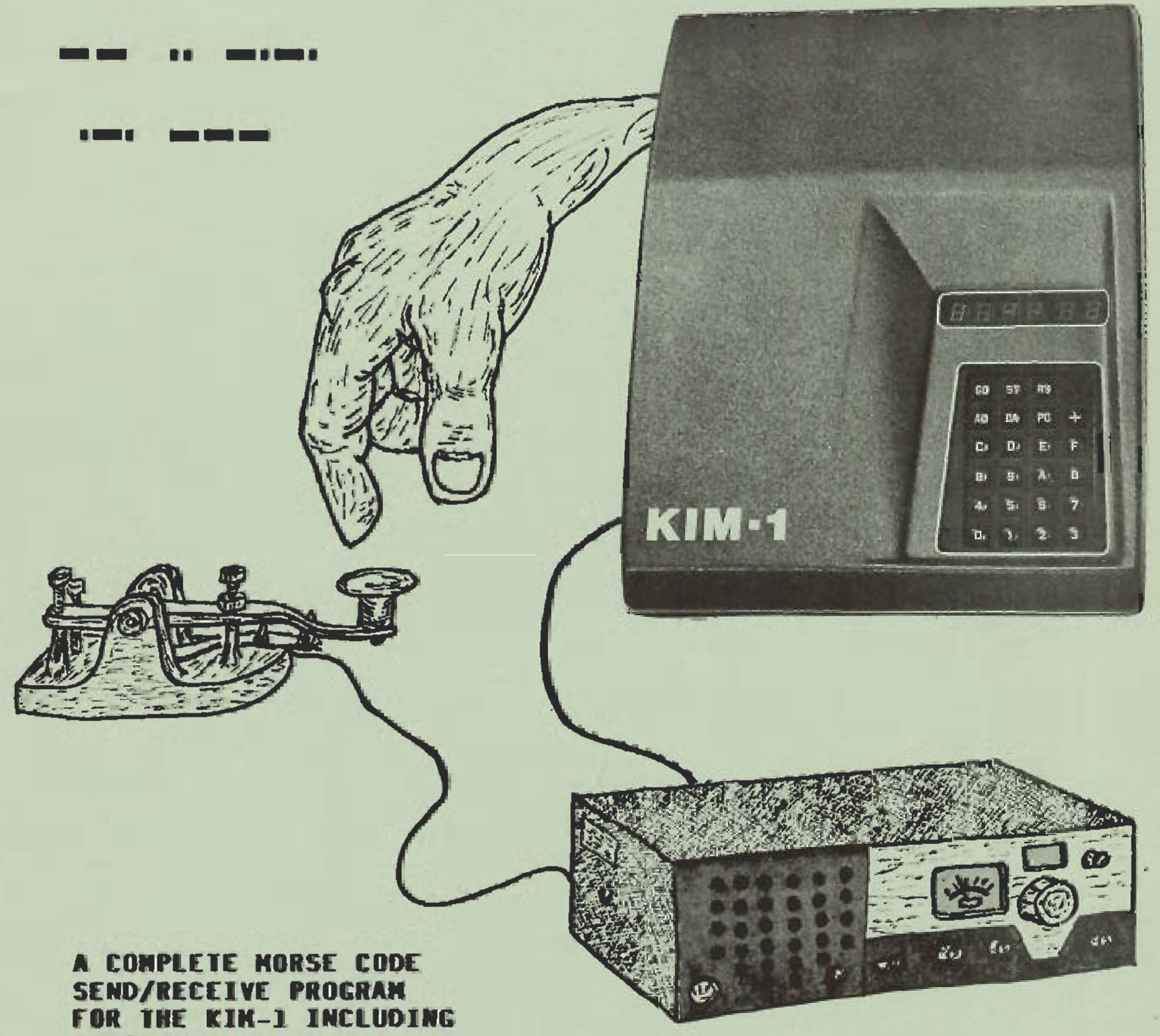
MICRO™

THE 6502 JOURNAL



A COMPLETE MORSE CODE
SEND/RECEIVE PROGRAM
FOR THE KIM-1 INCLUDING
COMPLETE SOURCE LISTINGS

NO 4 April-May 1978 $1.50

MICRO

MICRO

APRIL/MAY 1978

ISSUE NUMBER FOUR

Advertisers Index

The COMPUTERIST	IFC	The COMPUTERIST	26
The Computer Store	2	Computer Components	29
Riverside Electronics	12	A B Computers	30
CGRS	21	K L Power Supplies	33
Micro Technology Unlimited	21	the enclosures group	44

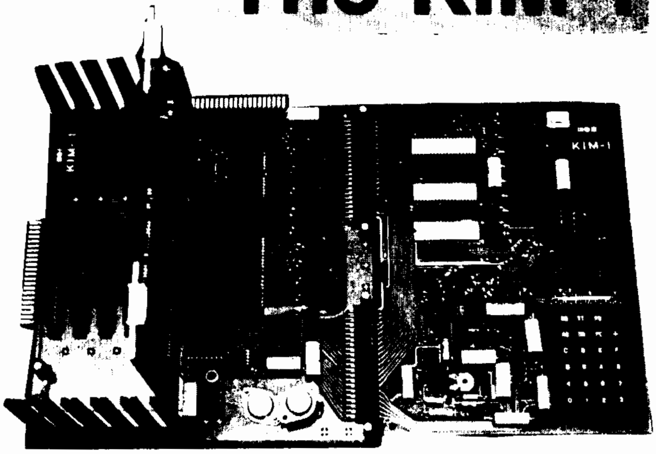
MICRO is published bi-monthly by The COMPUTERIST, 8 Fourth Lane, So. Chelmsford, MA 01824. Robert M. Tripp, Editor/Publisher. Controlled circulation postage paid at Chelmsford, Massachusetts.

Single Copy: $1.50 Annual Subscription: $6.00 (6 issues) in USA.

the Computer Store

63 SOUTH MAIN STREET, WINDSOR LOCKS, CONNECTICUT 06096
203-627-0188

The KIM 1



The Computer Store is pleased to announce off-the-shelf availability of **Apple II™, the personal computer.**

the Computer Store

63 SOUTH MAIN STREET • WINDSOR LOCKS, CONNECTICUT

GIFT CERTIFICATE

THIS COUPON GOOD FOR $2 OFF ANY PURCHASE OVER $5.

IN THIS ISSUE ...

The feature article in this issue is "A Complete Morse Code Send/Receive Program for the KIM-1" by Marvin L. De Jong [page 7]. Marvin has had two excellent articles in previous issues of MICRO [Digital-Analog and Analog-Digital Conversion Using the KIM-1, MICRO #2, and, Employing the KIM-1 Microcomputer as a Timer and Data Logging Module, MICRO #3]. His new article, which includes eight pages of source listings should be of interest to all 6502 programmers, even those with zero interest in ham radio. There are a number of useful techniques in the program:

a bit pattern conversion;
a table lookup;
some interrupt handling;
use of the KIM timer

just to mention a few. The ham radio enthusiast will, of course, find a lot of other good stuff, and will probably want to try it with their own equipment.

"The Apple II Chart" [page 4] was submitted by another MICRO regular, C. R. (Chuck) Carpenter. Chuck recommends that the chart be used to layout and keep track of strings for Applesoft BASIC. He suggests making two copies of the page, one for alphabetic and one for numeric variables, placing them between two sheets of plastic, and writing on the plastic with a felt tip pen so that the setup can be erased and used over again.

Chuck has also written the "Apple II Printing Update" [page 27] as a follow on to his article on "Printing with the Apple II", MICRO #3. Here he presents solutions to a couple of problems he encountered, plus a short note on how to let BASIC do hex-to-decimal conversions for you.

Charles Floto, with a little help from his friends, continues to provide info about the PET. "The PET Vet Examines some BASIC Idiosyncrasies" [page 5] has a discussion of some of the features of a Mailing List Program which was written by Richard Rosner. Charles also discusses some "Early PET-compatible Products" [page 22]. Roy O'Brien assembled a short list of "PET Software from Commodore" [page 21] which covers software and documentation which you may be able to get directly from Commodore if you ask for it nicely.

The extensive "6502 Bibliography" being compiled by William Dial, is continued. Part I [MICRO #1] covered references 1 through 128; Part II [MICRO #3] covered 129 through 179; and Part III continues through reference 300. Suddenly there seems to be a lot of material being written on the 6502. It looks like the secret of what a great little processor it is has gotten "out of the bag". If you know of any source of regular info on 6502s that Bill is not covering, how about letting him know about it and perhaps he can get on the subscription or distribution list and include the material in future "6502 Bibliography" parts.

Since a "beeper" for the PET is mentioned in one of this issues articles, and since the Apple II already has a built in beeper, it only seemed fair to give the KIM-1 a voice too. Gerald C. Jenkins presents "A Kim Beeper" [page 43] that is easy to build and provides the software to run it.

"The MICRO Software Catalog" [page 23], begins in this issue, and will probably become a regular department. A number of items were received too late for inclusion in this issue, and will be held over for the next issue. Certain items were considered to be too small or of limited interest to be included. We will return these to the senders so that they will know the status of their submission.

While MICRO likes to "accentuate the positive", we would be remiss if we would totally "eliminate the negative". A potentially serious problem with the Apple II has been raised, and a brief discussion is presented in "A Worm in the Apple" [page 32]. We will follow up on this item and present more info next issue.

Rick Auricchio presents "An Apple II Programmer's Guide" [page 45] which contains a lot of information he has discovered which the manual did not cover. Included in the article are a pair of tables which Apple programmers will find useful.

```
NUMERICAL VARIABLES     STRING VARIABLES - ADD ($)     APPLE II VARIABLES  FOR APPLE SOFT BASIC

A A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF AG AH AI AJ AK AL AM AN AO AP AQ AR AS AT AU AV AW AX AY AZ
B B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF BG BH BI BJ BK BL BM BN BO BP BQ BR BS BT BU BV BW BX BY BZ
C C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF CG CH CI CJ CK CL CM CN CO CP CQ CR CS CT CU CV CW CX CY CZ
D D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF DG DH DI DJ DK DL DM DN DO DP DQ DR DS DT DU DV DW DX DY DZ
E E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF EG EH EI EJ EK EL EM EN EO EP EQ ER ES ET EU EV EW EX EY EZ
F F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF FG FH FI FJ FK FL FM FN FO FP FQ FR FS FT FU FV FW FX FY FZ
G G0 G1 G2 G3 G4 G5 G6 G7 G8 G9 GA GB GC GD GE GF GG GH GI GJ GK GL GM GN GO GP GQ GR GS GT GU GV GW GX GY GZ
H H0 H1 H2 H3 H4 H5 H6 H7 H8 H9 HA HB HC HD HE HF HG HH HI HJ HK HL HM HN HO HP HQ HR HS HT HU HV HW HX HY HZ
I I0 I1 I2 I3 I4 I5 I6 I7 I8 I9 IA IB IC ID IE IF IG IH II IJ IK IL IM IN IO IP IQ IR IS IT IU IV IW IX IY IZ
J J0 J1 J2 J3 J4 J5 J6 J7 J8 J9 JA JB JC JD JE JF JG JH JI JJ JK JL JM JN JO JP JQ JR JS JT JU JV JW JX JY JZ
K K0 K1 K2 K3 K4 K5 K6 K7 K8 K9 KA KB KC KD KE KF KG KH KI KJ KK KL KM KN KO KP KQ KR KS KT KU KV KW KX KY KZ
L L0 L1 L2 L3 L4 L5 L6 L7 L8 L9 LA LB LC LD LE LF LG LH LI LJ LK LL LM LN LO LP LQ LR LS LT LU LV LW LX LY LZ
M M0 M1 M2 M3 M4 M5 M6 M7 M8 M9 MA MB MC MD ME MF MG MH MI MJ MK ML MM MN MO MP MQ MR MS MT MU MV MW MX MY MZ
N N0 N1 N2 N3 N4 N5 N6 N7 N8 N9 NA NB NC ND NE NF NG NH NI NJ NK NL NM NN NO NP NQ NR NS NT NU NV NW NX NY NZ
O O0 O1 O2 O3 O4 O5 O6 O7 O8 O9 OA OB OC OD OE OF OG OH OI OJ OK OL OM ON OO OP OQ OR OS OT OU OV OW OX OY OZ
P P0 P1 P2 P3 P4 P5 P6 P7 P8 P9 PA PB PC PD PE PF PG PH PI PJ PK PL PM PN PO PP PQ PR PS PT PU PV PW PX PY PZ
Q Q0 Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 QA QB QC QD QE QF QG QH QI QJ QK QL QM QN QO QP QQ QR QS QT QU QV QW QX QY QZ
R R0 R1 R2 R3 R4 R5 R6 R7 R8 R9 RA RB RC RD RE RF RG RH RI RJ RK RL RM RN RO RP RQ RR RS RT RU RV RW RX RY RZ
S S0 S1 S2 S3 S4 S5 S6 S7 S8 S9 SA SB SC SD SE SF SG SH SI SJ SK SL SM SN SO SP SQ SR SS ST SU SV SW SX SY SZ
T T0 T1 T2 T3 T4 T5 T6 T7 T8 T9 TA TB TC TD TE TF TG TH TI TJ TK TL TM TN TO TP TQ TR TS TT TU TV TW TX TY TZ
U U0 U1 U2 U3 U4 U5 U6 U7 U8 U9 UA UB UC UD UE UF UG UH UI UJ UK UL UM UN UO UP UQ UR US UT UU UV UW UX UY UZ
V V0 V1 V2 V3 V4 V5 V6 V7 V8 V9 VA VB VC VD VE VF VG VH VI VJ VK VL VM VN VO VP VQ VR VS VT VU VV VW VX VY VZ
W W0 W1 W2 W3 W4 W5 W6 W7 W8 W9 WA WB WC WD WE WF WG WH WI WJ WK WL WM WN WO WP WQ WR WS WT WU WV WW WX WY WZ
X X0 X1 X2 X3 X4 X5 X6 X7 X8 X9 XA XB XC XD XE XF XG XH XI XJ XK XL XM XN XO XP XQ XR XS XT XU XV XW XX XY XZ
Y Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7 Y8 Y9 YA YB YC YD YE YF YG YH YI YJ YK YL YM YN YO YP YQ YR YS YT YU YV YW YX YY YZ
Z Z0 Z1 Z2 Z3 Z4 Z5 Z6 Z7 Z8 Z9 ZA ZB ZC ZD ZE ZF ZG ZH ZI ZJ ZK ZL ZM ZN ZO ZP ZQ ZR ZS ZT ZU ZV ZW ZX ZY ZZ
```

THE PET VET EXAMINES SOME BASIC IDIOSYNCRASIES

Charles Floto
325 Pennsylvania Ave., S.E.
Washington, DC 20003

Richard Rosner has supplied a program listing produced using his RS-232 printer interface for the PET. As it's well commented I'll only point out examples of some of the unusual features of PET BASIC.

Line 1 is an example of the OPEN statement. The first number specifies that it applies to logical file number 5. This is the name by means of which other statements in the program will use this data file. The second number specifies that physical device number 5 is being used. Which device is number 5 is determined by the wiring of the system.

The PET, as sold, is wired for device 0 the keyboard; 1, the built-in tape drive; 2, the auxiliary drive connector on the back; and 3, the screen. Referring to a physical device that hasn't been electrically connected will result in a DEVICE NOT PRESENT ERROR. Richard's system does contain a physical device 5: his RS-232 output port.

If the third number in the OPEN statement is 0, reading the file is enabled. Writing is prepared for by 1, while a 2 here enables file writing with an end-of-tape character to be added when the file is CLOSEd.

Line 2 illustrates the use of CMD. It allows program commands to be applied to a device specified by the logical file connected with it (not by the physical device number). Note that RUN will merely cause a listing to be produced. RUN 5 calls the rest of the program into action.

Line 2000 demonstrates use of the OPEN statement with a variable. Lines 2000-2300 print data either on the tape drive or on the screen depending on which device number is the current value of variable D. In each case logical file 8 is used.

Another idiosyncrasy comes up here: while PRINT may be entered as ?, PRINT# cannot be entered as ?# - it must be spelled out. Otherwise a SYNTAX ERROR will result when the program is run, even though the listing will look alright.

But you can still save a good deal of typing entering these lines. Once 2110 is in simply move the cursor up to change the line number to 2111 and NA to AD. Then hit RETURN and you'll have both 2110 and 2111 in memory.

I suggest you make a few changes in Richard's program. Add 105 DIM ST$(CO) Consider storing the zip code as a string rather than as an integer. Repeat lines 2000-2300 as 5000-5300 (by changing the first digit in each line number) and change line 4500 accordingly. Then you can alter the display format without messing up the tape format. And remember that you can slow screen printing by holding the RVS key down.

A final note: I understand Commodore is now using a different tape drive and recording system. This may create compatibility problems in exchanging programs between the early PETs and the later ones.

```
1 OPEN 5,5,1,"Mailing List Program (Incomplete)"
2 CMD5:PRINT"":LIST:END
5 REM THE ABOVE LINES LIST THE PROGRAM ON THE HARD COPY UNIT
10 REM
11 REM WRITTEN BY RICHARD ROSNER
12 REM             BROOKFIELD, CONN.
13 REM FOR THE COMMODORE PET.
14 REM PRINTED ON A GE PRINTER
15 REM USING A PET ADA AVAILABLE FROM THE AUTHOR.
49 REM D=DEVICE CODE
```

```
 50 D=1:REM TAPE DRIVE #1
 90 CO=50
 91 REM CO=MAX NO. OF RECORDS IN LIST
 100 DIM NA$(CO),AD$(CO),CI$(CO)
 101 REM NA$=NAME,AD$=ADDRESS,CI$=CITY
 102 REM ST$=STATE,Z=ZIP CODE
 103 REM KC=KEY CODE. UP TO 10 FOR EACH ADDRESS
 110 DIM Z(CO),KC%(10,CO)
 997 REM ENTER RECORDS FOR MAILING LIST
 998 REM EXIT ON '!' FOR NAME
 1000 FOR N=0 TO CO
 1010 INPUT"NAME";NA$(N)
 1020 IF NA$(N)="!" GOTO 2000
 1025 LN=N
 1030 INPUT"ADDRESS";AD$(N)
 1040 INPUT"CITY,STATE";CI$(N),ST$(N)
 1050 INPUT"ZIP CODE"; Z(N)
 1060 FOR N1=0 TO 10
 1070 PRINT "KEY#";N1;:INPUT KC%(N1,N)
 1080 IF KC%(N1,N)=0 GOTO 1180
 1100 NEXTN1
 1180 NEXT N
 1998 PRINT ON TAPE DRIVE(D=1) OR SCREEN (D=3)
 2000 OPEN 8,D,1,"ADDRESS FILE"
 2009 REM LN=NUMBER OF RECORDS
 2010 PRINT#8,LN
 2100 FOR N=0 TO LN
 2110 PRINT#8,NA$(N)
 2111 PRINT#8,AD$(N)
 2112 PRINT#8,CI$(N)
 2113 PRINT#8,ST$(N)
 2115 PRINT#8,Z(N)
 2120 FOR N1=0 TO 10
 2130 PRINT#8,KC%(N1,N)
 2150 NEXT N1
 2200 NEXT N
 2300 CLOSE 8
 3000 END
 3997 REM ENTER AT 4000 TO READ IN FROM TAPE
 3998 REM DRIVE NO. 1 AND THEN PRINT ON SCREEN
 4000 OPEN 8,1,0,"ADDRESS FILE"
 4010 INPUT#8,LN
 4011 PRINTLN:REM PRINT RECORD COUNT
 4100 FOR N=0 TO LN
 4110 INPUT#8,NA$(N)
 4120 REM IF ST1 AND 64 GOTO 4300
 4130 INPUT#8,AD$(N)
 4131 INPUT#8,CI$(N)
 4132 INPUT#8,ST$(N)
 4135 INPUT#8,Z(N)
 4140 FOR N1=0 TO 10
 4150 INPUT#8,KC%(N1,N)
 4160 NEXTN1
 4190 PRINTN:REM PRINT RECORD NO. AS READ
 4200 NEXT N
 4300 CLOSE 8
 4500 D=3:GOTO 2000
READY.
```

A COMPLETE MORSE CODE SEND/RECEIVE PROGRAM FOR THE KIM-1

Marvin L. De Jong, KOEI
Dept. of Math-Physics
The School of the Ozarks
Point Lookout, MO 65726

I. INTRODUCTION

The program described below will convert ASCII from a keyboard to a Morse code digital signal which can be used to key a transmitter. It will also convert a Morse code digital signal to ASCII for display on the user's video system. Suitable references for circuits to convert the audio signal from a communications receiver to a digital Morse signal are also given. [1,2]

The entire program resides in the memory on the KIM-1, and has the following features:

1. The precise code speed in words per minute can be entered at any time from the keyboard. Key in CONTROL S followed by any two-digit decimal number from 05 to 99 words per minute.

