P1 (CUMULATIVE) FOR K = 4, N = 2

Q	ALPHA = 0.	.125	.25	.5	1	2	4
	RHO = 0.	.5	1	2	4	8	16
.05	.01425	.009084	.005788	.002348	.000384	.000010	0.
.1	.04125	.027168	.017882	.007733	.001437	.000048	0.
.15	.0795	.054073	.036744	.016922	.003556	.000152	0.
.2	.1275	.089520	.062769	.030748	.007286	.000393	.000001
.25	.18375	.133113	.096259	.050097	.013354	.000905	.000003
.3	.24675	.184334	.137399	.075886	.022702	.001920	.000012
.35	.315	.242528	.186226	.109026	.036524	.003843	.000036
.4	.387	.306890	.242600	.150374	.056293	.007343	.000105
.45	.46125	.376454	.306162	.200670	.083772	.013505	.000294
.5	.53625	.450075	.376292	.260451	.120989	.024037	.000789
.55	.6105	.526417	.452062	.329933	.170164	.041539	.002047
.6	.6825	.603934	.532179	.408871	.233544	.069831	.005153
.65	.75075	.680856	.614922	.496369	.313095	.114267	.012580
.7	.81375	.755166	.698075	.590643	.409995	.181859	.029745
.75	.870000	.824586	.778845	.688731	.523808	.280823	.067864
.8	.918000	.886549	.853775	.786115	.651201	.418694	.148293
.85	.956250	.938185	.918646	.876272	.784011	.597341	.305922
.9	.983250	.976292	.968362	.950105	.906358	.801642	.578264
.95	.997500	.997315	.996833	.995255	.990437	.975744	.928547

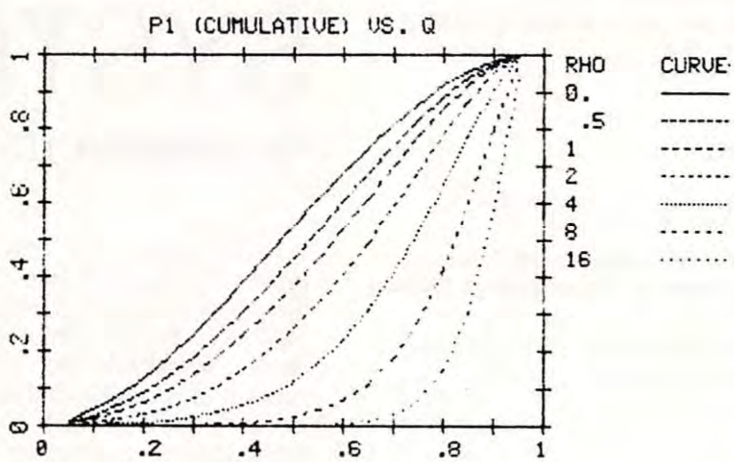


Figure 3

PE FOR K = 4, N = 2

M	ALPHA = 0.	.125	.25	.5	1	2	4
	RHO = 0.	.5	1	2	4	8	16
2	.509553	.454439	.408615	.338910	.259034	.218670	.288887
4	.759892	.708395	.661886	.582550	.469299	.362902	.359455

8 bytes, 6 or 7 bits high. These routines are made available in this manner in case you would like to mix text and graphics in a more sophisticated manner than a simple dump.

Following the transfer vector is the data area. The values here specify how the Visible Memory is to be dumped and where it is. SDUMP is assembled to work with the 460 Paper Tiger, but by making the changes described in lines 25-27, the code will work on the Paper Tiger 440. Presumably, with similarly minor changes, SDUMP will also work on the newest Paper Tiger, the 445.

The following should be noted about SDUMP and its use: The only code in SDUMP that is specific to a particular version of BASIC is that in OUTCH, lines 235-280. This code was given to me by Greg Yob — thanks Greg. It outputs the character in the ACC directly to the device whose number is in RDEV, at location $600E in the data area. Because this code bypasses the PET's file system and directly accesses the IEEE-488 routines, the device does not even have to be opened.

Each routine in SDUMP checks to see if the stop key is pressed, using the routine STOPTS at lines 281-292. If so, the routine quits and returns to the routine which called it. Because of the way the Paper Tigers enter and exit graphics mode, it is possible for them to be left in graphics mode when the stop key is pressed. If this happens, you will know it when it does, the easiest method of recovering is to turn the printer off, then on.

You should not have a CMD operation open

```
4000 REM PLOT X IN (XA,XB) NX WIDE, Y IN (YA,YB) NY HIGH, KEY PK
4005 IF PF=0 THEN RETURN
4010 CLEAR
4060 CX=NX/(XB-XA):REM CONVERSION CONSTANTS
4070 CY=NY/(YB-YA)
4090 FOR JR=1 TO NR:REM GET THE DATA POINTS
4100 DOTL DO(JR,1),DO(JR,2)
4110 FOR JQ=1 TO NQ
4120 X=QS(JQ)
4130 IX=INT(.5+(X-XA)*CX)
4140 IF IX<0 THEN IX=0:REM MAKE SURE X OK
4150 IF IX>NX THEN IX=NX
4160 IX=IX+OX
4210 IF PK=1 THEN Y=DD(JQ,JR):REM PK=1 FOR P1 DENS DIST
4220 IF PK=2 THEN Y=CD(JQ,JR):REM PK=2 FOR P1 CUM DIST
4300 REM CONVERT Y LIKE X
4310 IY=INT(.5+(Y-YA)*CY)
4320 IF IY<0 THEN IY=0:REM FORCE ON PLOT
4330 IF IY>NY THEN IY=NY
4335 IY=IY+OY
4340 IF JQ=1 THEN MOVE IX,IY
4350 DRAW IX,IY
4480 NEXT JQ
4490 NEXT JR
4495 DOTL 1,0
4500 REM PRODUCE THE PLOT
4510 MOVE OX,OY:REM BORDER
4520 DRAW OX+NX,OY:DRAW OX+NX,OY+NY
4530 DRAW OX,OY+NY:DRAW OX,OY
4590 MOVE OX+NX/2-3*(LEN(PL$)+6),OY+NY+10
4592 CHAR PL$;" VS. Q"
4594 PL$=""
4600 REM DISPLAY RHO AND DOTS
4610 IX=OX+NX+10:IY=OY+NY-7
4620 MOVE IX,IY:CHAR "RHO    CURVE"
4630 FOR I=1 TO NR
4640 V=RS(I):GOSUB2002
4650 IY=IY-12:MOVE IX,IY
4660 CHAR V$
4670 DOTL DO(I,1),DO(I,2):LINE IX+42,IY+3,319,IY+3
4680 NEXT I
4690 DOTL 1,0
4700 REM DRAW A GRID
4702 TL=3:REM TIC LENGTH
4705 DX=.1:REM X GRID SPACING (ALWAYS)
4710 DY=10:REM Y SPACING - HAVE TO SEARCH
4715 IF YB/5<DY THEN DY=DY/10:GOTO4715
4720 EX=INT(XB/DX+.01):EY=INT(YB/DY+.01):REM POINTS ON GRID
4725 FX=1:IF EX>5 THEN FX=2:IF EX>10 THEN FX=5:IF EX>20 THEN FX=10
4730 FY=1:IF EY>5 THEN FY=2:IF EY>10 THEN FY=5:IF EY>20 THEN FY=10
4735 FOR I=0 TO EY:OZ=OY+I*DY*CY:LINE OX-TL,OZ,OX+TL,OZ:REM Y AXIS
4737 LINE OX+NX-TL,OZ,OX+NX+TL,OZ
4740 IF I=FY*INT(I/FY) THEN CHROT 1:MOVE OX-TL-5,OZ-3:CHAR MID$(STR$(I*DY),2)
4745 NEXT I
4750 FOR I=0 TO EX:OZ=OX+I*DX*CX:LINE OZ,OY-TL,OZ,OY+TL:REM X AXIS
4752 LINE OZ,OY+NY-TL,OZ,OY+NY+TL
4755 IF I=FX*INT(I/FX) THEN CHROT 0:MOVE OZ-3,OY-TL-10:CHAR MID$(STR$(I*DX),2)
4760 NEXT I
4900 REM PRODUCE THE PLOT
4910 PRINT:PRINT
4920 CMD3:REM REGULAR OUT TO THE SCREEN
4930 SYS(LP):REM THERE IT GOES
4940 CMD1:REM BACK TO THE PRINTER
4950 RETURN
```

Listing 1

to the Paper Tiger when SDUMP is called, because of the way this command is interpreted in the IEEE-488 system. To avoid this, open a unit to the screen (device 3) and switch to this unit before invoking SDUMP. For example:

```
OPEN 1,4:REM PRINTER FILE
OPEN 2,3:REM SCREEN FILE
CMD 1:REM OUTPUT TO PRINTER
.......
CMD 2:REM DIVERT OUTPUT
SYS(96*256):REM DUMP VISIBLE MEMORY
CMD 1:REM RESUME PRINTER OUTPUT
```

The byte in the data area called EORVAL (at $6011) is exclusive-ored with each Visible Memory byte when it is accessed for dumping. This gives a visible indication of the progress of the dump which I find entertaining. It is actually an instance

of Cohen's first law of interactive computing — "Always let the operator know that something is going on." However, this leaves the screen reversed when the dump finishes. If you do not like this, there are (at least) two possibilities: (1) Set EORVAL to zero ($00); the exclusive or will than not change anything. (2) If you are using the Keyword Graphics Package, follow the call to SDUMP with a 'SCFLIP 0,0,319,199'; this will reverse the whole screen, restoring its original condition.

To load SDUMP together with the MTU keyword Graphics Package, when reserving memory space, do a 'POKE 53,96' instead of 'POKE 53,98' for a 32K system, and similarly for smaller systems. This will reserve the two pages needed by SDUMP.

Listing 2

```
9100 REM MTU VISIBLE MEMORY TO IDS 460 PAPER TIGER SCREEN DUMP
9110 PRINT#U:REM SPACE
9120 PRINT#U,CHR$(3);:REM ENTER GRAPHICS MODE
9130 VM=256*PEEK(832):REM START OF VISIBLE MEMORY
9140 PVMEM
9142 GRSHRT
9145 S=7:REM ROWS PER GROUP
9150 FOR R0=0 TO 199 STEP S:REM S ROWS AT A TIME
9160 :R1=R0+S-1:REM END OF ROW GROUP
9170 :IF R1>199 THEN R1=199
9180 :FOR C=0 TO 39:REM A BYTE (8 BIT COLUMNS) AT A TIME
9190 ::FOR I=0 TO 7:REM CLEAR VALUES TO BE PRINTED
9200 :::P(I)=0
9210 ::NEXT I
9220 ::V=VM+C:REM LOC OF BYTE
9225 ::P2=1:REM POWER OF 2 TO ADD
9230 ::FOR R=R0 TO R1:REM SCAN THE ROWS
9235 :::PRINT C;R
9240 :::B=PEEK(V):REM GET THE BYTE (8 BITS)
9250 :::V=V+40:REM LOC OF BYTE BELOW
9260 :::IF B=0 THEN 9315:REM FASTER IF EMPTY
9270 :::M=1:REM MASK (2^(7-I))
9280 :::REM ACCUMULATE VALUES FOR PRINTING
9290 :::FOR I=7 TO 0 STEP -1
9295 ::::IF (B AND M)<>0 THEN P(I)=P(I)+P2
9300 ::::M=M+M
9310 :::NEXT I
9315 :::P2=P2+P2
9320 ::NEXT R:REM DO THE ROWS
9330 ::REM NOW, PRINT THE 8 COLUMNS OF ROWS
9340 ::FOR I=0 TO 7
9350 :::PRINT#U,CHR$(P(I));
9360 :::IF P(I)=3 THEN PRINT#U,CHR$(P(I));:REM 3 IS SPECIAL
9370 ::NEXT I
9390 :NEXT C:REM END OF COLUMN LOOP
9400 :PRINT#U,CHR$(3);CHR$(14);:REM GRAPHICS LINE FEED/RETURN
9410 :VM=VM+S*40:REM DOWN S ROWS
9420 NEXT R0:REM END OF ROW GROUP LOOP
9430 PRINT#U,CHR$(3);CHR$(2):REM LEAVE GRAPHICS MODE
9439 ]X
9440 VISMEM
9450 RETURN:REM DONE
```

Listing 3

```
0000                   ; SDUMP.ASM - MTU TO IDS PAPER TIGER 460 (440) SCREEN DUMP
0000                   ;
0000                   ; BY MARTIN J. COHEN, DECEMBER 1980
0000                   ;
0000                   ; ANYONE WHO WANTS TO CAN USE THIS PROGRAM,
0000                   ; ALTHOUGH SOME ACKNOWLEDGEMENT WOULD BE APPRECIATED
0000                   ;
0000                   ;
0000                   ; APPROXIMATE TIME NEEDED TO DUMP VISIBLE MEMORY:
0000                   ;
0000                   ; AT 1200 BAUD, 1 MIN, 30 SEC
0000                   ; AT 9600 BAUD, 45 SEC (WITH 3 MS DELAY SET BY NMSDLY, BELOW)
0000                   ;
0000                   ; THE ACTUAL CPU TIME NEEDED IS ABOUT 3 SECONDS!!
0000                   ;
0000                   ;
0000                           * = $6000        ; TWO PAGES BELOW KGP CODE
6000                   ;
6000  4C 18 60                 JMP OUTVM        ; SKIP DATA AREA AND DUMP THE VIS MEM
6003  4C 8D 60                 JMP OUTROW       ; OUTPUT ROW STARTING AT VM
6006  4C DF 60                 JMP OUTCOL       ; OUTPUT A COLUMN OF 8 BYTES
6009                   ;
6009                   ; DATA AREA ...
6009                   ;
6009                   ; NOTE - TO RUN THIS ROUTINE ON A 440 INSTEAD OF A 460,
6009                   ; CHANGE THE FOLOWING VALUES AS INDICATED (VALUES IN DECIMAL):
6009                   ; RPFXC=0, RREP=33, RVAL=6, REND=2, RXGR=11.
6009                   ;
6009  14               RPFXC   .BYTE 20         ; NUMBER OF BLANK PREFIX COLUMNS (440:0)
600A  02               RPFXR   .BYTE 2          ; NUMBER OF BLANK PREFIX ROWS
600B  1C               RREP    .BYTE 28         ; MAIN REPETITION COUNT (440:33)
600C  07               RVAL    .BYTE 7          ; ROWS TO OUTPUT IN MAIN LOOP (440:6)
```

NEW LIFE for old Pets™?

Is it a dream ... is it fantasy?

Switch from old ROMs to new

- Not sure about the ROM Retrofit Kit from Commodore? Now you can use all three sets of Commodore ROMs and others as well.
- The Basic Switch allows switch selection of either ROM set (your original set or your retrofit set) from Commodore. Plus, Models 15-A and 15-B include an additional zero insertion force socket allowing easy use of ROMs like the BASIC Programmer's Toolkit ... concurrently.
- Models 15-A and 15-B The Basic Switch plus ... includes expanded cable assembly and zero insertion force socket. Your 15th ROM simple plugs in ... enabled while either ROM set is selected Socket 15 may be readdressed by the user for additional flexibility.
- The Basic Switch is sold in assembled form only. All models are designed for easy attachment to your PET with a convenient cable assembly. No soldering or drilling is required. The Basic Switch mates with a cable assembly at your primary board, and does not use the physical connectors of any PET ports.
- Price List:
  Old PETs $129.95*
  New PETs $139.95*
  *Case optional, $15 additional.
- Our prices and complete product specifications are available by contacting APPLIED MICRO SYSTEMS, Mishawaka, Indiana; or any Commodore Dealer.

Now available for Commodore® Basic 4.0

**Dealer inquiries are encouraged.
Free MACHINE LANGUAGE MONITOR COMMANDS List will be included with pricing and product specification requests.**

APPLIED MICRO SYSTEMS
3502 Home Street, Mishawaka, Indiana 46544
1-219-259-3787 (Indiana) • 1-800-348-7208

COMPUTE!'s Book Corner

Back Issues:

COMPUTE! January, 1981, Issue 8 .... $2.50 ☐
COMPUTE! February, 1981, Issue 9 .... $2.50 ☐
COMPUTE! March, 1981, Issue 10 .... $2.50 ☐

For The Single Board Computer Owners: (KIM, SYM, AIM, OSI)

COMPUTE II, All Three Issues
(The only ones: **COMPUTE II** merged into **COMPUTE!**)
Issues 1, 2, and 3 .... $5.00 ☐

Individual Issues

COMPUTE II, #1 .... $2.00 ☐
COMPUTE II, #2 .... $2.00 ☐
COMPUTE II, #3 .... $2.00 ☐

Miscellaneous:

Best of the PET Gazette .... $10.00 ☐
(Note...an old anthology primarily for old ROM, 8K original PETs.)
Collected PET User Notes (7 issues) .... $10.00 ☐
(Above note applies)
Collected 6502 User Notes (17 issues) .... $20.00 ☐
(Specifically for Single-Board Computer Owners)

Ordering information:

Address orders to:
Compute's Book Corner
P.O. Box 5406
Greensboro, NC 27403 USA

Payment, in US funds, required with order. *Master Charge/VISA accepted. All items subject to availability. Please add the following amount(s) for shipping/handling:
US $2.00 (UPS or mail) • Canada $2.00 (mail)
Foreign Air Mail $5.00

The $180 Programmable character generator that performs above and beyond its price.

