allen "Two APPLE II Assemblers: A comparative Software Review". Advantages and disadvantages of the Microproducts and S-C Assemblers for the Apple II.
Rowe, Mike "The MICRO Software Catalog: V". Reviews of about one dozer programs for 6502 based systems.
Rittmann, Russell "Expand Your 6502-Based TIM Monitor". A modification of the TIM system to expand the command set so that ROM resident programs or routines can be executed from within TIM.
Dial, Wm R. "6502 Bibliography— Part VIII". The 6502 literature continues to expand.
Sandberg, Gary P "How Does 16 Get You 10?". Hexadecimal/Decimal conversions for the Apple.
Herman, Harvey B. "How Does Your ROM Today". Programs and techniques for testing the KIM and PET ROMs.
Bridge, Theodore E. "Life for the KIM-1 and an XITEX Video Board". Program runs on a 16K Kim.

431. CONTACT Newsletter No 4 (Dec. 1979)

Anon "Apples work PIA's". A note to the effect that the problems reported earlier by END magazine on the apparent incompatibility of the Apple with PIA's have been resolved and that EDN now believes this long saga must have had its source in human error. (See EDN Magazine, Sept. 20, 1978)
Anon, "The Colon as a Listing Formatter for Applesoft". How to indent your listings for neatness and easy reading.
Annon "Disk Operating System Notes". Includes Notes on Data Format, Using Random-Length Records, Using Fixed-Length Records, Appending Files, DOS Error Codes, Getting Commas into Applesoft, etc.

432. Dr. Dobb's Journal 4 No 32 Issue 2 (Feb., 1979)

Gordon, H.T. "An Unusual Pseudorandom Number Generator Program". Program for the KIM-1.
Carpenter, Chuck "Reset Adapter". How to avoid accidental loss of programs involving the reset button on the Apple II
Prigot, Jonathan M. "Loading Kim's Cassettes". How to load OSI cassettes into the KIM.

433. Byte 4 No 2 (Feb., 1979)

Libes, Sol "Byte News". Atari has two new 6502 based computers. According to the latest sales reports, more 6502 microprocessors are being manufactured than any other uP. Most of the volume goes to high volume game use.
Mathews, Dr. Randall S. "An Apple and the Queens". An Eight Queens program for the Apple.
Raskin, Jef "Unlimited Precision Division". A BASIC program for unlimited integer division.

434. Kilobaud No 26 (Feb., 1979)

Green, Wayne "Publisher's Remarks". A review of OSI's new units the Ip and IIP Challengers.
Lindsay, Len "PET Pourri". Accessories for the PET include a voice input module, a sound output module, Joysticks, a digital plotter, a light pen, and an S-100 adapter for the PET. How to add sound to your PET and sound programming instructions. New languages to supplement Basic are PILOT and PETFORTH. New sources of information on the PET are the PET Manual and a manual called PETABLE, as well as a newsletter called Sphinx. Programming tips cover the GET, ON, .GOSUB, and others. A new wrinkle for recovering programs from faulty tapes is given.

Flogel, Ekkehard "Apple and the PIA". Contradicting the troubles reported by EDN magazine, a board was developed with a PIA 6520 on it to put an Apple II and a KIM together. Programs can be sent from one unit to the other and vice versa.
 Price, David "Music, Maestro". The AD8 is a computer-controlled synthesizer system using a 6502 microprocessor and a 6820 I/O port.
 Bishop, Robert J. "The Apple Speaks... Softly". Apple II Voice digitizer.

435. **Calculators/Computers Magazine 3 No 1 (Jan./Feb., 1979)**
 Day, Jim "High-Resolution Apple Art". Applesoft II program for various shapes.
 Albrecht, Bob and Karl "PET BASIC for Parents and Teachers". PET Conventions in a simple animation program.
436. **73 Magazine No 221 pg 21 (Feb., 1979)**
 Birman, Paul "Petting". How to find the end of a program on tape when you want to load a new program into your PET.
437. **Personal Computing 3 No 2 pg 63-74 (Feb., 1979)**
 Gerue and McNeil, "Chess Challenger-10 Wins Microchess Tourney". Microchess 2.0, Peter Jennings entry, took fourth place. This is 6502 based.
437. **Creative Computing 5 No 1 (Jan., 1979)**
 Yob, Gregory "Personal Electronic Transactions". New products described are Expandapet memory, PET ROM disassemblies, a useful book on what the PET rom is all about, Some data on the User Port, PET Video Slave display, Exploring PET random numbers, PET sounds and music, etc.
 Wells, Ralph "HOW about a 'Counterfeit Cursor' For your PET?". Enables one to use the cursor in games or under better control.
 Heuer, Randy "Ohio Scientific Superboard II and Challenger 1P". A review of OSI's new lost cost micro.
 Rugg and Feldman "Speed Reading Made Easy...via Your PET". This program turns your computer into a tachistoscope to teach improved reading habits.
 Milewski, Richard A. "Apple-Cart". All about Data Files on diskettes. Simple file accessing statements, Sample serial access programs, and some software reviews.

NOW AVAILABLE

Basic Software

For SOL-IIA and PET-8K

General Pack 1

(Checkbook Balancer, Tic Tac Toe, Metric Conversion) \$10.95

General Pack 2

(Space Patrol, Biorhythm, Battlestar, One-Armed Bandit) 18.95

Financial Pack 1

(loans, Depreciation, Investments) 12.95

Financial Pack 2

(Mortgage & Loan Amortization, Future Projections, Risk Analysis) 12.95

Statistics Pack 1

(Mean & Deviation, Linear Correlations & Regression, Distribution, Contingency Table Analysis) 18.95

Game Pack 1

(Basketball, Object Removal, Bowling, Darts, Gopher) 22.95

Game Pack 2 -

(children - educational) 12.95
 (Arithmetic God, Addition Dice, Distance = Rate X Time) 12.95

Tape Data Query

(File Management System) - For the KIM-1 50.00

PCROS - a Real-Time Operating System in 1K KIM RAM

Assembly listing 24.95

Cassette tape with user's manual 14.95
 Schematic for relay control board 9.95

All programs on high-quality cassette tape.

Send self-address, stamped envelope for complete software catalogue.

Send check or money order to:

H. GELLER COMPUTER SYSTEMS

Dept. M

P.O. Box 350

New York, New York 10040

(New York residents add applicable sales tax)

THE ULTIMATE PET RENUMBER

Don Rindsberg
The Bit Stop
Box 973
Mobile, Alabama 36601

This article can be of help to the BASIC programmer in providing a fast, fool-proof renumbering system, but it also includes details on how to use the PET BASIC interpreter's own machine-language routines to do some useful chores.

Renumbering programs written in BASIC, such as Jim Butterfield's (see MICRO Dec 78 - Jan 79) are very slow in renumbering long programs, and because BASIC is cumbersome in performing such routine chores, the machine-language approach has some major advantages. This routine will renumber a 300-line program in around 20 seconds, as compared to more than 300 seconds for Jim's BASIC version. Further, Jim is forced to duck the issue of providing space for extra-digit line numbers, whereas by calling BASIC's line insertion routine, this program provides enough space for five digits for every GOTO, GOSUB, etc.

