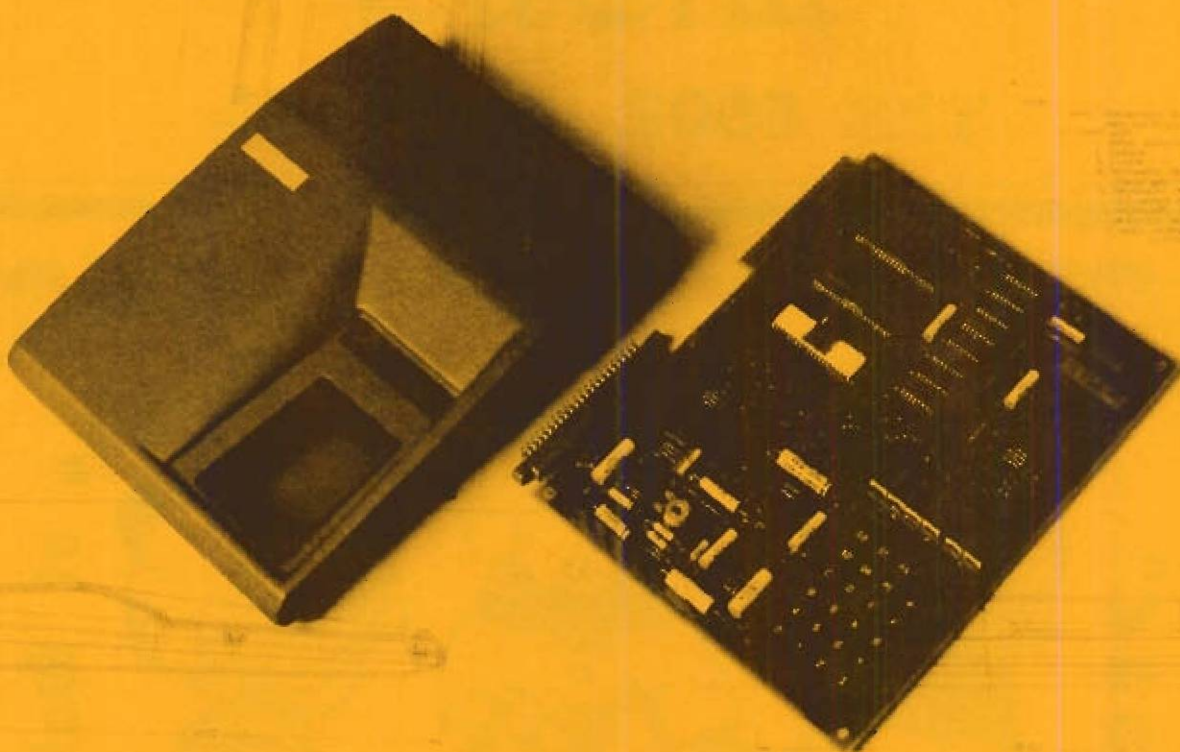
MICRO™

THE 6502 JOURNAL

NO 2 Dec 77-Jan 78 $1.50

QUICK CHANGE ARTISTRY



ENGINEERED SPECIFICALLY FOR THE KIM-1 MICRO COMPUTER
- Protection of Chips and Other Components
- Viewing Angle of Readout Enhanced
- Improved Keyboard Position for Easier Operation

EASILY ASSEMBLED
- Absolutely No Alteration of KIM-1 Required
- 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 _____ SKE 1-1(s)
@ $23.50 Each
California Residents please pay
$25.03 (Includes Sales Tax)

the enclosures group

55 stevenson, san francisco 94105

Color Desired blue ☐ beige ☐ black ☐ white ☐

* TM Rohm & Haas

Patent Applied For

MICRO

DECEMBER 1977/JANUARY 1978

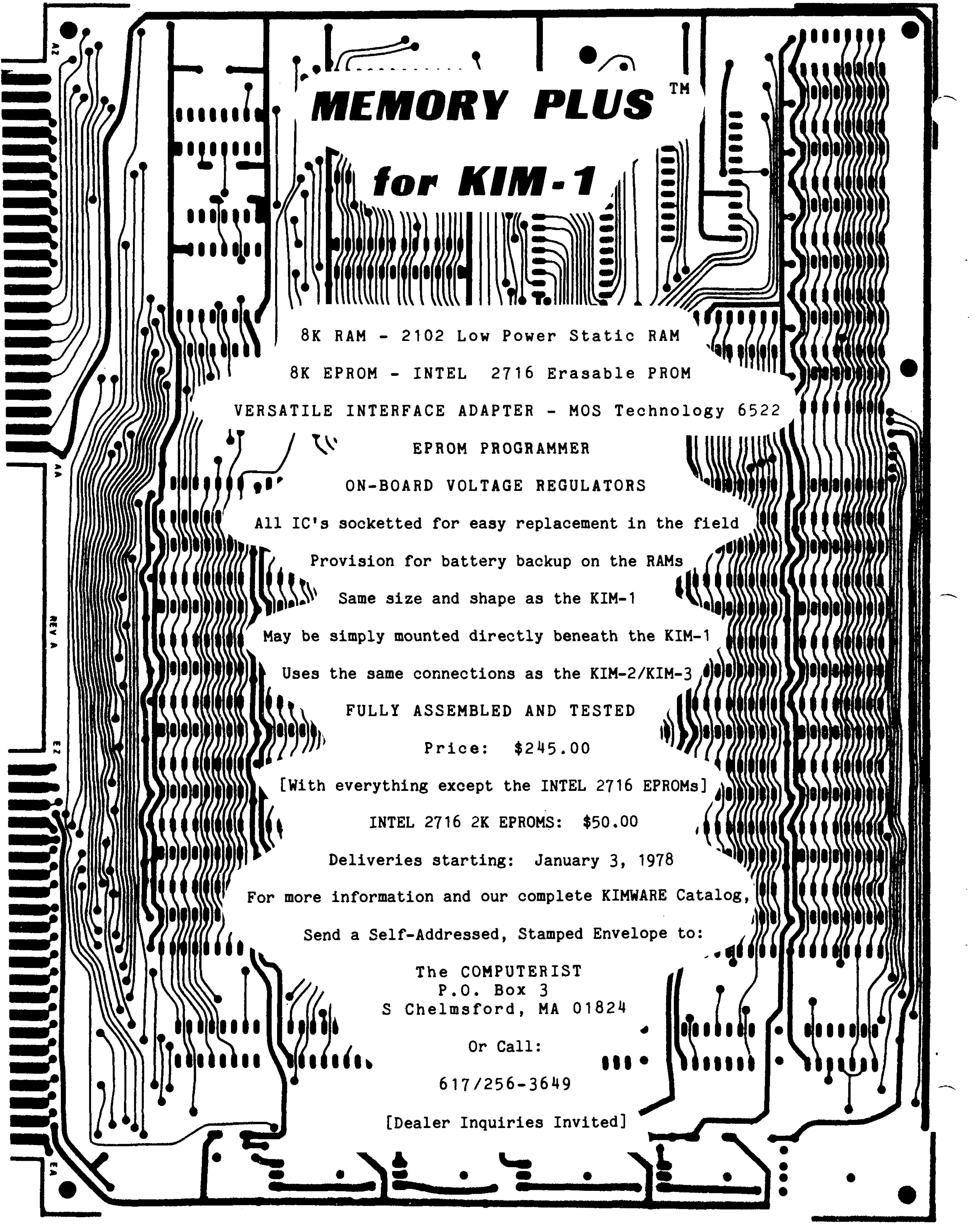
ISSUE NUMBER TWO

Advertisers Index

Subscription Rate: $6.00 per year (six issues) in U.S.A.

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

Deadline for February/March 1978 issue: January 15, 1978.


MEMORY PLUS™
for KIM-1
8K RAM - 2102 Low Power Static RAM
8K EPROM - INTEL 2716 Erasable PROM
VERSATILE INTERFACE ADAPTER - MOS Technology 6522
EPROM PROGRAMMER
ON-BOARD VOLTAGE REGULATORS
All IC's socketted for easy replacement in the field
Provision for battery backup on the RAMs
Same size and shape as the KIM-1
May be simply mounted directly beneath the KIM-1
Uses the same connections as the KIM-2/KIM-3
FULLY ASSEMBLED AND TESTED
Price: $245.00
[With everything except the INTEL 2716 EPROMs]
INTEL 2716 2K EPROMS: $50.00
Deliveries starting: January 3, 1978
For more information and our complete KIMWARE Catalog,
Send a Self-Addressed, Stamped Envelope to:
The COMPUTERIST
P.O. Box 3
S Chelmsford, MA 01824
Or Call:
617/256-3649
[Dealer Inquiries Invited]

Making Music with the KIM-1

Armand L. Camus
P.O. Box 294
Westford, MA 01886

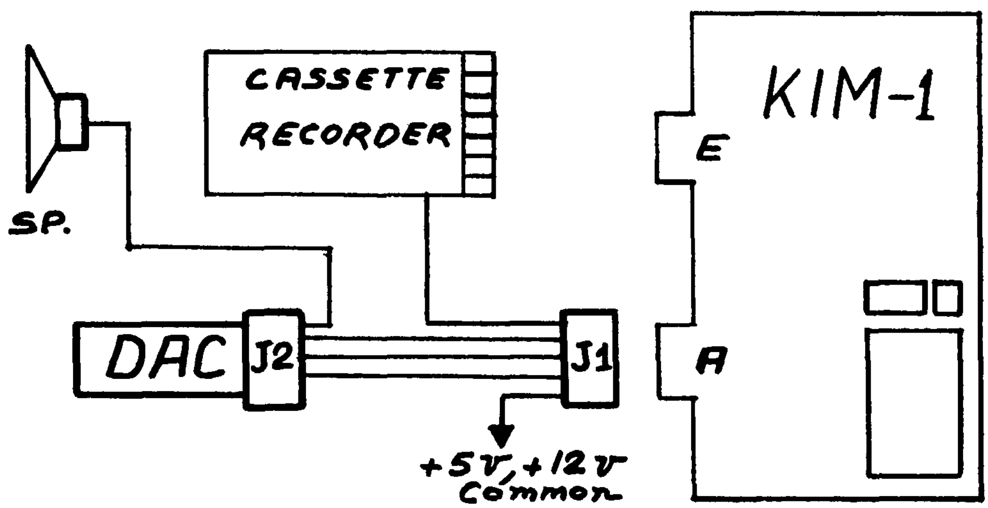
What kind of music can you make with the help of a microcomputer, namely the KIM-1 with its 1.1K bytes of memory? Well, it certainly will not sound like the Boston Symphony Orchestra, live or on records, but with the right type of music it will give an acceptable rendition of a chosen piece of music. Many elements of good music will be missing, especially the timbre of the different instruments of the orchestra, but on the positive side the notes will be on tune, you will be able to compose in four-part harmony, the tempo will be adjustable, and the whole process will permit some of the artistic creativity which may hide in each of us to emerge to the surface. Last, but not least, it will be a lot of fun.