It's the NEW HAL PCG 6500 for your PET. With HAL, you control each dot in the 8 x 8 matrix so you get sharp, clear graphics. HAL offers:

- Storage for 64 user-programmable characters
- Built-in CB2 sound amplifier
- PCG Manual and demo program
- Interface to PET/CBM with 24 pin character generator ROMS. (If your PET uses 28 pin ROMS, Conversion Kits are available for $45. To simplify ordering, please indicate memory RAM #2114 or #6550.)

The HAL PCG 6500 is available exclusively at Systems Formulate...

Call today for more information or to place your phone order: (415) 326-9100 39 Town & Country Village, Palo Alto, CA 94301
We honor Master Charge, Visa, Check or Money Order (California residents add 6½% sales tax)

Dealer Inquiries Invited

SYSTEMS FORMULATE CORPORATION
39 Town & Country Village • Palo Alto, California 94301

```
600D  04          REND   .BYTE 4           ; ROWS TO OUTPUT AT END (440:2)
600E              ; THE TOTAL NUMBER OF ROWS OUTPUT = RREP*RVAL + REND = 200
600E  04          RDEV   .BYTE 4           ; OUTPUT DEVICE
600F  0E          RXGR   .BYTE 14          ; GRAPHICS RETURN (440:11)
6010  03          NMSDLY .BYTE 3           ; MS TO DELAY AFTER EACH BYTE
6011  FF          EORVAL .BYTE $FF         ; VALUE TO EOR WITH SCREEN LOC ($FF TO
FLIP, 0 TO SKIP)
6012  90          VMPAGE .BYTE $90         ; STARTING PAGE OF VISIBLE MEMORY
6013              ;
6013  00 00       VM     .WORD 0           ; LOCAL STORAGE - LOC OF A VIS MEM ROW
6015  28          BYTEPL .BYTE 40          ; BYTES PER VM LINE
6016  00          RREPX  .BYTE 0           ; STORAGE FOR REP COUNT
6017  00          RVALX  .BYTE 0           ; STORAGE FOR ROW COUNT
6018              ;
6018              ; OUTVM - OUTPUT THE WHOLE VISIBLE MEMORY
6018              ;
6018  AD 12 60    OUTVM  LDA VMPAGE        ; SET LOC OF VM
601B  8D 14 60           STA VM+1
601E  A9 00              LDA #0
6020  8D 13 60           STA VM
6023  AD 0B 60           LDA RREP         ; SET MAJOR REP COUNT
6026  8D 16 60           STA RREPX
6029  20 88 60           JSR ENTRGR       ; ENTER GRAPHICS MODE
602C  AE 0A 60           LDX RPFXR         ; SEE IF ANY PREFIX ROWS
602F  F0 06              BEQ OUTVM1
6031  20 C5 60    OUTVM0 JSR OUTRET        ; IF SO, OUTPUT THEM
6034  CA                 DEX
6035  D0 FA              BNE OUTVM0
6037              ;
6037              OUTVM1 = *
6037  20 BA 61           JSR STOPTS       ; CHECK FOR STOP KEY
603A  B0 3A              BCS OUTVMF
603C  20 B1 60           JSR OUTPFX       ; OUTPUT A LINE P
603F  AD 0C 60           LDA RVAL         ; SET ROW COUNT
6042  8D 17 60           STA RVALX
6045  20 8D 60           JSR OUTROW       ; OUTPUT A ROW
6048  20 C5 60           JSR OUTRET       ; OUTPUT A RETURN
604B  AE 0C 60           LDX RVAL         ; SET VM = VM+RVAL*40
604E  18          OUTVM2 CLC
604F  AD 13 60           LDA VM
6052  6D 15 60           ADC BYTEPL
6055  8D 13 60           STA VM
6058  AD 14 60           LDA VM+1
605B  69 00              ADC #0
605D  8D 14 60           STA VM+1
6060  CA                 DEX
6061  D0 EB              BNE OUTVM2
6063              ;
6063  CE 16 60           DEC RREPX        ; COUNT ROWS
6066  D0 CF              BNE OUTVM1
6068              ;
6068  20 B1 60           JSR OUTPFX       ; START OF LAST ROW
606B  AD 0D 60           LDA REND         ; NUMBER OF ROWS
606E  F0 06              BEQ OUTVMF        ; SKIP IF NONE
6070  8D 17 60           STA RVALX
6073  20 8D 60           JSR OUTROW       ; THERE IT GOES
6076              OUTVMF = *
6076  20 C5 60           JSR OUTRET
6079  20 7D 60           JSR EXITGR        ; LEAVE GRAPHICS MODE
607C  60                 RTS               ; DONE
607D              ;
607D              ; EXITGR - LEAVE GRAPHICS MODE
607D              ;
607D  A9 03       EXITGR LDA #3
607F  20 6D 61           JSR OUTCH
6082  A9 02              LDA #2
6084  20 6D 61           JSR OUTCH
6087  60                 RTS
6088              ;
6088              ; ENTRGR - ENTER GRAPHICS MODE
6088              ;
6088  A9 03       ENTRGR LDA #3
608A  4C 6D 61           JMP OUTCH
608D              ;
608D              ; OUTROW - OUTPUT THE ROW POINTED TO BY VM, RVALX DEEP
608D              ;
608D  AD 13 60    OUTROW LDA VM            ; SET WHERE TO START
6090  8D D1 60           STA V
```

```
6093   AD 14 60             LDA VM+1
6096   8D D2 60             STA V+1
6099   AE 15 60             LDX BYTEPL       . DO 40 COLUMNS
609C   AD 17 60    OUTR1    LDA RVALX        ; SET DEPTH COUNT
609F   8D D3 60             STA R
60A2   20 DF 60             JSR OUTCOL       ; OUTPUT THOSE 8
60A5   EE D1 60             INC V            ; BUMP LOC
60A8   D0 03                BNE OUTR2
60AA   EE D2 60             INC V+1
60AD   CA          OUTR2    DEX              ; COUNT
60AE   D0 EC                BNE OUTR1
60B0   60                   RTS              ; DONE
60B1               ;
60B1               ; OUTPFX - OUTPUT RPFXC SPACES TO START LINE
60B1               ;
60B1   AE 09 60    OUTPFX LDX RPFXC
60B4   F0 0E                BEQ OUTPF2       ; CHECK FOR NONE
60B6   20 7D 60             JSR EXITGR
60B9   A9 20                LDA #32          ; LOAD THE SPACE
60BB   20 6D 61    OUTPF1 JSR OUTCH          ; OUTPUT IT
60BE   CA                   DEX              ; UNTIL DONE
60BF   D0 FA                BNE OUTPF1
60C1   20 88 60             JSR ENTRGR
60C4   60          OUTPF2 RTS                ; THAT'S ALL
60C5               ;
60C5               ; OUTRET - OUTPUT A GRAPHICS RETURN
60C5               ;
60C5   A9 03       OUTRET LDA #3
60C7   20 6D 61             JSR OUTCH
60CA   AD 0F 60             LDA RXGR
60CD   20 6D 61             JSR OUTCH
60D0   60                   RTS
60D1               ;
60D1               ; OUTCOL - OUTPUT 8 COLUMNS OF BITS
60D1               ;
60D1               ; PARAMETERS (BELOW) ARE V AND R
60D1               ;
60D1   00 00       V        .WORD 0          ; LOC IN VISIBLE MEMORY
60D3   00          R        .BYTE 0          ; NUMBER OF ROWS TO PROCESS
60D4               PO       * = *+8          ; RESULT TO OUTPUT
60DC               ;
60DC   00          P2       .BYTE 0          ; POWER OF 2 BIT
60DD   00          B        .BYTE 0          ; STORAGE FOR A BYTE
60DE   00          M        .BYTE 0          ; A MASK
60DF               ;
60DF               PGZ      = 1              ; PAGE ZERO LOCATION TO USE
60DF               ;
60DF   48          OUTCOL PHA                ; SAVE REGS
60E0   8A                   TXA
60E1   48                   PHA
60E2   98                   TYA
60E3   48                   PHA
60E4   A5 01                LDA PGZ          ; SAVE PAGE ZERO AREA
60E6   48                   PHA
60E7   A5 02                LDA PGZ+1
60E9   48                   PHA
60EA   20 BA 61             JSR STOPTS       ; SEE IF STOP PRESSED
60ED   B0 72                BCS MOVEPF       ; IF SO, QUIT NOW
60EF   A9 00                LDA #0           ; ZERO P(0:7)
60F1   A2 07                LDX #7
60F3   9D D4 60    CLP2     STA PO,X
60F6   CA                   DEX
60F7   10 FA                BPL CLP2
60F9
60F9   A9 01                LDA #1           ; SET P2 TO 1
60FB   8D DC 60             STA P2
60FE   AD D1 60             LDA V            . STORE VM LOC
6101   85 01                STA PGZ
6103   AD D2 60             LDA V+1
6106   85 02                STA PGZ+1
6108   A0 00       RLOOP    LDY #0
610A   B1 01                LDA (PGZ)Y       ; GET VM BYTE
610C   8D DD 60             STA B            ; SAVE IT
610F   4D 11 60             EOR EORVAL       ; REVERSE IT FOR SHOW
6112   91 01                STA (PGZ)Y
6114   A5 01                LDA PGZ          ; POINT TO NEXT ROW
6116   18                   CLC
6117   6D 15 60             ADC BYTEPL
```

Get Fireworks From Your PET*!

TREK-X

TREK-X Command the Enterprise as you scour the quadrant for enemy warships. This package not only has superb graphics, but also includes programming for optional sound effects. A one-player game for the PET 8K. **Order No. 0032P $7.95.**

DUNGEON OF DEATH

DUNGEON OF DEATH Battle evil demons, cast magic spells, and accumulate great wealth as you search for the Holy Grail. You'll have to descend into the Dungeon of Death and grope through the suffocating darkness. If you survive, glory and treasure are yours. For the PET 8K. **Order No. 0064P $7.95.**

ARCADE I

ARCADE I This package combines an exciting outdoor sport with one of America's most popular indoor sports:

•**Kite Fight** – It's a national sport in India. After you and a friend have spent several hours maneuvering your kites across the screen of your PET, you'll know why!

•**Pinball** – By far the finest use of the PET's exceptional graphics capabilities we've ever seen, and a heck of a lot of fun to boot.

Requires an 8K PET. **Order No. 0074P $7.95.**

CODE NAME: CIPHER

CODE NAME:CIPHER

Enjoy that same feeling of intrigue and discovery with the Code Name: Cipher package. Included are:

•**Memory Game** – Would you like to match your memory against the computer's? You can with the Memory Game.

•**Codemaster** – One player types in a word, phrase, or sentence, and the PET translates that message into a cryptogram. The other player must break the code and solve the cryptogram in the shortest time possible.

•**Deceitful Mindmaster** – This isn't your ordinary Mastermind-type game. You must guess the five letters in the hidden code word.

•**Code Breaker** – Cracking this code won't be as easy as cracking walnuts. You'll need to flex your mental muscles to win this game.

If you want a mental challenge, then Code Name: Cipher is for you. For the 8K PET. **Order No. 0112P. $7.95.**

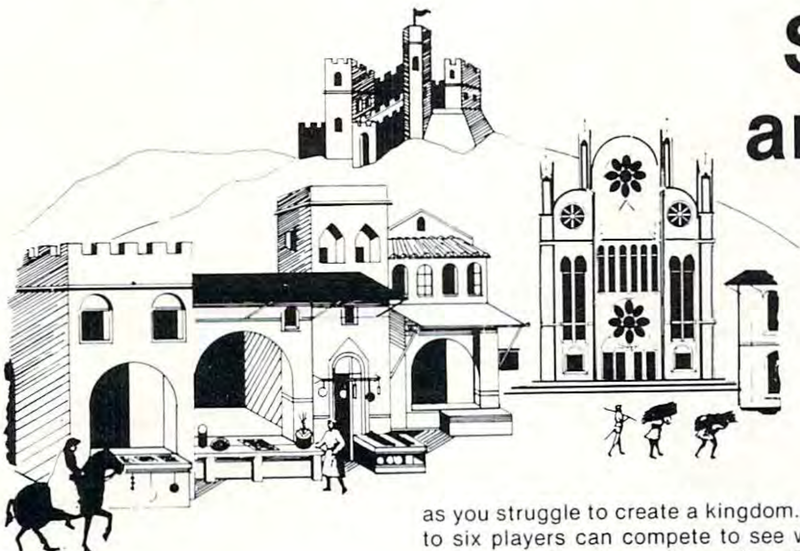
Instant Software™

*A trademark of Commodore Business Machines

PETERBOROUGH, N.H. 03458
603-924-7296

Captivate Yourself.

Santa Paravia and Fiumaccio



SANTA PARAVIA AND FIUMACCIO Become the ruler of a medieval city-state as you struggle to create a kingdom. Up to six players can compete to see who will become the King or Queen first. This program requires a PET 16K. **Order No. 0175P.$9.95.**

The most captivating and engrossing program ever made for the PET*

It is the dawn of the 15th Century; you rule a tiny Italian city-state. Your goal: The Crown!

Up to six players can compete as rulers of neighboring cities. You control the grain harvest, feed your serfs, set tax rates, dispense justice and invest in public works.

The future of your realm will depend on your decisions. If they are wise, your city-state will grow and you will acquire loftier titles. If your rule is incompetent, your people will starve and you may be invaded by your neighbors.

How will *you* rule your kingdom? Will you be an enlightened leader—or an unscrupulous despot? Only you can answer that question—with Santa Paravia and Fiumaccio.

CHIMERA

CHIMERA If you think the legendary Chimera was hard to handle, wait until you try the Chimera package. Included are:

- **Reflex**—Round and round the little white ball rolls. Only fast reflexes can guide it into the center of the maze.
- **Dragon**—You'll have to shoot down those pesky, fire-breathing dragons with your cannon. If you succeed your castle will be safe, if not it will mean a call to your fire insurance company. For one player.
- **Dungeon**—A very punctual guard comes down to the dungeon every day to torture you. This means that you have only thirty seconds to dig your way under the castle to freedom. For one player.
- **Dragon Hunt**—You must go forth and slay a fire-breathing dragon. The only thing that will protect you from the flames is your shield, if you know when to use it. For one player.
- **Dropoff**—You must make your opponent's men "dropoff" the board by moving and firing your own men. For one or two players. **Order No. 0110P. $9.95.**

PET DEMO I

PET DEMO I You can give yourself, your family, and your friends hours of fun and excitement with this gem of a package.

- **Slot Machine**—You won't be able to resist the enticing messages from this computerized one-armed bandit.
- **Chase**—You must find the black piece as you search through the ever-changing maze.
- **Flying Pheasant**—Try to shoot the flying pheasant on the wing.
- **Sitting Ducks**—Try to get your archer to shoot as many ducks as possible for a high score.
- **Craps**—It's Snake Eyes, Little Joe, or Boxcars as you roll the dice and try to make your point.
- **Gran Prix 2001**—Drivers with experience ranging from novice to professional will enjoy this multi-leveled race game.
- **Fox and Hounds**—It's you against the computer as your four hounds try to capture the computer's fox.

For true excitement, you'll need a PET 8K. **Order No. 0035P $7.95.**

DOW JONE$

DOW JONES Up to six players can enjoy this exciting stock market game. You can buy and sell stock in response to changing market conditions. Get a taste of what playing the market is all about. Requires a PET with 8K. **Order No. 0026P $7.95.**

We Guarantee It!

Instant Software Guarantee

OUR PROGRAMS ARE GUARANTEED TO BE QUALITY PRODUCTS. IF NOT COMPLETELY SATISFIED YOU MAY RETURN THE PROGRAM WITHIN 60 DAYS. A CREDIT OR REPLACEMENT WILL BE WILLINGLY GIVEN FOR ANY REASON.

Prices subject to change without notice.