2. The operator can type as many as 256 characters ahead of the character currently being sent. One page of memory is devoted to a FIFO buffer.

3. When there are less than 16 characters left in the buffer, the KIM-1 display indicates how many characters are left (F to 0 hex).

4. Backspace capability is provided. CONTROL B erases the last character entered into the buffer, and the operator then enters the correct character.

5. The buffer can be pre-loaded with as many characters (up to 256) as desired while the program is in the receive mode. Pressing CONTROL G starts the program sending code as soon as the operator is ready.

6. CONTROL R sends the program from the send mode to the receive mode.

7. While in the receive mode the display on the KIM-1 informs the operator to either increase the code speed (F, for faster, on the display) or decrease (S, for slower) the speed for proper reception. The receive program actually tolerates a large range in code speeds with no adjustment.

8. The feature just mentioned can be used to measure the "other guy's" code speed.

9. If the receive mode is not used, any CONTROL key not mentioned above will put the program in an idle loop so the buffer can be loaded. CONTROL G starts the message.

10. The carriage return key restarts the send program, or it can be returned from the receive mode to the send mode with CONTROL G.

The KIM-1 was first programmed to send code by Pollock [3], and some of the features of his program are found here. Pollock [4] has also described a microprocessor controlled keyboard using the 6504. It has more features than his original program written for the KIM-1, but the program described here has some additional features which are very attractive, especially the receive program.

II. BACKGROUND

A. Sending Morse Code (ASCII to Morse)

A negative going 10 microsecond strobe pulse from the keyboard is connected to the NMI pin on the KIM-1. Whenever a key is pressed an NMI interrupt occurs and the ASCII code from the keyboard is read at the lowest 7 pins of port A (PAD). The eighth bit is held high, so the number read is actually the ASCII code plus 80 hex. This number is stored in the FIFO buffer which is page 2 of memory on the KIM-1. The send routine uses the numbers in the FIFO memory to index a location in page zero which contains the information to construct the Morse character.

An illustration will make this clear. The ASCII hex representation of the letter C is 43. The strobe pulse causes port A to be read, which results in the number C3 (C3 = 43 + 80) being stored in the FIFO. When the send routine gets to the location in the FIFO where C3 is stored, it uses it to locate the contents of address 00C3. In location C3 in zero page is found 1A which is 00011010 in binary. The most significant 1 is simply a bit which indicates that all lesser significant bits contain the code information, namely 1 = dash and 0 = dot. Thus, C is dash-dot-dash-dot (1010).

The program causes the 00011010 to be rotated left (ROL) until a 1 appears in the carry position. The carry flag set causes the program to analyze the remaining bits for their code content. It does this by successively rotating them (ROL) into the carry position. If a 1 appears in the carry position, PB0 is held at logical 1 for the appropriate time followed by a space while PB0 is at logical 0. If a 0 appears in the carry position a dot is sent, followed by a space. When a total of 8 ROL commands have been completed, counting those needed to find the leading 1, then PB0 is held at logical 0 for an additional time to give a character space. The space bar produces still more time at logical 0 to produce a word space.

CONTROL S changes the NMI interrupt vectors so that the next two characters (hopefully decimal digits) from the keyboard are read, converted from base ten to hex [5], and converted to the basic time unit (see below). The interrupt vectors are then restored so that further characters from the keyboard are read as usual. Control characters are obtained by pressing the control key followed by the appropriate control character.

B. Timing Considerations.

Before going much further, the timing calculations will be described. Morse code is a variable length code. That is, the number of bits is variable as contrasted to a fixed bit-length code such as ASCII. Its structure is based on the time duration of the various components as follows:

Mark Elements:

Dot = 1t
Dash = 3t

Space Elements

Element space = 1t
  (time between dots and dashes)
Character space = 3t
  (time between letters)
Word space = 7t
  (time between words)

The time t depends on the code speed. According to The Radio Amateur's Handbook a code speed of 24 words per minute (wpm) corresponds to 10 dots per second. Since there are 10 element spaces included in the 10 dots per second, there are a total of 20 t in one second: that is, t = 1/20 second at 24 wpm. At any other speed then

t = (1/20)(24/S)
  = (50 ms)(24/S)
  = (1200/S) in milliseconds (ms)

where S is the code speed in wpm. If the divide-by-1024 timer on the KIM is used, 1 count corresponds to 1.024 ms. The number T (called TIME in the program) to be loaded into the timer is then

T = (1172/S) base ten or
  = (494/S) hex.

The speed S in wpm is entered in decimal from the keyboard, converted to base 16 (hex), sent to a divide routine to find T, and T is stored at 0000 in memory. 99 wpm gives 0C hex in TIME while 05 wpm gives EB hex. Care was taken in developing the above calculations because of a discrepancy between it and the results given by Pollock[4].

The system timing was tested by comparing it with code sent by W1AW. The speeds are the same to better than one word per minute from 5 wpm to 35 wpm.

In the receiving program a word space is detected when a space counter exceeds 5T. At moderate code speeds 5T is greater than 255 resulting in an overflow. Consequently, in the receive program 1/2T is used as the basic time unit. In this case, speeds as low as 12 wpm can be received. At slower speeds the system still works, but word spaces occur between each letter.

C. Receiving Morse Code (Morse to ASCII)

To receive Morse code and convert it to ASCII, the inverse of the above process is carried out. It is assumed that a suitable audio detection circuit [1,2[produces a logical 1 for a space element and a logical 0 for a mark element. This digital Morse signal is applied to PB7 and the IRQ pin on the KIM-1. A character register begins with a 1 in the zero bit position. Each time a dot is received the character register is shifted left and a zero is loaded into the character register. Each time a dash is received the character register is shifted left and a one is loaded into the zero bit position. Thus, when a character space is detected, and a C (for example) has been received, the character register will contain 1A, just as in sending a C. However, the 1A is used to index a zero page location which contains the ASCII code for C, namely 43. The various components are identified by timing their duration.

III. THE PROGRAMS

A detailed listing of the programs is given below. The detailed comments should allow the reader to understand, modify, and trouble-shoot the program.

A. The Send Program

Some important variables, their meanings, and their locations in zero page are given:

Name	Location	Use
TIME	0000	TIME is the quantity T mentioned in the section on timing considerations. It is the time, in units of 1.024 ms, of the dot or element space components.
SPEED	0013	SPEED is the hex equivalent of the number entered for the speed by the operator.
PNTR	0015	PNTR is a number which points to the location in the FIFO memory which contains the character currently being sent. The program idles as long as Y = PNTR, but begins to send when Y exceeds PNTR.
LO	001E	Scratchpad location for division of 494 by SPEED to give TIME.
HI	001F	Same use as LO.
CNTR	0022	CNTR keeps track of how many characters are left in the FIFO memory. A character entered decrements CNTR; a character sent increments CNTR.
CHEK	0024	Scratchpad location to count the number of numbers which have been entered after the control S has been entered.
YREG	00F4	The Y register is used to point to the location in the FIFO memory where the last character entered from the keyboard is, namely 0200,Y.

B. The Receive Program

Some important variables, their meanings, and their locations are given:

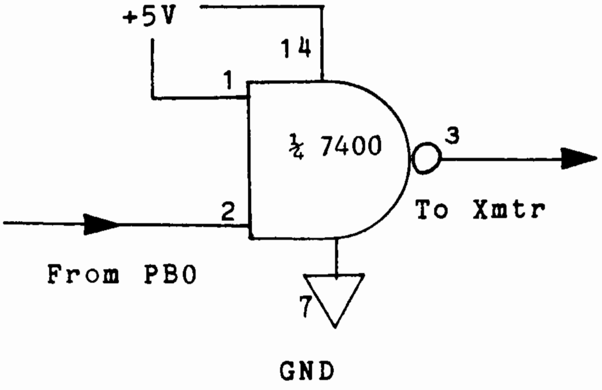
Name	Location	Use
XREG	00F5	The X register is the character register. It begins with a 1 in the 0-bit. It is shifted left for each mark element received and loaded with a 1 for a dash and a zero for a dot. Later it is used to index a table in zero page which has the ASCII code for the character.
MCNTZ	0054	If a mark element (dot or dash) is being received (PB7 and IRQ at logical 0) the mark counter is incremented at a rate of 1 count every 2.048 ms.
SCNTZ	00EE	Same as mark counter except the incrementing occurs when a space is being detected (PB7 high and IRQ high). Rate is also 1 count every 2.048 ms.
HALFT	0051	If the SPEED is set correctly, the number of counts during a dot should be exactly 1/2 TIME. This is the "dot length". If MCNTZ exceeds 1/2 the dot length the program decides that a valid mark character has been received. HALFT is 1/2 the dot length. A valid space element occurs when SCNTZ exceeds HALFT.

Name	Location	Use
TWOT	0052	TWOT is twice the dot length and is used to decide if a dot or a dash has been received. If MCNTZ exceeds TWOT the element is a dash, otherwise it is a dot.
FIVET	0053	FIVET is five times the dot length and is used to decide when a word space has been received.

IV. INTERFACE

The keyboard strobe is connected to the NMI pin on the expansion connector on the KIM-1, and the 7 bit ASCII code from the keyboard goes to pins PA0-PA6, the low order bit to PA0 and the high order bit to PA6. PA7 should be pulled up with a 10K resistor.

The author's transmitter is a solid-state Triton IV and can be keyed with TTL IC's. The circuit diagram below indicates how it was connected to the KIM-1. Transmitters using grid-block keying or cathode keying cannot use this circuit. A relay driven by a Darlington pair connected to pin PB0 should work. The KIM-1 manuals give the appropriate details.



The audio from the receiver must produce a logical 0 at pin PB7 and the IRQ pin when a tone is detected, and a logical 1 at the same pins when a space is detected. The reader is urged to try either of the circuits found in references 1 and 2. I used a half-baked scheme in which the audio from the receiver was fed to a half-wave rectifier (diode), filtered slightly, and connected to the inverting input of a CA3140 op amp. The voltage at the non-inverting input was adjustable. The op amp was operated as an open-loop comparator with the output connected to pin PB7 and IRQ. An oscilloscope was necessary to monitor the output and make the necessary adjustments for various signal levels. I am not recommending this circuit for general use.

I have also tried using the tape-input PLL system on the KIM-1 to convert the receiver audio to a digital signal. To lower the free-running frequency of the VCO a shunt capacitor must be added. The digital signal appears at address 1742, bit 7. I had only marginal success, the problem being that the digital signal tends to drop out for very short periods of time, which clears the mark counter (instructions 039F-03A2). Substituting NOP's for these instructions seems to improve the performance, but receiver tuning and volume control adjustments are sensitive. Some users may wish to experiment with deleting the aforementioned instructions in whatever interface circuit they may use.

V. MISCELLANEOUS REMARKS

To get the entire Send/Receive program in the KIM-1 memory extensive use was made of page 1. This is also used as the stack. Care was taken to leave enough room for the stack operations, and for insurance, there are several points in the program where the stack pointer is initialized to FF. No problems should be encountered once the program is up and running. If you have any debugging to do I suggest using the single-step mode (be sure to set the NMI vectors) to check the jumps and branches. My experience has been that errors in branches generally result in about half the program being wiped out, especially if it is in page 1 of memory.

Wouldn't it be nice if some outfit like The COMPUTERIST would offer an interface board which would provide an audio to digital Morse circuit, a relay driver and relay (reed type) for transmit, a DIP socket for a ribbon cable from the keyboard, and a DIP socket for the ASCII out (see appendix), all on a single board which would mate with the KIM-1 application socket.

The first time I operated the system, I answered a CQ on 40 meters from WB2GMN, Hank, who has Army Signal Corps experience. Even though he rated his speed at 55 wpm he copied me at 60 wpm. Hank reported that the code sounded like perfect code (which it should be) and that it was very crisp at 60 wpm. It was a real coincidence to contact someone who had the capability to appreciate the keyboard system and to give an evaluation of its performance.

I hope that you enjoy working these programs. If you do not want the receive program, simply put in a JMP 0300 instruction (4C 00 03) starting at 0300. If you have any questions, feel free to write, enclosing a SASE for a response. I will try to answer any questions about interfacing the system to your station.

References:

[1] Steber, G. R., and Reyer, S. E., "The Morse-A-Letter", Popular Electronics, January, 1977.

[2] Riley, T. P., "A Morse Code to Alphanumeric Converter and Display", in three parts, QST for October, November and December, 1975.

[3] Pollock, James W., "1000 WPM Morse Code Typer", 73 Magazine, January, 1977.

[4] Pollock, James, W., "A Microprocessor Controlled CW Keyboard", Ham Radio, January, 1978.

[5] Ward, Jack, "Manipulating ASCII Data", Kilobaud, February, 1978.

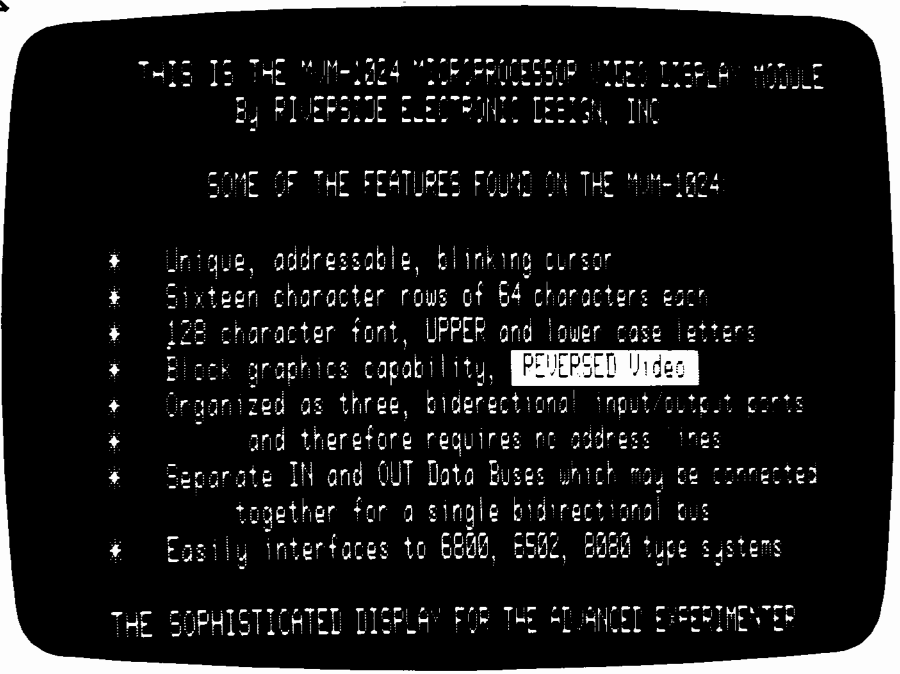
ACSII to MORSE and MORSE to ASCII
Lookup Tables in Page Zero

```
00   XX 20 45 54 49 41 4E 4D 53 55 52 57 44 4B 47 4F
10   48 56 46 XX 4C XX 50 4A 42 58 43 59 5A 51 XX XX
20   35 34 XX 33 XX XX XX 32 XX XX XX XX XX XX XX 31
30   36 3D 2F XX XX XX XX XX 37 XX XX XX 38 XX 39 30
40   XX XX XX XX XX XX XX XX XX XX XX XX 3F XX XX XX
50   XX XX XX XX XX 2E XX XX XX XX XX XX XX XX XX XX

A0   80 XX XX 2A 45 XX XX XX XX XX XX XX 73 XX 55 32
B0   3F 2F 27 23 21 20 30 38 3C 3E XX XX XX 31 XX 4C
C0   XX 05 18 1A 0C 02 12 0E 10 04 17 0D 14 07 06 0F
D0   16 1D 0A 08 03 09 11 0B 19 1B 1C XX XX XX XX XX
```

Special Morse Characters	Keyboard Character
$\overline{BT}$	=
$\overline{SK}$	$
$\overline{AR}$	#
Space (Word)	Space Bar

THIS IS WHAT YOUR KIM-1 CAN DO WITH RIVERSIDE'S KEM (KIM-1 EXPANSION MODULE) AND MVM-1024 VIDEO DISPLAY DRIVER. DISPLAY/MONITOR SOFTWARE AVALABLE IN EPROM (PLUGS INTO KEM) OR BURN YOUR OWN WITH OUR 2708/16 PROGRAMMER