The entire program for renumbering is given in hexadecimal in listing 1. More later about how to enter it into your machine. With your BASIC program and the renumber routine in RAM, press SYS8181 (by coincidence, the name of the program) and you will either get a message of reassurance that all has gone well, or will get an error message, such as "line too long". In no case will the program bomb, because this is a two-pass program; during the first pass, nothing is done to the Basic text, other than making sure there is enough space for five-digit line numbers. If any problem exists, the BASIC text is unchanged.

DEVELOPING THE PROGRAM

Commodore made it a formidable task to decipher the code of BASIC sufficiently to be able to make patches for a short renumber system. The first obstacle is that the PEEK statement is disabled for the area of memory where BASIC resides. But, by sleight-of-hand, a little PUNCHing and POKEing and addition of a simple output port on PET's memory-expansion connector, the PET disgorged the contents of its ROMs into my homebrew machine and onto a disk; now, with the capability of having the programs in RAM, where breaks could be inserted for diagnosis, the job became a little easier.

Programming a renumber routine is made tedious by the fact that, in the BASIC text, the line numbers following the GOTO tokens are coded in ASCII, whereas the line numbers at the beginning of a line of text are coded as two-byte hex numbers. Fortunately, the BASIC interpreter has routines built in to do these conversions back and forth between ASCII and hex. The locations of these and other routines called by this program are given in TABLE 1. Another problem encountered was locating some page zero registers, essential to 6502 programming, which are not altered by the BASIC itself. In some cases, I use space in the line buffer at 000A-0059, but this cannot be done in the section of the Program which uses the line buffer for its original purpose, i.e., inserting a line in its proper place in the BASIC text.

This program uses very little RAM, since no tables are created.

PROGRAM OPERATION

The program first sets or clears a flag, depending upon entry point (DCM 8181 or 8184), since entry point determines whether the

renumber job is standard or custom. It then checks to see if sufficient memory exists to allow for insertion of spaces for as many as five digits for GOTO line numbers. An error message (see TABLE 2) is generated if there is less than one page available for this enlargement of the program. Then, each line of text is moved into the line buffer, and if a GOTO, GOSUB, or THEN (followed by a number) is present, spaces are inserted and the expanded line is inserted by BASIC's own line-insertion routine into the text area, just as though you retyped the line on the keyboard. Any lines too long for this expansion produce an error message before any harm is done to the program. BASIC's own error routine is called to print these messages! The "TOO LONG" message is a shortened version of "STRING TOO LONG" used by BASIC.

In the text, all statements are compressed into single-byte tokens, which I have listed for your reference in TABLE 3. For example, GOSUB is hex 8D, THEN is A7, etc. This program searches out all the 89, 8D and A7 tokens. Getting the proper ASCII numbers after these tokens requires conversion of the ASCII to hexadecimal and searching for a matching line number in the text area. If no match is found, the guy evidently had a GOTO pointing to a non-existent line number, so we flag this in the text by an opening parenthesis, such as:

```
GOTO(  
GOSUB( :X=X+1  
IF A=B THEN(  
ON X GOTO 1234,( ,5678,9987
```

When the progra is listed or run, the need for correction is obvious. While we are searching for a matching line number, we keep track of the new line number which corresponds to the current position in the text, so that when the match is found, the new line number can be converted to ASCII and placed directly into the text. The actual resequencing process which follows is an anticlimax, because it requires so little coding (1E16-1E3E). When the entire renumbering job is done, we jump back to BASIC's warm start location.

USING THE PROGRAM

If you would like to renumber your program with the standard starting line number 100 and increment by 10, simply type SYS8181, which directs the program to hex address 1FF5. If you would like to choose a different starting line number or increment, POKE the desired values at the addresses shown in LISTING 2, and type SYS8184 to enter the program at 1FF8. If your BASIC program is long, it may take 3-4 seconds to complete the renumbering job. After renumbering, running the program will generally write over the renumber code, since it occupies the same space as some BASIC variables. The only precaution to be taken in renumbering is to avoid line numbers which exceed PET's limit of 63999.

ROUTINE ENTRY POINT (HEX)	FUNCTION AND IMPLEMENTATION
C359	Print an error message from the message table. Enter with X containing the location of the message relative to C190. Message terminator is ASCII having bit 7 on.
1F00	A duplicate of the original BASIC line insertion routine located at C3B4, except for the exit jump. Enter with the line assembled in the line buffer 000A-0059 with 00 as line terminator. Also, the character count must be in 005C, and the line number (hex) at 0008/9.
CCA4	Evaluate an expression whose beginning address is in 00C9/CA. We use this sub to convert from ASCII to binary, with the result appearing in the floating accumulator 00B0†.
DB1B	Convert fixed number in 00B1/2 to floating number. Enter with X=90 and carry set.
D6D0	Convert binary value, such as line number, in floating accumulator to two-byte fixed number and place in 0008/9.
DCAF	Convert floating number at 00B0† to ASCII and place in a string starting at 0101, preceded by a space or minus sign at 0100 and terminated by 00.
C38B	BASIC warm start. Prints READY.
CA27	Print message. Enter with ADH in Y, ADL in A. Message is ASCII string enough with 00.
DC9F	Print the decimal integer whose hex value is in microprocessor registers A and X, for example, a line number.

TABLE 1- BASIC ROUTINES USED

MESSAGE	INTERPRETATION
CHECK FOR GOTO(ETC	Successful renumbering.
120 ? TOO LONG ERROR	Line 120 is too long to renumber. Break into two or more lines, and renumber again.
? OUT OF MEMORY ERROR	Program too long to renumber.
? SYNTAX ERROR	Attempt to RUN program with GOTO(remaining in program, or attempt to renumber with one of these in program text.
GOTO(GOSUB(ON X GOTO(IF A=B THEN(The opening parenthesis in the text represents attempt to reference a non-existent line number.

Note: Lines of the following form are likely to cause a TOO LONG error:

100 ON X GOSUB 1,2,3,4,5,6,7,8,9,10,11,12

TABLE 2- MESSAGES

RENUMB ORG \$1D00
 DON RINDSBERG
 (C) 1978 N.A.I.L.