TO ORDER

SEE YOUR LOCAL INSTANT SOFTWARE DEALER

OR

call now Toll-Free 1-800-258-5473

Instant Software™

*A trademark of Commodore Business Machines

PETERBOROUGH, N.H. 03458
603-924-7296

```
611A  85 01            STA PGZ
611C  A5 02            LDA PGZ+1
611E  69 00            ADC #0
6120  85 02            STA PGZ+1
6122           ;
6122  A9 01            LDA #1          ; SET MASK TO 1
6124  8D DE 60         STA M
6127  A2 07            LDX #7           ; FOR I = 7 TO 0 STEP -1
6129  AD DD 60  ILOOP  LDA B            ; IF B AND M () 0
612C  2D DE 60         AND M
612F  F0 09            BEQ ILOOP1
6131  BD D4 60         LDA PO,X        ; P(I)=P(I)+P2
6134  0D DC 60         ORA P2
6137  9D D4 60         STA PO,X
613A  0E DE 60  ILOOP1 ASL M            ; SHIFT MASK LEFT
613D  CA               DEX             ; SEE IF DONE
613E  10 E9            BPL ILOOP
6140           ;
6140  0E DC 60         ASL P2          ; DOUBLE P2
6143  CE D3 60         DEC R           ; SEE IF OUTER LOOP DONE
6146  D0 C0            BNE RLOOP
6148           ;
6148           ; OUTPUT PO(0:7)
6148           ;
6148  A0 00            LDY #0
614A  20 BA 61  MOVEP  JSR STOPTS       ; SEE IF STOP PRESSEDD
614D  B0 12            BCS MOVEPF      ; IF SO, QUIT HERE
614F  B9 D4 60         LDA PO,Y
6152  20 6D 61         JSR OUTCH       ; OUTPUT A CHARACTER
6155  C9 03            CMP #3          ; SEE IF 3
6157  D0 03            BNE MOVEP1
6159  20 6D 61         JSR OUTCH       ; IF SO, DO IT AGAIN
615C  C8        MOVEP1 INY
615D  C0 08            CPY #8          ; ONLY DO 8
615F  D0 E9            BNE MOVEP
6161           ;
6161  68        MOVEPF PLA              ; RESTORE PAGE ZERO AREA
6162  85 02            STA PGZ+1
6164  68               PLA
6165  85 01            STA PGZ
6167  68               PLA              ; RESTORE REGS
6168  A8               TAY
6169  68               PLA
616A  AA               TAX
616B  68               PLA
616C           ;
616C  60               RTS
616D           ;
616D           ; OUTCH - OUTPUT A CHARACTER TO DEVICE RDEV
616D           ;
616D           ; THIS ROUTINE WAS SUPPLIED BY GREG YOB - THANKS MUCH
616D           ;
616D  8E B7 61  OUTCH  STX OUTCHX        ; SAVE REGS
6170  8C B8 61         STY OUTCHY
6173  48               PHA
6174  A5 D4            LDA $D4          ; SAVE CURRENT DEVICE
6176  8D B9 61         STA TMPDEV
6179  AD 0E 60         LDA RDEV        ; SET MY DEVICE
617C  85 D4            STA $D4
617E  20 BA F0         JSR $F0BA       ; LISTEN
6181  20 2D F1         JSR $F12D       ; ATTENTION
6184            OUTCH1 = *
6184  20 BA 61         JSR STOPTS       ; SEE IF STOP PRESSED
6187  B0 11            BCS OUTCH2       ; IF SO, EXIT FROM HERE
6189  A9 00            LDA #0          ; CLEAR STATUS
618B  85 96            STA $96
618D  68               PLA             ; REGET CHAR
618E  48               PHA
618F  85 A5            STA $A5         , STORE WHERE IT SHOULD BE
6191  20 EE F0         JSR $F0EE       ; OUTPUT
6194  A5 96            LDA $96         ; SEE IF TIMED OUT
6196  25 01            AND 1
6198  D0 EA            BNE OUTCH1      ; IF SO, TRY AGAIN
619A            OUTCH2 = *
619A  20 83 F1         JSR $F183       ; UNLISTEN
619D  AD B9 61         LDA TMPDEV      ; RESTORE DEVICE
61A0  85 D4            STA $D4
61A2  AE 10 60         LDX NMSDLY       ; DELAY A FEW MS
```

```
00266  61A5  F0 08                BEQ OUTCHF
00267  61A7  A0 C8         OUTCH3 LDY #200          ; 1 MS INNER LOOP (LOOP IS 5 CYCLES LONG)
00268  61A9  88            OUTCH4 DEY
00269  61AA  D0 FD                BNE OUTCH4
00270  61AC  CA                   DEX
00271  61AD  D0 F8                BNE OUTCH3
00272  61AF                OUTCHF = *
00273  61AF  AE B7 61             LDX OUTCHX       ; AND REGS
00274  61B2  AC B8 61             LDY OUTCHY
00275  61B5  68                   PLA
00276  61B6  60                   RTS              ; DONE
00277  61B7                ;
00278  61B7  00            OUTCHX .BYTE 0
00279  61B8  00            OUTCHY .BYTE 0
00280  61B9  00            TMPDEV .BYTE 0
00281  61BA                ;
00282  61BA                ; STOPTS - SET CARRY IF STOP KEY PRESSED
00283  61BA                ;
00284  61BA                ; TAKEN FROM PAGE 5 OF MTU DOCUMENTATION FOR K-1002-6C
00285  61BA                ;
00286  61BA  AD 12 E8      STOPTS LDA $E812         ; LOOK AT KEYBOARD
00287  61BD  C9 EF                CMP #$EF         ; TEST FOR STOP KEY
00288  61BF  18                   CLC              ; CARRY CLEAR FOR NO STOP
00289  61C0  D0 01                BNE STOPT1
00290  61C2  38                   SEC              ; CARRY SET FOR YES STOP
00291  61C3  60            STOPT1 RTS
00292  61C4                ;
00293  61C4                       .END
ERRORS = 00000
SYMBOL TABLE
SYMBOL VALUE
 B        60DD    BYTEPL   6015    CLP2     60F3    ENTRGR   6088
 EORVAL   6011    EXITGR   607D    ILOOP    6129    ILOOP1   613A
 M        60DE    MOVEP    614A    MOVEP1   615C    MOVEPF   6161
 NMSDLY   6010    OUTCH    616D    OUTCH1   6184    OUTCH2   619A
 OUTCH3   61A7    OUTCH4   61A9    OUTCHF   61AF    OUTCHX   61B7
 OUTCHY   61B8    OUTCOL   60DF    OUTPF1   60BB    OUTPF2   60C4
 OUTPFX   60B1    OUTR1    609C    OUTR2    60AD    OUTRET   60C5
 OUTROW   608D    OUTVM    6018    OUTVM0   6031    OUTVM1   6037
 OUTVM2   604E    OUTVMF   6076    P2       60DC    PGZ      0001
 PO       60D4    R        60D3    RDEV     600E    REND     600D
 RLOOP    6108    RPFXC    6009    RPFXR    600A    RREP     600B
 RREPX    6016    RVAL     600C    RVALX    6017    RXGR     600F
 STOPT1   61C3    STOPTS   61BA    TMPDEV   61B9    V        60D1
 VM       6013    VMPAGE   6012
END OF ASSEMBLY
```

```
CROSS REFERENCE......PAGE   1

B       $60DD   152   182   196
BYTEPL  $6015    42    72   112   187
CLP2    $60F3   170   172
ENTRGR  $6088    54   103   132
EORVAL  $6011    38   183
EXITGR  $607D    90    95   127
ILOOP   $6129   196   204
ILOOP1  $613A   198   202
M       $60DE   153   194   197   202
MOVEP   $614A   213   222
MOVEP1  $615C   218   220
MOVEPF  $6161   167   214   224
NMSDLY  $6010    37   265
OUTCH   $616D    96    98   104   129
                138   140   216   219   240
OUTCH1  $6184   249   260
OUTCH2  $619A   251   261
OUTCH3  $61A7   267   271
OUTCH4  $61A9   268   269
OUTCHF  $61AF   266   272
OUTCHX  $61B7   240   273   278
OUTCHY  $61B8   241   274   279
OUTCOL  $60DF    21   115   157
OUTPF1  $60BB   129   131
OUTPF2  $60C4   126   133
OUTPFX  $60B1    64    83   125
OUTR1   $609C   113   120
OUTR2   $60AD   117   119
OUTRET  $60C5    57    68    89   137
OUTROW  $608D    20    67    87   108
OUTVM   $6018    19    48
OUTVM0  $6031    57    59
OUTVM1  $6037    56    61    81
OUTVM2  $604E    70    78
OUTVMF  $6076    63    85    88
P2      $60DC   151   175   200   206
PGZ     $0001   155   162   164   177   179
                181   184   185   188   189
                191   225   227
PO      $60D4   149   170   199   201   215
R       $60D3   114   148   207
RDEV    $600E    35   245
REND    $600D    33    84
RLOOP   $6108   180   208
RPFXC   $6009    29   125
RPFXR   $600A    30    55
RREP    $600B    31    52
RREPX   $6016    43    53    80
RVAL    $600C    32    65    69
RVALX   $6017    44    66    86   113
RXGR    $600F    36   139
STOPT1  $61C3   289   291
STOPTS  $61BA    62   166   213   250   286
TMPDEV  $61B9   244   263   280
V       $60D1   109   111   116   118   147
                176   178
VM      $6013    41    49    51    71    73
                 74    76   108   110
VMPAGE  $6012    39    48
```

Machine Language: Getting To The Machine Language Program

Jim Butterfield
Toronto, Canada

Your PET/CBM is a Basic machine. To run machine language you have to leave Basic – perhaps for a temporary period – and enter the machine language program. You'll often want Basic and Machine Language to work together. Where time is not critical, many things code easily into Basic. But where speed is important, or the job is beyond Basic's normal powers, you'll want to use machine language inserts. At that time, your computer will want to go into machine language.

There are four standard methods of doing this: some are more complex than others. Each has its own advantages and drawbacks.

The SYS command and the USR function call machine language whenever Basic desires to do so. This may be done with a direct command or from a program. The machine language program acts as a subroutine, and may return to the Basic calling point when it has done the job.

The more complex "wedge" method calls a machine language routine frequently whenever Basic is running. It doesn't wait for the Basic program to call it in; it seems to run simultaneously with Basic.

The interrupt method taps the PET's internal interrupt scheme. Every sixtieth of a second – whether Basic is running or not – PET's interrupt kicks in and does a number of quick jobs, such as checking the keyboard and flashing the cursor. Machine language programs which tap the interrupt seem to run continuously, even when Basic is not active.

The Machine Language Monitor has a Go (.G) command which allows you to start a machine language program directly. The program is not called as a subroutine, so it must find its own way back to the MLM when it is finished.

Each of the four methods will be discussed briefly here.

SYS And USR

SYS and USR create direct calls from Basic to a machine language program. This program runs only when called, and when it is finished it will hopefully return control to Basic and allow Basic execution to continue.

SYS is a command. You say SYS 7143, for example, as a direct command or within a program, and machine language at decimal address 7143 will start executing. SYS is quite convenient when you have several machine language programs to be run at different times: you just give the address of each one as you call it.

USR is a function, not a command. You cannot say USR(0) alone any more than you can say SQR(0): it must be part of a command. You might say any of: PRINT USR(0); X=USR(99); IF USR(7)=3 THEN ... or any similar syntax.

When Basic encounters the USR function within a Basic statement, it will start to execute machine language at a present address. Hopefully you will have set the address to point at the program you want to run; you do this by POKEing the desired address into locations 1 and 2. Once you've done this, USR will fire you into the desired machine code every time you use it.

The argument of the USR function — that's the value enclosed in brackets — is available to the machine language program if it wants to use it. This value may be found in the floating point accumulator, which is at hexadecimal B0 to B45 in original ROMs or at 5E to 63 in subsequent PET/CBM machines. It's store in floating point notation, which is devilishly hard to read if you don't know the system and not that easy if you do. When a simmple number like 5 comes up as hexadecimal 83 A0 00 00 00 20 you may be happy to reach for a built-in conversion routine that yields a much more readable fixed-point value of 00 05.

If you use the USR argument you may also leave a value in the same floating point accumulator just before you return to Basic. This value will be picked up by Basic as the value computed by the USR function.

To summarize: SYS lets you pick any of several machine language programs. USR takes you to a fixed location and allows you to pass a single value to and from machine language if you want. The SYS command seems simpler to the beginner, but USR is also straightforward once you get used to it.

The Wedge, Or Infiltrating Basic

This advanced technique gives the effect of a Basic "supervisor" which watches Basic run and occasionally kicks in with some of its own activities. It doesn't need to be called from Basic: once implanted, it will be there and active any time Basic is running.

It's a powerful method of extending Basic. Many systems use it: disk monitors, including the Commodore "wedge" DOS; Basic enhancers such as the Programmer's Toolkit or Basic Aid; and Brett Butler's TRACE as published in **COMPUTE!**,

issue 1.

How does it work? It's done by infiltrating a Basic subroutine called CHRGET which is located in page zero. This subroutine is called every time the Basic interpreter wants to get a character from your Basic program. By making very careful changes to this subroutine, you can force the Basic interpreter to do a little extra work for you.

It's not simple. But with a little persistence and a lot of bravery, you can train Basic to do some clever new tricks.

Interrupt

Sixty times a second, PET's normal activity freezes. An interrupt signal causes a completely independent program to run. When the interrupt program completes, the computer's normal programs unfreeze and continue exactly where they left off.

This powerful mechanism allows PET to do several important jobs. The jiffy clock is updated; the keyboard is checked for activity; the stop key is checked and its condition logged; the cursor is flashed when necessary; and the cassette motors are started or stopped. All of this is invisible to the main program, which clanks along happily without even noticing the interruptions.

The interrupt mechanism works all the time, even when Basic isn't running. If you add your own machine language program to the interrupt sequences, it too will work all the time — sixty times a second. It's ideal for watching special input/output ports, flashing parts of the screen, and similar jobs.

You can get at the interrupt routine quite easily. There is a memory location called the Hardware Interrupt Vector: in original ROMs, it's at hexadecimal 0219 and 021A; in new ROM systems it's at hex 0090 and 0091. In either case, the locations contain an address which points to the interrupt routine. If you change the address, the interrupt mechanism will go wherever you say, sixty times a second. At the end of your coding, don't forget to jump to the regular interrupt program so that the keyboard, clock, etc. still work properly.

Changing the address of the Hardware Interrupt vector has a small problem. Like all addresses, it comes in two chunks: a low order byte and a high order byte. If you have just changed the low order part and are about to change the second part when the interrupt strikes, you have a disaster on your hands. The address that the interrupt finds at that moment will be nonsense — part old address and part new.

Avoid this problem by making use of the SEI (Set Interrupt disable) instruction to lock out the interrupt while you are changing the vector. Don't forget to restore the interrupt with a CLI (Clear Interrupt disable) when you've finished putting the address in place.

It seems odd, but cassette tape can neither read or write after you have changed the interrupt vector from its usual address; and LOADs from disk may "hang" without saying READY. Be sure to make provision to restore the vector if you do much input or output.

Machine Language Monitor

In the Machine Language Monitor, you can type .G for Go and go directly to any machine language program you like. You will go with a direct jump (JMP) command, which means that the program is not treated as a subroutine. You can't get back with a return (RTS) instruction; instead, you will likely use a Break (BRK) command to reconnect with the monitor.

The Go command and associated BRK instructions are useful in debugging programs. After your program is written, replace several of the instructions in your program with Break commands. Try to scatter the Break commands evenly throughout your program, especially at the start of logical program "modules". Now perform Go to the start of your program. You should come back to the monitor almost instantly with the first Break point. If so, you've reached that program step safely; replace the Break instruction with the command that originally belonged there. Now you can Go to that address, and the program will resume and continue to the next Break. As you go through the program piece by piece, check that the registers contain the values you expect; if appropriate, check key memory locations, too.

If the PET misbehaves or goes terribly quiet, at least you will have isolated the portion of the program that is doing it to you. On the next test, you can set your break points closer together in that area, and pin the problem down step by step.

Summary

There are several ways to link your PET to machine language programs. Beginners will want to stay with the SYS command and the USR function until they have gained confidence. They should learn the Machine Language Monitor (.G) and Break (BRK) functions as quickly as possible to help in checking out programs.

The advanced functions — wedge and interrupt — will be there when they are needed. ©

Odds & Ends on the 2040 Disk

Jim Butterfield

The disk has almost more brains than the PET. It contains two separate microprocessors, each of which has its own ROM program; the micros talk to each other via a shared block of memory.

* * * * * * * * * *

80 COLUMN GRAPHICS

The Integrated Visible Memory for the PET has now been redesigned for the new 12" screen 80 column and forthcoming 40 column PET computers from Commodore. Like earlier MTU units, the new K-1008-43 package mounts inside the PET case for total protection. To make the power and flexibility of the 320 by 200 bit mapped pixel graphics display easily accessible, we have designed the Keyword Graphic Program. This adds 45 graphics commands to Commodore BASIC. If you have been waiting for easy to use, high resolution graphics for your PET, isn't it time you called MTU?

K-1008-43M Manual only $10 (credited toward purchase)
k-1008-43 Complete ready to install package $495

Mastercharge and Visa accepted

Write or call today for our full line catalog describing all MTU 6502 products, including our high speed 8" Floppy Disk Controller for up to 4 megabytes of PET storage.

The image on the screen was created by the program below.

```
 10 VISMEM: CLEAR
 20 P=160: Q=100
 30 XP=144: XR=1.5*3.1415927
 40 YP=56: YR=1: ZP=64
 50 XF=XR/XP: YF=YP/YR: ZF=XR/ZP
 60 FOR ZI=-Q TO Q-1
 70 IF ZI<-ZP OR ZI>ZP GOTO 150
 80 ZT=ZI*XP/ZP: ZZ=ZI
 90 XL=INT(.5+SQR(XP*XP-ZT*ZT))
100 FOR XI=-XL TO XL
110 XT=SQR(XI*XI+ZT*ZT)*XF: XX=XI
120 YY=(SIN(XT)+.4*SIN(3*XT))*YF
130 GOSUB 170
140 NEXT XI
150 NEXT ZI
160 STOP
170 X1=XX+ZZ+P
180 Y1=YY-ZZ+Q
190 GMODE 1: MOVE X1,Y1: WRPIX
200 IF Y1=0 GOTO 220
210 GMODE 2: LINE X1,Y1-1,X1,0
220 RETURN
```

MTU
Micro Technology Unlimited
2806 Hillsborough Street
P.O. Box 12106
Raleigh, NC 27605, U.S.A.
(919) 833-1458

NOW 80 COLUMN PETS CAN HAVE MTU HIGH RESOLUTION GRAPHICS

A Thirteen Line BASIC Delete

Arthur C. Hudson
Ottawa, Ontario

Here is a short program written entirely in BASIC, which allows you to delete any group of lines from an existing program. Typically the increment is 1, so that all lines in the group are deleted, but this is not necessary.

To use the Basic Delete, just screen merge it with your existing program. Hopefully no conflict of line numbers will occur, if there is conflict, then some renumbering will be required. After the merge, RUN7878, and as instructed, modify the listed line 7892 to define the start, the end and the increment. Then press return twice, and the delete process will begin. The line number being deleted is displayed and you may press BREAK (RUN/STOP) at any time.