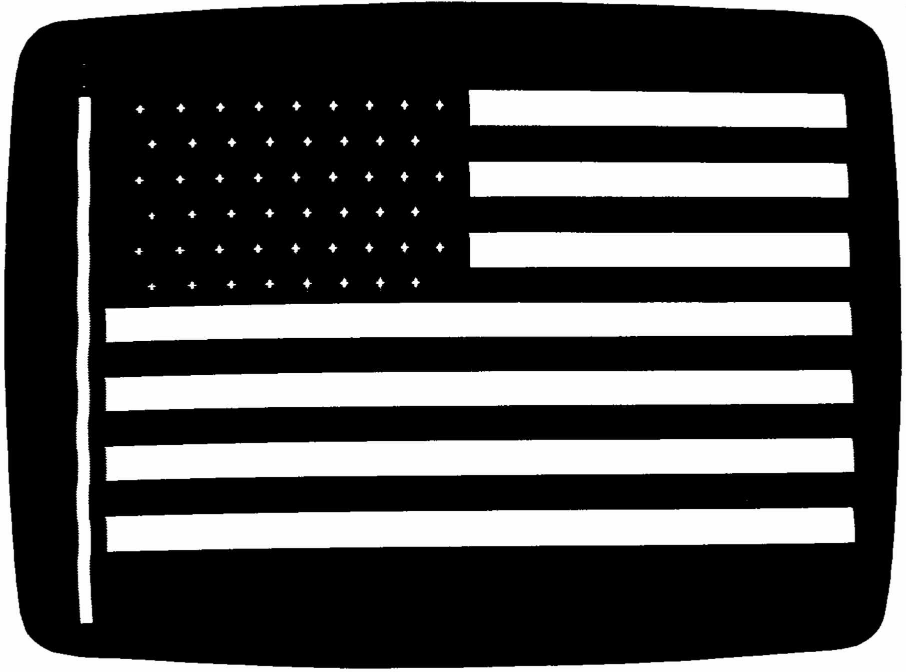


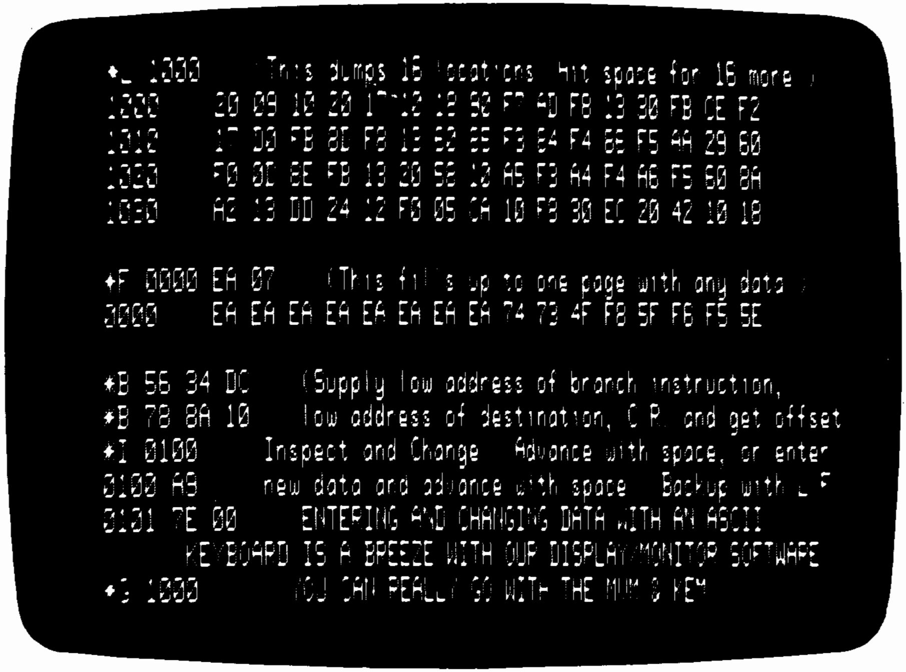
```
        THIS IS THE MVM-1024 MICROPROCESSOR VIDEO DISPLAY MODULE
              By RIVERSIDE ELECTRONIC DESIGN, INC

                SOME OF THE FEATURES FOUND ON THE MVM-1024

  *   Unique, addressable, blinking cursor
  *   Sixteen character rows of 64 characters each
  *   128 character font, UPPER and lower case letters
  *   Block graphics capability, REVERSED Video
  *   Organized as three, biderectional input/output ports
  *        and therefore requires no address lines
  *   Separate IN and OUT Data Buses which may be connected
           together for a single bidirectional bus
  *   Easily interfaces to 6800, 6502, 8080 type systems

THE SOPHISTICATED DISPLAY FOR THE ADVANCED EXPERIMENTER
```





```
*L 1000     (This dumps 16 locations. Hit space for 16 more.)
1000     20 09 10 20 17 12 12 50 F7 4D F8 13 30 FB CE F2
1010     17 D0 FB 8D F8 13 62 E5 F3 E4 F4 6E F5 AA 29 60
1020     F0 0C 8E FB 13 20 5E 10 A5 F3 A4 F4 A6 F5 60 8A
1030     A2 13 DD 24 12 F0 05 CA 10 F3 30 EC 20 42 10 18

*F 0000 EA 07     (This fills up to one page with any data.)
3000     EA EA EA EA EA EA EA EA 74 73 4F F8 5F F6 F5 5E

*B 56 34 DC     (Supply low address of branch instruction,
*B 78 8A 10      low address of destination, C.R. and get offset
*I 0100     Inspect and Change. Advance with space, or enter
0100 A9     new data and advance with space. Backup with L.F.
0101 7E 00      ENTERING AND CHANGING DATA WITH AN ASCII
        KEYBOARD IS A BREEZE WITH OUR DISPLAY/MONITOR SOFTWARE
*G 1000         YOU CAN REALLY GO WITH THE MVM & KEM
```

Riverside ELECTRONIC DESIGN INC.
1700 NIAGARA ST. BUFFALO, NY 14207

MVM-1024 $235; KEM $155; PROGRAMMER $75; EPROM $35
DIRECT FROM RIVERSIDE OR THROUGH YOUR DEALER

```
                   TIME    *    $0000        MORSE CODE SEND PROGRAM
                   ZTB     *    $0000
                   SPEED   *    $0013
                   PNTR    *    $0015
                   LO      *    $001E
                   HI      *    $001F
                   CNTR    *    $0022
                   CHEK    *    $0024
                   HALFT   *    $0051  1/2 DOT TIME
                   TWOT    *    $0052  TWICE DOT TIME
                   FIVET   *    $0053  FIVE TIME DOT TIME
                   MCNTZ   *    $0054
                   SCNTZ   *    $00EE
                   FIFO    *    $0200
                   CULO    *    $13F9  AUTHORS DISPLAY DEVICE
                   CUHI    *    $13FA  REGISTERS
                   DATA    *    $13FB
                   NMIL    *    $17FA  NON-MASKABLE INTERRUPT LOW
                   NMIH    *    $17FB  NON-MASKABLE INTERRUPT HIGH
                   IRLO    *    $17FE  INTERRUPT REQUEST LOW
                   IRHI    *    $17FF  INTERRUPT REQUEST HIGH
                   PAD     *    $1700  PORT A DATA
                   PADD    *    $1701  PORT A DATA DIRECTION
                   PBD     *    $1702  PORT B DATA REGISTER
                   PBDD    *    $1703  PORT B DATA DIRECTION REGISTER
                   SAD     *    $1740  KIM DISPLAY
                   SADD    *    $1741  KIM DISPLAY DIRECTION
                   SBD     *    $1742
                   SBDD    *    $1743
                   TIM     *    $1706  DIVIDE BY 64 TIMER
                   TMER    *    $1707  DIVIDE BY 1024 TIMER
                   TAB     *    $1FE7  KIM ROM CHARACTER TABLE

0056                       ORG  $0056

0056 D8            INIT    CLD         INIT SEQUENCE.  CLEAR DECIMAL
0057 A9 FF                 LDAIM $FF
0059 85 00                 STAZ  TIME  INITIAL CODE SPEED OF 18 WPM
005B 78            RTN     SEI         PREVENT INTERRUPTS
005C A2 FF                 LDXIM $FF   FROM RECEIVER
005E 9A                    TXS         SET STACK POINT TO TOP $01FF
005F A9 20                 LDAIM VCTL  SET NIM VECTORS FOR KEYBOARD
0061 8D FA 17              STA   NMIL
0064 A9 01                 LDAIM VCTL
0066 8D FB 17              STA   NMIH
0069 A9 00                 LDAIM $00
006B 8D 01 17              STA   PADD  PORT A IS INPUT PORT
006E 8D 02 17              STA   PBD   PORT B, PIN PB0, WILL BEGIN AT 0
0071 A9 01                 LDAIM $01   PORT B, PIN PB0, IS OUTPUT PIN
0073 8D 03 17              STA   PBDD
0076 A9 7F                 LDAIM $7F   SET UP DISPLAY PORTS
0078 8D 41 17              STA   SADD  PINS 0 - 6 ARE OUTPUT PINS
007B A9 1E                 LDAIM $1E
007D 8D 43 17              STA   SBDD  PINS 1 - 4 ARE OUTPUT PINS
0080 A9 08                 LDAIM $08   INIT LEFTMOST DIGIT
```

```
0082 8D 42 17          STA   SBD    ON KIM-1 DISPLAY
0085 A9 80             LDAIM $80    BLANK DISPLAY BY PUTTING 80
0087 8D 40 17          STA   SAD    IN PORT SAD
008A A0 FF             LDYIM $FF    INIT Y POINTER
008C 84 15             STYZ  PNTR   INIT SEND POINTER
008E 84 22             STYZ  CNTR   INIT BUFFER COUNTER
0090 C4 15       LOOP  CPYZ  PNTR   IS Y = PNTR?
0092 F0 FC             BEQ   LOOP   YES, IDLE UNTIL DIFFERENT
0094 E6 15             INCZ  PNTR   NO, INCR PNTR TO LOOKUP
0096 A6 15             LDXZ  PNTR   CHARACTER. PNTR = X INDEX
0098 BD 00 02          LDAX  FIFO   GET CHARACTER FROM FIFO
009B 4C 15 01          JMP   LOOPX  CONTINUE AT LOOPX

                 DISPLAY SUBROUTINE

0100                   ORG   $0100

0100 A6 22       DISP  LDXZ  CNTR   TRANSFER CNTR TO X
0102 E0 10             CPXIM $10    IS CNTR LESS THAN 10 HEX
0104 90 08             BCC   OVER   YES, DISPLAY CNTR
0106 A9 80             LDAIM $80    NO, BLANK DISPLAY
0108 8D 40 17          STA   SAD
010B 4C 14 01          JMP   THER
010E BD E7 1F    OVER  LDAX  TAB    FIND VALUE FROM KIM ROM
0111 8D 40 17          STA   SAD    TO DISPLAY CNTR
0114 60          THER  RTS          RETURN

0115 20 80 17    LOOPX JSR   SEND   GO TO SEND TO OUTPUT CODE
0118 E6 22             INCZ  CNTR   INCR CNTR
011A 20 00 01          JSR   DISP   DISPLAY IF LESS THAN 10
011D 4C 90 00          JMP   LOOP   CONTINUE LOOP

                 INTERRUPT ROUTINES

0120 48          VCTL  PHA          SAVE A, X AND STATUS
0121 8A                TXA          ON STACK
0122 48                PHA
0123 08                PHP
0124 AD 00 17          LDA   PAD    READ KEYBOARD
0127 48                PHA          SAVE ON STACK
0128 29 60             ANDIM $60    MASK ALL BUT TOP BITS
012A F0 0F             BEQ   CNTRL  CONTROL CHARACTER?
012C 68                PLA          NO. RECALL A AND INCR Y
012D C8                INY
012E 99 00 02          STAY  FIFO   STORE A CHAR IN FIFO
0131 20 00 01          JSR   DISP   DISPLAY CNTR IF LESS THAN 10
0134 C6 22             DECZ  CNTR   UPDATE CNTR
0136 28          BACK  PLP          RESTORE REGISTER
0137 68                PLA
0138 AA                TAX
0139 68                PLA
013A 40                RTI          RETURN FROM INTERRUPT

013B 68          CNTRL PLA          RECALL A FROM STACK
013C 29 7F             ANDIM $7F    MAKS OFF HIGHEST BIT
013E C9 02             CMPIM $02    BACKSPACE?
```

```
0140 D0 06            BNE   CNTX   TEST OTHER CHARACTER
0142 88               DEY          YES.  DECR Y TO DELETE CHARACTER
0143 E6 22            INCZ  CNTR   FIX COUNTER
0145 4C 36 01         JMP   BACK   RETURN

0148 C9 13     CNTX   CMPIM $13    CONTROL S = SPEED
014A D0 58            BNE   ARND   NO TEST OTHERS
014C A9 58            LDAIM FIX    CHANGE INTERRUPT SO NEXT
014E 8D FA 17         STA   NMIL   INTERRUPTS GO TO FIX
0151 A9 00            LDAIM $00    INIT CHEK TO 00
0153 85 24            STAZ  CHEK
0155 4C 36 01         JMP   BACK   RETURN

0158 48        FIX    PHA          SAVE REGISTERS
0159 8A               TXA
015A 48               PHA
015B 08               PHP
015C AD 00 17         LDA   PAD    READ FIRST DIGIT
015F 29 0F            ANDIM $0F    MASK TO DIGIT
0161 AA               TAX          MOVE TO X
0162 A5 24            LDAZ  CHEK   CHEK = 0 = FIRST DIGIT
0164 C9 01            CMPIM $01    CHEK = 1 = SECOND DIGIT
0166 F0 10            BEQ   AHD    FIRST DIGIT BRANCH
0168 8A               TXA          GET DIGIT BACK
0169 0A               ASLA         TIMES 2
016A 85 13            STAZ  SPEED  SAVE
016C 0A               ASLA         TIMES 4
016D 0A               ASLA         TIMES 8
016E 18               CLC          PREPARE TO ADD SPEED
016F 65 13            ADCZ  SPEED  *8 + *2 = *10
0171 85 13            STAZ  SPEED  STORE
0173 E6 24            INCZ  CHEK   SET FOR SECOND DIGIT
0175 4C 36 01         JMP   BACK   RETURN

0178 C6 24     AHD    DECZ  CHEK   RE-INIT CHEK
017A 8A               TXA
017B 18               CLC
017C 65 13            ADCZ  SPEED  ADD ONES DIGIT TO
017E 85 13            STAZ  SPEED  TENS DIGIT ANS STORE
0180 38               SEC          DIVIDE 494(HEX)/SPEED
0181 A2 00            LDXIM $00    CLEAR X FOR QUOTIENT
0183 A9 94            LDAIM $94    LOW ORDER BYTE OF DIVIDEND
0185 85 1E            STAZ  LO
0187 A9 04            LDAIM $04    HIGH ORDER BYTE OF DIVIDEND
0189 85 1F            STAZ  HI
018B A5 1E     UP     LDAZ  LO     START SUB. FROM DIVIDEND
018D E5 13            SBCZ  SPEED  UNTIL BORROW
018F 85 1E            STAZ  LO     FROM HIG BYTE, IE CARRY IS SET
0191 A5 1F            LDAZ  HI     IF BORROW OCCURS FROM LOW ORDER
0193 E9 00            SBCIM $00    BYTE, SUB 1 FROM HIGH
0195 85 1F            STAZ  HI     ORDER BYTE
0197 E8               INX          INCR X FOR EACH SUB.
0198 B0 F1            BCS   UP     BORROW FROM HI? NO. GO BACK
019A 86 00            STXZ  TIME   AND SUB. OTHERWISE DONE
019C A9 20            LDAIM VCTL   RESET NMI VECTORS FOR VCTL
019E 8D FA 17         STA   NMIL
```

```
01A1 4C 36 01           JMP   BACK   RETURN TO MAIN PROGRAM

01A4 C9 12      ARND    CMPIM $12    REMAINDER OF VCTL
01A6 D0 03              BNE   TREE   CONTROL R?
01A8 4C 00 03           JMP   RCV    YES. GO TO RECEIVE PROGRAM
01AB C9 0D      TREE    CMPIM $0D    CARRAIGE RETURN?
01AD D0 03              BNE   BUF    BRANCH IF NOT
01AF 4C 5B 00           JMP   RTN    YES. START MAIN PROGRAM
01B2 C9 07      BUF     CMPIM $07    CONTROL G?
01B4 F0 03              BEQ   BRR    YES. RESET STACK POINTER AND GO
01B6 4C B6 01   IDLE    JMP   IDLE   TO LOOP. OR, IDLE HERE
01B9 A2 FF      BRR     LDXIM $FF    WHILE BUFFER IS LOADED
01BB 9A                 TXS          RESET STACK TOP
01BC 4C 90 00           JMP   LOOP   AND CONTINUE


                MORSE CODE RECEIVE PROGRAM

                        ORG   $0300

0300 A9 90      RCV     LDAIM IRQ    SET IRQ VECTORS
0302 8D FE 17           STA   IRLO
0305 A9 03              LDAIM IRQ    / PAGE ADDRESS
0307 8D FF 17           STA   IRHI
030A A5 00      CRK     LDAZ  TIME   SET DOT LENGTH BY GETTING
030C 4A                 LSRA         TIME AND DIVIDING BY 2
030D 85 51              STAZ  HALFT
030F 46 51              LSRZ  HALFT  HALFT HALFT IS 1/2 DOT LENGTH
0311 85 52              STAZ  TWOT
0313 06 52              ASLZ  TWOT   TWOT IS TWICE DOT LENGTH
0315 85 53              STAZ  FIVET
0317 0A                 ASLA         MULTIPLY BY 4
0318 0A                 ASLA
0319 18                 CLC
031A 65 53              ADCZ  FIVET  AND ADD 1 TIMES TO GET
031C 85 53              STAZ  FIVET  5 TIMES DOT LENGTH
031E A9 00              LDAIM $00    CLEAR MARK AND SPACE
0320 85 54              STAZ  MCNTZ  COUNTERS
0322 85 EE              STAZ  SCNTZ
0324 58                 CLI          ALLOW INTERRUPTS TO START
0325 A2 01              LDXIM $01    INIT CHARACTER REGISTER
0327 4C 27 03   IDL     JMP   IDL    IDLE HER UNTIL MARK OCCURS
032A 20 8A 03   AGN     JSR   TIMSET START TIMER FOR SPACE COUNT
032D E6 EE              INCZ  SCNTZ  INCR SPACE COUNTER
032F A5 EE              LDAZ  SCNTZ  DOES IT EXCEED 1/2 DOT LENGTH?
0331 C5 51              CMPZ  HALFT
0333 B0 08              BCS   CHECK  YES, JUMP TO SET CHAR REGS
0335 AD 07 17   WAIT    LDA   TMER   OTHERWISE WAIT FOR TIMER
0338 10 FB              BPL   WAIT
033A 4C 2A 03           JMP   AGN    AND COUNT SPACES

033D 8A         CHECK   TXA          SHIFT CHAR REGISTER LEFT
033E 0A                 ASLA
033F AA                 TAX
```

```
0340 A5 54            LDAZ  MCNTZ  IF MARK COUNTER EXCEEDS TWICE
0342 C5 52            CMPZ  TWOT   THE DOT LENGTH, PUT ONE IN
0344 90 03            BCC   SKIP   CHAR REGISTER, OTHERWISE A ZERO
0346 E8               INX
0347 B0 11            BCS   FAT    IF A DASH, SKIP DISPLAY
0349 0A        SKIP   ASLA         IF A DOT, COMPARE WITH TIME
034A C5 00            CMPZ  TIME   FOR SPEED INDICATOR
034C B0 07            BCS   CAT
034E A9 F1            LDAIM $F1    SHOW "F" IS DISPLAY
0350 8D 40 17         STA   SAD
0353 90 05            BCC   FAT
0355 A9 ED     CAT    LDAIM $ED    SHOW "S" IN DISPLAY
0357 8D 40 17         STA   SAD
035A A9 00     FAT    LDAIM $00    CLEAR MARK COUNTER
035C 85 54            STAZ  MCNTZ
035E AD 07 17  HOLD   LDA   TMER   WAIT FOR TIMER
0361 10 FB            BPL   HOLD
0363 20 8A 03         JSR   TIMSET START TIMER AGAIN
0366 E6 EE            INCZ  SCNTZ  INCR SPACE COUNTER AGAIN
0368 A5 EE            LDAZ  SCNTZ
036A C5 52            CMPZ  TWOT   DOES SPACE COUNTER EXCEED TWICE
036C 90 F0            BCC   HOLD   THE DOT LENGTH. IF NOT, HOLD
036E 20 CA 03         JSR   CHAR   IF YES, PRINT CHARACTER
0371 A2 01            LDXIM $01    RESET CHAR REGISTER
0373 AD 07 17  DOZE   LDA   TMER   WAIT FOR TIMER
0376 10 FB            BPL   DOZE
0378 20 8A 03         JSR   TIMSET START TIMER AGAIN
037B E6 EE            INCZ  SCNTZ  INCR SPACE COUNTER
037D A5 EE            LDAZ  SCNTZ
037F C5 53            CMPZ  FIVET  DOES SPACE COUNTER EXCEED FIVE TIMES
0381 90 F0            BCC   DOZE   DOT LENGTH.  IF LESS, DOZE AGAIN
0383 20 CA 03         JSR   CHAR   OTHERWISE PRINT SPACE
0386 78               SEI          PREVENT INTERRUPTS WHILE
0387 4C 0A 03         JMP   CRK    CHECKING SPEED SETTING

038A A9 20     TIMSET LDAIM $20    LOAD TIMER FOR 2.048 MS
038C 8D 06 17         STA   TIM
038F 60               RTS          RETURN TO RCV PROGRAM

0390 08        IRQ    PHP          SAVE REGISTERS
0391 48               PHA
0392 20 8A 03         JSR   TIMSET START TIMER

0395 AD 07 17  LOAF   LDA   TMER   WAIT FOR TIMER
0398 10 FB            BPL   LOAF
039A AD 02 17         LDA   PBD    IS MARK SIGNAL PRESENT
039D 10 09            BPL   OVER   YES, GO TO OVER
039F A9 00            LDAIM $00    NO, MUST HAVE BEEN NOISE
03A1 85 54            STAZ  MCNTZ  WHICH CAUSED INTERRUPT.  RETURN
03A3 E6 EE            INCZ  SCNTZ  TO COUNT SPACE AFTER RESETTING
03A5 68               PLA          MARK COUNTER TO ZERO
03A6 28               PLP
03A7 40               RTI          RETURN FROM INTERRUPT
```

```
03A8 20 8A 03  OVER   JSR   TIMSET START TIMER AGAIN
03AB E6 54            INCZ  MCNTZ  INCR MARK COUNTER
03AD A5 54            LDAZ  MCNTZ  DOES MARK COUNTER EXCEED
03AF C5 51            CMPZ  HALFT  1/2 THE DOT LENGTH?
03B1 90 E2            BCC   LOAF   NO, GO LOAF AND CHECK MARK
03B3 A9 00            LDAIM $00    YES.  CLEAR SPACE COUNTER
03B5 85 EE            STAZ  SCNTZ
03B7 AD 07 17  KILTIM LDA   TMER   CHECK TIMER
03BA 10 FB            BPL   KILTIM KILL TIME
03BC AD 02 17         LDA   PBD    CHECK MARK SIGNAL ON PB7
03BF 10 E7            BPL   OVER   LOOP AGAIN IF STILL ON
03C1 8A               TXA          SAVE S WHILE STACK POINTER IS SET
03C2 A2 FF            LDXIM $FF    RESET TO TOP OF STACK
03C4 9A               TXS
03C5 AA               TAX          RESTORE X
03C6 58               CLI          CLEAR INTERRUPT FLAG SET EARLIER
03C7 4C 2A 03         JMP   AGN    RETURN TO COUNT SPACE

03CA B5 00     CHAR   LDAZX ZTB    LOOKUP ASCII SYMBOL
03CC 8D FB 13         STA   DATA   DATA IS VIDEO PORT IN AUTHORS
03CF A9 3F            LDAIM $3F    SYSTEM.  THE REMAINDER OF THIS
03D1 2D F9 13         AND   CULO   SUBROUTINE INCREMENTS THE
03D4 C9 3F            CMPIM $3F    POSITION OF THE CURSOR TO PREPARE
03D6 90 11            BCC   AHD    FOR THE NEXT CHARACTER
03D8 A9 1F            LDAIM $1F
03DA 2D FA 13         AND   CUHI
03DD 18               CLC
03DE 69 01            ADCIM $01
03E0 C9 20            CMPIM $20
03E2 90 02            BCC   UP
03E4 A9 10            LDAIM $10
03E6 8D FA 13  UP     STA   CUHI
03E9 EE F9 13  AHD    INC   CULO
03EC 60               RTS

               SEND SUBROUTINE

1780                  ORG   $1780

1780 AA        SEND   TAX          A CONTAINS CHAR FROM FIFO
1781 B5 00            LDAZX ZTB    USE THIS TO LOOKUP MORSE
1783 30 3F            BMI   WDSP   SPACE BAR CHAR HAS 1 IN BIT 7
1785 18               CLC          IF NOT MINUS, CLEAR CARRY FLAG AND
1786 A2 08            LDXIM $08    SET UP X FOR 8 ROL INSTRUCTIONS
1788 2A        RPT    ROLA         ROTATE LEFT UNTIL 1 APPEARS IN CARRY
1789 B0 06            BCS   DWN    BRANCH IF 1 IN CARRY
178B CA               DEX          ELSE, DECREMENT X
178C F0 35            BEQ   OUT    IF X = 0, THEN DONE
178E 4C 88 17         JMP   RPT    ELSE CONTINUE
1791 CA        DWN    DEX          KEEP TRACK OF BITS TESTED
1792 2A        BACK   ROLA         ROTATE A LEFT AND SAVE ON STACK
1793 48               PHA
1794 8A               TXA          SAVE X ON STACK ALSO
1795 48               PHA
```

```
1796 B0 18              BCS   DASH   DID ROTATE SET CARRY? IF YES,
1798 A2 01              LDXIM $01    SEND DASH, ELSE SEND DOT
179A EE 02 17  DAH      INC   PBD    PB0 WILL BE LOGICAL 1 FO 1 T
179D 20 C9 17  SPA      JSR   TIMER  TIME GIVES DELAY OF TIME (1.024MS)
17A0 CA                 DEX          ONE TIME UNIT IS UP
17A1 D0 FA              BNE   SPA    IS X = 0? DELAY ANOTHER UNIT
17A3 AD 02 17           LDA   PBD    YES. NOW CHECK PB0.  IF A 1
17A6 4A                 LSRA          A SHIFT WILL SET CARRY FLAG
17A7 90 0C              BCC   DONE   IF CARRY CLEAR, THEN DONE
17A9 CE 02 17           DEC   PBD    OTHERWISE, SET PB0 = 0 FOR ELEMENT
17AC E8                 INX          SPACE FOR A DELAY OF 1 UNIT BY
17AD 4C 9D 17           JMP   SPA    RESETTING X AND LOADING TIMER

17B0 A2 03     DASH     LDXIM $03    DASH TAKES 3 TIME UNITS
17B2 4C 9A 17           JMP   DAH    SEND 3 UNITS FOLLOWED BY SPACE

17B5 68        DONE     PLA          THEN ELEMENT IS DONE SO
17B6 AA                 TAX          RESTORE A AND X AND GO BACK
17B7 68                 PLA          IF X IS NOT ZERO
17B8 CA                 DEX          OTHERWISE ADD CHARACTER SPACE
17B9 D0 D7              BNE   BACK   BY RUNNING TIMER FOR
17BB A2 02              LDXIM $02    2 MORE TIME UNITS
17BD 20 C9 17  AGAIN    JSR   TIMER
17C0 CA                 DEX
17C1 D0 FA              BNE   AGAIN  IF X = 0, THEN DONE
17C3 60        OUT      RTS          OR ELSE DELAY MORE

17C4 A2 04     WDSP     LDXIM $04    WORDSPACE REQUIRES 4 MORE TIME UNITS
17C6 4C BD 17           JMP   AGAIN  SO USE TIMER FOR THIS

17C9 A5 00     TIMER    LDAZ  TIME   GET TIME FROM ZERO PAGE
17CB 8D 07 17           STA   TMER   LOAD DIVIDE BY 1024 TIMER
17CE 2C 07 17  CHK      BIT   TMER   IS TIMER FINISHED?
17D1 10 FB              BPL   CHK    NO, WAIT FOR IT
17D3 60                 RTS          YES, RETURN
```