This elementary article explains the "HOW-TO" rather than the "WHY" in making music with a microcomputer. Many of the hobbyists who may find it too simple may refer to the excellent article by Hal Chamberlin which dwells in detail on the subject.

An easy way for the beginner to start his musical career is to acquire a minimum of equipment besides the KIM-1 and cassette recorder it is assumed are already in his possession.

The DAC unit is a printed circuit board containing a complete audio output system for the KIM-1. This board also comes with a cassette tape, an instruction sheet listing the songs which can be loaded in the KIM, and a reprint of the reference article including the interconnections to be made between the two connectors.

Now that we have described the hardware we will concentrate on what to do in order to get some music out of the system. The simplest way at this time is to load File 1 and File 2 of the tape and to see if the Star Spangled Banner comes out clear and patriotic. The procedure is simple:



J1, J2 connectors: Vector R644, Winchester HKD2250, or equivalent. J2 will be too long, but will work just the same.

Speaker, 2½", 8 ohm, 0.3W, from Radio Shack, or equivalent.

DAC Digital-to-analog converter from THE COMPUTERIST, P.O. Box 3, S. Chelmsford, MA 01824 or MTU, P.O. Box 4596, Manchester, NH 03103.

Start the KIM-1 and press the appropriate keys to get:

```
AD  00F1  DA  00
AD  17F9  DA  01
AD  1873  Press  GO
```

Start the cassette until you get 0000 in the address display, which indicates that the loading was done properly. After stopping the cassette, press the keys to get:

```
AD  17F9  DA  02
AD  1873  Press  GO
```

Start the cassette again until you get 0000. Stop the cassette. Now you are ready. Press AD 0100, press GO and the song will be played. As it stops, the program resets the address AD to 0100, so by pressing GO again, the song will repeat itself.

In the same manner you could load Files 3 and 4 to get a rendition of Exodus. The sound quality may be changed by loading File 5 or File 6. Personally, I prefer File 6 which has a much more mellow timbre.

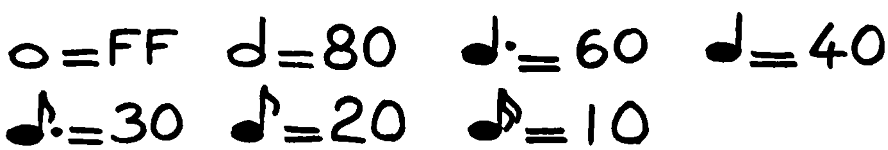
Transcribing A Song

Now that we have gone through the above steps, we will learn to code a song. For our purpose, a particular note of music will have two characteristic elements:

its pitch, represented by its position on the staff
its duration, relative to other notes.

What we mean is that a half note lasts twice as long as a quarter note, a quarter note lasts twice as long as an eighth note, etc. . . We are not talking about tempo yet, this will come later.

1. Duration Code:
   We will assign a two-digit code to the duration of a note:



2. Pitch Code:

With the help of this lookup table we can find quickly the code for any note within the limits of C6 and C2, the high and low C's. However, the very low notes may not be reproduced too well with a small speaker and it may not be advisable to go below C3 (Code 1A).

3. Coding A Song:

The program given at the end of this article is a coding of the well-known carol "Deck The Halls", which we thought would be appropriate for this issue. [Editor's Note: It inspired this issue's cover!] If you look at this coding, you will observe that it is done line by line. Each line is composed of six elements. For example, the first line is:

```
0200  60 4A 44 32 24
```

- the 0200 is the memory address of the element 60. The next element, 4A, would then have an address 0201, and so on.
- the 60 is the duration of the group of four notes which follow. A 60 means a dotted quarter note.
- the 4A is the note C, for the first voice.
- the 44 is the note A, for the second voice.
- the 32 is the note C, for the third voice.
- the 24 is the note F, for the fourth voice.

This is an F major chord which could be represented as in (1), and it corresponds to the word "DECK" of the song.

Now we will code the first bar of the song. Remember that each line will have the same format:

> address (4 digits), duration (2 digits), 1st voice (2 digits), 2nd voice (2 digits), 3rd voice (2 digits), and 4th voice (2 digits) for a total of fourteen (14) digits. If a voice is quiet, use 00 at the appropriate location.

The first vertical group of notes (C,A,C,F) corresponding to the word "DECK" has already been explained above.

The second vertical group of notes corresponding to the word "THE" is made of B flat, G, C, and E. Looking up the pitch code table, we find the following codes:

> Bb = 46, G = 40, C = 32, and E = 22. Each note is an eighth note so the duration code is 20. The address of the duration code is 0205 so our second line will be:

```
0205  20 46 40 32 22
```

In the same fashion the two other vertical groups are made of quarter notes (code 40) and we get for the first bar:

```
0200  60 4A 44 32 24 (DECK)
0205  20 46 40 32 22 (THE)
020A  40 44 3C 32 24 (HALLS)
020F  40 40 3A 2E 1A (WITH)
```

Remember that there is a Key Signature in this carol and that all the B's, wherever located on the staff, are flat, unless otherwise indicated, which explains the 46 of the second line and the 2E of the fourth line.

Another part of that song is shown in the example (3). The first voice plays two notes (A and B natural), while the other voices play only one. We solve this problem by writing two lines, one for the A and one for the B natural, repeating the other notes to extend their duration to a quarter note. We get:

```
02D2  20 44 3C 32 24
02D7  20 48 3C 32 24
```

Both A note (code 44) and B natural note (code 48) have only the duration of one eighth note each (code 20), and we have to write two separate lines for them, but the three other notes will be repeated so that their total duration is a quarter note. Fortunately, the lower notes, even when repeated, will blend together and sound more like a quarter note than two consecutive eighth notes.

Now we should be able to code a song, but as a preliminary exercise, you may want to load "Deck the Halls" and see how it works out. Here is the procedure:

Load Files 01 and 02 of the DAC tape, as explained at the beginning of this article. You may also want to load File 06 to give a more mellow timbre. Then go to address 0200 and start inputting the data. The addresses in the left side give you a check on your progress and catch possible omissions of data. What we are doing here is using the main program and writing over the song already in memory. At any time it is possible to go back to AD 0100, push GO and listen to what is already in memory. Somewhere at about 2/3 of the song, we run out of memory (0200 to 02F9), but we have enough left to tell our microcomputer that it is the end of that particular segment (02FA 01), and that we wish to continue at address 0083 (02FB and 02FC). At the very end, check address 00DD 00. The data 00 indicates the end of the piece and this will reset the KIM-1 to address 0100, ready to "GO", so to speak.

After you have loaded the code and pushed the GO key, the carol should start, sounding good if no mistake was made, but perhaps a little bit on the slow side. To change the tempo, either way, go to address 001D and the data will probably show 60. Change the data to 40, go back to address 0100, push GO and the tempo will be much faster. Experiment with the data at AD 001D and find the tempo you prefer.

I have found out that while I am coding I like to listen to what is already in memory, because a simple mistake at the beginning, especially forgetting one voice or the duration code, will throw the rest out of whack. Starting the song at the beginning, when it is already correct is a waste of time, but it is possible to start the song at some other point. However, it must always be at one of the duration addresses shown at the end of this article. If not, the KIM-1 would interpret the duration code as a musical note and vice-versa! The starting address is contained in locations 0017 and 0018. To start, for example, at address 0237, go to address 0017 read 00, 0018 read 02. This means that the song normally starts at 0200. All we have to do is change the data to read:

AD 0017 DA 37
AD 0018 DA 02

Then setting address 0100 and pushing GO will cause the song to start at location 0237 everytime.

Available Memory

The memory available to the user is divided in two groups, each group not necessarily in consecutive order. First group is associated with the music program, frequency table or the notes, KIM, etc... Second group is associated with the song. The actual layout of the memory is as follows:

0000 to 001E	Program variables
001F to 0082	Note frequency table
0083 to 00EE	Song, second part
00EF to 00FF	KIM variables
0100 to 01AA	Music program
01AB to 01F3	Song, third part
01F4 to 01FF	6502 Stack
0200 to 02FF	Song, first part
0300 to 03FF	Waveform (voice) table
1780 to 17E4	Song, fourth part

If your music score extends beyond the first part locations, you have to provide room for continuation. Assuming a score uses all of the available memory space for coding a song, the following locations are important:

Use of Location	Part 1	Part 2	Part 3	Part 4
Beginning of Part (Song)	0200	0083	01AB	1780
Beginning of Last Line	02F5	00E7	01EC	17DF
Last note of Last Line	02F9	00EB	01F0	17E3
End of Sequence (Song)	02FA (01)	00EC (01)	01F1 (01)	17E4 (00)
Low Address Next Segment	02FB (83)	00ED (AB)	01F2 (80)	
High Address Next Segment	02FC (00)	00EE (01)	01F3 (17)	

Reference: Chamberlin, Hal, "A Sampling of Techniques for Computer Performance of Music", BYTE Magazine, Sept. 1977, pp. 62-83.

Score for "Deck the Halls"

```
0200: 60 4A 40 3A 32
0205: 20 40 40 3A 22
020A: 40 44 3C 32 24
020F: 40 40 3A 2E 1A
0214: 40 3C 32 2C 1E
0219: 40 40 3A 32 1A
021E: 40 44 3C 32 24
0223: 40 3C 32 2C 24
0228: 20 40 3A 32 1A
022D: 20 44 3C 32 1A
0232: 20 46 40 32 1A
0237: 20 40 3A 32 1A
023C: 60 44 3C 32 24
0241: 20 40 36 2E 16
0246: 40 3C 32 2C 1A
024B: 40 3A 32 28 1A
0250: 80 3C 32 2C 24
0255: 60 62 5C 32 24
025A: 20 5E 58 32 22
025F: 40 5C 54 32 24
0264: 40 58 52 2E 1A
0269: 40 54 4E 2C 1E
026E: 40 58 52 32 1A
0273: 40 5C 54 32 24
0278: 40 54 4A 2C 24
027D: 20 58 52 32 1A
0282: 20 5C 54 32 1A
0287: 20 5E 58 32 1A
028C: 20 58 52 32 1A
0291: 60 5C 54 32 24
0296: 20 58 4E 2E 16
029B: 40 54 4A 2C 1A
02A0: 40 52 4A 28 1A
02A5: 80 54 4A 2C 24
02AA: 60 40 3A 32 1A
02AF: 20 44 3C 32 1A
02B4: 40 46 40 32 1A
02B9: 40 40 3A 32 1A
02BE: 60 44 3C 32 24
02C3: 20 46 3C 28 24
02C8: 40 4A 3C 2C 24
02CD: 40 40 40 32 22
02D2: 20 44 3C 32 24
02D7: 20 48 3C 32 24
02DC: 40 4A 40 32 22
02E1: 20 4E 44 32 24
02E6: 20 52 44 32 24
02EB: 40 54 44 32 1E
02F0: 40 52 40 32 28
02F5: 40 4E 3C 30 28
02FA: 01
02FB: 83
02FC: 00
```

```
0083: 80 4A 3A 28 1A
0088: 60 4A 44 32 24
008D: 20 46 40 32 24
0092: 40 44 3C 32 24
0097: 40 40 3A 2E 1A
009C: 40 3C 32 2C 1E
00A1: 40 40 3A 2E 1A
00A6: 40 44 3C 32 24
00AB: 40 3C 32 2C 24
00B0: 20 4E 3C 2E 16
00B5: 20 4E 3C 2E 16
00BA: 20 4E 3C 2E 16
00BF: 20 4E 3C 2E 16
00C4: 60 4A 3C 2C 24
00C9: 25 46 3C 36 28
00CE: 50 44 3C 32 1A
00D3: 60 40 3A 2E 1A
00D8: 95 3C 32 2C 24
00DD: 00
```


Writing for MICRO

MICRO is interested in all aspects of microcomputers based on the 6502 microprocessor family. Our primary coverage is aimed at factual, useful information. This may be "How To" articles, useful programs and subroutines, descriptions of working applications, special interest groups such as Hams, reviews of products and literature, technical tutorials, and so forth. Authors will receive a small honorium plus reprints of their article. Help spread the 6502 word.