As an example of using an increment other than unity, you could write all or part of a program using even numbers for the useful statements and odd numbers for the remarks. Save on tape or disk, and then automatically delete all the remarks and save again. Finally the original can be brought back, and all even numbered statements deleted. This gives a program consisting only of the remarks. Each of these three versions can have its uses.

Somewhat complicated programming techniques are used here, and the statements must be entered carefully. Note that after you have modified the automatically listed line 7896 and pressed RETURN, the cursor will rest on a direct statement, RUN7882. In this way, a second RETURN will initiate the delete process.

The program uses the dynamic keyboard feature of the PET. (See **COMPUTE!** Issue 4 page 58 and the earlier reference - Louder - cited therein). It uses bins 834 and 835 in the second cassette buffer, but this does not prevent use of the second cassette.

One of the more interesting problems in this type of program is that PET suffers from amnesia the moment it executes a delete (all variables set to 0). It is for this reason that parameters have to be embedded in a program statement, and also N, the number of the line currently being deleted, must be poked into memory before the deletion and retrieved after it.

Note that in line 7892 the word 'INCREMENT' is spelled incorrectly. Don't try to fix it, or PET will see thé word REM inside it and bomb out. Don't think you can get away with substituting 'step' for 'incrment', because PET will object to the use of ST, a reserved word. Finally don't try incr'ment, PET doesn't like this either, (not alphanumeric).

The first time that the Basic Delete is used, the asterisks in SN7896 will be replaced by numbers. There is of course no need to replace the asterisks when executing a SAVE.

My version of this program uses about 330 bytes. It is certainly possible to trim this down by about 50 bytes.

If You Have OLD ROM

Referring to statements numbered 7884 and 7886; for 623 and 624 substitute 527 and 528. For 158 substitute 525. These bins relate to the keyboard buffer. Note that Harvey Davis's article is written for old ROM, so the conversions given above apply in reverse, if you have new ROM.

Reference:
Algebraic Input for the PET, Harvey Davis, **COMPUTE!** Vol. 1, Issue 4, page 58.

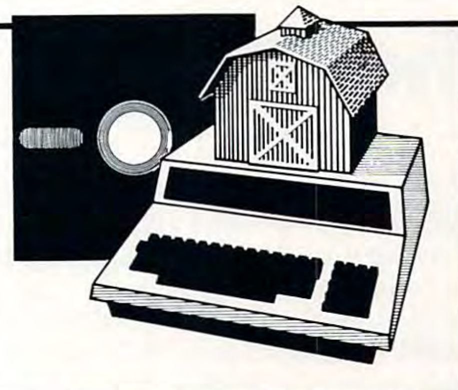
```
 10 PRINT"ĥ↓A THIRTEEN LINE BASIC DELETE
 12 PRINT"↓ARTHUR C. HUDSON
 14 PRINT"↓11 AMBERLY PLACE
 16 PRINT"↓OTTAWA,ONT.
 18 PRINT"↓CANADA
 20 PRINT"↓K1J 7J9
 22 PRINT"↓PHONE (613) 749 5475
 30 PRINT"↓↓KEY IN CONT":STOP
 7878 PRINT"ĥ→MODIFY SN7896,THEN CR
 7880 PRINT"↓↓RUN7882↑↑↑":LIST7896
 7882 POKE835,0:POKE834,0:GOTO7894
 7884 POKE623,13:POKE624,13
 7886 POKE158,2:PRINT"ĥ↓↓↓GOTO7894
 7888 PRINT"h↓↓"N"↑↑↑";:N=N+IN
 7890 D=INT(N/256):POKE835,D
 7892 POKE834,N-D*256:END
 7894 N=256*PEEK(835)+PEEK(834)
 7896 FIRST=0000:LAST=0000:INCRMENT=01
 7898 IF N > LA THEN STOP
 7900 IF N < FI THEN N=FI
 7902 GOTO7884
READY.
```

©

Agricultural Software from Cyberia™

FOR PET/CBM SYSTEMS

Cyber-Farmer™ The complete accounting package for today's farm operations



- Records, sorts, combines and prints the results of the farm operation.
- Account headings and numbers are pre-assigned for nearly every type of farm income or expense, but any account may be deleted, altered or added.
- Keeps personal, family and house-hold accounts as well.
- Cyber-Farmer management tools include cash-flow analyses, depreciation schedules, inventory and budget reports.
- No computer experience is necessary to operate this system.

Farrow-Filer™

Record-keeping and management system for the pork producer

- Helps the pork producer to lower their cost-per-pig by producing more animals.
- High visual impact of reports allows for better culling of freeloaders.
- Alerts the operator to possible medical problems in the herd.
- Forecasts farrowings for a 3 month period.
- Keeps individual records on each animal.

Farmer's Workbook™

The most powerful management tool ever for the agricultural producer. Farmers Workbook combines the power of Visicalc® * with the knowledge of a major midwestern university. The Farmers Workbook is a collection of templates that are designed to be run on the Visicalc® program. The templates include lables, formulas, sample data, test cases and full documentation. Template titles include: Cattle Feeder, Pig Production, Sheep Production, Grain Management, Loan Payments, Market Average, Land Purchase and many others.

*Visicalc is a trademark of Personal Software, Inc.

Also...Two Other Enhancements for PET/CBM Systems

SuperBus™ Greatly multiplies system capabilities

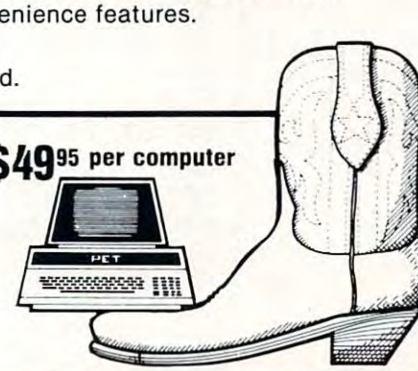
$195 per computer

This is an active, integrated system—not just a passive network. SuperBus gives schools, banks, laboratories and businesses control of the way their networks operate.

- Up to 18 computers, disk drives or printers can be interconnected.
- Complete file security (program and data) and BASIC security.
- Built-in error detection and convenience features.
- Can both read and write to disk.
- All BASIC commands can be used.

Auto-Boot™ Simplifies PET/CBM operation

$49.95 per computer



- Auto-Boot is a ROM that automatically loads and runs the first program on the disk (initializing if necessary).
- Completely compatible with most other programs.
- Just insert the disk in the drive, turn on the computer, and Auto-Boot does the rest.

MasterCard, VISA and C.O.D. orders accepted. Specify computer model when ordering SuperBus or Auto Boot. Dealer inquiries invited.

CYBERIA 515-292-7634
2330 LINCOLN WAY, AMES, IOWA 50010

Calculated Bar-graph Routines On The PET

Edward F. Heite
Camden-Wyoming, Delaware

To exploit the CBM printer's graphic potentials, programmers need a few routines that haven't been published yet. The "Keyprint" program (**COMPUTE!**, issue 7, page 84) is okay for dumping 40 columns to the printer, if you have the right ROM. But if you want to manipulate the full 80 columns and create complicated graphs, you need a way to calculate the length of the bar.

A calculated bar can be created as a string variable, by concatenating a graphic string to the desired length with a FOR ... NEXT loop. Listing 1 is a dummy program to demonstrate this process.

Line 1 sets B$ to an empty value. Line 2 defines C$ as a single graphic character. In line 3, the value of the bar is set at 20; in actual programs, this would be a calculated value. Line 4 sets the FOR ... NEXT loop to the value of J, and thus determines the length of the bar. Line 5 concatenates B$–C$ to create a new value for B$. Line 5 keeps adding symbols to B$ until the loop reaches the value of J. After the loop has cycled the required number of times, B$ is a bar of length J, which in this case is 20.

In normal program use, a series of these routines would create the bars. Then the printer routine would use them in a report; lines 7-9 are a typical printer routine.

LISTING 1

```
1 B$=""
2 C$="▒"
3 J=20
4 FOR X=1 TO J
5 B$=C$+B$
6 NEXT X
7 OPEN 1,4,0
8 PRINT#1, B$
9 CLOSE 1
```

©

The Revised Pet/CBM Personal Computer Guide

Jim Butterfield
Toronto, Canada

This article deals with the changes and new features of the well-known Osborne/McGraw-Hill guide. As such, it isn't a full scale review. Many PET/CBM users are familiar with the first edition; it was the first truly comprehensive user guide for their machine. As such, they were less likely to complain about its faults, which were few, and more likely to be thankful that such a book finally existed.

A Stronger Style

The new edition is a major revision. The previous casual, almost folksy style ("Assuming you have just brought your PET home in a box, you must unpack it") has been replaced by a much tougher down-to-business style. The name PET has been almost universally replaced by CBM. The new book socks it to you with a much more hard-hitting style.

The organization of the book is stronger. Chapters have been reorganized, and additional Basic programming material inserted. There's a stronger grouping of data with headings, subheadings and detail. The Preface suggests, "Even if you have never programmed a computer before, this book will teach you how to write your own Basic programs ... Chapters 4, 5 and 8 teach BASIC programming." That's 190 pages of Basic material, the last 50 of which are essentially reference. It may be rather too terse for many learners, but it's all there.

File Foulup?

The book covers the newer 4.0 ROM system. This is quite a feat considering how recent this system is. Unfortunately, some of this new material appears to have been prepared hastily.

The new Relative data files are discussed, but the book gets the whole thing wrong. It would be well for readers to stay away from this section entirely: relative files are easy to handle, but not in the manner the book suggests. It seems that the authors have confused the carriage return character with the IEEE-488 EOI line; somehow the comma gets dragged in as a field delimiter and we end up with a mess. Worse and worse: playing with the comma makes numeric file variables difficult to handle, and we end up with pages of explanation on how to cope with this. It would have been so easy if we'd started off on the right foot: for writing, one PRINT# statement writes one record; for reading, EOI (as detected in the ST value) signals the last field within a record. And no commas, please.

Appendices

The tables in Appendix A do a fair job of trying to sort out the various codes used by the PET. Between screen formats, PET ASCII, true ASCII and keywords, they take some unscrambling.

For a book which makes some effort to be up-to-date on such things as 4.0 machines, I was surprised to see the out-of-date list of CBM newsletters and references given in appendix D. The PET Gazette and PET User Notes were still listed, and there was no sign of **COMPUTE!** magazine. And I really thought that Commodore Canada's excellent Transactor should have been on the list.

Table F-3 near the end of the book is a curious piece of work. It seems that the authors got hold of the symbol table from Commodore/Microsoft's assembly and sorted and printed it for both Upgrade and 4.0 ROMs. It's fascinating: I suspect that it

shows the original Commodore/Microsoft symbolic names for memory addresses: for example, the Floating Point Accumulator at hexadecimal 005E seems to be called FAC. But mixed in with these is a series of values which don't represent addresses at all. For example, hex 35 is the memory address of part of the top-of-Basic-memory pointer. But 35 is shown in the table as ERRFC, which happens to be the value loaded into the X register just before printing an ?ILLEGAL QUANTITY error message. Oddities in this computer-generated table: non-existent addresses are printed as 0000 rather than being left blank; and locations for which the authors apparently had no explanation are marked "X". It's a lovely table — I wish I could figure out why it's there.

Summing up.

Like its predecessor, the new book is a prodigious work. Its stronger style will improve its value as a reference, although some readers may miss the more casual approach of the first edition.

It's certainly the most comprehensive guide to using CMB/PET machines that is available today. The book is well organized and clearly written. It's generously fitted with examples, programs, diagrams and tables. Apart from the problems dealing with Relative files, the book is a sound approach to using the computer.

[**PET/CBM Personal Computer Guide**, Second Edition: by Adam Osborne and Carroll S. Donahue. Published by Osborne/McGraw-Hill, Berkeley, California.] ©

CBM/PET Computers & Such!

Features for Mar/Apr 81:

Reconditioned SELECTRICs
- 11" carriage $ 449
- 15" carriage $ 549
- 15" carr,h/d,PET i/f $1295

Micro Software Systems
- Billboard for 8032 $ 39
- Billboard for 2001 $ 29

Cursor Magazine (on tape, specify issues. Write for catalog) 10/$44.95
Cable, PET-IEEE $ 37
Cable, IEEE-IEEE $ 47
Modem + terminal software $ 279
NEC Spinwriter #5530 $2695
Xymec typewriter/printer $2499
Centronics 737, PET i/f $ 899
Epson MX-80, PET i/f $ 599
Watanabe MIPLOT $1149
Uncrasher (for new 2001) $ 15
Free cassette drive with 16/32k CBM

CALL FOR CURRENT PRICES ON CBM/PET EQUIPMENT

CONNECT PET/CBM TO PRINTER: Simply plug into the interface and use your PET/CBM computer with popular "standard" printers, or add a video monitor for a second display. All interfaces are assembled, tested, and waranteed. No software is required.

TU-65C Use NEC, Centronics, Xymec, or other industry standard parallel printers. Works with disk, other IEEE devices attached. $129.95

TU-6514 Use RS-232 (serial) printers, like Heathkit H-14. 300 to 4800 baud (please specify rate desired). Includes power pack. $79.95

TU-PVE Connect your Video Monitor to PET/CBM for second display. Our unique contact extensions leave the user port available for connection of other accessories. $39.95 (NOT for 8016/32)

ESCON Interfaces:

SELECTRIC to PET/CBM	$649
SELECTRIC to Apple, Atari, TRS-80: Parallel	$549
SELECTRIC to Apple, Atari, TRS-80: RS-232 Serial	$579
MTST-I/O to Parallel	$425
Cables extra, $20 to $90. Installation	$100

Virginia Micro Systems
14415 Jefferson Davis Highway
Woodbridge, Virginia 22191 (703) 491-6502

VISA/MC, VA +4%
Factory auth sales & service
MWF 12-8, Sa 9-3

IEEE-488 BUS
SYSTEM BUILDING BLOCKS
For Commodore PET/CBM and other computers...

TNW-2000

TNW-1000 Serial Interface: $129
1 channel output only

TNW-2000 Serial Interface: $229
1 channel input and output

TNW-232D Dual Serial Interface: $369
2 channels input and output plus RS-232 control lines

TNW-103 Telephone Modem: $389
Auto answer/auto dial. Use with DAA

SOFTWARE

PTERM: A program that turns your PET into a terminal (Use with TNW-2000, TNW-232D, or TNW 103)
SWAP: Allows storage of up to 8 programs in PET memory at once. Run them in any order.
PAN: A sophisticated electronic mail program (use with TNW-103)

PLUS Most popular computers, disks, printers, etc.

TNW CORPORATION

Write or call for information today:

TNW Corporation
3351 Hancock Street
San Diego CA 92110
(714) 225-1040

"The ROM Rabbit"

ROM and MANUAL — $49.95
Specify 3.0 or 4.0 Basic (Works with or without toolkit)

EASTERN HOUSE SOFTWARE (919) 924-2889
3239 Linda Drive (919) 748-8446
Winston-Salem, N. C. 27106

PET' MACHINE LANGUAGE GUIDE

Contents include sections on:
- Input and output routines.
- Fixed point, floating point, and Ascii number conversion.
- Clocks and timers.
- Built-in arithmetic functions.
- Programming hints and suggestions.
- Many sample programs.

While supply lasts:
Guides for Old ROMS
only $5.00 inc. postage
New ROMS order below

If you are interested in or are already into machine language programming on the PET, then this invaluable guide is for you. More than 30 of the PET's built-in routines are fully detailed so that the reader can immediately put them to good use.

Available for $6.95 + .75 postage. Michigan residents please include 4% state sales tax. VISA and Mastercharge cards accepted - give card number and expiration date. Quantity discounts are available.

ABACUS SOFTWARE
P. O. Box 7211
Grand Rapids, Michigan 49510

Un-Compactor

Robert W. Baker
Atco, NJ

Since my Compactor program was published in the Nov./Dec. '80 issue of **COMPUTE!**, I've had several requests for a companion program to un-Compact programs. The program shown here will do just that!

The program reads a BASIC program file from disk on drive 0 and creates a new copy on drive 1. The new program filename is the same as the original except for a "/U" suffix to indicate an un-compacted version. As with Compactor, load the newly created program file and enter a CLR command from the keyboard to correct the program links. Then save the program back to disk as usual. Un-compactor does not generate correct link values when writing the new program file, it merely writes a dummy value to reserve space for a link. This saves a fair amount of extra work not really needed in the program. The CLR command will force BASIC to correct the program links for you.

The program takes any multi-statement lines (statements separated by colons) and breaks them into separate program lines with new line numbers. The new line numbers are generated by adding one to the original line for each new line generated. This procedure is followed for however many statements exist in the line, as long as new line numbers can be generated without reaching the next line number in the original program. If that point is reached, the remainder of the original line is then copied as part of the last line generated with any appropriate separating colons.

The program must take into account certain BASIC tokens or keywords since they effect whether or not a particular line can be broken into separate lines. Thus, any data following a GOTO, END, RUN, IF, RETURN, REM, STOP, LIST, or CONT token is copied unchanged to the end of the current program line. Also, once a quote is detected, the line must be copied until another quote or end of the program line is reached.