APPENDIX:
Using the KIM-1 Ports to Output the ASCII

Most readers will not have the same addressable video system used by the author. To use the receive portion of the program, some provision must be made to output the ASCII along with a strobe pulse. Below you will find a suggested program to do this. It makes use of ports SAD and SBD addresses 1740 and 1742 respectively. These are available on the application connector. The ASCII code appears at the KB COL A-G pins, while the strobe should appear at the TTY PTR pin.

NOTE: While this program should work it has not been tested.

ALTERNATIVE ASCII OUTPUT

```
                        ORG   $03CA

                    *** THIS ROUTINE HAS NOT BEEN TESTED ***

03CA           ZTB     *      $0000
03CA           SAD     *      $1740
03CA           SADD    *      $1741
03CA           SBD     *      $1742
03CA           SBDD    *      $1743

03CA A9 20     CHAR   LDAIM $20    ENABLE OUTPUT PULSE PINS
03CC 8D 42 17         STA   SBD
03CF A9 21            LDAIM $21
03D1 8D 43 17         STA   SBDD
03D4 AD 40 17         LDA   SAD    SAVE CONTENTS OF CURRENT
03D7 48               PHA          DISPLAY ON KIM-1
03D8 AD 41 17         LDA   SADD
03DB 48               PHA
03DC B5 00            LDAZX ZTB    GET ASCII CODE
03DE 8D 40 17         STA   SAD    OUTPUT ASCII
03E1 A9 FF            LDAIM $FF
03E3 8D 41 17         STA   SADD   ENABLE OUTPUT PORT
03E6 EE 42 17         INC   SBD    STROBE PULSE WILL BE
03E9 EA               NOP          LENGTHEN PULSE
03EA CE 42 17         DEC   SBD    NEGATIVE
03ED 68               PLA          RESTORE SADD AND SAD
03EE 8D 41 17         STA   SADD
03F1 68               PLA
03F2 8D 40 17         STA   SAD
03F5 A9 1E            LDAIM $1E    RESTORE SBDD AND SBD
03F7 8D 43 17         STA   SBDD
03FA A9 08            LDAIM $08
03FC 8D 42 17         STA   SBD
03FF 60               RTS
```

PET SOFTWARE FROM COMMODORE

Roy O'Brien
P.O. Box 187
Somerset, NJ 08873

It appears that in response to specific questions, Commodore is sending out selected Application Notes. The software consists of the following:

Machine Language Monitor - (9 pages)
A discussion of the TIM program as adapted to the PET. Early PET owners are supposed to receive TIM on cassette and later PETs will have TIM in ROM.

PET Cassette Files - (31 pages)
A learn-by-doing mini-course in file management with the PET.

IEEE-488 Devices - (5 pages)
A listing of available equipment which directly interfaces to the PET. Gives device, model number, manufacturer; includes printers, counters, measurers, ADCs, DACs, timers, synthesizers, analyzers, plotters, tapes, discs, etc.

BASIC Bugs - (4 pages)
Kinks, quirks and bugs in PET BASIC.

PET and ASCII - (4 pages)
Definitions and symbol codes, including a neat little program which shows graphics and codes on screen.

PET Uses Its Memory - (1 page)
A reprint of PET memory usage from PCCs Nov/Dec 1977 issue.

Animating Your PET - (2 pages)
How to use the programmable cursor controls to create moving graphics.

Some Questions and Answers - (11 pps)
Things you always wanted to know and weren't afraid to ask; summarized. A must for PET owners.

HIGH RESOLUTION GRAPHICS

In response to your requests, we now offer the K-1008, a Dot Matrix display board (320H x 200V) for the KIM-1.

But — we didn't stop there. We also call it an 8K memory board, directly connected to your KIM-1. Full read/write with no wait states or snow ever.

And — we made it low power to reduce system costs. In fact our 18 watt K-1000 power supply can typically power your KIM-1 plus 32K of K-1008 memory.

How to use it — The K-1008 visible memory only needs a power supply and a KIM-1 to function as memory. Add a standard monitor and you have high resolution graphics for diagrams, graphs, even variable font text up to 22 lines of 53 characters.

K-1008 Assembled/Tested
$289.00 board $40.00
Graphic Software Listing
$20.00
K-1000 power supply $40.00
K-1005 5 slot card file $69.00

Micro Technology Unlimited
P.O. Box 4596
Manchester, N.H. 03103

K-1000 POWER SUPPLY FOR KIM-1

The original power supply designed for home or office use.
5V-1.2A 12V-.1A regulated
8V-.75A 16V-.25A unregulated.
Enclosed in a black Bakelite box with terminal strip output, line cord and fused primary. This unit is standard industrial quality, designed to run at its rated outputs continuously at even low line voltage.

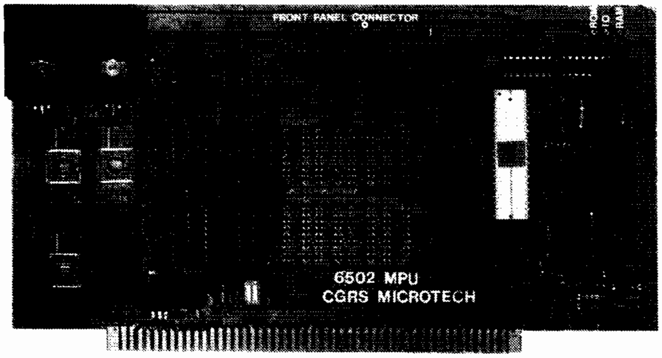
Price $40.00 prepay, plus shipping. Special-orders received during April, May, June which include this ad with the order, $10.00 discount.

Dealer & group inquiries invited.

Micro Technology
Unlimited
P.O. Box 4596
Manchester, N.H. 03103

FINALLY:

6502 ON THE S100



CGRS MICROTECH INTRODUCES A
6502 COMPUTER SYSTEM

- S100 STANDARD BUS COMPATIBLE
- MPU CARD WITH 2K RAM-4K ROM ON BOARD
- T.I.M. (6530) SYSTEM I/O CARD
- D.M.A. FRONT CONTROL PANEL

	KIT	ASSEM
✱ INTRODUCTORY SYSTEM MPU CARD: 1K RAM FRONT PANEL: HEX DISPLAY	$249.95	$299.95
✱ STANDARD SYSTEM MPU CARD: 1K RAM T.I.M. I/O CARD S100 MOTHERBOARD: 7 SLOT POWER SUPPLY 8V:10A ±16V:1A	$349.95	$449.95

SEND CHECK OR MONEY ORDER TO:
CGRS MICROTECH
P.O. BOX 368
SOUTHAMPTON, PA 18966

EARLY PET-COMPATIBLE PRODUCTS

Charles Floto
325 Pennsylvania Ave., S.E.
Washington, DC 20003

Throughout the five months I've had my PET, I've felt the biggest design oversight was leaving out a speaker. Commodore even went to the trouble of removing one, along with its amplifier, from the tape drive.

The versatility of the Apple II's audio output is nice, but I'd be satisfied with a simple beeper like the one in the Heath Company's H8. That's why I'm spending $19.95 for the PETsqueak from HUH Electronic Music Productions (P.O. Box 259, Fairfax, CA 94930 415/457-7598). This assembled and tested device doesn't just produce audible output under user control. It also beeps automatically during program loading or saving to indicate file headers and completion of the operation. I look forward to being able to turn away from my PET and still keep track of what's happening.

PET-compatible products from HUH scheduled for April and May delivery include an 8-bit digital-to-analog converter, an adapter for a video monitor (so you can have a larger screen facing a different direction), and an S-100 bus interface.

While I'm looking forward to adding the beeper to my PET, the thing that will really enhance its value is a compatible printer. The big news this month is that you can now hook any RS-232 printer to your PET. The necessary adapter is sold by Connecticut microComputer (150 Pocono Rd., Brookfield, CT 06804). Assembled and tested, but without power supplies, case, or RS-232 connector, it goes for $103.50 with shipping and handling. The complete version is $174. The speed will be set at 300 baud unless another rate is requested at the time of ordering. This may be changed by the user later. With the PET ADApter model 1200 you can produce not only program listings, but also mailing labels, letters, etc. The appearance will naturally depend on the printer used. Lower case letters are substituted for the graphics character.

The third addition I plan to make to my PET is a 6502 assembler written in BASIC. I ordered this for $24.95 from Personal Software (P.O. Box 136-M3, Cambridge, MA 02138 617/783-0694).

While I'm content with the PET keyboard anyone who wants to hook up another one may be interested in the ASCII keyboard interface sold by Excel Co. (2241 Tamalpais Ave., El Cerrito, CA 94530 415/237-8114). Prices start at $65.

The makers of the KIMSI have announced the PETSI. In kit form with one S-100 connector it's $105. Assembled with the maximum of four S-100 slots it's $165. Neither version includes a power supply. Forethought Products (P.O. Box 386-D, Coburg, OR 97401 503/485-8575) is the manufacturer.

May delivery is scheduled for an RS-232 interface from The Net Works (5014 Narragansett #6, San Diego, CA 92107 714/223-1176). Single port version is $240; dual port $280.

The PET Vet will have more to say about these and other PET oriented products in future issues of MICRO. If you have information about PET products, as a manufacturer, dealer, or user, please send materials to:

The PET Vet
MICRO
P.O. Box 3
S. Chelmsford, MA 01824

THE MICRO SOFTWARE CATALOG

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

As a service to the 6502 community, MICRO will publish a continuing catalog of software available for 6502 based systems. The source of this information will normally be the authors or distributors of the software. Since there is only a limited amount of space which can be devoted to this effort, there will be some restrictions placed on what is published. To qualify for inclusion in the catalog the software must be currently available, should have been sold (or given) to at least twenty-five customers, must be of general interest, and must be significant. "Significant" means that the program is not just a short utility which could be presented as a one-page article in a magazine, or a simple game, etc. The intent of the catalog is not to promote everyone selling everything, but rather to highlight the important software packages which do exist.

Publication of information about any software in this catalog does not imply anything about its worth, capabilities, documentation, etc. We depend on the information supplied to us. We will not knowingly include any software that is not worthy, and we reserve the right to publish additional information about these products - be it good or bad - that we receive from our readers or any other valid source.

It is easy to get your package listed. Just write to the above address and provide the information required as shown in the listings below. Please write your own "description". If we have to write the description from general information you provide, we may miss points which you think are important and emphasize things you think are trivial. Also, material which is presented in the proper form will normally get priority over other material.