(& SIGN MEANS PLUS)

EXTERNAL ROUTINES

INSERT . \$1F00 INSERT A LINE INTO TEXT
 MESSG . \$1FCA DONE MESSAGE

TEMPORARIES

BUFF . \$0008 LINE BUFFER LOCATION
 POINT . \$0019 TEMP LINE BUFF POINTER
 POINTX . \$001A TEMP POINTER
 LINCNT . \$005C NO. CHAR. IN LINE
 PTRSO . \$007A ORIGINAL POINTERS
 PTRS . \$006A WORKING POINTERS
 FLAG . \$0069 FLAG THE GOTOS
 BUFPTR . \$006E LINE BUFF POINTER PAGE ZERO
 COUNT . \$006F COUNTER
 STARTC . \$00DB CUSTOM STARTING LINE NO.
 INTC . \$00DD CUSTOM INTERVAL
 CUSTOM . \$00DE FLAG CUSTOM JOB

BASIC PARAMETERS

FACC . \$00B0 BASIC FLOATING ACCUM
 BASICP . \$00C9 BASIC POINTER
 BERROR . \$C359 BASIC ERROR ROUTINE
 WARM . \$C38B BASIC WARM START
 PRINT . \$CA27 BASIC PRINT ROUTINE
 EVAL . \$CCA4 EXPRESSION EVALUATOR
 FIX . \$D6D0 CONVERT TO FIXED DP
 FLOAT . \$DB1B CONVERT FIXED NMBR TO FLOAT
 PNUMBR . \$DC9F BASIC PRINT NUMBER
 ASCII . \$DCAF CONVERT NMBR TO ASCII AT \$0100

MAINLINE

1D00	A5	7D	START	LDA	PTRSO	&03	GET END TEXT ADH
1D02	C9	1B		CMPIM	\$1B		ENOUGH ROOM TO EXPAND?
1D04	90	05		BCC	SPACE		
1D06	A2	52	BOMB	LDXIM	\$52		OUT OF MEMORY
1D08	4C	FC		JMP	ERROR		
1D0B	20	BD	SPACE	JSR	COPY		MAKE CC TEXT POINTERS
1D0E	20	3F	NEXT	JSR	DNTST		ARE WE DONE THIS SECTION?
1D11	F0	2B		BEQ	RENUM		
1D13	A2	08		LDXIM	\$08		LINE BUFFER START

1D15	A0	02		LDYIM	\$02	POINT TO LINE NMBR IN TEXT
1D17	B1	6A	GETBYT	LDAIY	PTRS	GET BYTE FROM TEXT
1D19	95	00		STAZX	\$00	STORE IN LINE BUFFER
1D1B	C0	04		CPYIM	\$04	ZERO HERE NOT TERMINATOR
1D1D	90	04		BCC	SKIPA	
1D1F	C9	00		CMPIM	\$00	
1D21	F0	04		BEQ	TERM	GOT THE TERMINATOR
1D23	C8		SKIPA	INY		
1D24	E8			INX		
1D25	D0	F0		BNE	GETBYT	FORCED BRANCH
1D27	20	47	1E	TERM	JSR	EDIT ONE LINE
1D2A	A5	69		LDAZ	FLAG	
1D2C	D0	0A		BNE	SKIPB	SKIP IF NO GOS FLAGGED
1D2E	38			SEC		
1D2F	A5	6E		LDA	BUFPTR	
1D31	E9	05		SBCIM	\$05	CORRECT BYTE COUNT
1D33	85	5C		STA	LINCNT	NEED CHAR COUNT
1D35	4C	00	1F	JMP	INSERT	BUT RETURN TO NEXT LINE
1D38	20	C7	1E	SKIPB	JSR	UPDATE POINT TO NEXT LINE
1D3B	4C	0E	1D	JMP	NEXT	
1D3E	20	BD	1E	RENUM	JSR	COPY THE POINTERS
1D41	20	3F	1E	NEXTR	JSR	DNTST ARE WE DONE THIS PORTION?
1D44	D0	03		BNE	NOTDON	
1D46	4C	16	1E	JMP	RESEQ	
1D49	20	AE	1F	NOTDON	JSR	STRTLN GET STARTING LINE NMBR
1D4C	A0	03		SCAN	LDYIM	\$03 POINT TO TEXT-1
1D4E	C8			SCANA	INY	
1D4F	B1	6A		SCANX	LDAIY	PTRS GET A BYTE
1D51	D0	06		BNE	GOTEST	BRANCH IF NOT TERMINATOR
1D53	20	C7	1E	JSR	UPDATE	GO TO NEXT LINE
1D56	4C	41	1D	JMP	NEXTR	
1D59	C9	89		GOTEST	CMPIM	\$89 GOT A GOTO?
1D5B	F0	15		BEQ	GOTO	
1D5D	C9	8D		CMPIM	\$8D	GOT A GOSUB?
1D5F	F0	11		BEQ	GOTO	
1D61	C9	A7		CMPIM	\$A7	GOT A THEN?
1D63	D0	E9		BNE	SCANA	
1D65	C8			THEN	INY	POINT TO NEXT
1D66	B1	6A		LDAIY	PTRS	
1D68	C9	20		CMPIM	\$20	IGNORE SPACES
1D6A	F0	F9		BEQ	THEN	
1D6C	20	E5	1E	JSR	TSTDGT	TEST FOR NUMBER
1D6F	B0	E8		BCS	GOTEST	
1D71	88			DEY		
1D72	C8			GOTO	INY	
1D73	84	19		STY	POINT	SAVE A MOMENT
1D75	98			TYA		
1D76	18			CLC		
1D77	65	6A		ADC	PTRS	POINT TO ASCII NMBRS
1D79	85	C9		STA	BASICP	
1D7B	20	ED	1F	JSR	PATCH	BUG FIX
1D7E	EA			NOP		
1D7F	20	A4	CC	JSR	EVAL	CALL BASIC EVALUATOR
1D82	20	D0	D6	JSR	FIX	AND BASIC FIX ROUTINE

1D85 A5 7A	SEARCH	LDA	PTRSO	SETUP SEARCH POINTERS
1D87 85 1A		STA	POINTX	
1D89 A5 7B		LDA	PTRSO &01	
1D8B 85 1B		STA	POINTX &01	
1D8D A0 00	SRCHLP	LDYIM	\$00	
1D8F B1 1A		LDAIY	POINTX	GET NEXT BYTE
1D91 C8		INY		
1D92 11 1A		ORAIY	POINTX	TEST FOR TWO ZERO BYTES
1D94 D0 10		BNE	NOTEND	ZEROES MARK EOT
1D96 A9 20		LDAIM	\$20	GET A SPACE
1D98 8D 00 01		STA	\$0100	ASCII WORKSPACE
1D9B A9 28		LDAIM	\$28	GET OPEN PAREN
1D9D 8D 01 01		STA	\$0101	
1DA0 88		DEY		
1DA1 8C 02 01		STY	\$0102	TERMINATE WITH ZERO
1DA4 F0 20		BEQ	MVASC	FORCED BRANCH
1DA6 A0 02	NOTEND	LDYIM	\$02	
1DA8 B1 1A		LDAIY	POINTX	GET LINE NO. LOW
1DAA C5 08		CMP	BUFF	MATCH?
1DAC D0 55		BNE	NOMAT	
1DAE C8		INY		
1DAF B1 1A		LDAIY	POINTX	GET LINE NO. HIGH
1DB1 C5 09		CMP	BUFF	&01
1DB3 D0 4E		BNE	NOMAT	
1DB5 A6 10	MATCH	LDX	BUFF	&08 GET CURRENT LINE NMBR
1DB7 86 B2		STX	FACC	&02
1DB9 A5 11		LDA	BUFF	&09 SECOND BYTE
1DBB 85 B1		STA	FACC	&01
1DBD A2 90		LDXIM	\$90	SETUP FOR FLOAT
1DBF 38		SEC		
1DC0 20 1B DB		JSR	FLOAT	
1DC3 20 AF DC		JSR	ASCII	TO \$0101 PLUS
1DC6 A2 FB	MVASC	LDXIM	\$FB	MINUS 5
1DC8 A4 19		LDY	POINT	
1DCA BD 06 00	LOOPA	LDAAX	\$0006	
1DCD F0 08		BEQ	BLANKS	TERMINATOR ZERO
1DCF 91 6A		STAIY	PTRS	
1DD1 C8		INY		
1DD2 E8		INX		
1DD3 D0 F5		BNE	LOOPA	
1DD5 F0 0C		BEQ	COMMA	
1DD7 A9 20	BLANKS	LDAIM	\$20	GET SPACE
1DD9 91 6A		STAIY	PTRS	STORE IT
1ddb C8		INY		
1DDC E8		INX		
1DDD D0 F8		BNE	BLANKS	
1DDF 88		DEY		
1DE0 D0 01		BNE	COMMA	
1DE2 C8	COMMX	INY		
1DE3 B1 6A	COMMA	LDAIY	PTRS	GET NEXT BYTE
1DE5 20 E5 1E		JSR	TSTDGT	TEST FOR NUMBER
1DE8 B0 06		BCS	NOTNUM	
1DEA A9 20		LDAIM	\$20	SPACE
1DEC 91 6A		STAIY	PTRS	STORE IT
1DEE D0 F2		BNE	COMMX	FORCED
1DF0 C9 20	NOTNUM	CMPIM	\$20	SPACE?