Hope this proves to be of help, especially to those currently using Compactor. This program allows you to effectively re-create programs that were compacted. Now you can get a compacted program in Un-compactor to help speed up program execution. As usual, I'll supply copies of the program on cassette for $2 to cover costs.

```
 10 FOR X=1 TO 10
 11 PRINT X
 12 NEXT
 20 PRINT
 21 PRINT
 22 PRINT
 30 REM TEST FILE FOR UNCOMPACTOR
 40 A=1
 41 B=2
 42 C=3
 43 D=4
 44 E=5:F=6:G=7
 45 X=10
 46 Y=20
 47 Z=30
 100 END:THAT ALL!
READY.
```

- SAMPLE LISTING
OUTPUT FILE FROM UNCOMPACTOR

```
 10 FOR X=1 TO 10: PRINT X: NEXT
 20 PRINT:PRINT:PRINT
 30 REM TEST FILE FOR UNCOMPACTOR
 40 A=1:B=2:C=3:D=4:E=5:F=6:G=7
 45 X=10:Y=20:Z=30
 100 END:THAT ALL!
READY.
```

- SAMPLE LISTING
INPUT FILE TO UNCOMPACTOR

```
30 REM         UN - COMPACTOR
50 REM      BY: ROBERT W. BAKER
70 REM 15 WINDSOR DR., ATCO, NJ 08004
100 :
110 GOTO 270
120 :
130 REM >>>>>> SUBROUTINES <<<<<<<
140 :
150 GOSUB 160: V1=V
160 GET#5,C$: GOSUB 190
170 IF C$="" THEN V=0: RETURN
180 V=ASC(C$): RETURN
190 INPUT#15,EN,EM$,ET,ES
200 IF EN=0 THEN RETURN
210 PRINT "ĥɼDISK ERROR":PRINT
220 PRINT EN;EM$;ET;ES
230 GOTO 1030
240 :
250 REM ***** INITIALIZATION *****
260 :
270 PRINT"ĥ";SPC(10);"ɼUN-COMPACTOR↓↓
280 PRINT" ɼINPUTr̂ FILE IN ɼDRIVE #0↓
290 PRINT"ɼOUTPUTr̂ FILE IN ɼDRIVE #1↓↓
300 INPUT"ɼINPUT FILENAMEr̂";FL$
310 DIM C(256)
320 OPEN 15,8,15
330 OPEN 5,8,5,"0:"+FL$+",P,R"
340 GOSUB 190
350 PRINT:PRINT"OK, WORKING ON LINE# ¬
      ¬.....↓
360 FO$=LEFT$(FL$,14)+"/U"
```

```
370 PRINT#15,"S1:"+FO$
380 OPEN 6,8,6,"1:"+FO$+",P,W"
390 GOSUB 190
400 GOSUB 150: PRINT#6,CHR$(V1);C$;
410 F=1: GOTO 580
420 :
430 REM ***** OUTPUT THIS LINE#
440 :
450 LN=NL: IF LK=0 THEN 1010
460 PRINT LN,
470 PRINT#6,CHR$(1);CHR$(1);
480 PRINT#6,CHR$(LL);CHR$(LH);
490 :
500 REM ***** READ THIS BASIC PGM LINE
510 :
520 X=1
530 GOSUB 160: C(X)=V
540 IF V>0 THEN X=X+1: GOTO 530
550 :
560 REM ***** GET NEXT LINK & LINE#
570 :
580 GOSUB 150: LK=V+V1: IF LK=0 THEN 600
590 GOSUB 150: NL=V1+(256*V): LL=V1:
     ¬ LH=V
600 IF F THEN F=0: GOTO 450
610 :
620 REM ***** BREAK UP LINE IF POSSIBLE
630 :
640 X=1
650 :
660 REM SKIP IF NOT COLON
670 :
680 IF C(X)<>58 THEN 810
690 IF X=1 THEN 950
700 LN=LN+1: IF LN>=NL THEN 950
710 PRINT#6,CHR$(0);CHR$(1);CHR$(1);
720 H=INT(LN/256): L=LN-(256*H)
730 PRINT#6,CHR$(L);CHR$(H);
740 X=X+1: IF C(X)=32 OR C(X)=58 THEN ¬
     ¬740
750 GOTO 680
760 :
770 REM COPY REST OF LINE IF ---
780 REM  GOTO, END, RUN, IF, RETURN
790 REM  REM, STOP, LIST, CONT
800 :
810 IF C(X)<128 OR C(X)>155 THEN 910
820 IF C(X)=128 OR C(X)>153 THEN 850
830 IF C(X)<137 OR C(X)>144 THEN 910
840 IF C(X)=140 OR C(X)=141 THEN 910
850 PRINT#6,CHR$(C(X));
860 IF C(X)>0 THEN X=X+1: GOTO 850
870 GOTO 450
880 :
890 REM SKIP IF NOT QUOTE
900 :
910 IF C(X)<>34 THEN 950
920 PRINT#6,CHR$(C(X)); : X=X+1
930 IF C(X)=34 OR C(X)=0 THEN 950
940 GOTO 920
950 PRINT#6,CHR$(C(X));
960 IF C(X)>0 THEN X=X+1: GOTO 680
970 GOTO 450
980 :
990 REM *** END OF BASIC PROGRAM
1000 :
1010 PRINT#6,CHR$(0);CHR$(0);
1020 PRINT"ĥṟDONE":PRINT:PRINT
1030 CLOSE 5: CLOSE 6: CLOSE 15
READY.
```

©

Using the Hardware Interrupt Vector on the Pet

Eric Brandon

The operating system of the PET is divided into several distinct parts. Some of these get and process your BASIC statements; others deal with all Input/Output operations, and some update the clock, flash the cursor and take care of other sundry details every 60th of a second. This article will show you how to change the operation of the latter to suit your own needs.

Every 60th of a second the PET gets a signal on its IRQ interrupt. When this occurs, it saves all registers and goes to the memory locations specified in locations 537 and 538 (144 and 145 on new ROMs). It executes the machine language program there, and upon hitting an RTI instruction, reloads all of its registers and continues with whatever it was doing. By changing the hardware Interrupt Vector at 537 and 538 (144 and 145 new ROMs) we can make the PET execute our program every 60th of a second, while BASIC operates normally.

I have included here two sample programs using this technique, one is for ROM 2.0 (old ROMs) and the other is for ROM 3.0. What they do, is that after a SYS 826, the contents of the BASIC Input Buffer are constantly displayed on the top two lines of the screen. Hitting the ampersand (&), BREAKs the machine language program, and hitting the slash to the right of the ampersand on the keyboard, clears the buffer if you find that it is getting too cluttered. These programs were written only as examples of using the Hardware Interrupt Vector and are meant to show you how to use this with your own programs. Before we proceed, I wish to clarify just one feature of my assembler; the plus sign on lines 11 and 37 means add one to the value of the symbol. On most assemblers this should be substituted with HIV 1.

Lines 2-6 simply set the values of some symbols. INBUF is the first memory location of the BASIC Input Buffer. KEY is the location that contains the keyboard matrix value of the key presently depressed. INTRPT is the routine to which the Hardware Interrupt Vector usually points. HIV is the location of the first byte of the two byte Hardware Interrupt Vector. Finally, SCRN is the top lefthand corner of the screen.

Lines 7-13 are essential and should be looked at in detail. Line 7 has the Set Interrupt Mask instruction. This is necessary to prevent the PET from being interrupted with only one byte of the pointer changed. Line 12 clears the interrupt mask. If the mask wouldn't be cleared, the PET would "hang up" and need to be turned off. Lines 8–11 make the pointer point to 0347 (0345 new ROMs). Note that the least significant byte goes in 537 (144 new ROMs), and that the most significannt byte goes in 538 (145 new ROMs). The RTS in line 13 returns you to BASIC after your SYS. The effect of an SYS 826 is to make the cursor reappear nearly immediately, but now the PET executes the machine language program at 0347 (0345 new ROMs) every 60thy of a second. The actual operation of the program is quite straightforward to anyone familiar with machine language programming.

Lines 33–39 are the standard procedure for setting the Hardware Interrupt Vector back to normal. Note that POKE 537,133:POKE 538,230 (POKE 144,46:POKE 145,230 new ROMs) has the same effect. This procedure must be done before any cassette I/O.

The last thing that deserves notice are lines 25,32, and 39. The only safe way to leave a program that has been called by the Hardware Interrupt Vector is to jump somewhere into the interrupt handling routine. Since it begins at E685 (E62E new ROMs), that is where you will most often go. You <u>cannot</u> end your program with a RTS or a BRK.

I learned this technique from disassembling KEYPRINT by Charles Brannonn, a program in a previous issue of **COMPUTE!**. I hope you find this useful, and if you have any questions, you can write me at:

Eric Brandon
36 Hartfield Road
Islington, Ontario
Canada
M9A 3C9

```
INPBUF =   $0200
KEY =      $0097
INTRPT =   $E62E
HIV =      $0090
SCRN =     $8000
START =    $0345
LOOP =     $0351
CLEAR =    $035F
LOOP2 =    $0363
NORMAL =   $036E
                        *       =    $33A
                        INPBUF  =    $200
                        KEY     =    151
                        INTRPT  =    $E62E
                        HIV     =    144
                        SCRN    =    $8000
    033A 78                     SEI
    033B A9 45                  LDA  #$45
    033D 85 90                  STA  HIV
    033F A9 03                  LDA  #$03
    0341 85 91                  STA  HIV+
    0343 58                     CLI
    0344 60                     RTS
    0345 A5 97          START   LDA  KEY
    0347 C9 45                  CMP  #69
    0349 F0 14                  BEQ  CLEAR
    034B C9 4D                  CMP  #77
    034D F0 1F                  BEQ  NORMAL
    034F A2 00                  LDX  #0
    0351 BD 00 02       LOOP    LDA  INPBUF,X
    0354 9D 00 80               STA  SCRN,X
    0357 E8                     INX
    0358 E0 50                  CPX  #80
    035A D0 F5                  BNE  LOOP
    035C 4C 2E E6               JMP  INTRPT
    035F A2 00          CLEAR   LDX  #0
    0361 A9 20                  LDA  #32
    0363 9D 00 02       LOOP2   STA  INPBUF,X
    0366 E8                     INX
    0367 E0 50                  CPX  #80
    0369 D0 F8                  BNE  LOOP2
    036B 4C 2E E6               JMP  INTRPT
    036E 78             NORMAL  SEI
    036F A9 2E                  LDA  #$2E
    0371 85 90                  STA  HIV
    0373 A9 E6                  LDA  #$E6
    0375 85 91                  STA  HIV+
    0377 58                     CLI
    0378 4C 2E E6               JMP  INTRPT
```

NEW ROM (3.0) VERSION

```
INPBUF =   $000A
KEY =      $0203
INTRPT =   $E685
HIV =      $0219
SCRN =     $8000
START =    $0347
LOOP =     $0354
CLEAR =    $0361
LOOP2 =    $0365
NORMAL =   $036F
                        *       =    $33A
                        INPBUF  =    $0A
                        KEY     =    515
                        INTRPT  =    $E685
                        HIV     =    537
                        SCRN    =    $8000
    033A 78                     SEI
    033B A9 47                  LDA  #$47
    033D 8D 19 02               STA  HIV
    0340 A9 03                  LDA  #$03
    0342 8D 1A 02               STA  HIV+
    0345 58                     CLI
    0346 60                     RTS
    0347 AD 03 02       START   LDA  KEY
    034A C9 45                  CMP  #69
    034C F0 13                  BEQ  CLEAR
    034E C9 4D                  CMP  #77
    0350 F0 1D                  BEQ  NORMAL
    0352 A2 00                  LDX  #0
    0354 B5 0A          LOOP    LDA  INPBUF,X
    0356 9D 00 80               STA  SCRN,X
    0359 E8                     INX
    035A E0 50                  CPX  #80
    035C D0 F6                  BNE  LOOP
    035E 4C 85 E6               JMP  INTRPT
    0361 A2 00          CLEAR   LDX  #0
    0363 A9 20                  LDA  #32
    0365 95 0A          LOOP2   STA  INPBUF,X
    0367 E8                     INX
    0368 E0 50                  CPX  #80
    036A D0 F9                  BNE  LOOP2
    036C 4C 85 E6               JMP  INTRPT
    036F 78             NORMAL  SEI
    0370 A9 85                  LDA  #$85
    0372 8D 19 02               STA  HIV
    0375 A9 E6                  LDA  #$E6
    0377 8D 1A 02               STA  HIV+
    037A 58                     CLI
    037B 4C 85 E6               JMP  INTRPT
```

OLD ROM VERSION ©

Odds & Ends on the 2040 Disk

Jim Butterfield

The disk's ID is written over 600 times. There's no quick way of changing it — just copy everything over to a new disk if you really need to do this.

* * * * * * * * *

Odds & Ends on the 2040 Disk

Jim Butterfield

Yes, the disk checks every read by using a checksum. You can depend on a good read being correct.

* * * * * * * * * *

Pet As An IEEE-488 Logic Analyzer

Jim Butterfield
Toronto, Canada

If you'd like to see what's going on on the GPIB — and if you can borrow an extra PET and IEEE interface cable — this program will help.

It shows the current status of four of the GPIB control lines, plus a log of the last nine characters transmitted on the bus.

The four control lines are NRFD, NDAC, DAV, and EOI. It would be nice to show ATN too, but I couldn't fit this in: it's detected in a rather odd way in the PET so that fitting it in is rather too tricky for this simple program.

The last nine characters are shown in "screen format". This means that you'll have to do a little translation work to sort out what some of them mean. On the other hand, it allows you to see characters that otherwise wouldn't be printed. A carriage return, for example, shows up as a lower case m; this is a little confusing at the start, but you'll quickly get used to it and it's handy to see everything that goes through. Don't forget that original model PETs may show upper and lower case reversed.

I had hoped to show which characters were accompanied by the EOI signal. It turned out that time is critical — the bus works very fast — and that adding this feature would cut down the number of displayed characters from nine to five. I opted for the bigger count, and dropped the EOI log feature.

The high speed of the bus makes it difficult to watch the control lines in real time. When the "active" PET is exchanging information with disk or printer everything is happening very fast, and the "logic analyzer" PET will show an amazing flurry of activity on the control lines. Only when the activity stops or hangs up will you be able to see the lines in their static conditions.

You may use the program to chase down real GPIB problems, or just to gain insight on how the bus works. Either way, it will come in handy if you can borrow that extra PET unit.

Even at the speed of program operation, a few signals come too fast to catch on the fly. If you must see everything in the select and unselect sequences, you'll have to cut down the number of characters displayes. Try changing the contents of $04F0 to, say, 5 if you want to do this.

```
                              ; IEEE WATCH   JIM BUTTERFIELD
04B0                                *=     $4B0
04B0                   DFLAG        =      $B1
04B0                   DNNSAV       =      $B2
04B0                   EOISAV       =      $B3
04B0 46 B1             START        LSR    DFLAG
04B2 78                             SEI
04B3 AD 12 E8          MAIN         LDA    $E812
04B6 C9 EF                          CMP    #$EF
04B8 D0 02                          BNE    CONT
04BA 58                             CLI
04BB 60                             RTS
04BC AC 10 E8          CONT         LDY    $E810          ;EOI
04BF AD 40 E8                       LDA    $E840          ;DAV, NRFD, NDAC
04C2 AE 20 E8                       LDX    $E820          ;DATA
04C5 29 C1                          AND    #$C1           ;EXTRACT BITS
04C7 C5 B2                          CMP    DNNSAV
04C9 D0 11                          BNE    DNN
04CB 98                             TYA
04CC 29 40                          AND    #$40           ;EXTRACT EOI
04CE 0A                             ASL    A
04CF 49 A0                          EOR    #$A0
04D1 C5 B3             EOI          CMP    EOISAV
04D3 F0 DE                          BEQ    MAIN
04D5 85 B3                          STA    EOISAV
04D7 8D 61 80                       STA    $8061
04DA D0 D7                          BNE    MAIN
                              ;ACTIVITY SEEN - UPDATE SCREEN
04DC 85 B2             DNN          STA    DNNSAV
```

```
440:    04DE 29 80          AND  #$80
450:    04E0 49 A0          EOR  #$A0
460:    04E2 8D 52 80       STA  $8052
470:    04E5 10 1D          BPL  NDAV         ;NO DAV SEEN
500:    04E7 A4 B1          LDY  DFLAG
510:    04E9 30 1B          BMI  DCONT        ;DAV SEEN BEFORE
520:    04EB 85 B1          STA  DFLAG
530:    04ED 85 B2          STA  DNNSAV
540:    04EF A0 00          LDY  #0
550:    04F1 B9 A2 80 SCROL LDA  $80A2,Y
560:    04F4 99 A1 80       STA  $80A1,Y
570:    04F7 C8             INY
580:    04F8 C0 08          CPY  #8
590:    04FA D0 F5          BNE  SCROL
600:    04FC 8A             TXA
600:    04FD 49 FF          EOR  #$FF
600:    04FF 8D A9 80       STA  $80A9
610:    0502 B0 AF          BCS  MAIN
640:    0504 85 B1    NDAV  STA  DFLAG
650:    0506 A5 B2    DCONT LDA  DNNSAV
660:    0508 29 40          AND  #$40         ;NRFD
670:    050A 0A             ASL  A
680:    050B 49 A0          EOR  #$A0
690:    050D 8D 57 80       STA  $8057
700:    0510 A5 B2          LDA  DNNSAV
710:    0512 29 01          AND  #$1          ;NDAC
720:    0514 4A             LSR  A
730:    0515 6A             ROR  A
740:    0516 49 A0          EOR  #$A0
750:    0518 8D 5C 80       STA  $805C
760:    051B D0 96          BNE  MAIN
```

```
 10 REM IEEE WATCH
 20 REM JIM BUTTERFIELD
 30 POKE59468,14:PRINT"□ DAV NRFD
NDAC  EOI":PRINT"  ↑    ↑    ↑    ↑▮"
 40 PRINT"=123456789=▮"
 50 SYS1200
READY.
```

©

Odds & Ends on the 2040 Disk

Jim Butterfield

Yes, the disk checks every write by reading the block back and verifying every byte for correctness. You can depend on a good write having gotten to the diskette correctly.