Name: ASSM/TED
System: Preconfigured for TIM
Can be modified for other systems.
Memory: 4K RAM
Language: Assembler
Hardware: CRT and Keyboard, tapes and printer optional.
Description: A resident Assembler/Text Editor. Syntax very similar to MOS Technology. Produces relocatable object code on tape and can store directly executable code in memory during assembly. Programs can be assembled from memory of tape. Includes 17 operating commands and 16 pseudo ops. Editor has auto line numbering, file formating, and a manuscript feature.
Copies: Information not provided.
Price: $25.00
Includes: Hex Dump of ASSM/TED and Relocating Loader, and Operators Manual. No tape provided.
Ordering Info: Specify memory limits: 0200-1200, 0400-1400, 1000-2000, or 2000-3000. Select one.
Author: C. W. Moser
Available from:
C. W. Moser
3239 Linda Drive
Winston-Salem, NC 27106

Name: COSMAC 1802 Simulator
System: KIM-1
Memory: Less than 1K RAM
Language: Assembler
Hardware: Basic KIM-1
Description: Permits the KIM-1 to simulate the COSMAC 1802 by executing its instruction set. The simulator does this by interpretting the COSMAC instructions in a normal program sequence and making all internal COSMAC registers available for examination at any time. They may be viewed statically in a single step mode or dynamically in a trace mode. All COSMAC software features are supported with the exception of DMA.
Copies: Just released. Will be discussed in an article in Kilobaud.
Price: $10.00
Includes: KIM-1 cassette tape, user manual, and complete source listing.
Ordering Info: None required
Author: Dann McCreary
Available from:
Dann McCreary
4758 Mansfield St, #2M
San Diego, CA 92116

Name: PLEASE
System: Basic KIM-1
Memory: Basic KIM-1 memory
Language: Assembler/PLEASE
Hardware: Basic KIM-1
Description: A collection of games and demos. Includes a 24 hour clock, HiLo game, Mastermind, Shooting Stars, Drunk Test, Reaction Time Tester, Adding Machine, and more. Written in a "high-level" language - PLEASE. Permits the user to modify and create his own programs. Let's you show off your KIM-1, and teaches you how to use it.
Copies: Over 800 have been sold
Price: $15.00
Includes: Operators manual, complete source listings, PLEASE language description, with object code on Hypertape.
Ordering Info: None
Author: Robert M. Tripp
Available from:
The COMPUTERIST
P.O. Box 3
S. Chelmsford, MA 01824

Name: Micro-ADE
System: KIM-1 (easily modified for use with other 6502 based systems)
Memory: 8K RAM or 4K EPROM + 4K RAM
Language: Assembler
Hardware: Terminal - CRT or TTY, cassette units optional
Description: A combination Assembler, Editor, and Disassembler. Uses MICRO 6502 syntax. With automatic cassette controls, any length file may be edited and assembled. Object files may be automatically dumped to cassette and for short programs may be dumped to and executed from memory. Includes many useful commands for handling cassettes, moving data in memory, and so forth.
Copies: Hundreds
Price: $25.00 without source listings
$25.00 for source listings
Includes: Extensive user manual which includes source listings for the I/O to permit user modification. Object on Hypertape cassette.
Ordering Info: Specify with or without the optional source listings.
Author: Peter Jennings
Available from:
Micro-Ware Ltd.
27 Firstbrooke Road
Toronto, Ontario
Canada M4E 2L2

The COMPUTERIST
P.O. Box 3
S. Chelmsford, MA 01824

Name: The 6502 Program Exchange
System: TIM and KIM-1
Memory: Depends on Program
Language: Assmebler, BASIC, FOCAL
Hardware: Depends on Program
Description: A large collection of programs for 6502 based systems. These include utilities, games, subroutines, an assembler, editor, and a high level language: FOCAL.
Copies: Few to Many depending on the particular program.
Price: Depends on program. Many are based purely on number of pages of code. Major packages are priced separately.
Includes: Normally includes source listings, documentation, sheets of sample run, and paper tape. KIM-1 cassettes at no additional charge if user supplies cassettes.
Ordering Info: Write for catalog.
Author: Many different authors.
Available from:
The 6502 Program Exchange
2920 Moana
Reno, NV 89509

Name: Personal Savings Investment
Loan Repayment
Direct Reduction Loan Info.
System: APPLE II
Memory: At least 16K
Language: APPLESOFT BASIC
Hardware: Standard APPLE II
Description: Three separate programs. PSI - compute future value of your investments; monthly amount needed to get to a certain goal at a certain time. LP - determine monthly payments for a car, house or other type of load. DRLI - find the total interest paid and remaining balance is for a loan.
Copies: Over 25 combined
Price: $3.75 (including handling) each of the three programs.
Includes: Object on cassette tape. A listing of the program and examples of program useage.
Ordering Info: Specify which program.
Author: Les Stubbs
Available from:
Les Stubbs
23725 Oakheath Place
Harbor City, CA 90710

Name: TINY BASIC
System: KIM, TIM, Jolt, Apple I
Memory: Minimum of 3K
Language: Assembler
Hardware: User defines I/O
Description: TINY BASIC is a subset of regular BASIC, limited to 16-bit integer arithmetic [+, -, *, /, ()]. There are 26 variables (A-Z), no stirngs and no arrays. The following commands are functional: LET PRINT INPUT IF-THEN GOTO GOSUB RUN LIST CLEAR RETURN REM END. TINY BASIC does not contain any I/O instructions; three JMPs link TINY to the user's I/O routines. These are well documented in the manual.
Copies: "Several hundred 6502 version"
Price: $5.00
Includes: 26 page User Manual and a paper tape in standard hex loader format. Hex Dump may be substituted upon request for paper tape.
Ordering Info: Specify version:

```
TB650K (0200-0AFF)  KIM, TIM, ....
TB650J (1000-18ff)  Jolt
TB650T (2000-28FF)  KIM with 4K RAM
```

Author: Tom Pittman
Available from:
ITTY BITTY COMPUTERS
P.O. Box 23189
San Jose, CA 95153

Name: HELP Mailing List Package
System: Basic KIM-1
Memory: Basic KIM-1
Language: Assembler/HELP
Hardware: Terminal, Cassettes, Relays
Description: A complete package for creating, maintaining, and printing mailing list information. A high speed cassette routine reads/writes at 800 baud (twelve times the KIM-1 rate) and can store about 900 names on one side of a 60 minute tape. Selective printing of mailing list. This package is used to maintain the MICRO mailing list This package is written in HELP, a "high-level" language which makes it easy to customize the package for your own requirements.
Copies: Over 100
Price: $15.00
Includes: An extensive user manual, a detailed discussion of the HELP language, and complete source listings. Object on Hypertape.
Ordering Info: None
Author: Robert M. Tripp
Available from:
The COMPUTERIST
P.O. Box 3
S. Chelmsford, MA 01824

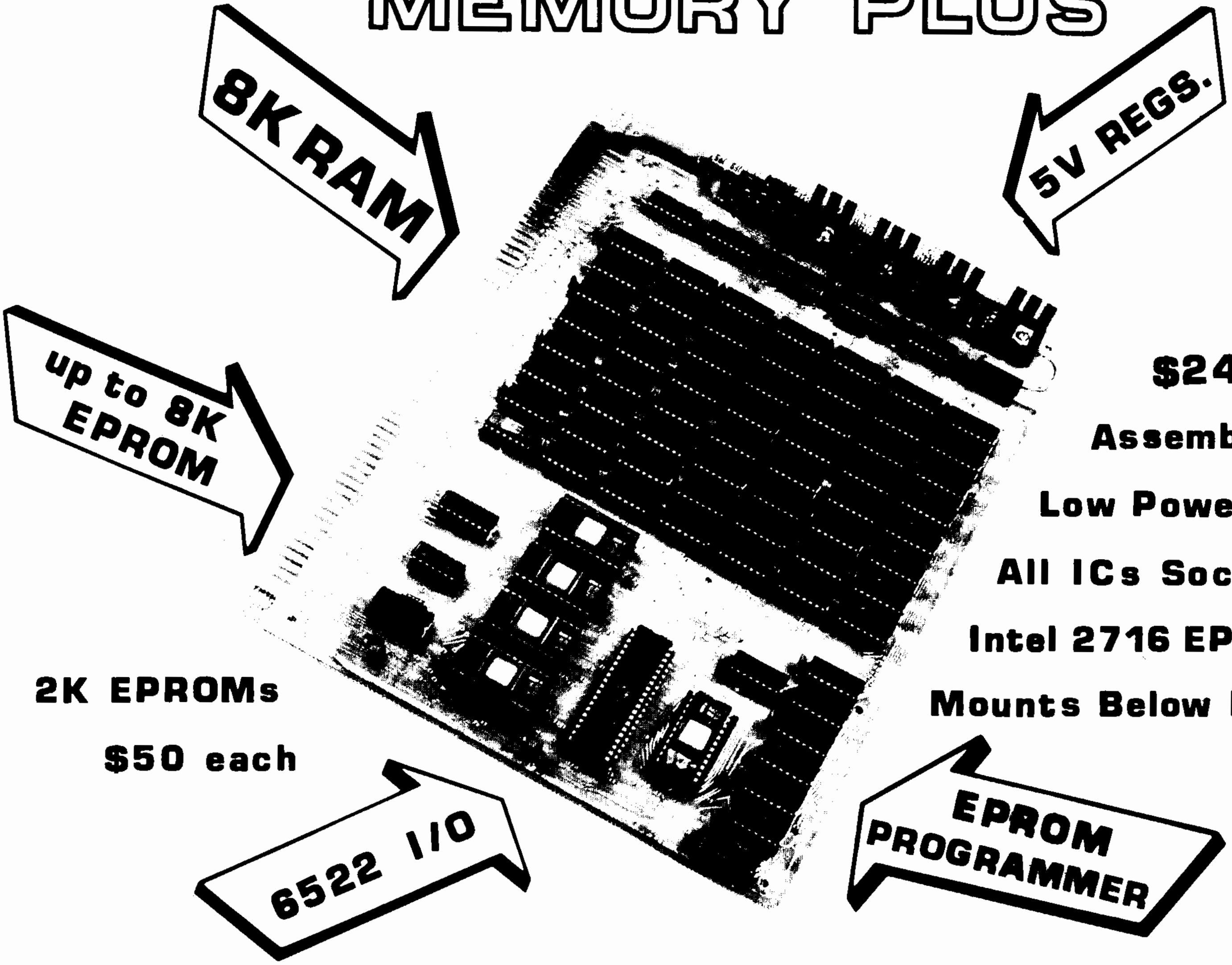
Name: ASM/TED
System: KIM-1 (may be modified for use with other 6502 based systems)
Memory: 6K RAM
Language: Assembler
Hardware: TTY
Description: The text editor performs line editing in RAM and can dump/load to paper tape or audio cassette. The resident assembler is single-pass using the standard MOS Technology syntax. Source code may be paper tape or memory resident and object code is always to memory.
Copies: Information not provided.
Price: $70.00
Includes: 50 page manual, source listings, and object on KIM cassette or paper tape.
Ordering Info: Send $2.00 for current catalog of available software.
Author: Not specified
Available from:
ARESCO
450 Forest Ave., Q-203
Norristown, PA 19401

Name: MicroChess
System: Basic KIM-1
Memory: Basic KIM-1
Language: Assembler
Hardware: Basic KIM-1
Description: Plays a reasonably good game of chess on a basic KIM-1. Has programmed openings. User enters his move via the KIM keypad and the KIM Display shows the move. The computer then makes its move and displays it. Program may be set to play at different speeds: 3, 10, or 100 seconds per move average. A great way to demo your KIM.
Copies: Hundreds
Price: $10.00 without cassette
$15.00 with cassette
Includes: Operator's manual, source listings, and a detailed discussion of the operation of the program. Object on cassette tape optional.
Ordering Info: Specify tape or not.
Author: Peter Jennings
Available from:
Micro-Ware Ltd.
27 Firstbrooke Road
Toronto, Ontario
Canada, M4E 2L2

The COMPUTERIST
P.O. Box 3
S. Chelmsford, MA 01824

Three PLUSes for the KIM-1:

MEMORY PLUS™



$245⁰⁰

Assembled

Low Power RAM

All ICs Socketted

Intel 2716 EPROMs

Mounts Below KIM-

2K EPROMs

$50 each

ENCLOSURE PLUS™ $30⁰⁰

Made by "The Enclosures Group" especially for the KIM-1/MEMORY PLUS combination. The MEMORY PLUS is mounted directly below the KIM-1 providing a compact package about 2.5" high which affords your system a high degree of protection from damage, dust, curious fingers, etc.

POWER PLUS™ $40⁰⁰

Designed specifically for the KIM-1. It has regulated +5V and +12V for the KIM-1 and more than enough unregulated +8V to power the MEMORY PLUS. It is completely enclosed in a black bakelite case measuring about 6.8" by 5.6" by 3". It is fully assembled and tested and weighs about 3 lbs.

MEMORY PLUS is $245 with everything except EPROMs.
KIM-1/MEMORY PLUS Cables are $10.00
Includes 60 page manual, cassette tape, connectors.

The COMPUTERIST
P.O. Box 3
S Chelmsford, MA 01824
617/256-3649

APPLE II PRINTING UPDATE

C. R. (Chuck) Carpenter W5USJ
2228 Montclair Place
Carrollton, TX 75006

"Printing with the Apple II" [MICRO #3] included information that has been revised. Since the article was written, I've improved some things and I'd like to pass them along.

The Adapter Didn't

After using the adapter circuit for a couple of months, I took a good look at what was happening. The conclusion was nothing! Initially, it didn't work when I connected it to the RS-232 receiver on the PS-40. I connected it to the serial TTL input (pin A7) and it worked. The voltage swing wasn't excessive (clamped with some diodes), so I left it hooked-up. Should have been a clue. But at the time I didn't see it, and anyway, it worked.

During one of our (infrequent) snowed-in days here in Texas, I had time to think about it. There wasn't any apparent reason not to hook it up directly; and I did. It worked the way it should so I had a no-interface-required computer to printer system. When I received my new Apple Operator's Manual I noticed a new interface circuit, not the one I used as originally provided.

All that is needed is to connect a signal lead and ground from the Apple to the printer. The signal lead connects to Pin 15 of Apple's game paddle connector. Also to Pin A7, TTL serial data in, on the printer. I soldered the game paddle connector to the 16 pin header. No other connections needed.

Now You Can Start and Stop

Ted Spradley, a programmer/engineer at work, helped me with the machine language print program. His analysis suggested restoring the page zero registers to make the print routine stop. As you more experienced programmers would know, it worked. I rewrote the program to store and restore the page zero data and now the routine turns on and off under program control. The program, shown in Figure 1, was a revelation to me. Again, my thanks to Ted for his assistance.

The Blues Are Gone

Most of my programs are printed on the paper that turns blue (and fades). Telpar has a black on off-white paper now. This new paper makes a much sharper copy too. The blue paper was also susceptible to smearing. This did not help the copy quality either, photographically or Xerographically.

There! Now that the problems are resolved, what's holding you back? Let's get printing.

Author's Note: Even if you don't have a printer, the print routine is useful. Use it to slow the screen speed down. This way you can read a listing during a slow scroll.

Getting Decimal Values From Hex Data

For some other program, POKE was used to enter machine language from BASIC. I did this for the print routine. All the HEX values have to be converted to decimal. At first I did this with the TI Programmer. Then I "discovered" what PEEK is all about. A BASIC program to print the decimal values simplifies the job. Convert the first and last addresses (to do a range of addresses) to their decimal values These values are 875 and 967 for the print program. Then use them in a FOR-NEXT routine like this:

```
100 FOR I=875 TO 967:PRINT PEEK(I);:
    PRINT" ";:NEXT I:END
```

This reduced a two hour job to about ten minutes. Hooray for progress.

Listing

```
*36BLLL

036B-   A5 36       LDA   $36
036D-   8D C6 03    STA   $03C6
0370-   A5 37       LDA   $37
0372-   8D C7 03    STA   $03C7
0375-   A9 89       LDA   #$89
0377-   85 36       STA   $36
0379-   A9 03       LDA   #$03
037B-   85 37       STA   $37
037D-   60          RTS
037E-   AD C6 03    LDA   $03C6
0381-   85 36       STA   $36
0383-   AD C7 03    LDA   $03C7
0386-   85 37       STA   $37
0388-   60          RTS
0389-   84 35       STY   $35
038B-   48          PHA
038C-   20 A5 03    JSR   $03A5
038F-   68          PLA
0390-   C9 8D       CMP   #$8D
0392-   D0 0C       BNE   $03A0
0394-   A9 8A       LDA   #$8A
0396-   20 A5 03    JSR   $03A5
0399-   A9 58       LDA   #$58
039B-   20 A8 FC    JSR   $FCA8
039E-   A9 8D       LDA   #$8D
03A0-   A4 35       LDY   $35
03A2-   4C F0 FD    JMP   $FDF0
03A5-   A0 0B       LDY   #$0B
03A7-   18          CLC
03A8-   48          PHA
03A9-   B0 05       BCS   $03B0
03AB-   AD 58 C0    LDA   $C058
03AE-   90 03       BCC   $03B3
03B0-   AD 59 C0    LDA   $C059
03B3-   A9 D3       LDA   #$D3
03B5-   48          PHA
03B6-   A9 20       LDA   #$20
03B8-   4A          LSR
03B9-   90 FD       BCC   $03B8
03BB-   68          PLA
03BC-   E9 01       SBC   #$01
03BE-   D0 F5       BNE   $03B5
03C0-   68          PLA
03C1-   6A          ROR
03C2-   88          DEY
03C3-   D0 E3       BNE   $03A8
03C5-   60          RTS
03C6-   F0 FD       BEQ   $03C5
```

HEX Dump

```
*36B.3C7

036B- A5 36 8D C6 03
0370- A5 37 8D C7 03 A9 89 85
0378- 36 A9 03 85 37 60 AD C6
0380- 03 85 36 AD C7 03 85 37
0388- 60 84 35 48 20 A5 03 68
0390- C9 8D D0 0C A9 8A 20 A5
0398- 03 A9 58 20 A8 FC A9 8D
03A0- A4 35 4C F0 FD A0 0B 18
03A8- 48 B0 05 AD 58 C0 90 03
03B0- AD 59 C0 A9 D3 48 A9 20
03B8- 4A 90 FD 68 E9 01 D0 F5
03C0- 68 6A 88 D0 E3 60 F0 FD
*
```

Print Routine

START Print	STOP Print
*36BG	*37EG
>CALL 875	>CALL 894
]SP=USR(875)	]EP=USR(894)

Type in one of above and then type RETURN to activate the command.

* = from Apple Monitor
\> = from Integer BASIC
] = from Applesoft BASIC

Change 03B4 to 4D for 300 baud.

Figure 1

Listing and HEX Dump of Machine Language Print Routine

WE'RE THE APPLE EXPERTS

Check our low prices and large selection of computers, software and peripherals.