1DF2	F0	EE		BEQ	COMMX	
1DF4	C9	2C		CMPIM	\$2C	COMMA?
1DF6	08			PHP		DEFER TEST
1DF7	20	AE	1F	JSR	STRTLN	GET STARTING LINE NMBR
1DFA	28			PLP		NOW TEST
1DFB	D0	03		BNE	JSCANX	NOT COMMA
1DFD	4C	72	1D	JMP	GOTO	GOT A COMMA
1E00	4C	4F	1D	JMP	SCANX	
1E03	20	EE	1E	JSR	INCLIN	INCR NEW LINE NMBR
1E06	A0	00		LDYIM	\$00	
1E08	B1	1A		LDAIY	POINTX	GET NEXT LINE ADDRESS
1E0A	48			PHA		
1E0B	C8			INY		
1E0C	B1	1A		LDAIY	POINTX	
1E0E	85	1B		STA	POINTX	&01
1E10	68			PLA		
1E11	85	1A		STA	POINTX	
1E13	4C	8D	1D	JMP	SRCHLP	BACK TO SEARCH AGAIN
1E16	20	AE	1F	RESEQ	JSR	STRTLN SETUP STARTING LINE
1E19	20	BD	1E		JSR	COPY COPY THE POINTERS
1E1C	20	3F	1E	LOOPR	JSR	DNTST DONE?
1E1F	F0	13			BEQ	WINDUP
1E21	A0	02			LDYIM	\$02 POINT TO LINE NMBR
1E23	A5	10			LDA	BUFF &08 GET NEW ONE
1E25	91	6A			STAIY	PTRS STORE IT
1E27	C8				INY	
1E28	A5	11			LDA	BUFF &09
1E2A	91	6A			STAIY	PTRS
1E2C	20	C7	1E		JSR	UPDATE ADVANCE TO NEXT LINE
1E2F	20	EE	1E		JSR	INCLIN INCREMENT LINE NMBR
1E32	90	E8			BCC	LOOPR FORCED
1E34	A0	1F		WINDUP	LDYIM	MESSG /100
1E36	A9	CA			LDAIM	MESSG
1E38	20	27	CA		JSR	PRINT END MESSAGE
1E3B	58				CLI	ALLOW KEYPRESSES
1E3C	4C	8B	C3		JMP	WARM BACK TO BASIC
1E3F	A0	00		DNTST	LDYIM	\$00
1E41	B1	6A			LDAIY	PTRS GET NEXT BYTE
1E43	C8				INY	ADVANCE TO NEXT
1E44	11	6A			ORAIY	PTRS OR WITH LAST TO FIND 0000
1E46	60				RTS	
1E47	A2	09		EDIT	LDXIM	BUFF &01
1E49	86	6E			STX	BUFPTR
1E4B	86	69			STX	FLAG SET FLAG
1E4D	E6	6E		EDITX	INC	BUFPTR
1E4F	A6	6E			LDX	BUFPTR
1E51	B5	00			LDAZX	\$00
1E53	F0	71			BEQ	RTS
1E55	C9	89		EDITY	CMPIM	\$89 GOTO?
1E57	F0	19			BEQ	SPACES

1E59	C9	8D		CMPIM	\$8D	GOSUB?
1E5B	F0	15		BEQ	SPACES	
1E5D	C9	A7		CMPIM	\$A7	THEN?
1E5F	D0	EC		BNE	EDITX	BACK FOR MORE
1E61	E6	6E	THENN	INC	BUFPTR	
1E63	A6	6E		LDX	BUFPTR	
1E65	B5	00		LDAZX	\$00	BYTE AFTER THEN
1E67	C9	20		CMPIM	\$20	IGNORE SPACES
1E69	F0	F6		BEQ	THENN	
1E6B	20	E5	1E	JSR	TSTDGT	IS IT NUMBER?
1E6E	B0	E5		BCS	EDITY	IF NOT, GO BACK
1E70	C6	6E		DEC	BUFPTR	
1E72	A2	09		LDXIM	BUFF	&01 TEXT-1
1E74	E8		SPACES SPACEX	INX		
1E75	B5	00		LDAZX	\$00	LOOK FOR TERMINATOR
1E77	D0	FB		BNE	SPACEX	
1E79	E0	54		CPXIM	\$54	LINE TOO LONG?
1E7B	90	0C		BCC	OKAY	
1E7D	A5	09		LDA	BUFF	&01
1E7F	A6	08		LDX	BUFF	GET BAD LINE NMBR
1E81	20	9F	DC	JSR	PNUMBR	PRINT IT
1E84	A2	BB		LDXIM	\$BB	TOO LONG MESSG
1E86	4C	FC	1E	JMP	ERROR	
1E89	A2	06		LDXIM	\$06	DIGITS PLUS ONE
1E8B	86	6F	OKAY	STX	COUNT	
1E8D	E6	6E	LOOP	INC	BUFPTR	
1E8F	C6	6F		DEC	COUNT	
1E91	F0	12		BEQ	COMMAS	
1E93	A6	6E		LDX	BUFPTR	
1E95	B5	00		LDAZX	\$00	
1E97	C9	20		CMPIM	\$20	TEST FOR SPACES
1E99	F0	F2		BEQ	LOOP	
1E9B	20	E5	1E	JSR	TSTDGT	TEST FOR NUMBER
1E9E	90	ED		BCC	LOOP	
1EA0	20	D5	1E	JSR	UPONE	MAKE ROOM FOR ONE DIGIT
1EA3	D0	E8		BNE	LOOP	FORCED BRANCH
1EA5	A0	00	COMMAS	LDYIM	\$00	
1EA7	84	69		STY	FLAG	WE WERE HERE
1EA9	A6	6E	FINDT	LDX	BUFPTR	
1EAB	B5	00		LDAZX	\$00	FIND TERMINATOR
1EAD	F0	17		BEQ	RTS	
1EAF	C9	20		CMPIM	\$20	SPACE?
1EB1	D0	04		BNE	TEST	
1EB3	E6	6E		INC	BUFPTR	
1EB5	D0	F2		BNE	FINDT	FORCED
1EB7	C9	2C	TEST	CMPIM	\$2C	COMMA?
1EB9	F0	B7		BEQ	SPACES	
1EBB	D0	90		BNE	EDITX	
1EBD	A2	04	COPY	LDXIM	\$04	COPY 4 BYTES
1EBF	B5	79	LP	LDAZX	\$79	
1EC1	95	69		STAZX	\$69	COPY POINTERS
1EC3	CA			DEX		
1EC4	D0	F9		BNE	LP	
1EC6	60		RTS	RTS		