* * * * * * * * * *

Odds & Ends on the 2040 Disk

Jim Butterfield

The first files written to disk will cluster around track 18, the directory track. This minimizes head movement on a lightly-used disk. By the same token, you might arrange to write your most-used programs and files first on the disk, to save both time and wear and tear.

* * * * * * * * * *

Running 40 Column Programs On A CBM 8032

Chuan Chee
St. Catherine's
Ontario Canada

Good news for those who own a Commodore 80 column CBM. I have developed a method of making the computer act almost like a 40 column PET.

Over the years, many programs have been developed for PET 2001 computers. There had been one ROM upgrade from BASIC 1.0 to 2.0 but many people and software companies got over that hurdle. Now Commodore has introduced a BASIC 4.0 for their PET 40XX and CBM 80XX computers. Again many programmers must change any SYS commands into the ROM locations. However, some programs can still run on the PET 40XX because the programmers were careful enough to avoid any of the ROM routines; especially the BASIC part as opposed to the Operating System. Luckily, most of the first 1024 bytes remained the same as promised by Commodore.

80 Column Problem

But hold on before you start attacking your programs, the CBM 80XX is a completely different animal — it has an 80 column CRT (or screen). All the programs are assuming that there are 40 bytes per line as in the case of a PET, but a CBM has 80. Therefore, any programs that store characters on the CRT memory will have every other line on columns 41 to 80. This is certainly a dissaster.

The Solution

In solving this problem, there must be some way of convincing the microcomputer that there are only 40 bytes per line as in the PET. Commodore was wise enough to implement their newly developed Video Interface Controller (or CRT Controller) into the CBM. They are also using this chip in the VIC 20 (Video Interface Computer). When the power is turned on, the operating system instructs the chip to do various functions such as the height of the 25 lines in normal or graphics modes. My program instructs the Controller to display 40 bytes per line and shift the first column to the right to center the display instead of being on the left side of the CRT.

That is just fine for the programs that store characters on the CRT. But what about those that simply PRINT. Now whenever the PRINT finishes a line (40 characters) of output, the ROM routines will make the next PRINT occur 80 bytes from the start of the current PRINT line. This will make the output appear on every other line.

Well, there just happens to be an "Output a byte on the CRT" jump vector at locations $00EB to $00EC. The CBM 4032 program will change this vector to intercept any character before it gets printed. The routines included in the program were modified from a PET 4032 Operating System ROM, set so that it will behave exactly like a 40 column PET. It will handle RETURN, cursor movements, INST, DEL, and even wrap around lines properly.

Bonuses Not Available On A Pet

There are several features that make this simulation of a PET 40XX even better since they are not available on any PET computer. Such bonuses include the automatic repeat of the cursor control and editing keys and the use of the REPEAT key with all other keys. There will also be the usual warning "bell" when six characters from the end of the line. To disable the "bell", type POKE 231.0. This RAM location contains the duration time of the "bell" which usually is 16. Try poking various values and notice how the duration changes.

I also decided to keep the functions of "◊" and ":" during scrolling the same as before because those who are used to them should not have to use the RVS key. Along the same lines, the ESC key is still fully functional. By the way, did you know that the ESC key not only gets you out of quote and insert modes but also the reverse mode, thus functioning similar to the OFF key?

The CBM 4032 Program

The program is in two parts — a BASIC and a data part. After turning on the computer or typing NEW, type in the BASIC part exactly as shown without any extra spaces. LIST it again to be sure. Next, get into the Machine Language Monitor by SYS4 and type in the data, making sure not to make any mistakes. The next important step is to save the program through the Monitor by .S "0:CBM 4032",08,0400,07A8 for a disk drive or .S "CBM 4032",01.0400,07A8 for a tape cassette drive. Now exit the Monitor and prepare to RUN the program.

The data is actually the machine language routines required. The BASIC portion will transfer it into the second half of the 2K CRT memory. As it transfers the data, you will see "garbage" appear on the CRT. This is an ideal spot to put the routines because the CRT will only use 40 bytes per line by 25 lines (= 1000 bytes), the second half of the CRT memory will never be printed on.

After the transfer, the BASIC portion will SYS 33876 ($8454) to have the routines set up the necessary parameters. It will give the CRT Controller the proper instructions and then CLR the CRT. A READY. will appear on the CRT and control is returned to the user. Now you are ready to RUN any programs meant for a 40 column PET with the proper ROM charges if necessary.

If for any reason you wish to go back to the original 80 column format, you can switch off and on the CBM. Alternatively, you can type SYS 58982 ($E666) and press both SHIFTs and the quote keys simultaneously. The latter method will again display the data on the second half of the CRT but you risk printing or typing over it.

Conclusion

Essentially, any program that can RUN on a PET 40XX, that is with BASIC 4.0, will work with this program. There is no need to alter the program to add anything extra to the programs to artificially perform what this program does. The only side effect is that the characters appear narrower than usual but the advantage of having the program displayed far exceeds this small deviation.

I would like to thank **Batteries Included**, in Toronto for allowing me to use their computers for the development of this program.

```
10 REM * CBM 4032 - BY CHUAN CHEE *
20 REM SEE ARTICLE IN COMPUTE!
30 A=32672:FORI=1136TO1998:
   POKEI+A,PEEK(I):NEXT:SYS33876
READY.

C*
     PC   IRQ   SR AC XR YR SP
.;   B780 E455 34 33 38 36 FA
.
.:   0470 31 14 1F 0F 28 05 19 21
.:   0478 00 07 00 00 10 00 00 00
.:   0480 00 00 00 28 50 78 A0 C8
.:   0488 F0 18 40 68 90 B8 E0 08
.:   0490 30 58 80 A8 D0 F8 20 48
.:   0498 70 98 C0 20 53 62 7D 80
.:   04A0 94 A0 B3 C2 20 02 19 20
.:   04A8 03 08 15 01 0E 20 03 08
.:   04B0 05 05 20 20 78 A9 6F A2
.:   04B8 84 85 EB 86 EC 86 A7 58
.:   04C0 20 75 84 A2 00 86 A7 A9
.:   04C8 10 A2 84 20 86 E0 60 20
.:   04D0 4F 85 4C 9D E1 A0 83 A2
.:   04D8 18 98 9D 3B 84 E0 14 F0
.:   04E0 08 E0 0D F0 04 E0 07 D0
.:   04E8 01 88 CA 10 EC E8 86 9F
.:   04F0 86 C4 A9 20 9D 00 80 9D
.:   04F8 00 81 9D 00 82 9D 00 83
.:   0500 CA D0 F1 A0 00 84 C6 84
.:   0508 D8 A6 D8 BD 3B 84 09 80
.:   0510 85 C5 BD 22 84 85 C4 A9
.:   0518 27 85 D5 E0 18 F0 09 BD
.:   0520 3C 84 30 04 A9 4F 85 D5
.:   0528 A5 C6 C9 28 90 04 E9 28
.:   0530 85 C6 60 09 40 A6 9F F0
.:   0538 02 09 80 A6 DC F0 02 C6
.:   0540 DC 20 06 E6 E6 C6 A4 D5
.:   0548 C4 C6 B0 30 A6 D8 C0 4F
.:   0550 D0 0B 20 1D 85 20 67 86
.:   0558 A9 00 85 C6 60 E0 18 D0
.:   0560 09 20 8B 86 C6 A3 C6 D8
.:   0568 A6 D8 1E 3C 84 5E 3C 84
.:   0570 20 1D 85 A5 C6 48 20 A9
.:   0578 84 68 85 C6 60 E0 17 B0
.:   0580 08 BD 3D 84 09 80 9D 3D
.:   0588 84 60 A0 27 A6 D8 D0 05
.:   0590 86 C6 68 68 60 BD 3A 84
.:   0598 30 06 CA BD 3A 84 A0 4F
.:   05A0 CA 86 D8 85 C5 BD 22 84
.:   05A8 85 C4 84 C6 84 D5 60 A9
.:   05B0 00 85 AC A5 D9 29 7F C9
.:   05B8 1B D0 07 68 68 4C BD E3
.:   05C0 EA EA A4 C6 A5 D9 30 68
.:   05C8 C9 0D D0 03 4C 7E 86 C9
.:   05D0 20 90 08 29 3F 20 6A E1
.:   05D8 4C D5 84 A6 DC F0 03 4C
.:   05E0 D9 84 C9 14 D0 10 88 84
.:   05E8 C6 10 06 20 2A 85 4C 5C
.:   05F0 E2 68 68 4C 51 E2 A6 CD
.:   05F8 F0 03 4C D9 84 C9 12 D0
.:   0600 03 85 9F 60 C9 13 D0 03
.:   0608 4C A3 84 C9 1D D0 10 C8
.:   0610 84 C6 88 C4 D5 90 07 20
.:   0618 67 86 A9 00 85 C6 60 C9
.:   0620 11 D0 FB 18 98 69 28 C5
.:   0628 D5 90 F1 F0 EF 4C 67 86
.:   0630 29 7F C9 7F D0 02 A9 5E
.:   0638 C9 20 90 03 4C D3 84 C9
.:   0640 0D D0 03 4C 7E 86 A6 CD
.:   0648 D0 2F C9 14 D0 27 A4 D5
.:   0650 B1 C4 C9 20 D0 04 C4 C6
.:   0658 D0 07 C0 4F F0 16 20 ED
.:   0660 86 A4 D5 88 B1 C4 C8 91
.:   0668 C4 88 C4 C6 D0 F5 A9 20
.:   0670 91 C4 E6 DC 60 A6 DC F0
.:   0678 05 09 40 4C D9 84 C9 11
.:   0680 D0 2A A5 C6 C9 28 90 05
.:   0688 E9 28 85 C6 60 A6 D8 F0
.:   0690 FB BD 3A 84 10 07 C6 D8
.:   0698 20 A9 84 90 EF CA CA 86
.:   06A0 D8 20 A9 84 A5 C6 18 69
.:   06A8 28 85 C6 60 C9 12 D0 04
.:   06B0 A9 00 85 9F C9 1D D0 08
.:   06B8 88 84 C6 10 EE 20 2A 85
.:   06C0 C9 13 D0 E7 4C 75 84 38
.:   06C8 46 A3 A6 D8 E8 E0 19 D0
.:   06D0 03 20 8B 86 BD 3B 84 10
.:   06D8 F3 86 D8 4C A9 84 A9 00
.:   06E0 85 DC 85 9F 85 CD 85 C6
```

```
.:  06E8 4C 67 86 A0 00 84 C4 A9
.:  06F0 80 85 C8 85 C5 A9 28 2C
.:  06F8 3C 84 30 02 A9 50 85 C7
.:  0700 B1 C7 91 C4 C8 D0 F9 E6
.:  0708 C8 E6 C5 A9 84 C5 C8 D0
.:  0710 EF A9 E8 85 C4 C6 C5 A9
.:  0718 20 C6 C4 C6 C7 91 C4 D0
.:  0720 F8 A2 19 86 D8 A2 00 C6
.:  0728 D8 BD 3B 84 29 7F BC 3C
.:  0730 84 10 02 09 80 9D 3B 84
.:  0738 E8 E0 19 D0 EC A9 83 8D
.:  0740 53 84 AD 3B 84 10 DE 20
.:  0748 0B E4 A6 D8 60 A6 D8 E8
.:  0750 E0 18 F0 36 90 03 4C 01
.:  0758 85 A2 17 BD 3C 84 09 80
.:  0760 85 C8 BC 3B 84 30 02 29
.:  0768 7F 9D 3C 84 98 09 80 85
.:  0770 C5 A0 27 BD 23 84 85 C7
.:  0778 BD 22 84 85 C4 B1 C4 91
.:  0780 C7 88 10 F9 CA E4 D8 D0
.:  0788 D2 E8 BD 3B 84 09 80 85
.:  0790 C5 29 7F 9D 3B 84 BD 22
.:  0798 84 85 C4 A0 27 A9 20 91
.:  07A0 C4 88 10 FB 58 4C A9 84
.
.
READY.
```

```
READY.

C*
     PC   IRQ  SR AC XR YR SP
.;  B780 E455 34 33 38 36 FA
.
8454 78            SEI
8455 A9 6F         LDA #$6F
8457 A2 84         LDX #$84
8459 85 EB         STA $EB
845B 86 EC         STX $EC
845D 86 A7         STX $A7
845F 58            CLI
8460 20 75 84      JSR $8475
8463 A2 00         LDX #$00
8465 86 A7         STX $A7
8467 A9 10         LDA #$10
8469 A2 84         LDX #$84
846B 20 86 E0      JSR $E086
846E 60            RTS
.
.
846F 20 4F 85      JSR $854F
8472 4C 9D E1      JMP $E19D
.
.
8475 A0 83         LDY #$83
8477 A2 18         LDX #$18
8479 98            TYA
847A 9D 3B 84      STA $843B,X
847D E0 14         CPX #$14
847F F0 08         BEQ $8489
8481 E0 0D         CPX #$0D
8483 F0 04         BEQ $8489
8485 E0 07         CPX #$07
8487 D0 01         BNE $848A
8489 88            DEY
848A CA            DEX
848B 10 EC         BPL $8479
848D E8            INX
848E 86 9F         STX $9F
8490 86 C4         STX $C4
8492 A9 20         LDA #$20
8494 9D 00 80      STA $8000,X
8497 9D 00 81      STA $8100,X
849A 9D 00 82      STA $8200,X
849D 9D 00 83      STA $8300,X
84A0 CA            DEX
84A1 D0 F1         BNE $8494
.
.
84A3 A0 00         LDY #$00
84A5 84 C6         STY $C6
84A7 84 D8         STY $D8
84A9 A6 D8         LDX $D8
84AB BD 3B 84      LDA $843B,X
84AE 09 80         ORA #$80
84B0 85 C5         STA $C5
84B2 BD 22 84      LDA $8422,X
84B5 85 C4         STA $C4
84B7 A9 27         LDA #$27
84B9 85 D5         STA $D5
84BB E0 18         CPX #$18
84BD F0 09         BEQ $84C8
84BF BD 3C 84      LDA $843C,X
84C2 30 04         BMI $84C8
84C4 A9 4F         LDA #$4F
84C6 85 D5         STA $D5
84C8 A5 C6         LDA $C6
84CA C9 28         CMP #$28
84CC 90 04         BCC $84D2
84CE E9 28         SBC #$28
84D0 85 C6         STA $C6
84D2 60            RTS
.
.
84D3 09 40         ORA #$40
84D5 A6 9F         LDX $9F
84D7 F0 02         BEQ $84DB
84D9 09 80         ORA #$80
84DB A6 DC         LDX $DC
84DD F0 02         BEQ $84E1
84DF C6 DC         DEC $DC
84E1 20 06 E6      JSR $E606
84E4 E6 C6         INC $C6
84E6 A4 D5         LDY $D5
84E8 C4 C6         CPY $C6
84EA B0 30         BCS $851C
84EC A6 D8         LDX $D8
```

```
84EE C0 4F       CPY #$4F
84F0 D0 0B       BNE $84FD
84F2 20 1D 85    JSR $851D
84F5 20 67 86    JSR $8667
84F8 A9 00       LDA #$00
84FA 85 C6       STA $C6
84FC 60          RTS
84FD E0 18       CPX #$18
84FF D0 09       BNE $850A
.
.
8501 20 8B 86    JSR $868B
8504 C6 A3       DEC $A3
8506 C6 D8       DEC $D8
8508 A6 D8       LDX $D8
850A 1E 3C 84    ASL $843C,X
850D 5E 3C 84    LSR $843C,X
8510 20 1D 85    JSR $851D
8513 A5 C6       LDA $C6
8515 48          PHA
8516 20 A9 84    JSR $84A9
8519 68          PLA
851A 85 C6       STA $C6
851C 60          RTS
.
.
851D E0 17       CPX #$17
851F B0 08       BCS $8529
8521 BD 3D 84    LDA $843D,X
8524 09 80       ORA #$80
8526 9D 3D 84    STA $843D,X
8529 60          RTS
.
.
852A A0 27       LDY #$27
852C A6 D8       LDX $D8
852E D0 05       BNE $8535
8530 86 C6       STX $C6
8532 68          PLA
8533 68          PLA
8534 60          RTS
8535 BD 3A 84    LDA $843A,X
8538 30 06       BMI $8540
853A CA          DEX
853B BD 3A 84    LDA $843A,X
853E A0 4F       LDY #$4F
8540 CA          DEX
8541 86 D8       STX $D8
8543 85 C5       STA $C5
8545 BD 22 84    LDA $8422,X
8548 85 C4       STA $C4
854A 84 C6       STY $C6
854C 84 D5       STY $D5
854E 60          RTS
.
.
854F A9 00       LDA #$00
8551 85 AC       STA $AC
8553 A5 D9       LDA $D9
```

```
8555 29 7F       AND #$7F
8557 C9 1B       CMP #$1B
8559 D0 07       BNE $8562
855B 68          PLA
855C 68          PLA
855D 4C BD E3    JMP $E3BD
8560 EA          NOP
8561 EA          NOP
8562 A4 C6       LDY $C6
8564 A5 D9       LDA $D9
8566 30 68       BMI $85D0
8568 C9 0D       CMP #$0D
856A D0 03       BNE $856F
856C 4C 7E 86    JMP $867E
856F C9 20       CMP #$20
8571 90 08       BCC $857B
8573 29 3F       AND #$3F
8575 20 6A E1    JSR $E16A
8578 4C D5 84    JMP $84D5
857B A6 DC       LDX $DC
857D F0 03       BEQ $8582
857F 4C D9 84    JMP $84D9
8582 C9 14       CMP #$14
8584 D0 10       BNE $8596
8586 88          DEY
8587 84 C6       STY $C6
8589 10 06       BPL $8591
858B 20 2A 85    JSR $852A
858E 4C 5C E2    JMP $E25C
8591 68          PLA
8592 68          PLA
8593 4C 51 E2    JMP $E251
8596 A6 CD       LDX $CD
8598 F0 03       BEQ $859D
859A 4C D9 84    JMP $84D9
859D C9 12       CMP #$12
859F D0 03       BNE $85A4
85A1 85 9F       STA $9F
85A3 60          RTS
85A4 C9 13       CMP #$13
85A6 D0 03       BNE $85AB
85A8 4C A3 84    JMP $84A3
85AB C9 1D       CMP #$1D
85AD D0 10       BNE $85BF
85AF C8          INY
85B0 84 C6       STY $C6
85B2 88          DEY
85B3 C4 D5       CPY $D5
85B5 90 07       BCC $85BE
85B7 20 67 86    JSR $8667
85BA A9 00       LDA #$00
85BC 85 C6       STA $C6
85BE 60          RTS
85BF C9 11       CMP #$11
85C1 D0 FB       BNE $85BE
85C3 18          CLC
85C4 98          TYA
85C5 69 28       ADC #$28
85C7 C5 D5       CMP $D5
```

```
85C9 90 F1       BCC $85BC
85CB F0 EF       BEQ $85BC
85CD 4C 67 86    JMP $8667
.
.
85D0 29 7F       AND #$7F
85D2 C9 7F       CMP #$7F
85D4 D0 02       BNE $85D8
85D6 A9 5E       LDA #$5E
85D8 C9 20       CMP #$20
85DA 90 03       BCC $85DF
85DC 4C D3 84    JMP $84D3
85DF C9 0D       CMP #$0D
85E1 D0 03       BNE $85E6
85E3 4C 7E 86    JMP $867E
85E6 A6 CD       LDX $CD
85E8 D0 2F       BNE $8619
85EA C9 14       CMP #$14
85EC D0 27       BNE $8615
85EE A4 D5       LDY $D5
85F0 B1 C4       LDA ($C4),Y
85F2 C9 20       CMP #$20
85F4 D0 04       BNE $85FA
85F6 C4 C6       CPY $C6
85F8 D0 07       BNE $8601
85FA C0 4F       CPY #$4F
85FC F0 16       BEQ $8614
85FE 20 ED 86    JSR $86ED
8601 A4 D5       LDY $D5
8603 88          DEY
8604 B1 C4       LDA ($C4),Y
8606 C8          INY
8607 91 C4       STA ($C4),Y
8609 88          DEY
860A C4 C6       CPY $C6
860C D0 F5       BNE $8603
860E A9 20       LDA #$20
8610 91 C4       STA ($C4),Y
8612 E6 DC       INC $DC
8614 60          RTS
8615 A6 DC       LDX $DC
8617 F0 05       BEQ $861E
8619 09 40       ORA #$40
861B 4C D9 84    JMP $84D9
861E C9 11       CMP #$11
8620 D0 2A       BNE $864C
8622 A5 C6       LDA $C6
8624 C9 28       CMP #$28
8626 90 05       BCC $862D
8628 E9 28       SBC #$28
862A 85 C6       STA $C6
862C 60          RTS
862D A6 D8       LDX $D8
862F F0 FB       BEQ $862C
8631 BD 3A 84    LDA $843A,X
8634 10 07       BPL $863D
8636 C6 D8       DEC $D8
8638 20 A9 84    JSR $84A9
863B 90 EF       BCC $862C
863D CA          DEX
863E CA          DEX
863F 86 D8       STX $D8
8641 20 A9 84    JSR $84A9
8644 A5 C6       LDA $C6
8646 18          CLC
8647 69 28       ADC #$28
8649 85 C6       STA $C6
864B 60          RTS
864C C9 12       CMP #$12
864E D0 04       BNE $8654
8650 A9 00       LDA #$00
8652 85 9F       STA $9F
8654 C9 1D       CMP #$1D
8656 D0 08       BNE $8660
8658 88          DEY
8659 84 C6       STY $C6
865B 10 EE       BPL $864B
865D 20 2A 85    JSR $852A
8660 C9 13       CMP #$13
8662 D0 E7       BNE $864B
8664 4C 75 84    JMP $8475
.
.
8667 38          SEC
8668 46 A3       LSR $A3
866A A6 D8       LDX $D8
866C E8          INX
866D E0 19       CPX #$19
866F D0 03       BNE $8674
8671 20 8B 86    JSR $868B
8674 BD 3B 84    LDA $843B,X
8677 10 F3       BPL $866C
8679 86 D8       STX $D8
867B 4C A9 84    JMP $84A9
.
.
867E A9 00       LDA #$00
8680 85 DC       STA $DC
8682 85 9F       STA $9F
8684 85 CD       STA $CD
8686 85 C6       STA $C6
8688 4C 67 86    JMP $8667
.
.
868B A0 00       LDY #$00
868D 84 C4       STY $C4
868F A9 80       LDA #$80
8691 85 C8       STA $C8
8693 85 C5       STA $C5
8695 A9 28       LDA #$28
8697 2C 3C 84    BIT $843C
869A 30 02       BMI $869E
869C A9 50       LDA #$50
869E 85 C7       STA $C7
86A0 B1 C7       LDA ($C7),Y
86A2 91 C4       STA ($C4),Y
```

```
86A4 C8          INY
86A5 D0 F9       BNE $86A0
86A7 E6 C8       INC $C8
86A9 E6 C5       INC $C5
86AB A9 84       LDA #$84
86AD C5 C8       CMP $C8
86AF D0 EF       BNE $86A0
86B1 A9 E8       LDA #$E8
86B3 85 C4       STA $C4
86B5 C6 C5       DEC $C5
86B7 A9 20       LDA #$20
86B9 C6 C4       DEC $C4
86BB C6 C7       DEC $C7
86BD 91 C4       STA ($C4),Y
86BF D0 F8       BNE $86B9
86C1 A2 19       LDX #$19
86C3 86 D8       STX $D8
86C5 A2 00       LDX #$00
86C7 C6 D8       DEC $D8
86C9 BD 3B 84    LDA $843B,X
86CC 29 7F       AND #$7F
86CE BC 3C 84    LDY $843C,X
86D1 10 02       BPL $86D5
86D3 09 80       ORA #$80
86D5 9D 3B 84    STA $843B,X
86D8 E8          INX
86D9 E0 19       CPX #$19
86DB D0 EC       BNE $86C9
86DD A9 83       LDA #$83
86DF 8D 53 84    STA $8453
86E2 AD 3B 84    LDA $843B
86E5 10 DE       BPL $86C5
86E7 20 0B E4    JSR $E40B
86EA A6 D8       LDX $D8
86EC 60          RTS
.
.
86ED A6 D8       LDX $D8
86EF E8          INX
86F0 E0 18       CPX #$18
86F2 F0 36       BEQ $872A
86F4 90 03       BCC $86F9
86F6 4C 01 85    JMP $8501
86F9 A2 17       LDX #$17
86FB BD 3C 84    LDA $843C,X
86FE 09 80       ORA #$80
8700 85 C8       STA $C8
8702 BC 3B 84    LDY $843B,X
8705 30 02       BMI $8709
8707 29 7F       AND #$7F
8709 9D 3C 84    STA $843C,X
870C 98          TYA
870D 09 80       ORA #$80
870F 85 C5       STA $C5
8711 A0 27       LDY #$27
8713 BD 23 84    LDA $8423,X
8716 85 C7       STA $C7
8718 BD 22 84    LDA $8422,X
871B 85 C4       STA $C4
871D B1 C4       LDA ($C4),Y
871F 91 C7       STA ($C7),Y
8721 88          DEY
8722 10 F9       BPL $871D
8724 CA          DEX
8725 E4 D8       CPX $D8
8727 D0 D2       BNE $86FB
8729 E8          INX
872A BD 3B 84    LDA $843B,X
872D 09 80       ORA #$80
872F 85 C5       STA $C5
8731 29 7F       AND #$7F
8733 9D 3B 84    STA $843B,X
8736 BD 22 84    LDA $8422,X
8739 85 C4       STA $C4
873B A0 27       LDY #$27
873D A9 20       LDA #$20
873F 91 C4       STA ($C4),Y
8741 88          DEY
8742 10 FB       BPL $873F
8744 58          CLI
8745 4C A9 84    JMP $84A9
.
.
READY.
```