APPLE II PERIPHERAL INTERFACE CARDS:

- S-100 BUS INTERFACE ($160)†
  - Connect the Apple II to an S-100 Bus Motherboard
  - Will Run Almost All Memory, I/O, and Special Purpose S-100 Boards
  - All Interconnecting Cables and Plugs Supplied (S-100 Motherboard and Power Supply Not Included)
- PROGRAMMABLE PRINTER INTERFACE ($80)
  - Onboard EPROM Printer Driver
  - Full Handshake Logic
  - High Speed Parallel Output Port Capability
  - Provision for 256 Byte I/O Drive in EPROM
  - Printer, Driver Programs Available for Centronic, SWTPC-40, and Other Printers
- PROTOTYPING BOARD ($24)
  - Sixteen Sq. Inches of Development Surface
- EXTENDER BOARD ($24)
  - Completely Contained Inside Apple II
  - Compatible With Prototyping Board

†Delivery March, 1978

- **FLOPPY DISC SYSTEM** ($2100)
  - Programs Saved and Loaded by Name
  - Powerful Firmware DOS File Handling Capability
  - 252K Bytes Storage Capacity. 8 Inch Dia. Disc
  - Capable of Utilizing Up to 4 Drives (One Million Bytes)
  - File Handling as Easy as Inputing or Printing
  - Access Methods: Stream, Punctuated, Relative, Direct
- APPLE POWER CONTROL INTERFACE†
  - Up to Sixteen Control Channels
  - Control Room Lights, Stereo Equipment, Security Systems, Electrical Appliances
  - Handle Up to 1000 Watts per Channel Directly From Program Control
  - Complete Isolation of the Computer From the AC Line
  - PRICE:
    - Apple Power Interface Board and One Power Control Module ($85)
    - Additional Power Control Modules (Controls Four AC Circuits) ($25)
    - Appliance Control Module (Controls One AC Circuit) ($7.50)

SOFTWARE FOR APPLE II

■ Home Financial Record Program (Req. 16K Sys)	$ 20
■ Business Inventory (Req. 20K Sys)	$ 40
■ Bob Bishop's High Resolution Games (Req. 16K Sys)	$ 40
– Star Wars	$ 15
– Rocket Lander	$ 15
– Saucer Invasion	$ 15
■ Apple Music (Three Octaves)	$ 20
■ Data Save to Cassette	$ 20
■ Word Processor (Req. 20K System)	$ 50
■ High Res. Graphics for F.P. Basic (Req. 24K System)	$ 30
■ High Res. Paddle Drawing Routine	$ 20
■ Othello Game	$ 10
■ Send for Listing of Our Games, Color Graphics, and Sound Programs	

ADD ON MEMORY FOR APPLE II

■ Set of Eight	4K RAM CHIPS	$ 32
■ Set of Eight	16K RAM CHIPS	$320

BOOKS

■ 6502 Programming Manual	$ 10
■ 6502 Hardware Manual	$ 10

COMPUTER COMPONENTS, INC. OF ORANGE CO.
6791 WESTMINSTER AVE.
WESTMINSTER CA. 92683
(714) 898-8330

Prices subject to change without notice.

Mastercharge, Visa, B of A accepted. No C.O.D. Allow two weeks for personal check to clear. Add $1.50 for handling and postage. For computer system, please add $10.00 for shipping, handling, and insurance. California residents add 6% sales tax.

MICRO STUFF

Mailing Labels

Barring unforseen difficulties (last May we lost electricity for four days due to a snow storm), the mailing label on your copy of MICRO will have been generated on a KIM-1 with a Diablo type printer and the HELP Mailing List Package. Note near your name the two or three characters. The first two digits indicate the last issue you are scheduled to receive under your current subscription: 06 = issue number 6. The third character has particular meaning:

X = your name will appear on any mailing lists we sell, unless you notify us to remove it;

any other letter indicates you are getting MICRO free as an advertiser, exchange, or something;

no letter indicates that your name will not be included in mailing lists we sell, per your request.

Our New Printer

This issue of MICRO is being printed by a new printing company. We anticipate that the quality will be as good as the previous work.

Deadlines

With our new printer (he's cheaper but takes longer), deadlines are even more important than before. All ADs must be received by May 14 for the June/July issue. Articles should be received as soon as possible.

Calendar/Directory

If enough information is provided to make it worthwhile, we can publish a regular Calendar of 6502 related events and a Directory of 6502 Clubs. Since MICRO is only published every other month, remember to give information for several months at a time.

MICROBES

Tiny Bugs in Previous MICROs

EMPLOYING THE KIM-1 AS A TIMER ....

3:5 020E should be A9 99 LDAIM $99 since the processor is in decimal mode, not binary.

3:7 02A6 should be E4 03 not E0 03.

LIGHTING THE KIM-1 DISPLAY

Back cover There is no need to add Hex 80 to the sum of the individual LED segments to control PA7. It does hurt, but it is not required.

STANDARD 6502 ASSEMBLY SYNTAX?

Hal Chamberlin
29 Mead Street
Manchester, NH 03104

I could not help noticing the comment about MOS Technology's assembler syntax for the 6502 in MICRO #2. Judging from the force of that comment and the fact that every 6502 program I have seen uses a different assembler and systax there must be a great deal of discontent with MOS Technology's syntax.

Consideration of the history of 6502 development is all that is necessary to explain most of the features of its assembler syntax. The designers initially worked at Motorola with the goal of incorporating leading features of the PDP-11 instruction set into the 6800. Later, after leaving Motorola and designing the 6502 for MOS Technology, their PDP-11 experience served as a model for an assembler syntax to adequately handle the 13 addressing modes and other features of their creation. The result is the syntax described in about 10 square inches on the 6502 card and illustrated by the KIM assembly listings we all practically know by heart. The PDP-11 is one of the most used minicomputers ever and I have not heard of any significant group of '11 users abandoning DEC's syntax even though it can become a little cryptic.

So let us take a close look at the MOS Technology syntax, iterate what is right about it, and see how we can live with those features that are less than ideal. Note that I am not at all against extensions of what they have defined but I think it is important that an assembler be able to correctly assemble the KIM source as printed.

First we have the assembler directives and other statements that have nothing to do with the instruction set. For the most part these have been lifted directly from the PDP-11 assembler manual. The distinguishing feature about these statements is that they are preceeded by a period. I see nothing particularly wrong with these except perhaps that some of them are longer than three characters meaning that an opcode scanner might have to be a little more sophisticated than it would otherwise be. One definite problem though is the method that must be used to reserve areas of memory for data storage. I prefer the "DS 5" form rather than the ".=.+5" form for reserving five bytes probably because of an IBM background. But the real problem is that unless the assembler is carefully written, the location counter value printed to the left of such a statement gives the address of the first byte of memory used in the next statement rather than the address of the first byte of memory reserved in this one. However I think that the latter form can be lived with if one realizes that the expression ".=.+" is really the same as "DS" and provided the assembler prints the right address.

Now what about the machine instructions themselves? A tendency noted in several homebrew assemblers is to give every addressing mode variation of every instruction a different mnemonic. Although this is a good advertising ploy to swell the 57 listed op codes into 151 "variations", it does not make good sense. The operation code should merely specify the operation and the operand column should specify the operands. In my way of thinking the addressing mode is part of the operand (it tells where the operand is) and not the operation. Of course MOS Technology violated this somewhat by putting the register designation in the op code but that is not nearly as bad as putting everything in the op code.

One particularly nice feature of the existing syntax is the specification of the two indirect addressing modes. The designation "(SYMB,X)" clearly indicates that the value of SYMB is added to X before looking in the base page for the effective address and the designation "(SYMB),Y" says that the indirect cycle occurs before the contents of Y are added in to form the effective address. There should never be any problem with the use of parentheses for indicating indirect and the use of parentheses in arithmetic expressions. It is unfortunate however that indexed addressing is of the form "SYMB,X" rather than "SYMB(X)" as on most other systems but it can certainly be lived with.

With respect to the other addressing modes, the assembler should take care of determining whether the "zero page" form or the "absolute" form is to be used. Essentially the assembler would look at the value of the address and if it is less than 0100 (hex), use the appropriate zero page addressing form of the instruction. Besides always insuring the shortest possible program (both space and time), it frees the programmer from learning many of the addressing mode restrictions of certain instructions. The assembler will flag an error only when it is physically impossible to perform the requested operation.

One last minor gripe is the field separators (colon after symbols and semicolon before comments) required which adds (slightly) to typing effort and uses three valuable print column positions. Of course this is also straight out of the PDP-11 assembler. I know a powerful assembler can be written without this requirement and still have free format (IBM 360 assembler) but my programmer friends say that explicit delimiters can have important advantages. Anyway I live with it.

I can hear the cries now of "Sure it makes sense but it is so complicated to write a syntax analyzer for it". Of course our cross-town rivals (8080, Z-80) are already well into macro assemblers and linking relocating loaders and we are still working out the assembler syntax for our baby! If we believe that ours is a more powerful computer, surely an assembler with automatic address mode selection and conformance to our own manufacturer's assembly language is not too difficult a task to handle.

Editor's Note: While I do not want to use too much space in MICRO for debates on matters of personal preference, I will make space available in the next issue of MICRO for a rebuttal by a proponent of an alternative syntax. If no one writes such a rebuttal, I will do it myself, but I would much prefer to hear from one of you.

A WORM IN THE APPLE?

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

There may be a serious problem hidden deep within the Apple II according to John Conway and Jack Hemenway of EDN magazine. As part of their system design project based on a bare-board Apple - "Project Indecomp" - they tried to interface a 6820 PIA to the Apple, and uncovered a potentially serious problem. The normal way to operate a 6502 based system is to provide an external clock [phase 0] to the 6502 which then generates two non-overlapping clock signal [phase 1 and phase 2] which are used to control all system timing. For some reason, the design of the Apple II violated this basic clock scheme and uses the phase 0 external clock instead of the 6502 generated phase 2 clock. While these two clocks are very similar, they are not identical. Phase 1 and phase 0 have an overlap of about 50 nanoseconds. For many parts of the system this is not important, as indicated by the fact that the Apple II works. For other devices, however, such as the 6820 PIA, this difference is critical to the extent that the device simply will not work. A report in EDN scheduled for 20 May will cover this problem in detail, and we will try to get more info for the next issue of MICRO. Is the problem serious? Critical? Fatal? It is probably too early to judge the effect of this problem. It may not have an adverse effect in many systems. It may be possible to correct. Or it may be a very serious system problem.

WRITING FOR MICRO

One of the reasons I like the 6502 is that it seems to attract a lot of very interesting, active, enthusiastic users. I spend several hours on the phone each week talking to people who are so excited about what they are doing with their system that they just have to talk to someone. Oh, sometimes they pretend they have some "burning" question or want to order some small item, but really they mostly want to tell someone about all of the fun they are having or the discoveries they are making.

While I enjoy these conversations, and consider them one of the "fringe benefits " of editing MICRO, it disturbs me that many of these enthusiasts who are willing to spend five to ten dollars on a phone call to me, are not willing to spend a little time writing down their information for publication in MICRO where thousands can share it (and they can earn a few dollars).

MICRO, in order to serve its main purpose of presenting information about all aspects of the 6502 world, needs to receive information from a wide variety of sources. To achieve a more balanced content, we desperately need articles on: industrial, educational, business, home, and other real applications of systems; non-KIM, -Apple, -PET systems, homebrew and commercial; techniques for programming, interfacing, and expanding systems; and many other topics. Look to your own experience. If you have anything to share, then take the time to write it down. The "Manuscript Cover Sheet" on the next page should serve as a guide and make it a little easier to submit your article.

MICRO SUBSCRIPTIONS

MICRO is published bi-monthly, six issues per year. Single copy price is $1.50. Subscriptions are available at the following rates:

Surface Mail: All Countries	$6.00
Air Mail:	
Central America	$12.00
Europe & South America	$14.00
Other Countries	$16.00

All subscriptions start with the NEXT issue after receipt of your order.

Back issues are available, while they last, at $1.50 per copy (plus $1.25 for air mail postage overseas or $.50 for surface postage overseas).

The COMPUTERIST
P.O. Box 3
S Chelmsford, MA 10824

MANUSCRIPT COVER SHEET

Please complete all information requested on this cover sheet.

Date Submitted: ________________________________

Proposed Title: ________________________________

Author(s) Name(s): ________________________________

____

Mailing Address: ________________________________
(This will be published.)
____

Area Code: ____ Phone: ________________
(This will NOT be published.)

AUTHOR'S DECLARATION OF OWNERSHIP OF MANUSCRIPT RIGHTS: This manuscript is my/our original work and is not currently owned or being considered for publication by another publisher and has not been previously published in whole or in part in any other publication. I/we have written permission from the legal owner(s) to use any illustrations, photographs, or other source material appearing in this manuscript which is not my/our property. If required, the manuscript has been cleared for publication by my/our employer(s). Note any exceptions to the above (such as material has been published in a club newsletter but you still retain ownership) here:

Signature(s): ________________________________

____

Date: ________________

Any material which you are paid for by The COMPUTERIST, whether or not it is published in MICRO, becomes the exclusive property of The COMPUTERIST, with all rights reserved.

A Few Suggestions

All text material will be retyped. Therefore your format does not matter as long as it is readable. Double spaced, typed, is preferable, but not required. Any figures should be neatly drawn to scale as they will appear in MICRO. If we have to redraw the figures and diagrams, then we normally will pay less for that page. Photographs should be glossy prints either the same size as the final will be or twice the final size. We will re-assemble all programs to obtain clean listings using the syntax we have adopted (see inside back cover - MICRO #1). Since others will be copying your code, please try to thoroughly test it and make sure it is as error free as possible. Submit your articles early. We will try to get a proof back to you for final correction, but with our tight schedule this may not always be possible. Send your manuscripts to:

Robert M. Tripp, Editor, MICRO, P.O. Box 3, So. Chelmsford, MA 01824, U.S.A.

6502 BIBLIOGRAPHY
PART III

William Dial
438 Roslyn Avenue
Akron, OH 44320

180. Gordon, H.T., "Decoding 650X Opcodes", Dr. Dobbs Journal 2, No. 7, pp 20-22 (Aug. 1977)
    Subroutines that can be used with KIM.

181. Butterfield, Jim F., "A High-Speed Memory Test Program for the 6502" DDJ 2, No. 7, p 23 (Aug. 1977)
    A memory test program written for the KIM system.

182. Anon. "Ohio Scientific's New Disc Operating System", DDJ 2, No. 7, p 32 (Aug. 1977)
    The OS-65D is a complete operating system for all disc based OSI computer systems. Includes DOS, 8K Basic, Assembler, Editor, Extended Debugger and a Disassembler.

183. Anon.., "OSI offers Computer that thinks in Basic for $298", DDJ 2, No. 7, p 39 (Aug. 1977)
    OSI's new Model 500 CPU board can be used as a stand-alone computer or as the PCU in a larger system.

184. Moser, Carl W., 3239 Linda Dr., Winston-Salem, NC 27106, DDJ 2, No. 8, p 28 (Sept. 1977)
    Announcement of New Product: $25 for 6502 Editor and Assembler Hex Listing and Manual. Configured for TIM Systems.

185. Anon.,"1K Corner", OSI Small Systems Journal 1, No. 4, p 3 (Oct. 1977)
    Hex address and offset calculator program resides at 0DDE to 0EE4.

186. Anon., "Now You Can Play Star Wars", OSI Small Systems Journal 1, No. 4, pp 11-13, (Oct. 1977)
    Star Wars program by Robert L. Coppedge requires 8K Basic, OSI 440 Video Board and at least 4K of RAM.

187. Anon., "Conventional Typewriter", OSI Small Systems Journal 1, No. 4 pp 8-9 (Oct. 1977)
    Gary Smith's program for using the OSI-65V when interfaced to a printer to be used as a conventional typewriter and also modify the text for a data file.

188. Gordon, H.T., "OPLEGL Correction and a 6502 Scanning-Debugger", DDJ 2, No. 9, pp 42-44 (Oct. 1977)
    Gordon offers a corrected version of his 650X subroutine, OPLEGL, and gives a new byte-count subroutine, NUMBYT. A new scanning-debugger, SIMBUG, is submitted.

189. Swope, J., "6502 Goodies", DDJ 2, No. 9, Issue 19, p 45 (Oct. 1977)
    Swope, President of CGRS Microtech, PO Box 368, Southampton, PA 18966, announces that his company has finished a 6502 computer board for the S100 bus.

190. Wozniak, Stephen, "Sweet 16: The 6502 Dream Machine", Byte 2, No. 11, pp 150-159 (Nov. 1977)
    Sweet 16 is a 16 bit "metaprocessor" in software, intended as a 6502 enhancement package, not a stand-alone processor.

191. Shattuck, Bob and Schmidt, Bill, "Receive CW with a KIM-1", 73 Magazine, No. 206, pp 100-104 (Nov. 1977)
    A program for receiving CW with optional TTY or KIM display.

192. Johnson, Donald J., "KIM-1 Sidereal/Solar Clock Correction", Interface Age 2, No 12, p 9 (Nov. 1977)

A correction in the listing given in the August issue of Interface Age permits 24-hour operation.

193. KL Power Supplies, PO Box 86, Montgomeryville, PA 18936, Interface Age 2, N No. 12, p 140 (Nov. 1977)
The Model 512, 4.5 amp. power supply is designed for KIM-1.

194. Micro Technology Unlimited, Box 4596, Manchester, NH 03108, Interface Age 2, No. 12, p 140 (Nov. 1977)
The MTU Model K-1000 power supply is designed to power the KIM-1.

195. Wasson, Philip A., "Trace", KIM-1/6502 User Notes, Issue 7/8, pp 2-3 (Sept & Nov 1977)
With this program and about $2.00 worth of hardware you can see displayed on an oscilloscope screen, all of the registers in the 6502 and three consecutive memory locations.

196. Ohsiek, Charles C., "ID on Audio Cassette for SUPERTAPE", User Notes, Issue 7/8, p 4 (Sept & Nov 1977)
Program allows writing an ID on the audio cassette tape prefixing the data SUPERTAPE writes out.

197. Hawkins, George W., "2-Task Alternating Scheduler Routine", User Notes, Issue 7/8, p 5 (Sept & Nov 1977)
Program allows two programs to be run together in the KIM-1.

198. Gordon, Hal, "A Catalog of KIM-1 ROM Bytes", User Notes, Issue 7/8, p 5, (Sept. & Nov. 1977)
A table of the location of ROM bytes.

199. Anway, Allen, "Program BRANCH", User Notes, Issue 7/8, p 6 (Sept & Nov 1977)
With this program you can go through your program, find the Branch instructions and force the branch to see where you will end up.

200. Pollock, Jim, "KIM-1 to S-100 Bus Adapter", User Notes, Issue 7/8, p 7 (Sept. & Nov. 1977)
This adapter allows KIM-1 to be used with S-100 boards such as the $125 8K RAM board of Ithaca Audio.

201. Heinz, Harvey, "A Simple Music Program for KIM", User Notes, Issue 7/8, pp 8-9 (Sept. & Nov. 1977)
This is an excellent tutorial program with basic level explanations.

202. Hapgood, Will, "An A/D Converter", User Notes, Issue 7/8, pp 10-11, (Sept. & Nov. 1977)
A circuit for making very accurate A/D conversions using a Motorola dual-slope conversion chip, MC 1405 or 1505.

203. Butterfield, Jim, "KIM Blackjack", User Notes, Issue 7/8, pp 11-13, (Sept. & Nov. 1977)
Game uses the KIM display to good advantage in this program.

204. Strandtoft, B., "KIM-1 Resident Programs and Subroutines", User Notes, Issue 7/8, p 14 (Sept. & Nov. 1977)
A list of KIM Monitor routines with brief explanations.

205. Goenner, Markus P., "TTY Rapid Load", User Notes, Issue 7/8, p 15, (Sept. & Nov. 1977)
Program starts at 0000 and is fully relocatable.