1EC7 A0 00	UPDATE	LDYIM \$00	
1EC9 B1 6A		LDAIY PTRS	GET LINK ADL
1ECB 48		PHA	HOLD ON STACK
1ECC C8		INY	
1ECD B1 6A		LDAIY PTRS	GET LINK ADH
1ECF 85 6B		STA PTRS	&01 STORE LINK ADH
1ED1 68		PLA	
1ED2 85 6A		STA PTRS	STORE LINK ADL
1ED4 60		RTS	
1ED5 A2 59	UPONE	LDXIM BUFF	&51 END BUFFER
1ED7 CA	LOOPU	DEX	
1ED8 B5 00		LDAZX \$00	GET A BYTE
1EDA 95 01		STAZX \$01	MOVE UP ONE
1EDC E4 6E		CPX BUFPTR	
1EDE D0 F7		BNE LOOPU	
1EE0 A9 20		LDAIM \$20	INSERT SPACE
1EE2 95 00		STAZX \$00	
1EE4 60		RTS	
1EE5 C9 30	TSTDGT	CMPIM '0	
1EE7 90 03		BCC SET	
1EE9 C9 3A		CMPIM ':	
1EEB 60		RTS	WITH CARRY CLEAR
1EEC 38	SET	SEC	CARRY SET IF NON-NMBR
1EED 60		RTS	
1EEE 18	INCLIN	CLC	
1EEF A5 10		LDA BUFF	&08
1EF1 65 12		ADC BUFF	&0A
1EF3 85 10		STA BUFF	&08
1EF5 A5 11		LDA BUFF	&09
1EF7 69 00		ADCIM \$00	ADD INTERVAL
1EF9 85 11		STA BUFF	&09 TO CURRENT LINE
1EFB 60		RTS	
1EFC 58	ERROR	CLI	ALLOW KEYPRESS
1EFD 4C 59 C3		JMP	BERROR BASIC ERROR PROCESSOR
1FAE	ORG	\$1FAE	
1FAE A9 64	STRTLN	LDAIM \$64	DEFAULT 100
1FB0 85 10		STA BUFF	&08
1FB2 A9 00		LDAIM \$00	HIGH ORDER
1FB4 85 11		STA BUFF	&09
1FB6 A2 0A		LDXIM \$0A	INTERVAL 10
1FB8 A5 DE		LDA	CUSTOM TEST FOR CUSTOM

1FBA 10 0A
 1FBC A6 DD
 1FBE A5 DB
 1FC0 85 10
 1FC2 A5 DC
 1FC4 85 11
 1FC6 86 12
 1FC8 60
 1FC9 EA

BPL SKIPL
 LDX INTC CUSTOM INTERVAL
 LDA STARTC CUSTOM START
 STA BUFF &08
 LDA STARTC &01
 STA BUFF &09
 STA BUFF &0A
 SKIPL STA
 RTS
 NOP

FINAL MESSAGE \$1FCA THROUGH \$1FEC
 "CHECK FOR GOTOC ETC"

1FED PATCH ORG \$1FED

1FED A5 6B LDA PTRS &01
 1FEF 69 00 ADCIM \$00
 1FF1 85 CA STA BASICP &01
 1FF3 60 RTS
 1FF4 EA NOP

1FF5 18 ENTRY CLC CLEAR FOR STANDARD
 1FF6 90 01 BCC ALL
 1FF8 38 ENTRYA SEC SET FOR CUSTOM
 1FF9 78 ALL SEI DISABLE KEYS
 1FFA 66 DE RORZ CUSTOM FLAG IN BIT 7
 1FFC 4C 00 1D JMP START

Who regularly publishes more info on
 APPLES, PETS, KIMS, SYMS, AIMS, and
 other 6502 based systems, products and
 programs than

kilobaud **BYTE**
INTERFACE AGE
 creative computing
 COMBINED?

MICRO™ that's
 who

the full size magazine devoted to 6502
 information. Now published monthly \$12.00
 per year in USA.

Now you can get all of MICRO by buying
 "The BEST of MICRO Volume 1" for \$7.00
 (includes shipping) and starting your
 subscription with issue #7.

PO Box 3, S. Chelmsford, MA. 01824
 617/256-3649

CLASSIFIED ADS

ZIPTAPE loads 8K BASIC in 15 seconds!
 Slower than a speeding disc? Sure, but
 it only costs \$22.50 plus \$1.00 S&H.
 \$3.00 extra for software on KIM cassette.
 Described in MICRO #6. SASE for info.
 Lew Edwards, 1451 Hamilton Ave., Trenton,
 NJ 08629.

ADVERTISE in MICRO for only \$10.00 !!!
 A classified ad such as the one above,
 may be run in this new Classified Ad
 section for \$10.00. Ad may not exceed
 six lines, and only one ad per person,
 company, etc. Must relate to 6502
 type stuff, and ad must be prepaid.
 You will reach over 6000 readers!!!

The TARGET for users of Rockwell's
 AIM 65. Find out how to use the printer,
 keyboard and display. Reviews of
 upcoming Assembler and BASIC in ROMs.
 Six bimonthly issues for \$5.00 US and
 Canada (\$12.00 elsewhere). Contact:
 Don Clem, RR#2, Spencerville, OH 45887