©

Build Your Own Controllers

Nuts And Volts

Gene Zumchak

Part I

If you have a personal computer of any kind, you probably already appreciate the power of a general-purpose computer system to serve as a controller. While tying up your APPLE or PET to control the thermostat may not seem overly attractive, you can usually try out a control idea or scheme using your existing computer system and small amount of custom I/O. Eventually, however, you will want to dedicate a separate computer system to your controller application.

It wasn't so long ago that such a thought would be prohibitive. Computer systems were dream machines that cost several thousands of dollars. Of course, if you have a console type computer system which includes a CRT and perhaps one or more disks, then your console system can easily cost three or more thousand dollars. On the other hand, a great many controller applications require little more than a handful of chips that cost well under $100. In fact, if your application has any merit and a significant market, it may be quite possible to integrate the design into a single-chip microcomputer costing only a couple of dollars, and you can be on your way to making your first million.

While your particular application may never make you rich, it is fairly easy to put together a prototype or a one-of-a-kind microcontroller system for a reasonable price. A 6502 will cost less than $10. A 2716 will cost about the same. Figure $5 for a 128 x 8 RAM chip, (Motorola 6810), or $8 for a pair of 2114's for 1K of RAM. A 6522 for $8 will provide sixteen bits of I/O and a pair of timers (suitable for a real-time clock). Finally, a few more dollars for a crystal and some TTL for address decoding, and the electronic parts cost will come to not much more than $50.

If the parts really cost as little as mentioned, what's to prevent anyone with a little knowledge of computers from designing and building his own microcontrollers? The answer is absolutely nothing. But there is one small catch. While the cost of the end product may be minimal or even negligible, most companies or individuals who design microcomputer systems do it with the aid of a microprocessor "development system". Commercial development systems start at about $5,000, but typically range from $15,000 to $25,000.

In my book, **Microcomputer Design and Troubleshooting**, which is being typeset and will be in print in the Fall (Howard Sams, and the Blacksburg Series), I address the question of what comprises a typical development system, but more importantly, what is minimally required to put together your own low-budget development system. While the reader will want to read about the details in the book when it is published, the highlights of that discussion will be brought out here, in this first installment of several in which I will outline the procedures and equipment necessary to put together and bring up, your own microcomputer controllers.

The Development System

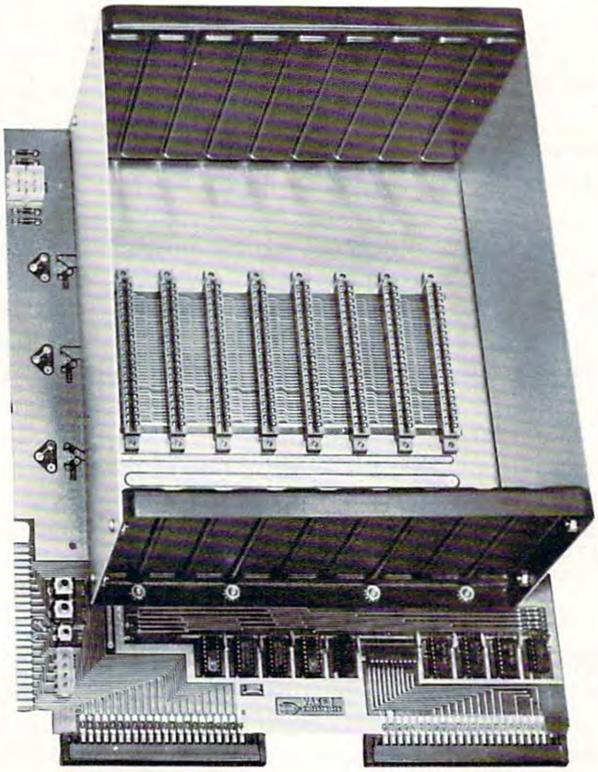
A development system is the hardware and software required to check out and debug both the hardware and software of a prototype microprocessor system. Ironically, the hardware and software debugging capabilities are not always reflected by the systems very high cost. Software debugging capabilities are usually satisfactory, provided that the system has an "optional" processor emulator module which typically costs $2,000 or more. Even with the emulator, the hardware debugging capabilities may be mediocre at best.

Typically, a commercial development system consists of the following items:

1. Microcomputer with software
2. Console device (CRT or Teletypewriter)
3. RAM memory blocks
4. Floppy disk(s)
5Printer
6. EPROM programmer (with software)
7. Emulator (processor)

Why should such a system cost $15,000? The reason for the high cost is the law of supply and demand; there just aren't that many people in the world who need a microprocessor development system. However, except for some specialized software like an editor and assembler, the first five items in the list are not appreciably different than what you get in a BASIC oriented console computer system like a PET or APPLE. And, of course,

VAK-1 MOTHERBOARD



We also carry:

SYM-1	**$229.00**
AIM-65 w/1K	**389.00**
AIM-65 w/4K	**439.00**

We also do custom hardware and software for the 6502 microprocessor

PRICE: **$139.00**

Call or write for shipping charges and our complete catalog.

The VAK-1 was specifically designed for use with the KIM-1, SYM-1 and the AIM 65 Microcomputer Systems. The VAK-1 uses the KIM-4* Bus Structure, because it is the only popular Multi-Sourced bus whose expansion boards were designed specifically for the 6502 Microprocessor.

SPECIFICATIONS:

- Complete with rigid CARD-CAGE
- Assembled (except for card-cage). Burned in and tested.
- All IC's are in sockets
- Fully buffered address and data bus
- Uses the KIM-4* Bus (both electrical Pin-out and card size) for expansion board slots
- Provides 8 slots for expansion boards on 1″ centers to allow for wire-wrap boards
- Designed for use with a Regulated Power Supply (such as our VAK-EPS) but has provisions for adding regulators for use with an unregulated power supply.
- Provides separate jacks for one audio-cassette, TTY and Power Supply.
- Board size: 14.5 in. Long x 11.5 in. Wide x 8 in. High
- Power requirements; 5V.DC @ 0.2 Amps.

**KIM-4 is a product of MOS Technology/C.B.M.*

RNB ENTERPRISES INCORPORATED

2951 W. Fairmount Avenue
Phoenix, AZ 85017
(602) 265-7564

editors and assemblers are easy to come by for most console systems. On the other hand, not many microprocessor development systems will allow you to run a program in BASIC, or Pascal, or FORTH. In other words, while a personal computer can be turned into a development system, a development system usually does not make a very good personal computer. It should be made clear, however, that a personal computer is not a development system without items 6 and 7 in the list above (or their equivalent). The EPROM programmer is easy. Such accessories are available for very reasonable prices. If you don't mind stuffing a blank board, you can put together your own universal EPROM programmer for less than $30. However, the "emulator" function is not quite so available.

The function of an emulator is to provide the prototype controller with the attributes of an operating system. Suppose you want to make a controller out of an existing single-board computer like a KIM or SYM. After attaching any additional I/O hardware required, you can hand assemble a controller program and enter it into the KIM or SYM's RAM using its built-in operating system. Programs under development can be saved on tape. Software debug functions are even available to get the program running. But what do you do if your prototype controller is not like a KIM or SYM? What if it has no keyboard or display, or any means (operating system) of entering a program into itself? There are two solutions to this problem. One is to use (abuse) an EPROM programmer. The second is to use some kind of emulator.

The first solution mentioned is actually used by owners of commercial development systems, who do not have an emulator module. It works as follows. First, a program is developed and entered into RAM in the development system's microcomputer. The RAM contents are now burned into an EPROM. The EPROM is now plugged into the prototype system and an attempt made to reset the prototype system and run the program. If the program does not run as expected, the program is modified and a second EPROM is programmed. In the meantime, the old EPROM is being erased. While this method can eventually produce a working program it is very tedious and ineffecient. To give you some idea of how really dumb this method is, consider using the same method to write a program in BASIC. That is, suppose you had to enter the program into RAM, burn the RAM contents into an EPROM and then plug the EPROM into a special socket to try out your program. Yet that is essentially what many, if not most, people do to bring up controllers. Clearly there must be a much better way.

The second approach is to give the prototype system a virtual operating system with some kind of emulator. Commercial development systems generally emulate the prototype's processor. Such a processor emulator is a very complex hardware and software system, usually requiring two or three large PC boards which live in the development system's card rack. The emulator physically connects to the prototype via a cable which plugs into the prototype's processor socket. The development system is used to create a program in a block of RAM. The emulator allows the block of RAM to be executed as if it resided in the prototype system. In addition, the program can be stepped, the register contents displayed, breakpoints set, etc. Effectively, the emulator runs considerable software "in the cracks" between prototype program instructions. One consequent limitation of this scheme, however, is that many emulators are unable to execute prototype programs at the full processor speed.

While a processor emulator can be quite useful for debugging software, it is somewhat less suitable for finding hardware bugs. Unfortunately, many users attempt to debug complicated software before even knowing whether the hardware is 100% functional. As mentioned, a processor emulator is very expensive, typically two or three thousand dollars, and cannot be used independently of the development system for which it was designed. Fortunately, another kind of emulator can be built that is usable with almost any computer system having an operating system, including one as simple as a KIM. Instead of emulating the prototype's processor, this emulator emulates the prototype system's ROM or EPROM. It is nothing more than a small block of RAM that can be alternately addressed as part of the host computer system, or via the EPROM socket in the prototype system. Aside from the fact that an EPROM emulator can be an efficient tool for debugging both hardware and software, the best thing about an EPROM emulator is that it can be put together for less than $100.