206. Parson, Charles H., "Read Temperature Once per Minute", User Notes, Issue 7/8, pp 16-18, (Sept. & Nov. 1977)
Program for temperature control systems.

207. Oliver, John and Hall, Williamson, "A KIM-1 Binary Dump and Load Routine", User Notes, Issue 7/8, pp 19-20, (Sept. & Nov. 1977)
SUPERDUMP/SUPERLOAD allows the use of the KIM-1 Cassette tape interface to read and write data blocks under program control. 1K bytes are dumped or loaded in less than 12 seconds.

208. The COMPUTERIST, PO Box 3, S Chelmsford, MA 01824, "MEMORY PLUS for KIM-1", New Product Announcement, MICRO, No. 2, p 2 (Dec 1977-Jan 1978)
New board for fitting directly beneath the KIM-1 has 8K RAM, 8K EPROM MOS Technology Versatile Interface Adapter, EPROM programmer, On Board Voltage Regulators; fully assembled and tested $245; Intel 2716 2K EPROMS extra $50 each.

209. Cole, Phyllis, "PET Update", Peoples Computers 6, No. 3, pp 6-7 (Nov-Dec1977)
Several rumors on the PET are answered.

210. Cole, Phyllis, "Our PET's First Steps", Peoples Computers 6, No. 3, pp 8-10, (Nov-Dec 1977)
An account of bringing a PET on stream in spite of a few initial bugs and limited documentation at the time.

211. Inman, Don, "The Data Handler Users Manual: Part 6", Peoples Computers 6, No. 3, pp 11-15, 44 (Nov-Dec1977)
The latest contribution in this series covers multiplication and division programs.

212. The 6502 Program Exchange, 2920 Moana, Reno, NV 89509, "Software Announcement:, On Line 2, No. 15, p 7 (Nov. 16, 1977)
Recent software includes an extended version of FOCAL, a 4K resident assembler and an efficient Mini-Editor.

213. MSS, Inc., "65XX Programs Available", New product announcement, On Line 2, No. 17, p 2 (Dec. 30, 1977)
Programs available include Disassembler, Loader, Punch, Dump, Memory Editor, Life Game, File Commands, Assembler/Text Editor, etc., MSS, Inc., 3201 East Pioneer Parkway, Suite 40, Arlington, Texas 76010.

214. Rychlewski, Walter J., III, "PET Demonstration Tape", On Line 2, No. 17, p 7, (Dec. 30, 1977), New Product Announcement.
Ten BASIC programs demonstrate most of the features of the PET; includes graphics and real time clock; $10 cassette. 603 Spruce St., Liberty, MO 64068.

215. Purser, Robert Elliott, PO Box 446, El Dorado, CA 95623, On Line 2, No. 17, p 9 (Dec. 30, 1977), New Product Announcement.
PET layout sheet with SASE, free.

216. Anon, "1K Corner: Cassette Loader and Memory Block Transfer", OSI Small Systems Journal 1, No. 5, p 3 (Nov. 1977)
With this program the user may record his own programs via the 430B Super I/O Board in a format that is recognizable to the auto-load function in the 65V Monitor PROM.

217. Anon, "Two New Software Packages", OSI Small Systems Journal 1, No. 5, pp 4-7 (Nov. 1977)
OSI has released two major new Disc software packages, Word Processor and 9-Digit BASIC which run under OS-65D version 2.0

218. Anon, "Two New Video Games", OSI Small Systems Journal 1, No. 5, pp 8-12 (Nov. 1977)
SAM (Surface-to-Air Missile) and BOMBER require OSI 8K BASIC, OSI 440 Video Board, terminal and Keyboard, and at least 4K of RAM.

219. Pfeiffer, Erich A., "Seasons Greetings", OSI Small Systems Journal 1, No. 5, p 12 (Nov. 1977)
Program using PEEK and POKE instruction to present a video message.

220. Anon, "ASCII Files under OS-65D", OSI Small Systems Journal 1, No. 5, pp 13-15 (Nov. 1977)
Auxilliary assistance program for a file system.

221. Anon, "BASIC in ROMS", New Product Announcement, OSI Small Systems Journal, 1, No. 5, p 15 (Nov. 1977)
The BASIC in ROM set No 65AB including 4 ROMS, one EPROM for the 6502 system; Another version 65VB for 440 Video system also available. Either version is $99.

222. Struve, Bill, "A $19 Music Interface", Byte 2, No. 12, pp 48-69, 170-171 (Dec. 1977)
Some theory and a KIM-1 interface for computer/music addicts.

223. Gordon, H.T., "The XF and X7 Instructions of the MOS Technology 6502", Byte Magazine 2, No. 12, p 72 (Dec. 1977)
A look at some of the unlisted instructions available in the 6502.

224. Forethought Products, PO Box 386, Coburg, OR 97401, Kilobaud, No. 12, p 15 (Dec. 1977), New Product Announcement.
A new board that makes S-100 (Altair/Imsai) type boards compatible with KIM. Motherboard has 8 slots, and does not alter the operation of KIM in any way.

225. Lancaster, Don, "TVT Hardware Design", Kilobaud, No. 12, pp 30-34 (Dec1977)
Part 1; instruction decoder and scan. Taken from Lancaster's new book, "The Cheap Video Cookbook" on the TVT-6L.

226. Blankenship, John, "Expand Your KIM!", Kilobaud, No 12, pp 36-42 (Dec 1977)
Part 2 discusses cabinet, nuts and bolts, in this series.

227. Byrd, David A., "TVT-6 Display Uncrowding", Popular Electronics 12, No. 6, p 6 (Dec. 1977)
Gives a technique for correction of a crowding of the display in Lancaster's TVT-6 Video Display.

228. Pittelkau, Clifton W., "The Bionic Clock!", 73 Magazine, No. 208, pp 102-105 (Jan. 1978)\
Software to add a real time clock to your KIM.

229. Eaton, John, "Growing with KIM", Kilobaud, No. 13, pp 36-39 (Jan. 1978)
Expansion PC Board provides compatibility with S-100 bus.

230. Chamberlin, Hal, "Software Keyboard Interface", Kilobaud, No. 13, pp 98-105 (Jan. 1978)
Shows how with a minimum of hardware and minimum cost.

231. Kraul, Douglas R., "Designing Multichannel Analog Interfaces", Byte 2, No. 2, pp 18-23 (June, 1977)
Hardware and software for an 8-channel analog I/O.

232. Fylstra, Dan, "Interfacing the IBM Selectric Keyboard Printer-Teaching KIM to Type", Byte 2, No. 6, pp 46-52, 133-139 (June 1977)
Hardware and software for hooking KIM up to a Selectric.

233. Jobs, Steven, "Interfacing the Apple Computer", Interface Age 1, No. 11, pp 65-66 (Oct. 1976)
Interfacing with a printer.

234. Wozniak, Steve and Baum, Allen, "A 6502 Disassembler from Apple", DDJ 1, No. 8, pp 22-25 (Sept. 1976)
Displays single or sequential 6502 instructions in mnemonic form.

235. Grater, Robert, "A Teletype Alternative", Kilobaud, No. 1, pp 114-116 (Jan77)
Convert parallel input TVT to serial operation, for KIM.

236. Anon."Errata to Zieglers 6502 Bug Program", DDJ 1, No. 8, p 33 (Sept. 1976)
Corrections for the listing given earlier in DDJ 1, No. 3.

237. Parks, Don, "Adding PLOP to your System", Kilobaud, No. 5, p 98 (May 1977)
A 6502 noisemaker for computer games.

238. Rankin, Roy, "Errata for Rankin's 6502 Floating Point Routines", DDJ 1, No. 10, p 57 (Nov/Dec, 1976)
Correction of a bug found in his earlier routine published in DDJ 1, No.7.

239. Lancaster, Don, "Build the TVT-6, Part II", Popular Electronics 12, No. 2 pp 49-55 (August, 1977)
System debugging, software, and how to interface to KIM and other systems.

240. The Data Mart, 914 East Waverly Drive, Arlington Heights, IL 60004, New Product Announcement, "Real Time Clock", On Line 2, No. 18, p 11 (Jan 18,1978)
Real Time Clock and Calendar for 6502. Assembled $95.

241. Optimal Technology, Inc., Blue Wood 127, Earlysville, VA 22936, Hardware Announcement: PROM Programmer, On Line 2, No. 18, p 11 (Jan 18, 1978)
Programmer for KIM programs both the 2708 and 2716 EPROMS. Runs on all 650X systems.

242. Trageser, Jim, "TVT-6L Correction", Kilobaud, No. 12, p 123 (Dec. 1977)
Corrections for the June 1977 article by Lancaster.

243. Meyers, Michael J., "Dedicated Controllers - There is Money to be Made", Kilobaud, No. 10, pp 84-92 (Oct. 1977)
Hobbyists should take advantages of opportunities to make money with their KIM or other micro.

244. Burhams, R.W., "Consider a MITE Printer", Kilobaud, No. 11, pp 38-42, (Nov. 1977)
At $276, the Mite Expandor is an alternative to the ASR-33 TTY.

245. Penhollow, Bert G.H., "Binary to BCD Conversion for Microprocessors", Electronic Design, p 212 (Oct. 11, 1977)
Packs the units and tens into one byte.

246. Chamberlain, Hal, "Computer Bits: Computer Music Part II", Popular Electronics 10, No. 4, pp 88-91 (Oct. 1977)
A description of music techniques which have been implemented on the KIM-1 DAC board. Also discusses generation of Touch Tone codes.

247. Chamberlain, Hal, "Computer Bits: Computer Music Part I", Popular Electronics 10, No. 3, pp 116-119 (Sept. 1977)
Timed loop techniques for computer music programs.

248. Anon., "74 Megabyte Disc Review", OSI Small Systems Journal 1, No. 6, pp pp 2-6 (Dec. 1977)
OSI offers the 74 megabyte CD-74 disc drive for small computers. Has four aluminum disc platters about 12" diameter. $6000. 6502 Related.

249. Anon., "Article Sponsorship Program", OSI Small Systems Journal 1, No. 6, p 7 (Dec. 1977)
OSI will pay for and provide technical assistance for articles on OSI equipment or programs to be published in computer magazines. 6502 Related.

250. Anon., "1K Corner", OSI Small Systems Journal 1, No. 6, p 7 (Dec 1977)
Short Program for PRIME NUMBER GENERATOR.

251. Owens, Gerald, "Shoot the Gluck", OSI Small Systems Journal 1, No. 6 pp 8-10 (Dec. 1977)
A game for the 12K Challenger with video.

252. Anon., "Floppy Disk Users Group", OSI Small Systems Journal 1, No. 6 p 10 (Dec. 1977)
OSI has formed a users group to redistribute user-contributed software on diskettes. The first group of 6502 machine code programs (12 listings) is now available.

253. Anon., "Terminal/Cassette DOS Input Routine", OSI Small Systems Journal 1, No. 6 pp 11-12 (Dec. 1977)
Program for reloading or transferring program source code.

254. Anon., "New Diskette Software packages", OSI Small Systems Journal 1, No. 6, p 12, (Dec. 1977)
Work Processor WP-1 and WP-1A is a complete word processor. OS-65D Version 2.0 with Nine-digit BASIC. Disk-Test provides a quick functional check of the 6502 computer system.

255. Anon., "Bank Accounts", OSI Small Systems Journal 1, No. 6, pp13-15(Dec 1977)
Two practical programs: CHECKBOOK ACCOUNT and SAVINGS ACCOUNT.

256. Fylstra, Dan, "SWEETS for KIM", Byte 3, No. 2, pp 62-77 (Feb. 1978)
SWEETS, a Simple Way to Enter, Edit and Test Software, is a small text editor and assembler which operates on hexadecimal code and which is designed to fit in the KIM-1's 1K byte small memory while leaving room for the user's programs.

257. Feagans, John, "A Slightly Sour SWEET 16", Byte 3, No. 2, p 93 (Feb. 1978)
Correction of a slight bug in the Wozniak article in Byte, Nov. 1977.

258. Leasia, John D., "Random Errors", Byte 3, No. 2, p 93 (Feb. 1978)
Correction of an error in the pseudorandom number generator shown earlier in Byte, Nov. 1977, p 218.

259. Kathryn Atwood Enterprises, P.O. Box 5203, Orange, CA 92667, Byte 3, No. 2, p 187 (Feb. 1978), New Product Announcement
4K RAM board, KIM interface and Mother Board.

260. Electronics Warehouse Inc., 1603 Aviation Blvd., Redondo Beach CA 90278, New Product Announcement.
Apple II I/O Board Kit plugs into slot of Apple II Mother Board.

261. Pittelkau, Clifton W., "KIM-1 Can Do It!", 73 Magazine, No. 209, pp 68-71 (Feb. 1978)
Adapting a KIM-1 to function as a versatile RTTY terminal at nominal cost.

262. O'Reilly, Francis J., "Looking for a Micro?", 73 Magazine, No. 209, pp 76-77, (Feb. 1978)
Pro's and Con's of the KIM-1 as a micro.

263. Bridge, Theodore E., "A KIM-1 Disassembler", DDJ 2, No. 10, Issue 20, pp 12-13 (Nov.-Dec. 1977)
A modification of Wozniak's earlier 6502 disassembler.

264. Eaton, John, "MATHPAC: A Kimath Supplement", DDJ 2, No. 10, Issue 20, pp 15-21 (Nov.-Dec. 1977)
MATHPAC is designed to increase the power of a 6502 system. It takes the power of KIMATH and gives it to the user. The user's I/O ASCII device turns the system into a scientific calculator.

265. Osborne, Adam, "War of the Processors", SCCS Interface 1, No. 6, pp 14-17, (May, 1976)
Traces evolution of major microprocessors, including 6502 and compares their computing power.

266. Anon., "KIM-1, A complete Microcomputer System for $245", SCCS Interface 1, No. 6, pp 44-45 (May, 1976)
A new products announcement for KIM-1.

267. Teener, Mike, "Bits and Byters", SCCS Interface 1, No. 6, p 58 (May, 1976)
Historical note recaps Motorola's suit against MOS Technology over the 6502's predecessor.

268. MOS Technology, Inc., 950 Rittenhouse Road, Norristown, PA 19401, KIM Application Note #107702, "S-100 to KIM-4 Bus Adapter",
Mechanical details of a simple adapter that will plug into the KIM-4 Mother Board and which will accept certain compatible S-100 boards such as the Kent-Moore No. 60083 video display board or the Kent-Moore No. 60082 4K static RAM board.

269. MOS Technology, Inc., 950 Rittenhouse Road, Norristown, PA 19401, KIM Application Note #111477, "Using KIM as a Dedicated Controller"
The KIM itself can be used as a very low cost controller with the addition of a PROM, a power-on-reset modification, and some additional circuitry to transfer control to the added PROM upon power-up.

270. MOS Technology, Inc., 950 Rittenhouse Road, Norristown PA 19401, KIM Application Note #117701, "Digital-Analog and Analog-Digital Conversion Using the KIM-1"
This is essentially the same as Reference #172 on DeJong's article in MICRO No. 2. Uses a 1408 D/A converter with KIM together with hardware and software for D-A and A-D as well as software to store the A/D converter output and recall converted data, emulating a storage oscilloscope.

271. MOS Technology, Inc., 950 Rittenhouse Road, Norristown, PA 19401, KIM Application Note #771121, "Software Routines for TVT"
Machine Language program to use with external keyboard.

272. Optimal Technology, Inc., Blue Wood 127, Earlysville,VA 22936, On Line 3, No. 1, p 1 (Feb. 8, 1978). New Product Announcement.
2708/16 EPROM PROGRAMMER for KIM-1. Requires 1-1/2 I/O Ports. Assembled and tested $59.95. Kit $49.95.

273. Purser, POB 466, El Dorado, CA 95623, On Line 3, No.1, p 3 (Feb. 8, 1978)
Free Guidelines for writing programs for the TRS-80 and PET and then selling them to Radio Shack and Commodore. Send SASE.

274. Personal Software, P.O. Box 136-03, Cambridge MA 02138 On-Line 3 No 1 pg 4 (Feb. 8., 1987) New Product Announcement.
Four full length games on cassette for PET or TRS-80. POKER, ONE QUEEN, KINGDOM, MATADOR; $9.95 for all four. STIMULATING SIMULATIONS by Dr. C.W. Engel, and additional entertainment personal finance/investment, and other systems programs including a 6502 Assembler in BASIC.

275. 6502 Program Exchange, 2920 Moana, Reno, NV 89509, Kilobaud, p 7 (Mar. 1978)
Announcement of new 6502 Software including an extended version of FOCAL called FCL 65E (6.5K). Also a Mini-Manual to get you started on TIM or KIM systems.

276. Eaton, John, "Corrections", Kilobaud, No. 15, p 12 (March, 1978)
Note on the availability of drilled PC boards for Eatons' KIM expansion article in January 1978 Kilobaud.

277. Scogin, Tom, "AppleSOFT Benchmarks: Fast!", Kilobaud, No. 15, p 12 (Mar 78)
Gives times for seven benchmark programs using Apple-II Integer and Apple-II AppleSOFT versions of BASIC.

278. Blankenship, John, "Expand Your KIM!", Part 4., Kilobaud, No. 15, pp 84-88 (March, 1978)
Part four of this series uses a $10 circuit board with a SWTP keyboard and a PR-40 printer as a miniature teletype.

279. Zaks, Rodney, "Micro History", Personal Computing 2, No. 2, pp 31-35, (Feb., 1978)
History of microprocessors. Has a very small paragraph on the MOS Technology 650X family.

280. DeJong, Marvin L., "Employing the KIM-1 Microcomputer as a Timer and Data Logging Module", MICRO No. 3, pp 3-7 (Feb. - Mar., 1978)
System for logging the time of up to 75 events to the nearest 100 microseconds or to other time increments, and later displaying these times on the KIM-1 display.

281. Carpenter, C.R., "Machine Language used in 'Ludwig von Apple II'", MICRO, No. 3, p 8 (Feb. - Mar. 1978)
Notes on an assembled version of the machine language used by Schwartz, MICRO, No. 2, p 19 in his music program.

282. Carpenter, C.R., "Printing with the Apple II", MICRO, No. 3, pp13-16, (Feb-Mar, 1978)
Hard-copy output from the Apple II using a TELPAR thermal printer, a simple one-transistor adapter circuit and a machine language printing routine.

283. Foreman, Evan H., P.O. Drawer F, Mobile, AL 36601, "The PET Shop", MICRO, No. 3, p 10 (Feb. - Mar. 1978)
Foreman offers to trade five game programs for the PET on a one-for-one basis.

284. Floto, Charles, "The PET VET Tackles Data Files", MICRO, No. 3, pp 9-10, (Feb. - Mar. 1978)
Discusses problems some have encountered in recording data files on tape and reading the information back in. Floto, in his capacity as the PET VET, offers his services on problems met with specific applications of PET.

285. Tater, Gary L., "Hold That Data", MICRO, No. 3, p 11 (Feb. - Mar. 1978)
Program to stop data on the video terminal by pressing a key. Handy for examining data during a disassembly or a long directory program.

286. Tripp, Robert M., "Typesetting on a 6502 System", MICRO, No. 3, pp 19-24, (Feb.-Mar. 1978)
A program for "justification" of copy to be printed.