Mixing Apples and Oranges

An Editorial

Often you have heard the injunction against mixing apples and oranges. MICRO proposes that we ignore the warning and mix APPLEs (I and II) with PETs, KIMs, Challengers, CGRS Microtechs, JOLTs, homebrews, and any other 6502 family based microcomputers. The hope is that mixing these devices together will help to show the similarities between them that are inherent by reason of their common microprocessor. This is the passive role of MICRO. The active role of MICRO is to promote commonality among these microcomputers. How about establishing a standard assembler syntax (since the MOS Technology syntax is so horrible)? How about defining some standard subroutines and subroutine calling sequences so that it will be easier to adapt programs from one machine to run on another? Would it be possible to develop some standard cassette tape format which would permit tapes generated by one microcomputer to be read into another? Wouldn't it be nice if the 6502 based BASICs, Disk Operating Systems, and other high level software was compatible across machines? Maybe it is only a dream, but maybe it isn't too late to seriously attempt to maximize the impact of the 6502 based systems by setting some standards so that the various 6502 systems can combine to strengthen our position in the micro world rather than weaken it by producing incompatible hardware and software. 6502 interests of the world unite! You have nothing to lose.

Meet the PET

Charles Floto
267 Willow Street
New Haven, CT 06511

This article is based on about a month's experience with PET model 2001-8 serial #0010081. Commodore indicates the only thing experimental about it is the color of the case, which is metal painted white.

In June I sent Commodore $595 for a PET with 4K of RAM to be delivered in late September. Toward the end of the latter month I was informed that initial production would be limited to the 8K version and that I could either send an additional $200 or get my money back. I sent the $200 and my PET was delivered October 25.

It made the trip from Palo Alto to New Haven well cushioned in a carton 21" x 23" x 19" high. Since being unpacked my PET's survived riding on a bus seat to Washington, D.C. and returning by car.

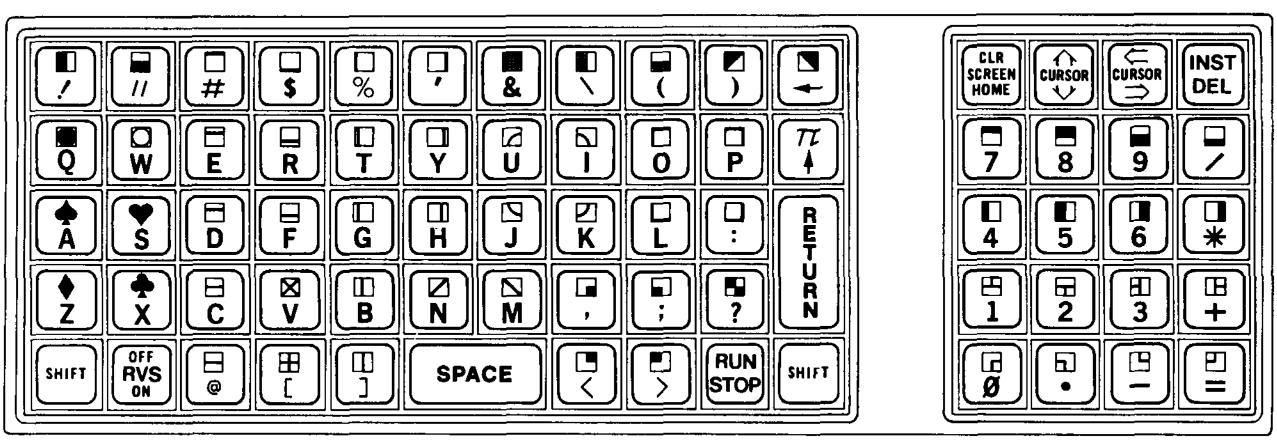
While the case has a maximum width of 17½" and a maximum depth of 19" the placement of the feet allows it to stand on anything at least a foot square. Maximum height is 15½" and the PET weighs about 37 pounds.

A glance at the PET reveals its distinguishing characteristic: everything's in one package -- including video display, keyboard, and tape drive. A standard 9" diagonal black and white TV tube is used. The display field measures about 4¾" high by 6" wide. This is divided into 25 lines of 40 columns. Each of these 1000 positions can be filled with one of about 300 different characters formed with an 8 x 8 matrix. Character width is about ¾ of character height. Characters available include those shown on the drawing of the keyboard. Any of these may also be displayed with black and white reversed on an individual basis. Furthermore, by changing the contents of memory location 59468 you can substitute lower case letters and four additional graphics for 30 of the graphics characters. Taking video inversion into account, this gives a total of 316 available characters.

The graphics characters have been chosen to be useful picture elements which give higher effective resolution than the 25 x 40 field would suggest. For example, a horizontal bar graph can be displayed with resolution better than 1/3 per cent of full scale.

Characters may be placed on the screen directly from the keyboard, by use of the PRINT command, or by a POKE into the screen memory. This extends from decimal address 32768 to 33767 (i.e., 32K to 32K + 999). If A is a number in this range and C is between 0 and 255 then POKE A, C will place a character on the screen.

The keyboard is the PET's most controversial feature. It takes up an area 9" by 2¾". Since I'm used to typing with only two fingers and a thumb the small size and rectangular layout don't bother me. Each keytop is about 3/8" square and can be depressed about 1/8". The feel is similar to a standard spring-loaded switch-



closure keyboard. The only trouble I've had with it is that occasionally one of the cursor control keys will insert a character rather than move the cursor. I've been able to rectify this by getting off the line and then coming back to it. Preliminary investigation suggests it should be quite easy to attach a standard keyboard to the PET.

The other prominent external feature of the PET is the tape drive which takes up an area just over 5" square. It has the useful function controls, but the motor is turned on and off under software control in the play and record modes. Prompts for operation of the controls are displayed on the screen as appropriate. The screen also displays status of the tape operation in progress. When attempting to load or verify a file with a specified name it displays the names of other files found. File names may have a maximum of 79 characters.

Short BASIC programs can be loaded from tape at an effective speed of 250-400 baud, with longer programs having a higher net rate. It should be possible to fill the entire 8K of RAM in under 2½ minutes -- once the proper program has been found.

Sticking out of the lower right side of the PET cabinet is a 40 line (plus 40 ground) printed circuit connection to the internal bus. As it does stick out about 1/8" metallic objects should be kept away from this area. The three groups of PC connectors in the rear present a lesser hazard as they're flush with the cabinet. These are: 1. A connector for the IEEE-488 bus (see MICRO No. One, page 11 for my discussion of this); 2. A parallel port with handshaking; 3. The interface for tape drive #2. I was able to verify this one by removing drive #1 from the cabinet and plugging it into the rear as #2. Commodore has exhibited the PET with a second drive connected, but hasn't offered to sell any yet.

So much for physical externals. How good is the firmware stored in that 14K of ROM? Since my instruction book hasn't arrived yet I'll limit my comments to the 8K BASIC interpreter described in the nine-page "temporary version" of the documentation.

My favorite feature of PET BASIC is the ease of editing a line within a program. Just move the cursor to the appropriate spot, make the change, and hit RETURN -- no need to retype the line. It's also handy to be able to turn on the machine, load a partially-written program from tape, work on it for a while, then save the new version on cassette, I'm glad I got the version with 8K of RAM as I've already written a program that leaves fewer than 5K bytes free. (The 4K model is said to use a different circuit card, but since there aren't any yet. . .)

I also appreciate the special variables TI and TI$. TI is incremented 60 times a second; it makes delays and timing applications easy. TI$ is a 24-hour clock whose 6 digits indicate hours, minutes, and seconds. As these suggest, variable names may be two letters -- as long as they're not reserved words such as OR, IF and ON. Variables may be integers, strings, real, or multi-dimensioned arrays of any one of these. Integers are limited to the range ±32767. Real variables are calculated with 10-digit precision, although only 9 digits are printed. For example, pi=3.14159265; twice pi=6.28318531.

Another distinctive feature is the GET statement which reads the keyboard without RETURN having been pressed. Unfortunately the random number function only works with positive arguments. RND(0)=.564705882 always, while RND(-1)=2.99196472E-08. The latter is typical of the values returned with negative integer arguments. This is the only bug I've discovered in PET BASIC.

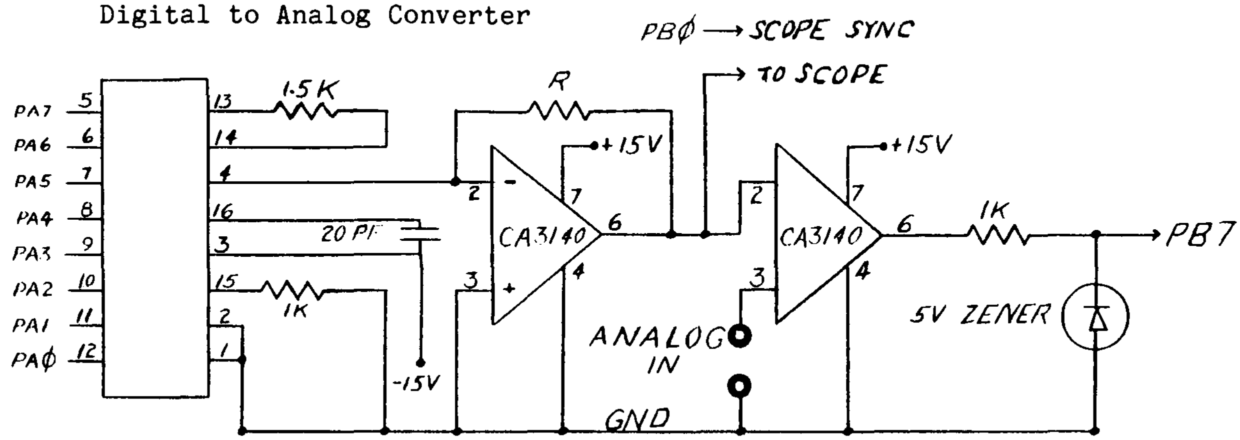
Editor's Notes: In MICRO #1, Charles Floto discussed the PET's IEEE-488 Bus. Since then Motorola and Intel have both announced new ICs that will make it easy to interface to this otherwise formidable bus structure. For more information on the PET and a comparison with the Radio Shack TRS-80, see "The PET Vs. the TRS-80" by Bob Wallace, MICRO #2, page 17.