INSERT ORG \$1F00
 DUPLICATE OF BASIC INSERT ROUTINE
 EXCEPT FOR EXIT JUMP

1F00	20	22	C5	JSR	\$C522	1F55	A5	7C	LDAZ	\$7C		
1F03	90	44		BCC	INSC	1F57	85	A9	STAZ	\$A9		
1F05	A0	01		LDYIM	\$01	1F59	65	5C	ADCZ	\$5C		
1F07	B1	AE		LDAIY	\$AE	1F5B	85	A7	STAZ	\$A7		
1F09	85	72		STAZ	\$72	1F5D	A4	7D	LDYZ	\$7D		
1F0B	A5	7C		LDAZ	\$7C	1F5F	84	AA	STYZ	\$AA		
1F0D	85	71		STAZ	\$71	1F61	90	01	BCC	INSD		
1F0F	A5	AF		LDAZ	\$AF	1F63	C8		INY			
1F11	85	74		STAZ	\$74	1F64	84	A8	INSB	STYZ	\$A8	
1F13	A5	AE		LDAZ	\$AE	1F66	20	DA	C2	JSR	\$C2DA	
1F15	C8			INY		1F69	A5	80		LDAZ	\$80	
1F16	F1	AE		SBCIY	\$AE	1F6B	A4	81		LDYZ	\$81	
1F18	18			CLC		1F6D	85	7C		STAZ	\$7C	
1F19	65	7C		ADCZ	\$7C	1F6F	84	7D		STYZ	\$7D	
1F1B	85	7C		STAZ	\$7C	1F71	A4	5C		LDYZ	\$5C	
1F1D	85	73		STAZ	\$73	1F73	88			DEY		
1F1F	A5	7D		LDAZ	\$7D	1F74	B9	06	00	INSE	LDAAY	\$0006
1F21	69	FF		ADCIM	\$FF	1F77	91	AE		STAIY	\$AE	
1F23	85	7D		STAZ	\$7D	1F79	88			DEY		
1F25	E5	AF		SBCZ	\$AF	1F7A	10	F8		BPL	INSE	
1F27	AA			TAX		1F7C	20	67	C5	INSF	JSR	\$C567
1F28	38			SEC		1F7F	A5	7A		LDAZ	\$7A	
1F29	A5	AE		LDAZ	\$AE	1F81	A4	7B		LDYZ	\$7B	
1F2B	E5	7C		SBCZ	\$7C	1F83	85	71		STAZ	\$71	
1F2D	A8			TAY		1F85	84	72		STYZ	\$72	
1F2E	B0	03		BCS	INSA	1F87	18			CLC		
1F30	E8			INX		1F88	A0	01		INSG	LDYIM	\$01
1F31	C6	74		DECZ	\$74	1F8A	B1	71		LDAIY	\$71	
1F33	18		INSA	CLC		1F8C	D0	03		BNE	INSH	
1F34	65	71		ADCZ	\$71	1F8E	4C	38	1D	JMP	\$1D38	
1F36	90	03		BCC	INSB	1F91	A0	04		INSH	LDYIM	\$04
1F38	C6	72		DECZ	\$72	1F93	C8			INSI	INY	
1F3A	18			CLC		1F94	B1	71		LDAIY	\$71	
1F3B	B1	71	INSB	LDAIY	\$71	1F96	D0	FB		BNE	INSI	
1F3D	91	73		STAIY	\$73	1F98	C8			INY		
1F3F	C8			INY		1F99	98			TYA		
1F40	D0	F9		BNE	INSB	1F9A	65	71		ADCZ	\$71	
1F42	E6	72		INCZ	\$72	1F9C	AA			TAX		
1F44	E6	74		INCZ	\$74	1F9D	A0	00		LDYIM	\$00	
1F46	CA			DEX		1F9F	91	71		STAIY	\$71	
1F47	D0	F2		BNE	INSB	1FA1	A5	72		LDAZ	\$72	
1F49	A9	0A	INSC	LDAIM	\$0A	1FA3	69	00		ADCIM	\$00	
1F4B	F0	17		BEQ	INSD	1FA5	C8			INY		
1F4D	A5	86		LDAZ	\$86	1FA6	91	71		STAIY	\$71	
1F4F	A4	87		LDYZ	\$87	1FA8	86	71		STXZ	\$71	
1F51	85	82		STAZ	\$82	1FAA	85	72		STAZ	\$72	
1F53	84	83		STYZ	\$83	1FAC	90	DA		BCC	INSG	

LOCATION

HEX	DECIMAL	VALUE TO BE POKED
00DB	219	Low order starting line number (weight 1)
00DC	220	High order starting line number (weight 256)
00DD	221	Increment desired (1-255)

Example: POKE 219,232
 POKE 220,3
 POKE 221,50

This will give a starting line number of $3 \times 256 + 232 = 1000$, and following lines will be incremented by 50.

LISTING 2 - NON-STANDARD LINE RENUMBER

STATEMENT	TOKEN	STATEMENT	TOKEN
END	80	FN	A5
FOR	81	SPC(A6
NEXT	82	THEN	A7
DATA	83	NOT	A8
INPUT#	84	STEP	A9
INPUT	85	+	AA
DIM	86	-	AB
READ	87	*	AC
LET	88	/	AD
GOTO	89	↑	AE
RUN	8A	AND	AF
IF	8B	OR	B0
RESTORE	8C	>	B1
GOSUB	8D	=	B2
RETURN	8E	<	B3
REM	8F	SGN	B4
STOP	90	INT	B5
ON	91	ABS	B6
WAIT	92	USR	B7
LOAD	93	FRE	B8
SAVE	94	POS	B9
VERIFY	95	SQR	BA
DEF	96	RND	BB
POKE	97	LOG	BC
PRINT#	98	EXP	BD
PRINT	99	COS	BE
CONT	9A	SIN	BF
LIST	9B	TAN	CO
CLR	9C	ATN	C1
CMD	9D	PEEK	C2
SYS	9E	LEN	C3
OPEN	9F	STR\$	C4
CLOSE	AO	VAL	C5
GET	A1	ASC	C6
NEW	A2	CHR\$	C7
TAB(A3	LEFT\$	C8
TO	A4	RIGHT\$	C9
		MID\$	CA

TABLE 3

TOKENS (shorthand used in BASIC text)

ENTERING THE PROGRAM

The hard way to load the program into your PET is to convert my hex listing into decimal and POKE each byte into memory. This is, of course, a challenge to your accuracy and diligence, although it may take only slightly longer than renumbering by hand. It is only a little easier to write a BASIC program which will accept the hex data and convert to decimal, with the hex incorporated in DATA statements and obtained by the READ statement. With this alternate, the program can be recorded for future use.

To make loading painless (except for the wallet), I have arranged to make tapes available through NAIL*, Drawer F, Mobile, Alabama 36601. These tapes load the machine-language program directly into high memory. Ask for "SYS8181" and send \$18.18. By the way, they also have a dandy PET monitor called SYS7171 for \$29.71, which has machine language capabilities, the ability to co-reside in RAM with BASIC programs, but also has the very helpful feature of being able to APPEND one BASIC program to another, just like the big boys do, with interleaving of lines. Like SYS8181, it uses the BASIC line-inserting routine to do the merging, just as though you typed all those new lines on your keyboard. I used a version of this monitor to develop SYS8181. If there is sufficient interest out there, I may develop a ROM version of SYS8181, but you will have to be a hardware buff to wire it into your PET.

Since PET BASIC was written by the same company who write APPLESOFT and is similar, some APPLE owners may wish to obtain a disassembled, documented listing of this renumbering program from me for \$5.00.

*National Artificial Intelligence Laboratory

P.S. SOFTWARE HOUSE
 FORMERLY PESHACK
PET™ SCHEMATICS

FOR ONLY \$24.95 YOU GET:
 24" X 30" schematic of the CPU board, plus oversized schematics of the Video Monitor and Tape Recorder, plus complete Parts layout - all accurately and painstakingly drawn to the minutest detail.

PET™ ROM ROUTINES

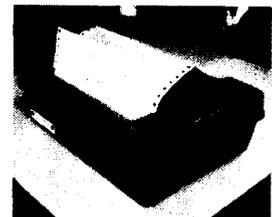
FOR ONLY \$19.95 YOU GET:
 Complete Disassembly listings of all 7 ROMs, plus identified subroutine entry points: Video Monitor, Keyboard routine, Tape Record and Playback routine, Real Time Clock, etc. To entice you we are also including our own Machine Language Monitor program for your PET using the keyboard and video display. You can have the Monitor program on cassette for only \$9.95 extra.