287. Tater, Gary L., "TIM Meets the S100 Bus", MICRO, No. 3, pp 25-26 (Feb.-Mar. 1978)
A bare-bones TIM S100 board to use with a terminal such as the CT-64 from SWTP.

288. Holt, Rod, "The Apple II Power Supply Revisited", MICRO, No. 3, p 28 (Feb.-Mar. 1978)
It is pointed out that the Apple II power supply, although small in physical size, is a switching type which runs cool and is sufficient to run an Apple II with several extra cards plugged into the system.

289. Anon, "Microbes-Tiny Bugs in Previous Micros", MICRO, No. 3, p 28 (Feb-Mar)
Corrections for Ultratape, MICRO No. 1, p 13; Making Music with the KIM, MICRO No. 2, p 7; and Important Addresses of KIM-1, MICRO No. 2, p 30.

290. Husbands, Charles R., "A Simple Frequency Counter Using the KIM-1", MICRO No. 3, pp 29-32 (Feb.-Mar. 1978)
The use of KIM-1 as a counter operating over the range of 500 Hz to above 15KHz.

291. Dial, William, "6502 Bibliography-Part II", MICRO, No. 3, pp 33-36 (Feb-Mar)
The second segment of this bibliography covers references 129 to 179 of the rapidly growing 6502 literature.

292. DeJong, Marvin L., "Lighting the KIM-1 Display", MICRO, No. 3, Back Cover.
Information on how to use the KIM-1 seven-segment display.

293. Anon, "Software Sources: 6502 Executive for KIM-1", Popular Electronics 13 No. 3, p 98 (March 1978)
Adaptable to any 6502 system, this Executive is designed for KIM-1 with 4K or more and TTY or TVT interface. $25 for listing. From Innovative Software, Inc., 3007 Casa Bonita Dr., Bonita, CA 92002.

294. Pollock, James W., "Microprocessors: A Microprocessor controlled CW Keyboard" Ham Radio 11, No. 1, pp 81-87 (Jan. 1978)
A preprogrammed microcomputer is designed to function as a Morse Code keyboard. Uses a MOS Technology MCS6504 which is a software compatible cousin to the 6502.

295. Connecticut Microcomputer, 150 Pocono Rd., Brookfield, CT 06804, New Product Announcement, "RS-232 Adapter for KIM", DDJ 3, No. 21, p 3 (Jan '78)
The ADApter converts KIM's 20 ma. current loop port to an RS-232 port without affecting the baud rate. $24.50

296. Schick, Paul, "Unsupported OPCODE Pitfalls", DDJ 3, No. 21, p 3 (Jan 1978)
Comments on the earlier article on 650X Opcodes: DDJ, Aug 1977.

297. Moser, Carl, "Memory Test for 6502", DDJ 3, No. 21, pp 4-5, (Jan 1978)
A program which tests RAM memory in a 6502 based system. I/O is arranged for 6502 TIM based system but can be easily changed.

298. Smith, Stephen P., "Challenging Challenger's ROMS", DDJ 3, No. 21, p 6 (Jan)
Using the PREK function of the OSI Microsoft BASIC, a disassembler to convert stored bytes in the PROMs or ROMs has been devised.

299. Computers One, PO Box 7148, Honolulu, HI 96821, New Product Announcement, On Line 3, No. 2, p 4 (March 1, 1978)
Pre-recorded programs for PET. "HUSTLERS" includes a number of business oriented programs for checking accounts, rent accounts, legal dairy and trust accounts.

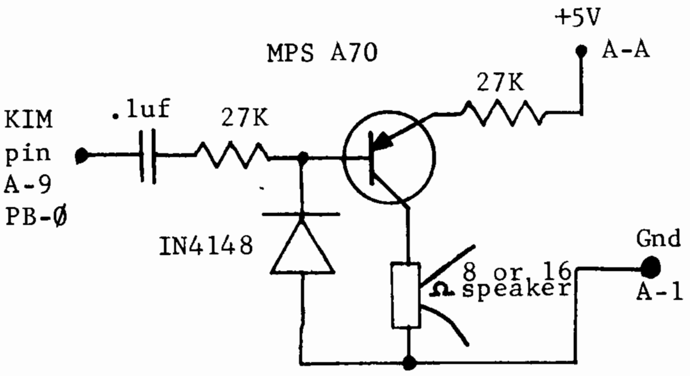
300. Lufkin, C.R., 315 Dominion Dr., Newport News, VA 23602, On-Line 3, No 2, p 5 (March 1, 1978)
FITABP is Federal Income Tax Program for PET owners with 8K. Prints out form 1040 Schedule A and B.

A KIM BEEPER

Gerald C. Jenkins
774 Twin Branch Drive
Birmingham, AL 35226

A short blast or two of audio for load errors, end-of-line, etc., is nice to have. This routine requires a simple audio amplifier such as the one in the KIM-1 User Manual, page 57, or the one shown below. Also needed is a latched output port, again such as those on the KIM-1, and a programmable timer.

Enter the routine with the number of blasts in the X register. Change the tone to suit by changing contents of NOTE, $0114.



```
0100 A9 FF      BEEP    LDAIM TIME    START TIMER FOR 1/4 SECOND TONE
0102 8D 07 17           STA   TIMER   USING INTERVAL TIMER
0105 A9 01              LDAIM $01     SET OUTPUT TONE OFF
0107 8D 02 17           STA   PBD
010A 8D 03 17           STA   PBDD

010D 4D 02 17   TONE    EOR   PBD     TOGGLE OUTPUT
0110 8D 02 17           STA   PBD
0113 A0 C8              LDYIM NOTE    SET TO COUNT FOR NOTE LENGTH
0115 88         TONEX   DEY           $C8 = 500 HZ
0116 D0 FD              BNE   TONEX   CYCLE IN DOWN COUNTER
0118 24 FF              BIT   TIME    TEST 1/4 SECOND UP
011A 10 F1              BPL   TONE    CONTINUE TONE IF NOT DONE
011C A9 01              LDAIM $01     TURN TONE OFF
011E 8D 02 17           STA   PBD
0121 A9 FF              LDAIM TIME    START WAIT BETWEEN BEEPS
0123 8D 07 17           STA   TIMER
0126 2C 07 17   NOTONE BIT    TIMER   WAIT FOR TIME OUT
0129 10 FB              BPL   NOTONE
012B CA                 DEX           DECREMENT NUMBER OF BEEPS COUNTER
012C D0 D2              BNE   BEEP    ANOTHER BEEP OR
012E 60                 RTS           DONE. RETURN TO CALLING ROUTINE
```

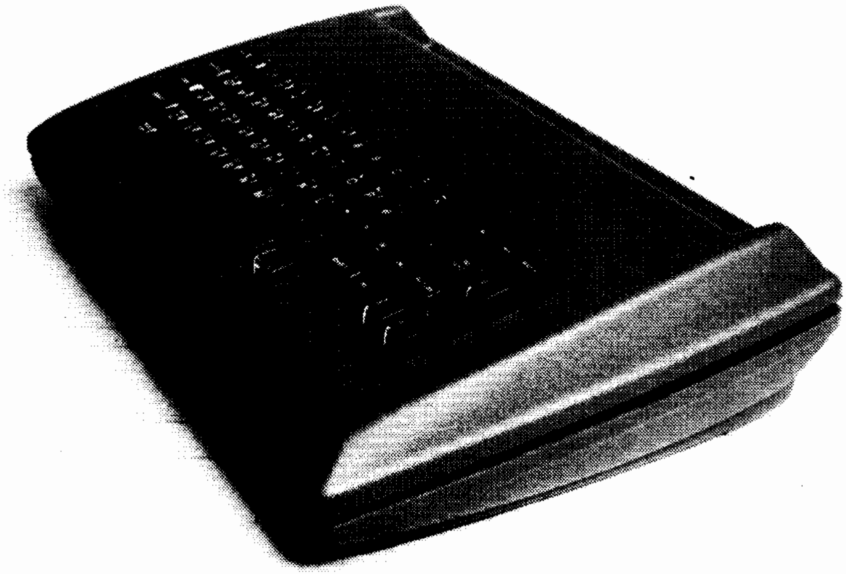
A Few Notes:

1. Although the above version is assembled at $0100, it is relocatable and can be placed anywhere in memory.

2. The calling sequence for BEEPER is:

```
put number of beeps into the X register
JSR   BEEPER
on return A = $FF, X = $00, and Y = $00
```

KEYBOARD WIZARDRY



ENGINEERED SPECIFICALLY FOR THE CHERRY-PRO KEYBOARD
- Space Provided for Power Supply and Additional Boards
- Easy Access to Connectors
- Keyboard Positioned for Ease of Operation

EASILY ASSEMBLED
- Requires Absolutely No Alteration of the PRO Keyboard
- All Fasteners Provided
- Goes Together in Minutes with a Small Screwdriver

ATTRACTIVE FUNCTIONAL PACKAGE
- Professional Appearance
- Four Color Combinations
- Improves Man/Machine Interface

MADE OF HIGH IMPACT STRENGTH THERMOFORMED PLASTIC
- Kydex 100*
- Durable
- Molded-In Color
- Non-Conductive

AVAILABLE FROM STOCK
- Allow Two to Three 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 _____ SKB 1-1(s)
@ $33.75 Each
California Residents please pay
$35.94 (Includes Sales Tax)

the enclosures group

55 stevenson, san francisco 94105

Color Desired blue ☐ beige ☐
black ☐ red ☐

Patent Applied For

* TM Rohm & Haas

AN APPLE-II PROGRAMMER'S GUIDE

[You Can Get There From Here!]

Rick Auricchio
59 Plymouth Avenue
Maplewood, NJ 07040

Most of the power of the APPLE-II comes in a "secret" form - almost undocumented software. After several months of coding, experimenting, digging, and writing to APPLE, most of the APPLE's pertinent software details have come to light.

Although most of the ROM software has been printed in the APPLE Reference Manual, its Integer Basic has not been listed; as a result, this article will be limited to Monitor software. Perhaps when a source listing of Integer Basic becomes available, we'll be able to interface with some of its many routines.

First Things First

When I took delivery of my Apple (July 1977), all I had was a "preliminary" manual - no goodies like listings or programming examples. My first letter to Apple brought a listing of the Monitor. Seeing what appeared to be a big jumble of instructions, I set out dividing the listing into logical routines while deciphering their input and output parameters. Once this was done, I could look at portions of the code without becoming dizzy.

The Monitor's code suffers from a few ills:

1 Subroutines lack a descriptive "preamble" stating function, calling sequences, and interface details.

2 Many subroutines have several entry points, each of which does something slightly different.

3 Useful routines are not documented in a concise form for user access.

I will concede that, while using a "shoehorn" to squeeze as much function as possible into those tiny ROM's, some shortcuts are to be expected. However, those valuable Comment Cards don't use up any memory space in the finished product - 'nuff said.

The Good Stuff

The best way to present the Apple's software interface details is to describe them in tabular form, with further explanation about the more complex ones. The following tables will be found on the back cover of this issue:

Table 1 outlines the important data areas used by the Monitor. These fields are used both internally by the Monitor, and in user communication with many Monitor routines. Not all of the data fields are listed in Table 1.

Table 2 gives a quick description of most of the useful Monitor routines: it contains Name, Location, Function, Input/Output parameters, and Volatile (clobbered) Registers.

Don't hesitate to experiment with these routines - since all the important software is in ROM, you can't clobber anything by trying them out (except what you might have in RAM, so beware).

Using the "User Exits"

The Monitor provides a few nice User Exits for us to get our hands into the Monitor. With these, it is a simple matter to "hook in" special I/O and command-processing routines to extend the Apple's capabilities.

Two of the most useful exits are the KEYIN and COUT exits. These routines, central to the function of the Monitor, are called to read the keyboard and output characters to the screen. By placing the address of a user routine in CSWH/L or KSWH/L, we will get control from the Monitor whenever it attempts to read the keys or output to the screen.

As an example of this exit's action, try this: with no I/O board in I/O Slot 5, key-in "Kc5" (control K, followed by 5, then Return). You'll have to hit Reset to stop the system.

Here's what happened: setting the keyboard to device 5 causes the Monitor to install $C500 as the "user-exit" address in KSWH/L. This, of course, is the address assigned to I/O Slot 5. Since no board is present, a BRK opcode eventually occurs; the Monitor prints the break and the registers, then reads for another command. Since we still exit to $C500, the process repeats itself endlessly. Reset removes both user exits; you must "re-hook" them after every Reset.

These two exits can enable user editing of keyboard input, printer driver programs, and many other ideas. Their use is limited to your ingenuity.

Another useful exit is the Control Y command exit. Upon recognition of Control Y, the Monitor issues a JSR to location $03F8. Here the user can process commands by scanning the original typed line or reading another. This exit is often very useful as a shorthand method of running a program. For example, when you're going back and forth between the Monitor and the Mini-Assembler, typing "F666G" is a bit tiresome. By placing a JMP $F666 in location $03F8, you can enter the Mini-Assembler via a simple Control Y.

Upon being entered from the Monitor at $03F8, the registers are garbage. Locations A1 and A2 contain converted values from the command (if any), and an RTS gets you neatly back into the Monitor. Figure 1 shows this in more detail.

Figure 1: Control Y Interface

Command typed:

```
*1234.F5A7Yc
```

Upon entry at $03F8,
the following exists:

A1L ($3C) contains $34
A1H ($3D) contains $12
A2L ($3E) contains $A7
A2H ($3F) contains $F5

Hardware Features

One of the best hardware facilities of the Apple-II, the screen display, is also the "darkest" - somewhat unknown. Here's what I've found out about it.

The screen buffer resides in memory pages 4 through 7, locations $0400 through about $07F8. The Secondary screen page, although not accessed by the Monitor, occupies locations $0800 through $0BF8. Screen lines are not in sequential memory order; rather, they are addressed by a somewhat complex calculation carried out in the routine BASCALC. What BASCALC does is to compute the base address for a particular line and save it; whenever the cursor's vertical position changes, BASCALC recomputes the base address. Characters are stored into the screen buffer by adding the base address to the cursor's horizontal position.

I haven't made too much use of directly storing characters into the screen buffer; usually just storing new cursor coordinates will do the trick via the Monitor routines. Be careful, though - only change vertical position via the VTAB routine since the base address must get recomputed!

Characters themselves are internally stored in 6-bit format in the screen buffer. Bit 7 ($80), when set, forces normal (white-on-black) video display for the character. If Bit 7 is reset, the character appears inverse (black-on-white) video. Bit 6 ($40), when set, enables blinking for the character; this occurs only if Bit 7 is off. Thus an ASCII "A" in normal mode is $81; in inverse mode, $01; in blinking mode, $41.

Reading the keyboard via location $C000 is easy; if Bit 7 ($80) is set, a key has been pressed. Bits 0 - 6 are the ASCII keycode. In order to enable the keyboard again, its strobe must be cleared by accessing location $C010. Since the keyboard is directly accessible, there is no reason you can't do "special" things in a user program based on some keyboard input - if you get keys directly from the keyboard, you can bypass ALL of the Control and Escape functions.

AN APPLE II PROGRAMMER'S GUIDE

Rick Auricchio
59 Plymouth Avenue
Maplewood, NJ 07040

MONITOR Data Areas in Page Zero

Name	Loc.	Function
WNDLEFT	20	Scrolling window: left side (0-$27)
WNDWDTH	21	Scrolling window: width (1-$28)
WNDTOP	22	Scrolling window: top line (0-$16)
WNDBTM	23	Scrolling window: bottom line (1-$17)
CH	24	Cursor: horizontal position (0-$27)
CV	25	Cursor: vertical position (0-$17)
COLOR	30	Current COLOR for PLOT/HLIN/VLIN functions
INVFLG	32	Video Format Control Mask: $FF=Normal, $7F=Blinking, $3F=Inverse
PROMPT	33	Prompt character: printed on GETLN CALL
CSWL	36	Low PC for user exit on COUT routine
CSWH	37	High PC for user exit on COUT routine
KSWL	38	Low PC for user exit on KEYIN routine
KSWH	39	High PC for user exit on KEYIN routine
PCL	3A	Low User PC saved here on BRK to Monitor
PCH	3B	High User PC saved here on BRK to Monitor
A1L	3C	A1 to A5 are pairs of Monitor work bytes
A1H	3D	
A2L	3E	
A2H	3F	
A3L	40	
A3H	41	
A4L	42	
A4H	43	
A5L	44	
A5H	45	
ACC	45	User AC saved here on BRK to Monitor
XREG	46	User X saved here on BRK to Monitor
YREG	47	User Y saved here on BRK to Monitor
STATUS	48	User P status saved here on BRK to Monitor
SPNT	49	User Stack Pointer saved here on BRK

Page 2 ($0200-$02FF) is used as the KEYIN Buffer.

Pages 4-7 ($0400-$07FF) are used as the Screen Buffer.
Page 8 ($0800-$08FF) is the "secondary" Screen Buffer.

Table 1.

MONITOR ROUTINES

Name	Loc.	Steps On	Function
PLOT	F800	AC	Plot a point. COLOR contains color in both halves of byte ($00-$FF). AC: y-coord, Y: x-coord.
CLRSCR	F832	AC,Y	Clear screen - graphics mode.
SCRN	F871	AC	Get screen color. AC: y-coord, Y: x-coord.
INSTDSP	F8D0	ALL	Disassemble instruction at PCH/PCL.
PRNTYX	F940	AC	Print contents of Y and X as 4 hex digits.
PRBL2	F94C	AC,X	Print blanks: X is number to print.
PREAD	FB1E	AC,Y	Read paddle. X: paddle number 0-3.
SETTXT	FB39	AC	Set TEXT mode.
SETGR	FB40	AC	Set GRAPHIC mode (GR).
VTAB	FC22	AC	VTAB to row in AC (0-$17).
CLREOP	FC42	AC,Y	Clear to end-of-page.
HOME	FC58	AC,Y	Home cursor and clear screen.
SCROLL	FC70	AC,Y	Scroll up one line.
CLREOL	FC9C	AC,Y	Clear to end-of-line.
NXTA4	FCB4	AC	Increment A4 (16 bits), then do NXTA1.
NXTA1	FCBA	AC	Increment A1 (16 bits). Set carry if result >= A2.
RDKEY	FD0C	AC,Y	Get a key from the keyboard.
RDCHAR	FD35	AC,Y	Get a key, also handles ESCAPE functions.
GETLN	FD6A	ALL	Get a line of text from the keyboard, up to the carriage return. Normal mode for Monitor. X returned with number of characters typed in.
CROUT	FD8E	AC,Y	Print a carriage return.
PRBYTE	FDDA	AC	Print contents of AC as 2 hex digits.
COUT	FDED	AC,Y	Print character in AC; also works for CR, BS, etc.
PRERR	FF2D	AC,Y	Print "ERR" and bell.
BELL	FF3A	AC,Y	Print bell.
RESET	FF59	--	RESET entry to Monitor - initialize.
MON	FF65	--	Normal entry to 'top' of Monitor when running.
SWEET16	F689	None	SWEET16 is a 16-bit machine language interpreter. [See: SWEET16: The 6502 Dream Machine, Steve Wozniak,] [BYTE, Vol. 2, No. 11, November 1977, pages 150-159.]

Table 2.