Digital-Analog and Analog-Digital Conversion Using the KIM-1

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

A Motorola 1408 8-bit digital to analog converter is connected as shown in the circuit diagram. (The 1408 is available from James Electronics, 1021 Howard Ave., San Carlos, CA 94070, as are the op amps used in these experiments.) The PAD port of the KIM is used to provide the digital input to the 1408. The analog output of the 1408 is a current sink at pin 4, which we converted to a voltage by means of the RCA CA3140 operational amplifier. The feedback resistor R is adjusted to give the desired voltage output. For example, an R of about 500 ohms gives a voltage range from 0 volts when PAD is 00000000 to 1 volt when PAD is 11111111.

Circuit Diagram for
Digital to Analog Converter



1. Generation of a Ramp Voltage Waveform

For the first experiment do not connect the second op amp, simply connect the output of the first op amp to an oscilloscope as shown. Load the following program.

Program to Generate a Ramp Voltage Waveform

ADDRESS	OPCODE	LABEL	INSTRUCTION		COMMENTS
0300	A9 FF	START	LDAIM	FF	255 in Accumulator
0302	8D 01 17		STA	PADD	Port A is Output Port
0305	EE 00 17	BACK	INC	PAD	Increment number in PAD
0308	4C 04 03		JMP	BACK	Increment in a Loop

Running this program should cause a ramp waveform to be observed on the oscilloscope screen. A close examination of the ramp will show that it consists of $2^8 = 256$ steps rather than a straight line.

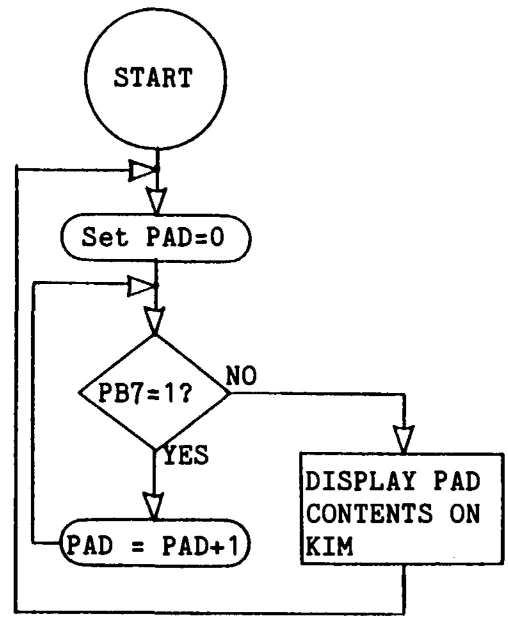
2. A DAC as an Analog to Digital Converter

The second op amp acts as a comparator. It compares the voltage from the output of the first op amp (which we shall call the digital signal) with a voltage from some source to be applied to pin 3 (which we shall call the analog signal). The output is connected to PB7 on the KIM. If PB7 = 1, the analog signal is greater than the digital signal. If PB7 = 0, the analog signal is less than the digital signal. The digital signal is, of course, produced by the contents of PAD.

A flow chart showing what we intend to do is shown below. Output port PAD is set to zero. If the analog signal is positive the PB7 = 1. PAD is now incremented until the comparator indicates that the analog signal is less than the digital signal, i.e. PB7 = 0. At that instant the digital and analog signals are the same to within one bit, the least significant bit, on PAD. The digital value of PAD is then displayed and the cycle continues.

If the feedback resistor is adjusted so that a value of PAD = 255_{10} = FF_{16} produces a voltage of 2.55 volts, then we have constructed a simple digital voltmeter with a full scale reading (in hex) of 2.55 volts. The extremely high impedance of the 3140 op amp makes this a rather good voltmeter. A simple program to convert from hex to base ten would make the meter easier to read.

Flow Chart for
Analog to Digital Converter



Program for Analog to Digital Converter
(Ramp Approximation)

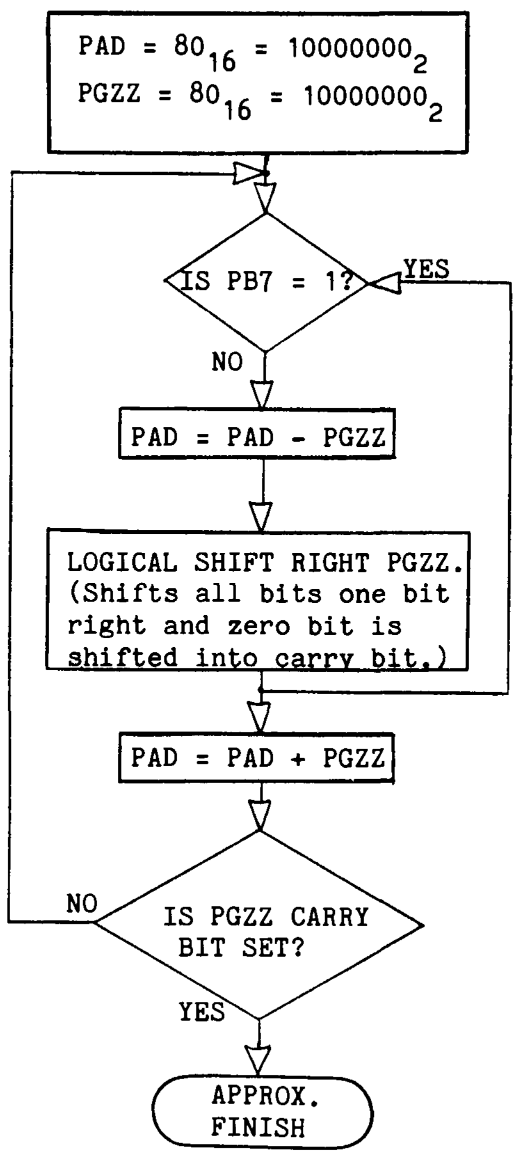
ADDRESS	OPCODE	LABEL	INSTRUCTION		COMMENTS
0300	A9 FF	START	LDAIM	FF	255 in Accumulator
0302	8D 01 17		STA	PADD	Make Port A Output Port
0305	A2 00	AGN	LDXIM	00	Start PAD at zero
0307	8E 00 17	RAMP	STX	PAD	Output Value of X register
030A	AD 02 17		LDA	PBD	Read Port B
030D	10 04		BPL	DISP	Branch if bit 7 = 0
030F	E8		INX		Increment X register
0310	4C 07 03		JMP	RAMP	Continue loop
0313	86 F9	DISP	STX	INH	Put X into Display register
0315	20 1F 1F		JSR	SCANDS	Use KIM Display Subroutine
0318	4C 05 03		JMP	AGN	and start again at zero

3. **Successive Approximation Analog to Digital Used as a Storage Scope.**

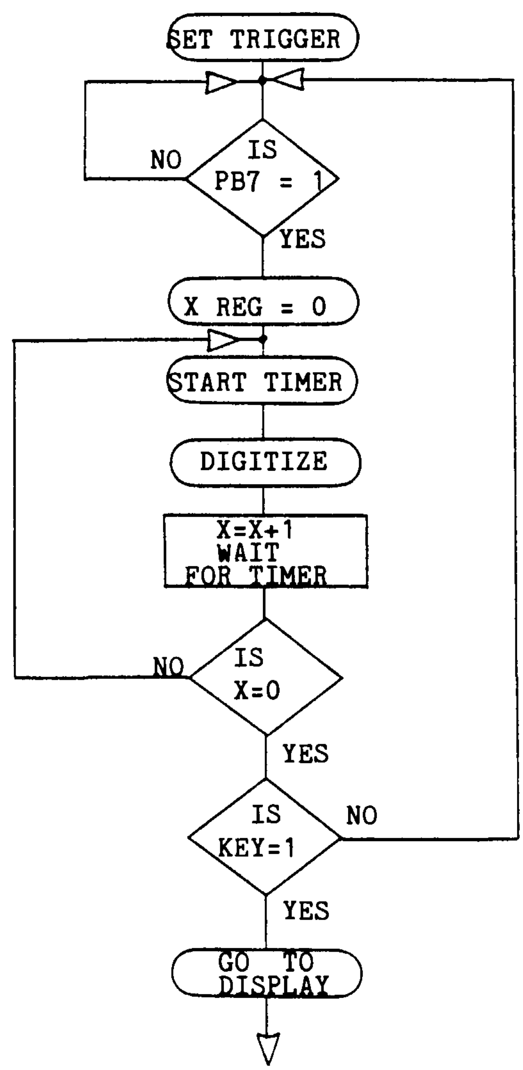
The ramp approximation is quite slow and there is a faster technique known as "successive approximation." It works as follows: the most significant bit to the DAC is set to one and all the others are set to zero. If the comparator indicates that the analog signal is greater than the digital signal, the next lower bit is set to 1 and the test is repeated. If the comparator indicates that the analog signal is less than the digital signal, the highest bit is made zero, and the next lower bit is set to 1 and the test is repeated. This iterative process is repeated until all eight bits have been tested, starting with the MSB and ending with the LSB. The flow chart indicates how this will be accomplished.

This analog to digital conversion scheme will be used in a program which digitizes 256 points on a waveform and then stores the results, to be displayed at a convenient time and with as many repetitions as desired on an oscilloscope. It is useful for examining slow waveforms with an oscilloscope with a low persistance screen, for example ECG waveforms, and it is useful for examining non-periodic waveforms such as a one-shot impulse from an accelerometer. The program has triggering built in, and the output scan portion synchronizes the oscilloscope with a sync signal, turning an inexpensive scope into something more useful. A flow chart for the program is given below.