PET to PARALLEL INTERFACE with 5V .8A power supply \$74.95
 PET to 2nd CASSETTE INTERFACE \$49.95

Send for our free SOFTWARE BROCHURE. Dealer inquiries welcome.

PET™ EXPANDOR PRINTER
PRINTER PRICE WITH PET INTERFACE \$525

- Small size of 4.5"H x 12½"W x 9½"D
- Impact printing - 3 copies
- Prints 80 columns wide
- Print Cylinder - not a matrix
- Uses 8½" paper, pressure or pin feed
- Easy to maintain yourself, or return to us
- Regular Paper - Coated paper not required
- Lightweight, 11½ lbs. with cover
- Prints 10 characters per second
- 64 Character ASCII Character Set
- Full Documentation Included



This is the ideal, low cost, reliable, self maintained printer with which to complete your PET™ system.

P.S. SOFTWARE HOUSE

P.O. Box 966 Mishawaka, IN 46544



Tel: (219) 255-3408

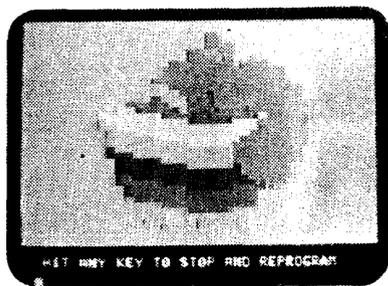


PET is a trademark of Commodore Business Machines.

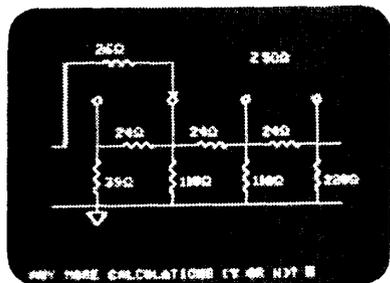
APPLE II®

QUALITY SOFTWARE

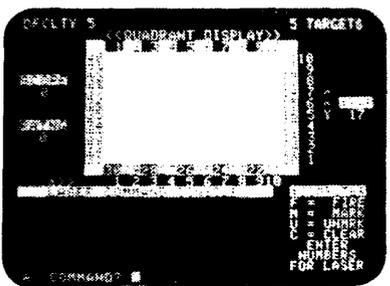
PET®



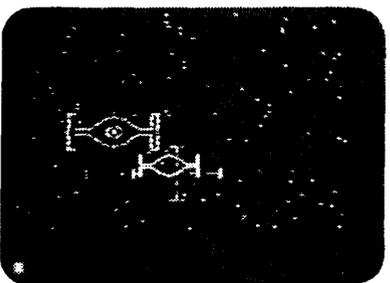
3-D ANIMATION A-\$24.95



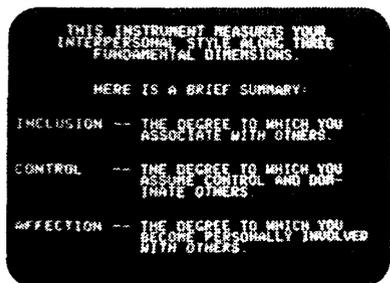
AUDIO ENGINEER A-\$29.95



STRATO LASER A-\$15.95



SUPER STARWARS A-\$15.95



ANALYST A-\$19.95

Apple FORTH
Pet FORTH

FORTH is a unique threaded language that is ideally suited for systems and applications programming on a micro-processor system. The user may have the interactive FORTH Compiler/Interpreter system running stand-alone in 4K to 6K bytes of RAM. The system also offers a built-in incremental assembler and text editor. Since the FORTH language is vocabulary based, the user may tailor the system to resemble the needs and structure of any specific application. Programming in FORTH consists of defining new words, which draw upon the existing vocabulary, and which in turn may be used to define even more complex applications. Reverse Polish Notation and LIFO stacks are used in the FORTH system to process arithmetic expressions. Programs written in FORTH are compact and very fast.

APPLE II COMPUTER \$34.95
PET 2001 COMPUTER \$34.95

Apple PIE

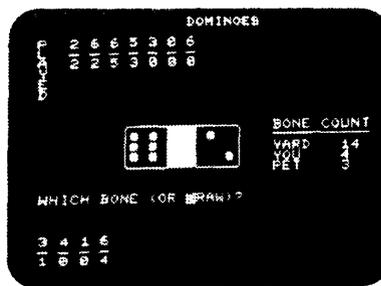
PIE (PROGRAMMA IMPROVED EDITOR) is a two-dimensional cursor-based editor designed specifically for use with memory-mapped and cursor-based CRT's. It is totally different from the usual line-based editors, which were originally designed for Teletypes. The keys of the system input keyboard are assigned specific PIE Editor function commands. PIE includes the following features: blinking cursor; cursor movement up, down, right, left, home, plus tabs; character insert and delete, string search forwards and backwards; page scrolling; GO TO line number, plus top or bottom of file; line insert and delete anywhere on screen; append and clear to end of line; move and copy buffer.

APPLE II COMPUTER \$19.95

All orders include 3% postage and handling. Apple II is a registered trademark of Apple Computer, Inc. Pet is a registered trademark of Commodore International and TRS-80 is a registered trademark of Radio Shack. California residents add 6% sales tax. VISA & MASTERCARD Accepted.



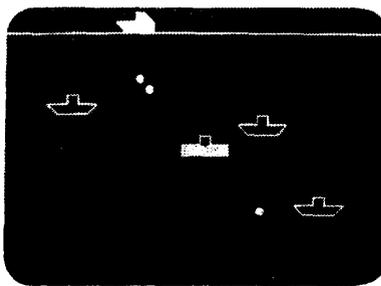
APPLE II LIGHT PEN \$34.95



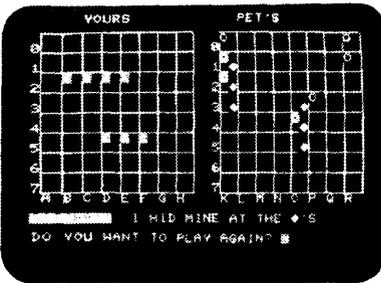
DOMINOES P-\$9.95



STARTREK P-\$9.95



DEPTH CHARGE P-\$9.95



BATTLESHIP P-\$9.95

PROGRAMMA INTERNATIONAL, Inc.
3400 Wilshire Blvd.
Los Angeles, CA 90010
(213) 384-0579
384-1116
384-1117

Dealer Inquiries Invited

PROGRAMMA

Software Products

softside software

305 Riverside Drive, New York, N.Y. 10025
212 - 866-8058

the pet program.