Flow Chart for Successive Approximation Analog to Digital Conversion



Flow Chart for Storage Scope

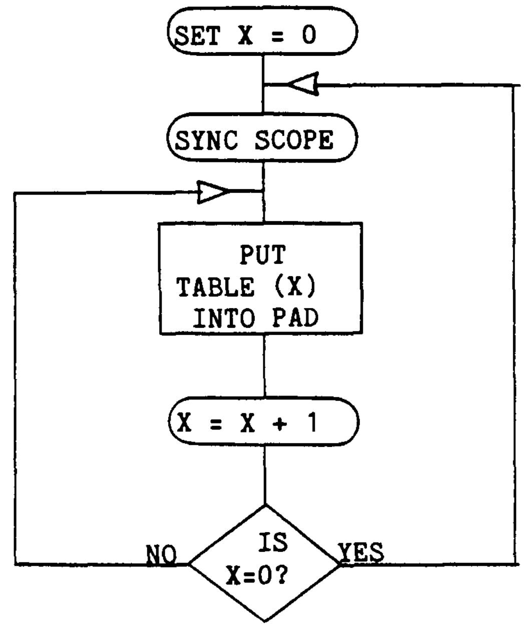


A short description of the behavior of the circuit and program follows. The experimenter chooses the desired trigger level and loads this into location 0306. When the analog signal is greater than this, the comparator makes PB7 go high and the scan begins. The sampling rate and the scan time is determined by the number loaded into the timer and the timer used; locations 0314 and 0316, respectively. It takes at least 200 microseconds to digitize so there is no point in choosing time intervals smaller than this. X is used as an index to identify each of the 256 points on the scan. After the timer is started the analog signal is digitized and the timer is watched until it is finished. X is then incremented and a new point is digitized until all 256 points are finished and stored in TABLE,X.

X is then zero again. This entire process will repeat unless the 1 key is depressed, in which case the program displays the data on the oscilloscope, connected as before to the output of the first op amp. The display will repeat, complete with SYNC signal output from PBO, until the program is halted. In our case we loaded the vector 17FA and 17FB with the starting address of the program (0300) so a depression of the ST key caused the entire program to start over.

A listing of the program is shown on the following page. Notice that the data is stored in TABLE,X located in page 2 of memory, PGZZ is at location 0000, the trigger level is in 0306 and the scan time variable is in 0314 and 0316. The scan time should not be shorter than 200 microseconds. As far as display is concerned, we found that a sweep rate of 200 to 500 microseconds per cm gave good results.

Flow Chart for Display



A few other comments may be in order. First, most of the ideas for this project were obtained in a KIM workshop offered by Dr. Robert Tinker. The software implementation is the author's work. There are some obvious improvements, such as a sample and hold device between the analog source and the comparator or a faster approximation routine. These improvements are left for the reader to implement. The author would be glad to be informed if such improvements are made.

MICRO Reviews: The First Book of KIM

This is one terrific book for anyone who has a KIM-1. It was assembled by Eric Rehnke (Publisher of "KIM-1/6502 User Notes"), Jim Butterfield ("Hypertape" and many other good utilities), and Stan Ockers (a regular "User Notes" contributor). Over half of the book is devoted to "Recreational Programs", games you can play on your basic KIM-1. The section on "Diagnostic & Utility Programs" is worth the price of the book by itself. The remainder of the book contains tutorial information on getting started with your KIM-1, expanding your system, and interfacing to the outside world. This well produced, 176 page resource is available for $9.50 (including postage in USA) from:

ORB
P.O. Box 311
Argonne, IL 60439

Program for Storage Scope

ADDRESS	OPCODE	LABEL	INSTRUCTION		COMMENTS
0300	A9 FF	BEGIN	LDAIM	FF	Initialize Port A to Output
0302	8D 01 17		STA	PADD	
0305	A9 10	START	LDAIM	TSET	Trigger Voltage Set
0307	8D 00 17		STA	PAD	
030A	A2 00		LDXIM	00	Initialize X register
030C	EA		NOP		
030D	EA		NOP		
030E	AD 02 17	TRIG	LDA	PBD	Tinput and test PB7
0311	10 FB		BPL	TRIG	Wait if PB7 = 0
0313	A9 C0	STIME	LDAIM	C0	Set Scan Time here
0315	8D 05 17		STA	TIMER	Select Interval Timer
0318	A9 80		LDAIM	80	Start Digitize Sequence
031A	85 00		STAZ	PGZZ	Store Initial Value
031C	8D 00 17	TEST	STA	PAD	Output Value
031F	AC 02 17		LDY	PBD	Test PB7
0322	30 03		BMI	FWRD	Branch if PB7 = 1
0324	38		SEC		Clear Borrow Flag
0325	E5 00		SBCZ	PGZZ	Subtract bit 7
0327	46 00	FWRD	LSRZ	PGZZ	Set PGZZ for Next Lower Bit
0329	B0 08		BCS	OUT	Out of Digitize Loop if Finished
032B	65 00		ADC	PGZZ	Set Next Lower Bit = 1
032D	4C 1C 03		JMP	TEST	Return to Test all Lower Bits
0330	8D 00 17	OUT	STA	PAD	Final Approximation in PAD
0333	9D 00 02		STAX	TABLE	and in TABLE(X) in Page 2
0336	E8		INX		Bump Table Index
0337	F0 08		BEQ	DISPLY	Go to Display if Table Complete
0339	AD 07 17	CHEK	LDA	TCHEK	Test if Timer is Finished
033C	10 FB		BPL	CHEK	If not, Wait in Loop
033E	4C 13 03		JMP	STIME	Digitize another Point
0341	20 6A 1F	DISPLY	JSR	GETKEY	Is Key 1 Depressed?
0344	C9 01		CMPIM	01	
0346	F0 03		BEQ	SYNC	Yes. Display the Data
0348	4C 05 03		JMP	START	No. Return to Start
034B	A9 01	SYNC	LDAIM	01	Set up PB0 as Sync
034D	8D 03 17		STA	PBDD	Output Pin
0350	A2 00		LDXIM	00	Init X to Display Table
0352	AD 02 17	RPT	LDA	PBD	Toggle PB0 for Sync
0355	49 01		EORIM	01	Signal to Scope
0357	8D 02 17		STA	PBD	
035A	BD 00 02	SCAN	LDAX	TABLE	Output Table(X) for
035D	8D 00 17		STA	PAD	Display on Scope
0360	E8		INX		Increment X register
0361	D0 F7		BNE	SCAN	Continue until all Points Done
0363	4C 52 03		JMP	RPT	Then Repeat

NOTE: This material was submitted by the author to the KIM-1 User Notes and has also been distributed by MOS Technology as "KIM Application Note #11701." It is printed here with the permission of the KIM-1 User Notes and MOS Technology.

The PET Vs. the TRS-80

Bob Wallace
P.O. Box 5415
Seattle, WA 98105

The $600 computer has arrived! The PET (Programmable Educational Terminal) by Commodore, and the TRS-80 by Radio Shack, usher in a new era of personal computing. Both are assembled computers, with video display, keyboard, audio cassette storage, and BASIC; both initially come with 4K bytes of user memory. There the similarity ends; each has significant advantages and disadvantages. I should mention my bias at the outset; I'm for the PET. Some cynical souls will think it's because the store where I work can carry the PET and not the TRS-80; however, the real reason is the PET's much better BASIC. More on this later.

Let's start with the hardware, which I consider about evenly balanced. The PET uses the 6502 microprocessor; as a matter of fact, Chuck Peddle, who designed the 6502 for Mos Technology, also designed the PET. Clock speed is 1 Mhz so a "load immediate" instruction takes 2 microseconds. The TRS-80 uses the Z-80 microprocessor, with a 1.776 Mhz clock. This gives a "load immediate" time of 3.94 microseconds. In general, the Z-80 has a more advanced instruction set (both manufacturers should have assemblers available in 1978). The actual speed of the BASIC depends more on the BASIC interpreter than the cycle times. If the speed is important to you, wait for the inevitable published benchmarks.

As mentioned, both units come with 4K of RAM; you can pay $200 more for the PET to get 8K, and $280 more for the TRS-80 to get 16K. Both could probably be expanded with user-installed memory chips for much less. Additional RAM for both units goes in an expansion box, not yet available. The TRS-80 comes with 4K of ROM, including BASIC, keyboard scanner, cassette and video handlers, etc. It has provision for a second 4K of ROM to get Level II BASIC, and a third 4K ROM to get Level III. The PET comes with 14K of ROM: 8K for BASIC, 4K for the operating system, 1K for a machine language monitor, and 1K for a diagnostic to test all the hardware.

The PET video display is 25 lines of 40 characters, or 1000 characters total. An 8 x 8 dot matrix is used for each character. The character set is 64 ASCII, upper case only. In addition, there are 64 graphics characters, including various lines for limited vector graphics, the playing card suit symbols, and others. Reverse field video (black on white instead of white on black) is available for all 128 characters by setting the high order bit. The TRS-80 video display is 16 lines of 64 characters, or 1024 characters. 5 x 7 dot characters are used in a 6 x 12 dot matrix. The character set is also 64 ASCII characters (upper case only), plus 64 graphics characters. For each character position, any of 6 dots may be turned on, giving flexible point graphics in a total field of 48 by 128 dots. The high order bit of a 7 bit byte determines whether a character or six dots is displayed. The PET display is 9 inches (diagonal), integral to the unit; the TRS-80 display is 12 inches, in a separate video monitor.

The TRS-80 keyboard is a straight, 53-key (Teletype) variety. The PET keyboard has 73 keys (53 in the center plus a 20 key numeric pad), but the keys are arranged in rows and columns, rather than offset like a typewriter. The PET keyboard is crowded, and is probably the worst feature of the unit.

The TRS-80 cassette interface operates at about 250 baud (25 cps); a 4K load would take about 160 seconds. It includes a motor on/off relay for reading and writing data files. Only one cassette recorder can be connected, making tape-to-tape copying and editing very difficult. The cassette recorder is a standard model sold by Radio Shack, connected with cables to the computer. The PET cassette interface operates at about 680 baud; a 4K load takes 60 seconds. One cassette recorder is integral to the unit; it uses a standard deck with special electronics. Another cassette recorder, available as an option in a month (the first option) for about $50, allows tape-to-tape work. Both cassette recorders have motor control.

Expansion from the "BASIC boxes" is planned for both units. Radio Shack is expecting a $700 minifloppy and a $1,500 dot matrix printer by the end of 1977, and later a serial I/O port and an acoustic coupler. A non-standard 40 pin bus connects peripherals (and the expansion memory) to the TRS-80. Commodore expects to have their second cassette box out in a month, a dot matrix printer in late 1977 or early 1978, a minifloppy in early 1978, and some "fun peripheral" soon after. The PET comes with an 8 bit I/O port, plus an IEEE-488 interface bus for peripherals. The IEEE-488 bus is used by instrumentation manufacturers, is standardized, and will probably be supported by some other manufacturers. Motorola is working on a one-chip controller for IEEE-488, since the interface logic is pretty complicated.

The PET is 16½ x 18½ x 14 inches and weighs 44 pounds. The TRS-80 consists of a 16½ x 8 x 3 box with the keyboard and electronics, a 16½ x 13½ x 12 video monitor, the cassette recorder, and a power supply (don't know the total weight).