1

GRAPHICS PAC

Quadruple your PET's graphic resolution. Do not be stuck with the PET's cumbersome 25X40 1000 point display. With the Graphics Pac you can *individually control 4000 points* on screen. It's great for *graphing, plotting, and gaming*. The Pac is a set of three programs with full documentation. PLOT places coordinate 0,0 in the screen's upper left hand corner. For more sophisticated applications the Pac includes GRAPH which plots point 0,0, in the center of the screen allowing you to *plot equations in all four quadrants*. As a *bonus* a Hi Res Doodle game is included. All this on a high quality cassette for \$9.95

2

ASSEMBLER 2001

is a full featured assembler for your PET micro-computer that follows the *standard 6502 set of machine language mnemonics*. Now you can write machine code programs. *Store* your assembled programs, *load* them, *run* them, and even *list* your programs and *various PET subroutines*. Unlike other assemblers this is one program! You do not have to go through a three tape process to edit and run a program. Of course to make more space you can trim out the features you do not need. Assembler 2001 allows you to run through the *USR of SYS commands*. This valuable program is offered at \$15.95.

3

BIKE

An *exciting new simulation* that puts you in charge of a bicycle manufacturing empire. Juggle inflation, breakdowns, seasonal sales variations, inventory, workers, prices, machines, and ad campaigns to keep your enterprise in the black. *Bike is dangerously addictive*. Once you start a game you will not want to stop. To allow you to take short rest breaks, Bike lets you store the data from your game on a tape so you can continue where you left off next time you wish to play. Worth a million in fun, we'll offer BIKE at \$9.95.

4

PINBALL

Dynamic usage of the PET's graphics features when combined with the fun of the *number 1 arcade game* equals an *action packed video spectacle* for your computer. Bumpers, chutes, flippers, free balls, gates, a jackpot, and a little luck guarantee a great game for all. \$9.95.

5

SUPER DOODLE

Give your PET a workout. This program really *puts the PET's graphics to work*. Super Doodle lets you use the screen of your PET like a *sketch pad*. Move a *cursor* in eight directions leaving a trail of any of the 256 characters the PET can produce. New features include an *erase key* that automatically remembers your last five moves, a return to center key, and clear control. *Why waste any more paper*, buy Super Doodle for only \$9.95.

6

DRIVING ACE

Non stop excitement with a fast moving, high paced version of your favorite video arcade racing games. Shift up! Shift Down! Watch your gas, and be careful on those hairpin turns. This dynamite tape has the two most common arcade racing games specially adapted to run on your PET computer. Driving Ace simulates an endless road packed with tight turns and gentle, but teasing, twists. Starting with fifty gallons of gas, how far can you go with a minimum of accidents? Grand Prix places you and your car on a crowded racing track. Race the clock and be careful steering around the fast but packed Grand Prix track. \$9.95

Dealer Rates On Request

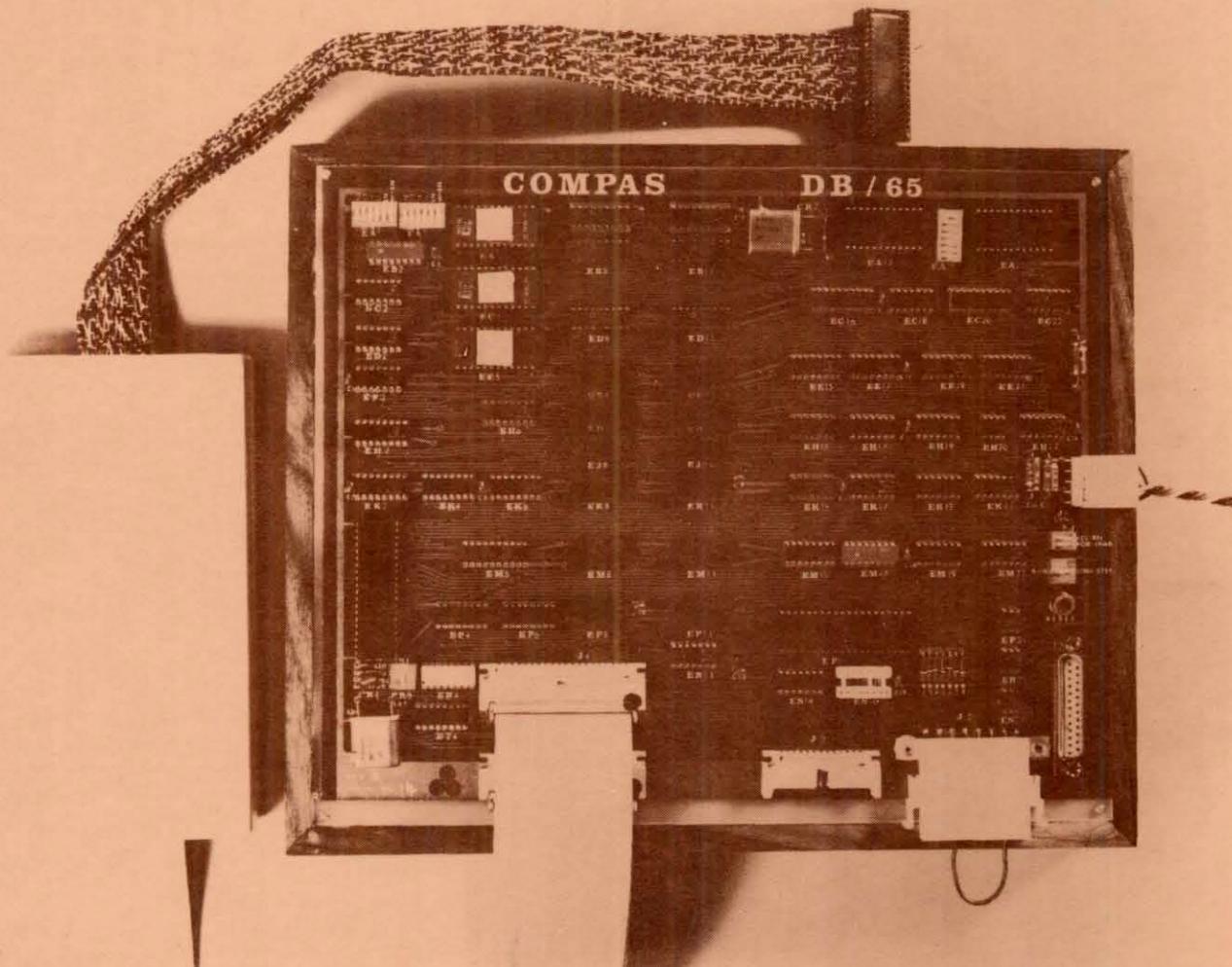
COMPAS microsystems

P.O. Box 687 224 S.E. 16th Street Ames, Iowa 50010

515/232-8187

DB/65

A complete hardware/software debug system for the Rockwell, Synertek, MOS/Technology 6500 microprocessor family.



Features

- * Standard in-circuit emulator
- * Hardware breakpoints
- * Single step mode
- * Eight software breakpoints
- * Real-time software breakpoints
- * RS 232C or current loop terminals
- * Symbolic disassembly of user program
- * Serial/parallel load capability
- * Program trace of instructions and registers
- * Prom resident debug monitor
- * Software history of instruction addresses
- * 2K ram standard with sockets for additional 6K if required
- * Scope sync output
- * User NMI and IRQ vectors supported
- * Write protect
- * User program may reside in high memory

SINGLE QUANTITY PRICE = \$1450