So far, the differences aren't outstanding. The BASIC's are very different, however. TRS-80 BASIC allows 26 numeric variables (floating point only, 7 digit significance), 2 string variables (A$ and B$, each with 16 characters maximum), and one array, A(i). PET BASIC allows 676 each of integer, floating point (10 significant digits), and string variables; strings can have a maximum length of 256 characters. PET arrays can have multiple dimensions; also, trig functions are included in PET BASIC but not TRS-80 BASIC. The PET BASIC also has a real time clock. TRS-80 BASIC includes commands to set and clear points on the graphic display. Also, more advanced BASIC on the TRS-80 will become available with two additional 4K ROM's. Both BASIC's include machine language functions, PEEK and POKE, and the ability to read and write data files with the cassette recorder. Neither includes the PRINT USING statement, although both have other line format control functions.

Both the PET and the TRS-80 are in production, and have been shipped to customers. There is a long line for both, however. I'm not sure about the TRS-80, but the PET has a 90 to 120 day waiting period, when ordered direct from the factory. The only currently authorized dealer for the PET is Mr. Calculator, a chain in California (I haven't found one up here); however, as soon as the back order situation improves, computer stores and large chains will be carrying it. The TRS-80, of course, is available at Radio Shack, and later at Tandy Computer stores (which will also carry the Processor Technology SOL-20, the Apple-II, and other brands). One other thing: manuals for the PET will be available separately, this fall, and I hope to order some of those if you're interested.

For more information, try "A PET for Every Home", Sept. Kilobaud, and "The Radio Shack TRS-80 Microcomputer System", Sept. Interface Age. Also, there's "Radio Shack's $600 Home Computer", Sept-Oct Creative Computing, "Birth of the PET Computer", Sept-Oct Personal Computing, and "Radio Shack's New Computer System", Oct Radio Electronics.

Several other new personal computers, in the PET/TRS-80 price range, are coming soon. Texas Instruments looks like the first giant corporation to jump into the ring; their new product announcement is expected this fall. Bally (an arcade game manufacturer) has a mail order unit available through JS&A. It's a combination video game and computer, with 12K of RAM/ROM, and a 160 by 100 dot video interface that connects to your TV (no cassette or keyboard) for $300. Atari (another video games manufacturer), and a European and Japenese company are also expected to enter the competition. National Semiconductor and Hewlett-Packard have the capability to produce personal computers, but I haven't heard any rumors from them yet. IBM is a rather distant possibility, I suppose. Zilog is announcing the Z-800 this fall, which is also a factor.

NOTE: This material was originally printed in "Northwest Computer Club News", October 1977 and is reprinted with the permission of the author.

Editor's Notes: One additional difference that may be important to some users is the fact that the PET Monitor supports assembly level code while the TRS-80 only supports BASIC. This means that you can write special routines in 6502 code on the PET. These may be called by the PET BASIC. This facility can greatly enhance the power of the computer. For a user's view of the PET, see "Meet the PET" by Charles Floto, MICRO #2, page 9.

Ludwig von Apple II

Marc Schwartz
220 Everit Street
New Haven, CT 06511

Owners of the Apple II know from the demonstration tapes that the Apple can make sounds. Not all know that it can make music. Having prepared a horse racing program, I decided that it would be fitting to start out the game with the bugle call heard at the track. The following program does just that!

A few words of explanation are in order. The series of "pokes" in lines 30 to 240 set up a musical tone subroutine that is called in line 460.

Each note is represented by a four digit code in A$. The first three digits of the code determine the note, and the last digit determines the length of the note. Line 410 decodes the first three digits by converting each digit to ASCII (Apple ASCII), subtracting 176 from each to give three numbers, from zero to nine, and then multiplying the first number by the second and adding the third. This is one of many possible ways of generating all the numbers from zero to a large number (ninety in this case) using single digits.

Line 420 takes the number just generated and subtracts it from forty. This is done because the subroutine as written is a bit confusing if you want to make music, since the tones go up as the numbers go down. This step corrects for that.

Line 440 determines how long each tone will be. AS "ASC(A$(Z + 3) - 176)" increases, the note lengthens: a "1" produces a very short note, and a "6" makes a very long note. For some reason, higher tones come out more brief than lower tones.

Line 450 determines the tempo. A larger number speeds up the tune; a smaller one slows it down. Tempo numbers can go from 1 to 255.

When the program reaches line 470, it returns to line 400 to begin decoding the next four digits and playing the next note.

I don't think that Chopin would need to worry about competition from anyone using this program, but it is fun to have a musical computer.

THE APPLE II BUGLE CALL

```
10  REM MAKING MUSIC WITH THE APPLE II
20  DIM A$(255)
30  POKE 2,173
40  POKE 3,48
50  POKE 4,192
60  POKE 5,165
70  POKE 6,0
80  POKE 7,32
90  POKE 8,168
100 POKE 9,252
110 POKE 10,165
120 POKE 11,1
130 POKE 12,208
140 POKE 13,4
150 POKE 14,198
160 POKE 15,24
170 POKE 16,240
180 POKE 17,5
190 POKE 18,198
200 POKE 19,1
210 POKE 20,76
220 POKE 21,2
230 POKE 22,0
240 POKE 23,96

300 A$="0011007152117201720172O1"
310 A$(25)="52115211521100715211007100l2"

400 FOR Z=1 TO LEN(A$)-3 STEP 4
410 Z1=(ASC(A$(Z))-176)*(ASC(A$(Z+1))-176)
    +ASC(A$(Z+2))-176
420 Z2=40-Z1
430 POKE 0,Z2
440 POKE 24,ASC(A$(Z+3))-176
450 POKE 1,75
460 CALL 2
470 NEXT Z
480 END
```

MICROBES

Tiny Bugs in Previous MICRO

1:13 HYPERTAPE and ULTRATAPE

00E9 3700 Hz Pulse Duration = C3
00EB 2400 Hz Pulse Duration = 7E

1:19 COMPUTER CONTROLLED RELAYS

Error in the Circuit Diagram. A single 7404 inverter may not have enough drive for two relays. Remove connection between Pin 2 of Write Mic Relay and Pin 2 of Write Remote Relay. Connect 7404 Pin 9 to 7404 Pin 11 and connect 7404 Pin 10 to Write Mic Relay Pin 2. Add a diode between Pin 2 and Pin 1 of the Write Mic Relay. Each relay is now driven by an independent 7404 inverter.

The Challenge of the OSI Challenger

Joel Henkel
Old County Road
Hillsborough, NH 03244

One of the factors that a purchaser of a microcomputer system must consider is the degree of "do it yourself" hardware and software effort he will have to exert to get his system doing what he wants. This effort, not evident from manufacturers' literature, can be critical for user satisfaction, as became evident in our experience with the OSI Challenger. These notes evaluating the Challenger may be helpful to prospective purchasers.

In any hobby industry, user skills are assumed. This is emphasized for microcomputer firms that formerly catered to electronic kit builders. OSI is one of these, having supplied special PC board kits to hams. They follow their own packaging philosophy that differs from the "standard" S-100 bus configuration. Their brochure explains that the 100 pin S-100 connectors were rejected because the fingers were subject to poor contact. Instead, OSI uses MOLEX connectors, which make positive contact. The brochure goes on to describe the rejection of on-board voltage regulators in favor of a self contained regulated power supply.

OSI circuit boards are larger than standard S-100 bus boards. This accommodates their design philosophy of packing many optional functions into one foil pattern. For example, their 430B I/O board supports: an eight channel multiplexed eight bit analog to digital converter, two channels of eight bit digital to analog conversion and a UART controlled cassette I/O interface or an RS232/twenty mil loop I/O interface.

Our system came without keyboard or video monitor. Interfacing for these is left to the user. The computer cabinet has two holes in its rear panel for user implemented I/O cabling from individual boards. The keyboard DIP socket and video output RCA connector are available at the edge of the 440 video board. MOLEX connectors on the edge of the 430 board provides access to the various I/O options.

Hardware documentation consists of kit construction manuals for individual boards, even if the boards are purchased assembled. Various options are treated separately. Overall hardware system documentation is completely lacking. For example, nowhere is there a description of the bus convention and pinout. One must generate these from actual inspection of board foil patterns. This exercise reveals interesting peculiarities, such as bringing the NMI (non-maskable interrupt line) and IRQ (interrupt request line) onto many boards and leaving them unconnected.

The software is sophisticated. One enters the system by resetting. A prompter, D/M, comes up on the video screen. To enter the video monitor, styled after the KIM, enter M and the six hex digits appear near the top of the screen. For DOS (disc operating system), enter D and the DOS is brought up through BASIC by a bootstrap ROM. (Earlier versions required loading a short sequence of memory locations using the video monitor.) From BASIC one can enter the DOS, from which it is possible to go to various modules, such as an extended monitor, back to BASIC, or to activate a few DOS commands, such as loading and recalling disc files, executing programs, or switching floppy disc drives (for dual floppy discs). The EXTENDED BASIC by MICROSOFT has many advanced features, such as string functions, and is apparently much faster than a comparable 8080 BASIC.

Software documentation is poorly organized. Perhaps with so many possible options, the job of creating well organized system documentation was beyond OSI's capability. Our experience with software documentation availability was sobering. The system comes with all OSI software on discette. However, only a BASIC users manual is included, beyond a general system description. One has to order software user manuals separately. We waited a long five months after order for ours.

We have used two versions of the DOS, an original 1.1 version and an updated 2.0 version. One interesting change has to do with copying the DOS itself. The original version could not be copied and an explicit notice to that effect was included. An unfortunate set of circumstances could come about, however, that would wipe out track one, completely disabling any further loads of the DOS. If computer power fails (or one turns off the computer) with the disc in its drive, out goes track one! Apparently a number of users had this happen (including us). Version 2.0 has complete copying capability. According to instructions the first thing one should do is copy the DOS and store away one copy in case of wipeout.

Another change from the original version is the serial display output rate to the video monitor, which was increased from ten characters per second to several times that rate. A third change in the DOS is an augmented facility to read and write disc files.

The 440 board video display format chosen is twenty four characters per line, which is too small. One can only speculate on the reason for the short line.

Many applications could readily use a real time operating system, (RTOS). OSI does not offer a RTOS, but has advertised that one, modelled on DEC's RTS11 is in the works. When contacted recently, however, OSI reported that it has indefinitely postponed development of its RTOS in favor of development of a business system. The contemplated RTOS may explain the interrupt lines found in the foil patterns of several boards mentioned earlier, and a foil pattern option on the 470 floppy disc controller board, a real time clock in the form of a divider chain driven by the on-board crystal clock.

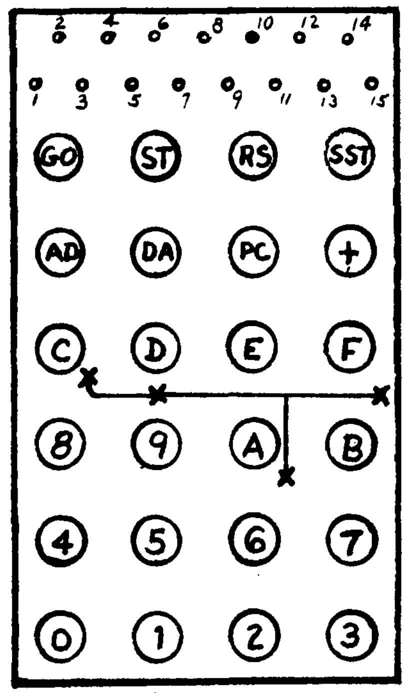
In summary, the OSI Challenger offers a lot of computer for the money. The tradeoff is the board orientation rather than system orientation, requiring a larger than average effort on the part of the user to bring his system up. This effort includes I/O interface cabling and "reading between the lines" in the supporting documentation.

Improving KIM-1 Keyboard Reliability

KIM Application Note
MOS Technology
950 Rittenhouse Road
Norristown, PA 19401

The keyboard on some KIM-1's has a "bouncy" key problem and the "9," "D," or "C" keys may fail entirely. The problem is due to the use of the outer edge of the snap-action discs to jump over the center contact line on the keyboard pc. Since the discs are only held against the pc board with tape, the contact is poor. There are five of these jump-overs in series for the "C" key (four for the "9" key), thereby compounding the problem. To check for the problem, measure the resistance from keyboard pin 3 to pin 15 (numbered from left to right as shown) with the "C" key depressed. It should be less than about 10 ohms.

Fortunately, this problem can be easily corrected. The solution is to solder a thin wire jumper across these poor contacts as follows. Disassemble the keyboard by first removing the four screws on the back of the keyboard at the corners. Then remove the two remaining screws that hold the keyboard to the KIM-1 (note for reassembly that they are longer), being careful not to pull the keyboard pc board away from the KIM-1 board -- it's only attached by the solder at one end. With the KIM-1 upside down, separate the black keyboard panel from the keyboard pc board. You may wish to cover the keyboard with masking tape to hold the keys in place. After cutting four small holes through the clear Mylar at the locations indicated by an X in the figure, the lines from "C" to "9," "D" to "9," "A" to "7" and the line to "B" are exposed. Connecting these points by soldering a thin wire between them routed as shown is sufficient to bridge the five potentially poor contacts.



COMPUTER SHOP

288 NORFOLK ST. CAMBRIDGE, MASS. 02139
corner of Hampshire & Norfolk St. 617-661-2670

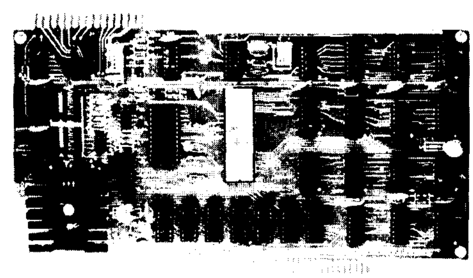
MEMORY — 4 K MEMORY BOARDS THAT CAN BE USED FOR YOUR KIM

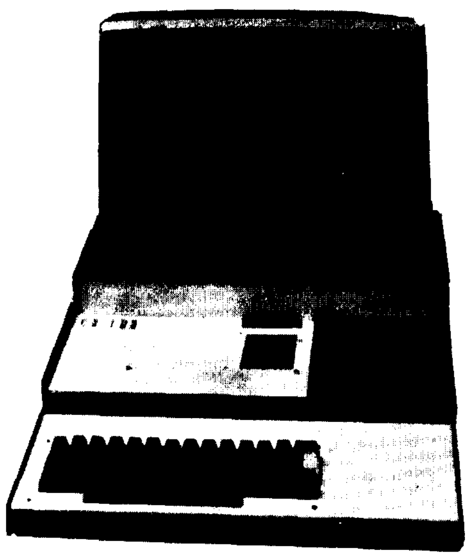
The VF8 4K memory boards have dual 22 pin .156 pinouts but with different pin assignments than your KIM. Instructions for jumpering between connectors are included. This board has been used as modified by many KIM owners.

FULLY ASSEMBLED AND TESTED REG. PWR. $129.00
LOW POWER RAM ADD.................................$ 10.00
KIT..$ 74.50
FULL SET SOCKETS FOR KIT ....$ 10.00
VF8 MOTHERBOARD-BUFFERED FOR 4 BDS. $ 65.00
CONNECTOR ASSEMBLY- KIM TO VF8........$ 20.00

CS 100 VIDEO TERMINAL BOARD FOR YOUR KIM

The CS100 VIDEO TERMINAL BOARD IS A 16 LINE BY 64 CHARACTER DISPLAY GENERATOR WITH CURSOR CONTROL AND EDITING Connect a 5V. ASCII Keyboard to it, a Regulated 5 Volt, Unregulat- 8 Volts, or 8 Volts AC, and your KIM Teletypeport to it along with a video monitor and away you go with all the convenience of a Video Terminal on your KIM.





The CS100-KIM is a fully enclosed portable cabinet that has cutouts for the KIM keyboard and hex display. Space is provided for your cassette recorder ASCII Keyboard, power supply, and extra memory boards. The display shown was generated by our Video Terminal Board connected to the KIM. The cabinet is heavy duty 1/8 inch aluminum finished in a blue spatter with white panels. An S100 low profile e 3 slot motherboard is available that fits under the KIM is available

CS100 Cabinet cut out for KIM $ 129.00( )
3 Connector S100 MB Assem. $ 75.00( )
VF8 4K Memory Board Assm. $ 129.00( )
V.F8 4K Ram Kit................$ 74.50( )
Low Power 2102 Option........$ 10.00( )
Socket Option....................$ 10.00( )
V.F8 Motherboard...............$ 65.00( )
Connector Board................$ 20.00( )
CS100 TIM Kit....................$ 129.95( )
CS100 6502 CPU Kit............$ 169.95( )
CS100 Front Panel Kit.........$ 129.95( )
CS100-VTB Vid. Term. Bd.Kit$ 155.00( )
CS100-VTB Vid. Term. Bd.Asm$ 185.00( )
KIM-1...............................$ 245.00( )
Total of Order.....................$...............
Mass. Res. Sales Tax...........$...............
Shipg. 1% ($2.00 minimum).....$...............
Total Remittance or Charge...$...............

BAC, VISA, MC NO. ....

SIGNATURE..

NAME..

ADDRESS..

CITY..................................STATE.........ZIP........

Important Addresses of KIM-1 and Monitor

William R. Dial
438 Roslyn Avenue
Akron, OH 44320

Address	Label	Function
00EF	PCL	Program Counter - Lo Byte
00F0	PCH	Program Counter - Hi Byte
00F1	P (PREG)	Status Register of Processor Set "00" for Binary
00F2	SP (SPUSER)	Stack Pointer
00F3	A (ACC)	Accumulator
00F4	Y	Y-Register
00F5	X	X-Register
00F6	CHKHI	Checksum on Tape, Hi
00F7	CHKSUM	Checksum on Tape, Lo
00F8	INL	Input Buffer, Lo - Display Buffer
00F9	INH	Input Buffer, Hi - Display Buffer
00FA	POINTL	Pointer, Lo - Display
00FB	POINTH	Pointer, Hi - Display
00FC	TEMP	Temporary Storage Byte
00FD	TMPX	Temporary Storage Byte
00FE	CHAR	Current Character for TTY
00FF	MODE	Byte Indicating KYBD or TTY Mode on KIM

Detail of Processor Status Register P (00F1)

Bit No.	Flag	Flags (1 = set)
7	N	Negative Result
6	V	Overflow
5	-	Reserved for Expansion
4	B	Break Command
3	D	Decimal Mode
2	I	Interrupt Disable
1	Z	Zero Result
0	C	Carry

01FF
01FE .... } STACK   Needed to Process Interrupts, save Addresses, etc.
01F8 etc.

I/O Ports, Interval Timers, and 6530 RAM Usage

Address		Label	Function
1700		PAD	Port A Data (user I/O)
1701		PADD	Port A Data Direction (1 = Output)
1702		PBD	Port B Data (User I/O)
1703		PBDD	Port B Data Direction (0 = Input)
1704 /	1744	CLKIT	INTERVAL TIMER
1705	1745	CLK8T	1704 et seq User
1706	1746	CLK64T	1744 et seq KIM MONITOR
1707	1747	CLK1024T	
1707	1747	CLKRDI	Read Time Out Bit
1706	1746	CLKRDT	Read Time
170C	174C	1T	TIMER USED when IRQ Interrupt at PB7 needed
170D	174D	8T	
170E	174E	64T	
170F	174F	1024T	
1740		SAD	Port A Data (KIM MONITOR)
1741		PADD (SADD)	Port A Data Direction
1742		SBD	Port B Data (KIM MONITOR)
1743		PBDD (SBDD)	Port B Data Direction
1780			Available Memory Block (Program PLEASE, etc.)
17E7		CHKL	Checksum for Tape Monitor
17E8		CHKH	
17E9		SAVX	Storage Location
17EA			" "
17EB			" "
17EC		VEB	Volatile Execution Block
17F2		CNTL 30	TTY Delay
17F3		CNTH 30	TTY Delay
17F4		TIMH	
17F5		SAL	Starting Address - Lo (Audio and Paper Tape)
17F6		SAH	- Hi
17F7		EAL	Ending Address - Lo
17F8		EAH	- Hi
17F9		ID	ID Number (Program No. on Tape)

17FA/FFFA	NMIV	(NMIL)	Stop Vector (Stop = 1C00)	Load 00
FB/FFFB		(NMIH)		1C
FC/FFFC	RSTV	(RSTL)	RST Vector	00
FD/FFFD		(RSTH)		1C
FE/FFFE	IRQV	(IRQL)	IRQ Vector (BRK = 1C00)	00
FF/FFFF		(IRQH)		1C

SUB-ROUTINES - 6530-003

Address	Label	Function
1800	DUMPT	Dump Memory to Tape
1873	LOADT	Load Memory from Tape
1932	INTVEB	Initiate Volatile Execution Block
194C	CHKT	Compute CHKSUM for Tape Load
195E	OUTBTC	Output One Byte
196F	HEXOUT	Convert LSD of A to ASCII and Output to Tape
197A	OUTCHT	Output to Tape One ASCII CHAR (Use Subs ONE & ZRO)
199E	ONE	Output to Tape = 1 (9 pulses 138 µ sec each)
19C4	ZRO	Output 0 to Tape (6 pulses 207 µ sec each)
19EA	INCVEB	Sub to INC VEB + 1, 2
19F3	RDBYT	Sub to read Byte from Tape
1A00	PACKT	Pack A = ASCII into SAVX as Hex Data
1A24	RDCHT	Get 1 Character from Tape and Return with Character in A (Use SAVX + 1 to ASM Char)
1A41	RDBIT	Gets one bit from Tape and returns it in sign of A
1A6B	PLLCAL	Diagnostics: PLL calibrate Output, 166 µ sec pulse string

SUB-ROUTINES - 6530-002

Address	Label	Function
1C00	SAVE	KIM Entry via STOP (NMI) or BRK (IRQ) Also SST
1C22	RST	KIM Entry via RST (Reset)
1C2A	DETCPS	Count Start Bit
1C4F	START	Make TTY/KB Selection
1CDC	PCCMD	Display Program Counter by Moving PC to POINT
1C64	CLEAR	Clear Input Buffer INL, INH
1C6A	READ	Get Character
1C77	TTYKB	Main Routine for Keyboard and Display

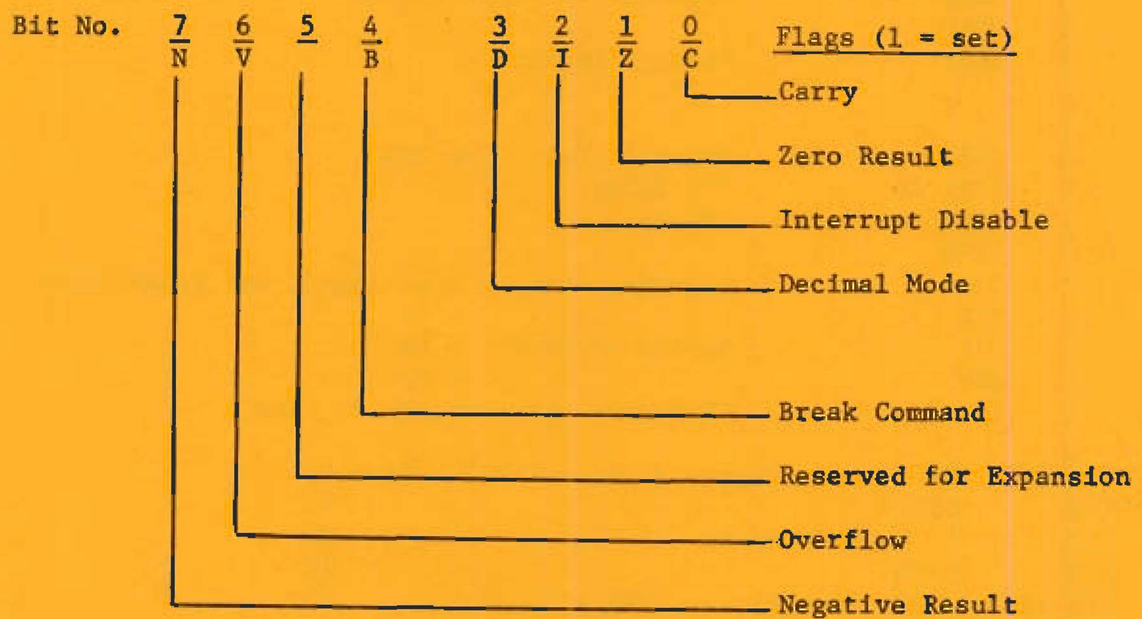
Address	Label	Function
1CE7	LOAD	Load Paper Tape from TTY
1D42	DUMP	Dump to TTY from Open Cell Address to LIMHL, LIMHH Limit High, H and L
1E1E	PRTPNT	Sub to Print POINTL, POINTH
1E2F	CRLF	Print String of ASCII Characters from TOP + X to TOP
1E3B	PRTBYT	Print 1 Hex Byte as Two ASCII Characters
1E5A	GETCH	Get 1 Character from TTY, Return from Sub with Char in A. X is preserved and Y returned = FF.
1E88	INITS	Initialization for SIGMA
1E9E	OUTSP	Print One Character CHAR = A. X is preserved, Y returned = FF. OUTSP Prints One Space.
1ED4	DELAY	This loop simulates DETCPS Section and will delay 1 Bit Time.
1EEB	DEHALF	Delay half Bit Time - Double right shift of Delay Constant for a Div by 2.
1EFE	AK	Sub to Determine if Key is depressed or Condition of SSW (Key not dep or TTY Mode A = 0) (Key dep or KB Mode A = not zero)
1F19	SCAND	Output to 7 Segment Display
1F1F	SCANDS (DISPLA)	Lights 7 Segment Display
1F48	CONVD	Convert and Display Hex - Used by SCAND only
1F63	INCPT	Sub to Increment POINT
1F6A	GETKEY	Get Key from Keyboard, Return with A = Key value. If A GT. than 15 then illegal or no Key.
1F91	CHK	Sub to Compute Check Sum
1F9D	GETBYT	Get 2 Hex Characters and Pack into INL, INH. X preserved, Y returned = 0.
1FAC	PACK	Shift Character in A into INL, INH
1FD5	TOP	Table
1FE7	TABLE	Table Hex to 7 Segment

Important Addresses of KIM-1 and Monitor

William R. Dial
438 Roslyn Avenue
Akron, OH 44320

Address	Label	Function
00EF	PCL	Program Counter - Lo Byte
00F0	PCH	Program Counter - Hi Byte
00F1	P (PREG)	Status Register of Processor Set "00" for Binary
00F2	SP (SPUSER)	Stack Pointer
00F3	A (ACC)	Accumulator
00F4	Y	Y-Register
00F5	X	X-Register
00F6	CHKHI	Checksum on Tape, Hi
00F7	CHKSUM	Checksum on Tape, Lo
00F8	INL	Input Buffer, Lo - Display Buffer
00F9	INH	Input Buffer, Hi - Display Buffer
00FA	POINTL	Pointer, Lo - Display
00FB	POINTH	Pointer, Hi - Display
00FC	TEMP	Temporary Storage Byte
00FD	TMPX	Temporary Storage Byte
00FE	CHAR	Current Character for TTY
00FF	MODE	Byte Indicating KYBD or TTY Mode on KIM

Detail of Processor Status Register P (00F1)



01FF 01FE .... 01F8 etc.	STACK	Needed to Process Interrupts, save Addresses, etc.

I/O Ports, Interval Timers, and 6530 RAM Usage

Address		Label	Function
1700		PAD	Port A Data (user I/O)
1701		PADD	Port A Data Direction (1 = Output)
1702		PBD	Port B Data (User I/O)
1703		PBDD	Port B Data Direction (0 = Input)
1704 /	1744	CLKIT	INTERVAL TIMER
1705	1745	CLK8T	1704 et seq User
1706	1746	CLK64T	1744 et seq KIM MONITOR
1707	1747	CLK1024T	
1707	1747	CLKRDI	Read Time Out Bit
1706	1746	CLKRDT	Read Time
170C	174C	1T	TIMER USED when IRQ Interrupt at PB7 needed
170D	174D	8T	
170E	174E	64T	
170F	174F	1024T	
1740		SAD	Port A Data (KIM MONITOR)
1741		PADD (SADD)	Port A Data Direction
1742		SBD	Port B Data (KIM MONITOR)
1743		PBDD (SBDD)	Port B Data Direction
1780			Available Memory Block (Program PLEASE, etc.)
17E7		CHKL	Checksum for Tape Monitor
17E8		CHKH	
17E9		SAVX	Storage Location
17EA			" "
17EB			" "
17EC		VEB	Volatile Execution Block
17F2		CNTL 30	TTY Delay
17F3		CNTH 30	TTY Delay
17F4		TIMH	
17F5		SAL	Starting Address - Lo (Audio and Paper Tape)
17F6		SAH	- Hi
17F7		EAL	Ending Address - Lo
17F8		EAH	- Hi
17F9		ID	ID Number (Program No. on Tape)

17FA/FFFA	NMIV	(NMIL)	Stop Vector (Stop = 1C00)	Load 00
FB/FFFB		(NMIH)		1C
FC/FFFC	RSTV	(RSTL)	RST Vector	00
FD/FFFD		(RSTH)		1C
FE/FFFE	IRQV	(IRQL)	IRQ Vector (BRK = 1C00)	00
FF/FFFF		(IRQH)		1C

SUB-ROUTINES - 6530-003

Address	Label	Function
1800	DUMPT	Dump Memory to Tape
1873	LOADT	Load Memory from Tape
1932	INTVEB	Initiate Volatile Execution Block
194C	CHKT	Compute CHKSUM for Tape Load
195E	OUTBTC	Output One Byte
196F	HEXOUT	Convert LSD of A to ASCII and Output to Tape
197A	OUTCHT	Output to Tape One ASCII CHAR (Use Subs ONE & ZRO)
199E	ONE	Output to Tape = 1 (9 pulses 138 μ sec each)
19C4	ZRO	Output 0 to Tape (6 pulses 207 μ sec each)
19EA	INCVEB	Sub to INC VEB + 1, 2
19F3	RDBYT	Sub to read Byte from Tape
1A00	PACKT	Pack A = ASCII into SAVX as Hex Data
1A24	RDCHT	Get 1 Character from Tape and Return with Character in A (Use SAVX + 1 to ASM Char)
1A41	RDBIT	Gets one bit from Tape and returns it in sign of A
1A6B	PLLCAL	Diagnostics: PLL calibrate Output, 166 μ sec pulse string

SUB-ROUTINES - 6530-002

Address	Label	Function
1C00	SAVE	KIM Entry vis STOP (NMI) or BRK (IRQ) Also SST
1C22	RST	KIM Entry via RST (Reset)
1C2A	DETCPS	Count Start Bit
1C4F	START	Make TTY/KB Selection
1CDC	PCCMD	Display Program Counter by Moving PC to POINT
1C64	CLEAR	Clear Input Buffer INL, INH
1C6A	READ	Get Character
1C77	TTYKB	Main Routine for Keyboard and Display

Address	Label	Function
1CE7	LOAD	Load Paper Tape from TTY
1D42	DUMP	Dump to TTY from Open Cell Address to LIMHL, LIMHH Limit High, H and L
1E1E	PRTPNT	Sub to Print POINTL, POINTH
1E2F	CRLF	Print String of ASCII Characters from TOP + X to TOP
1E3B	PRTBYT	Print 1 Hex Byte as Two ASCII Characters
1E5A	GETCH	Get 1 Character from TTY, Return from Sub with Char in A. X is preserved and Y returned = FF.
1E88	INITS	Initialization for SIGMA
1E9E	OUTSP	Print One Character CHAR = A. X is preserved, Y returned = FF. OUTSP Prints One Space.
1ED4	DELAY	This loop simulates DETCPS Section and will delay 1 Bit Time.
1EEB	DEHALF	Delay half Bit Time - Double right shift of Delay Constant for a Div by 2.
1EFE	AK	Sub to Determine if Key is depressed or Condition of SSW (Key not dep or TTY Mode A = 0) (Key dep or KB Mode A = not zero)
1F19	SCAND	Output to 7 Segment Display
1F1F	SCANDS (DISPLA)	Lights 7 Segment Display
1F48	CONVD	Convert and Display Hex - Used by SCAND only
1F63	INCPT	Sub to Increment POINT
1F6A	GETKEY	Get Key from Keyboard, Return with A = Key value. If A GT. than 15 then illegal or no Key.
1F91	CHK	Sub to Compute Check Sum
1F9D	GETBYT	Get 2 Hex Characters and Pack into INL, INH. X preserved, Y returned = 0.
1FAC	PACK	Shift Character in A into INL, INH
1FD5	TOP	Table
1FE7	TABLE	Table Hex to 7 Segment