An EPROM emulator is used as follows. A program is assembled and placed into the emulator RAM block using your computer's operating system. Throwing a switch on the emulator now causes the RAM block to be addressed from a cable and plug inserted into the empty EPROM socket on the prototype system. If a change is required, the switch is flipped back into the host system position and any changes made in the emulator RAM. There is never any need to burn an EPROM until the program is completely debugged. At any point along the way, the RAM contents can be preserved on tape or disk.

In the next column, we will see what it takes to put together an EPROM emulator, and use it to debug both hardware and software. A very workable microprocessor development system can be had with as little as a KIM, an EPROM programmer,

A BRILLIANT FUTURE FOR YOUR AIM-65 WITH THE BANKER MEMORY ™

Your 36K of free address space is the AIM's most valuable and limited resource. With today's large capacity RAM boards, ROM boards, disk systems, video boards, and other expansion accessories it is easy to deplete this resource before the application requirement is satisfied. MTU has solved this problem.

THE BANKER MEMORY contains 32K of RAM, 4 PROM sockets for 2716/2732/2332, a PROM programmer, 40 bits of parallel I/O, and 4 timers from two 6522 I/O chips. Addressing is extremely flexible with the RAM independently addressable in 4K blocks, PROM's independently addressable, and I/O addressable anywhere on a 64 byte boundary (even in AIM's I/O area at AXXX by adding a single jumper to the AIM).

This may sound familiar, but read on! Unlike other AIM compatible memory boards, THE BANKER MEMORY has on-board bank-switching logic! The four 8K blocks of RAM plus the 4 PROM sockets make up 8 **resources**, each associated with a bit in an Enable Register. Through this Enable Register resources may be turned on and off under software control. When a resource is off, its address space is freed for other uses. You can even put BANKER resources at the same address and switch among them for virtually unlimited RAM and PROM expansion! You can even have multiple page zero's and stacks! Do you need 160K byte of memory? It only takes 5 of THE BANKER MEMORY boards and you end up with 5 page zeros and stacks to boot!

There's more! The BANKER MEMORY also incorporates 18 bit addressing which allows for the 256K address spaces of the future. RAM, PROM, and I/O each has its own full 18 bit address decoder which allows these resources to be in different 64K banks. This board and other MTU products, such as our 320 by 200 dot VISIBLE MEMORY and Floppy Disk Controller with 16K DMA RAM, can turn your AIM into a truly powerful 6502 computer that far surpasses the packaged systems in functional performance.

INTRODUCTORY SPECIAL K-1032-1 32K BANKER MEMORY FULLY ASSEMBLED AND TESTED $395.00 ($450.00 as of March 1, 1980) or the K-1032-2 16K RAM only with bank switching and 18 bit address bus only $295.00

Isn't it time you took a closer look at MTU — we offer you power now with an eye to the future.

WRITE OR CALL TODAY FOR OUR 48 PAGE FALL 1980 6502 CATALOG
International requests include $1.00

VISA and MASTERCARD accepted

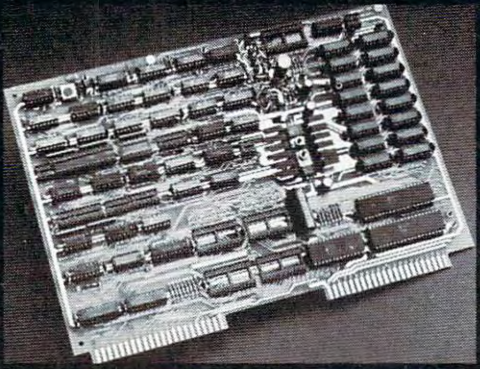


Photo credit:
SUPERNOVA CRAB NEBULA:
Palomar Observatory,
California Institute of Technology

MTU
Micro Technology Unlimited
2806 Hillsborough Street
P.O. Box 12106
Raleigh, NC 27605, U.S.A.
(919) 833-1458

and an EPROM emulator. With such a budget development system, you can bring up a controller based on ANY processor which can use EPROM. Of course, it's even easier if you have an assembler on your system for the processor you are using in the controller.

Perhaps you would like to control your electric train, or build a home environment control system. Or maybe you want to computerize your wife's loom. Maybe you'd like to cash in on your software experience and build custom controllers for local industrial clients. Whatever your bag, you probably already have with your present personal computer, most of what you need to put together a very practical microprocessor development system. Stay tuned for the next exciting installment. ©

A Kim-I Music File In Microsoft Basic

Part 1.

Anthony T. Scarpelli
N. Windham ME

Getting The System Together

If you have a KIM-I, don't have a printer, but do have a memory mapped video display, here's how I solved the problem of getting a software routine to cause an ASCII keyboard to act like a serial teleprinter with all the KIM-I's teletype operations. There's nothing that seems complicated about what I did, but it sure took some mental gyrations to get it working. Yet I did learn a lot about the KIM monitor routines which I'll tell you about. Also how to implement BASIC, and how to implement a Music File which I wrote for my wife. Here's the story.

I had a KIM-I up and running and was learning a lot about assembly language programming, when the opportunity of getting a high resolution video monitor for cheap came along. I bought a SWTP keyboard, and while I was at one of the computer fairs last year I purchased Microtechnology's 8K visible memory and a main frame. The price was good and it was completely compatible with the KIM. It's a dynamic memory system, but is completely invisible to normal computer use, and it has a standard video output. It works beautifully, and is fairly high in resolution with 64,000 bits as dots on the screen. Writing a "1" in a memory location lights up a dot, and, of course, a "0" turns the dot off. Microtechnology's SWIRL software routine shows the system off and provides hours of viewing enjoyment; and when company comes over it's great for showing off your computer.

Microtechnology also has a text display routine whereby, after an ASCII number is put into the accumulator, a subroutine call to the text display puts the ASCII character on the monitor screen. It provides a 53 character by 18 line display, with both upper and lower case letters. Having a software character generator gives you complete control over the configuration of the letters. For instance, I changed all my lower case letters, which I didn't need, into a table of 26 lines, dots, and other shapes for drawing on the screen. Also, the whole screen can be saved on tape. My wife was very pleased as a valentine message formed from a randomly patterned screen. Hypertape loaded the screen in under three minutes.

I also purchased from Microtechnology their bare board 16K memory, and purchased the I.C.'s and components at other sources. You can save about a hundred dollars this way, but you do have to get a few extra memory chips in case a bad one comes up and you do have to do all of your own soldering, and testing. If you go this route you might have a fault in the bare board. In the one I bought, a part of the PCB pattern wasn't etched away so I had no -5v supply. After I fixed the problem the board worked perfectly the first time running and onwards, and I have nothing but praises for the design.

Then came the job of getting my keyboard with parallel output ASCII to go serial. It turned out to be not too difficult when I found an interface in a series of articles by John Blankenship in Kilobaud. In the March '78 issue he shows how to build a parallel to serial interface for the KIM-I. It merely takes the parallel output of the keyboard, using three I.C.'s and a transistor, and the KIm's power and clock, and converts it to a serial output which is presented to the printer input of the KIM. It worked very well.

Then what? Well, here comes the hard part. In order to get the KIM to accept a printer input, you connected pins "21" and "V" on the applications connector, hit the RS button, press the RUBOUT key on your keyboard and type away. The only problem is that any ASCII characters that come in don't go anywhere except to the printer output of the same connector. The ASCII number is put into the accumulator, but how do you call up a subroutine in some other part of memory to display it? The solution wasn't too difficult. You write a little program that jumps to KIM's own GETCH subroutine which then puts the printer ASCII character into the accumulator, then jumps to the character display subroutine, then jumps back to the GETCH etc. You start out by going to the memory location where the program starts on the KIM keyboard, short the two pins together (best to get a switch to do this), hit RS, then RUBOUT, and G on the keyboard, and away you go. You're finally writing on the CRT. Now what?

With this method that's about all you can do because you are in a program of your own creation and are using KIM's ROM routines, and you have to stay there until you hit ST (stop). What I really wanted to do was have my keyboard act just like a printer: change memory, display it, and all the other things the user manual said you can do with a printer. I asked myself, how easily can this be done? More likely, how difficult is it. There were two possibilities open to me: hardware or software. My old teacher said you never learn enough by going the easy route. I didn't know whether hardware or software was the most difficult, but I chose software. You can judge the result; I probably would have bought a printer.

To go the software route meant rewriting some of the subroutines in the KIM's ROM. To show you what routines I had to include, let's go over what happens in the KIM when you hit RS. So get out your user manual, follow the diagrams and let's go.

First look at the listings starting at 1C22 in the User Manual and also at fig. 1.

1. When the RS (reset) button is pressed the data at locations 1FFC & 1FFD, which happens to be the address 1C22, is put into the program counter. This is the entry point for the program in ROM of the 6530-002. This address is fixed and cannot be changed. It is the KIM entry via RST.

2. The first thing that happens is the stack pointer is initialized to FF.

3. Then we go to a subroutine called INITS at 1E88. In INITS, the first thing done is to put 01 into the X register and then put it into the top of the stack at 00FF.

Next, the X-index gets 00 and is stored in PADD which is the 6530-002 A ports data direction register. This is at address 1741 and makes all the ports inputs so they can accept data from TTY or KB (keyboard).

Next X-index gets 3F and is stored at 1743 which is the 6530-002 B ports data direction register, PBDD, and it makes ports PB6 and PB7 inputs, and all the rest outputs. PB7 is connected to the audio tape interface circuits and is prepared to accept program loading from tape.

Next X-index is loaded with 07 and is stored in SBD (1742) which is the data to be sent out from the 6530-002 data ports. So PB 0, 1, & 2 now have 1's on them. PB0 is for TTY data out. PB 1, 2, 3, & 4 go to the 74145 I.C.'s inputs. With a 1 on 1 & 2 and 0 on 3 & 4, all the outputs of the 74145 are high except 03. This goes out to application connections A-V. When this pin is connected to A-21 (PA0), PA0 becomes low. This indicates TTY mode.

Next decimal mode is cleared and the interrupt disable status is set. Then a return from this subroutine.

4. Next back at 1C2A, FF is stored at 17F3 (CNTH 30) which is the TTY count, and 01 is stored in the accumulator. Then SAD (1740) is tested, specifically PA0. If it is not equal to zero, that is, if it's high, the program branches to START. PA7 is tested also. This is the input from the TTY keyboard. It tests for a rubout bit. PA7 is normally a one and the program will keep on testing this input until a zero is detected and also PA0 in case the TTY mode is not wanted any more.

If a zero is detected, the accumulator is loaded with FC and the carry flag is cleared, then 01 is

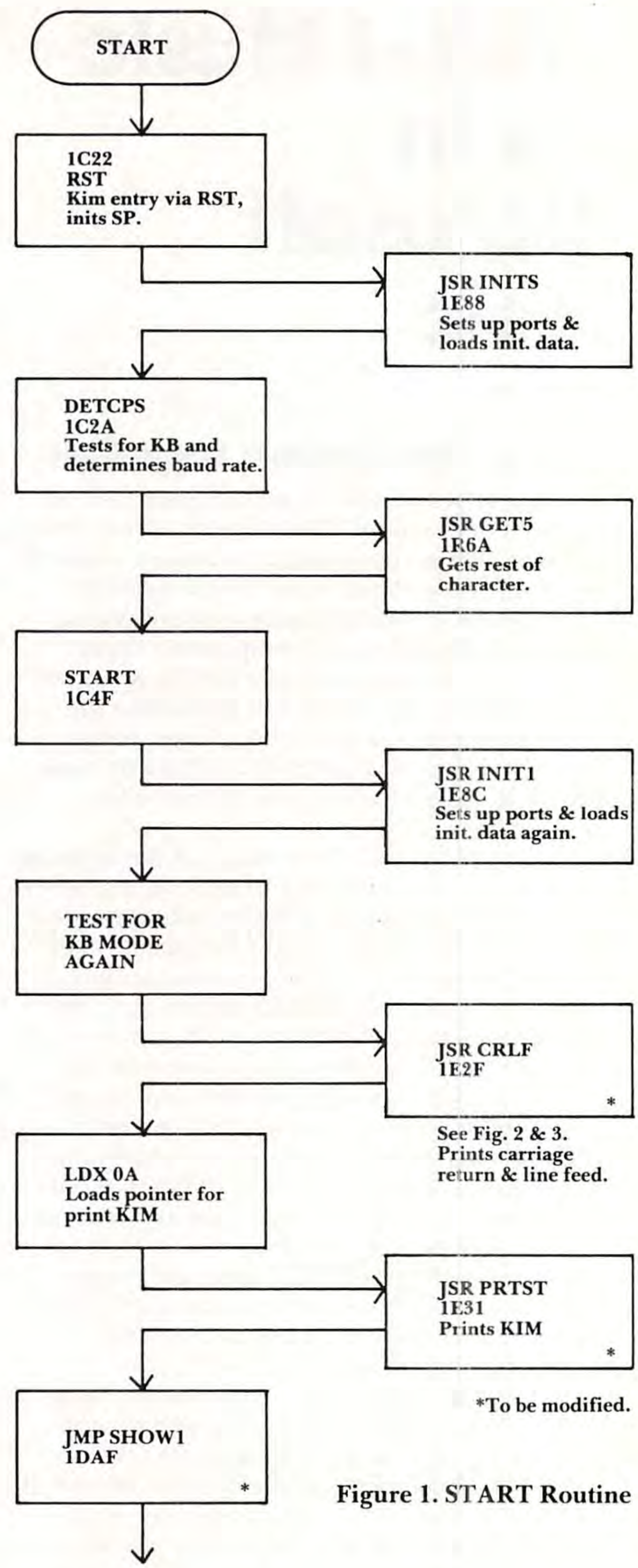


Figure 1. START Routine

added to the accumulator (FC). If the carry flag is not set it will branch to DET 2. It will the first time around anyway. This part (DET 2) first loads Y-index with SAD (1740) and if the rubout bit is still there (a 0 at PA7) then it goes back to DET 3 and another 01 is added to the accumulator. When the

accumulator reaches FF and 01 is added, the carry flag is set and CNTH30 (17F3) is incremented, it becomes 00. As long as the rubout bit is there the accumulator keeps on increasing and increases CNTH30. As soon as the bit ends the accumulator is stored in CNTL30 (17F2) and X-index gets an 08. Then the program goes to subroutine GET5 at 1E6A, where it goes to DEHALF (1EEB).

5. DEHALF first gets the high byte count time at CNTH30 and stores it in TIMH (17F4), then gets CNTL30. The accumulator and TIMH are shifted right (divides by two). If the 0 bit had a 0 the carry flag is cleared and a branch is taken to DE2, otherwise the accumulator is OR'd with 80 and it branches to DE4. If the DE2 branch was taken the carry flag has been set and next 01 is subtracted from the accumulator. The time is reduced and back with RTS. What is happening here is the keyboard baud rate in CNTL30 and CNTH30 is halved to get in the middle of the bit, then delayed one whole bit to read the next bit of the character. Cute, huh.

6. Back at 1E6D (GET2), the accumulator is loaded with SAD and the bit number 7 only is saved. 00FE is shifted right, then OR'd with the accumulator and stored in 00FE. Another delay and the process is repeated until the whole character is retreived, then another half delay, X-index is loaded with TMPX (00FD), and the accumulator gets CHAR which is the ASCII character. The accumulator is rotated left then shifted right, which gets rid of any parity bit that might be stuck on the character. Then a return to START.

7. START. First is a jump to subroutine INIT1 (1E8C) which is the same as before, it sets up the ports. The accumulator is loaded with 01, and SAD is tested again for TTY or KB mode. If there's a 1 in PA0 it branches to KB mode. If no KB mode, it then jumps to CRLF, Fig. 2 & 3, (1E2F), which

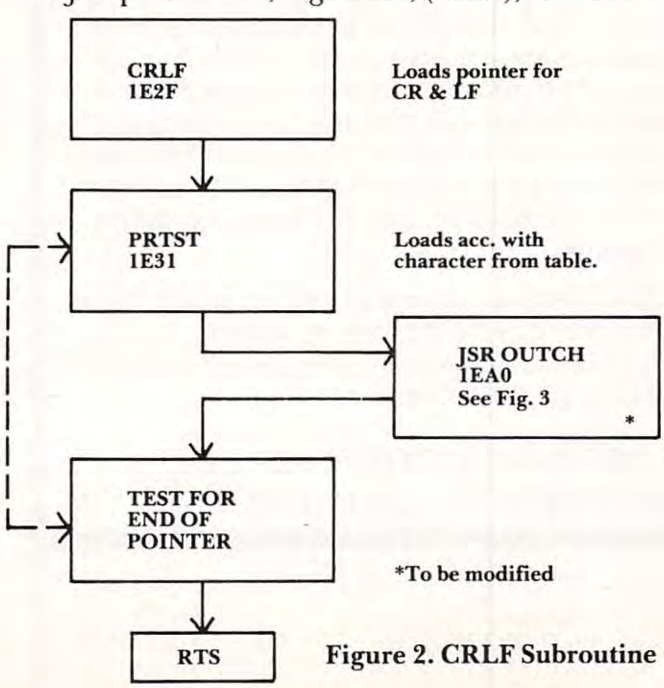


Figure 2. CRLF Subroutine

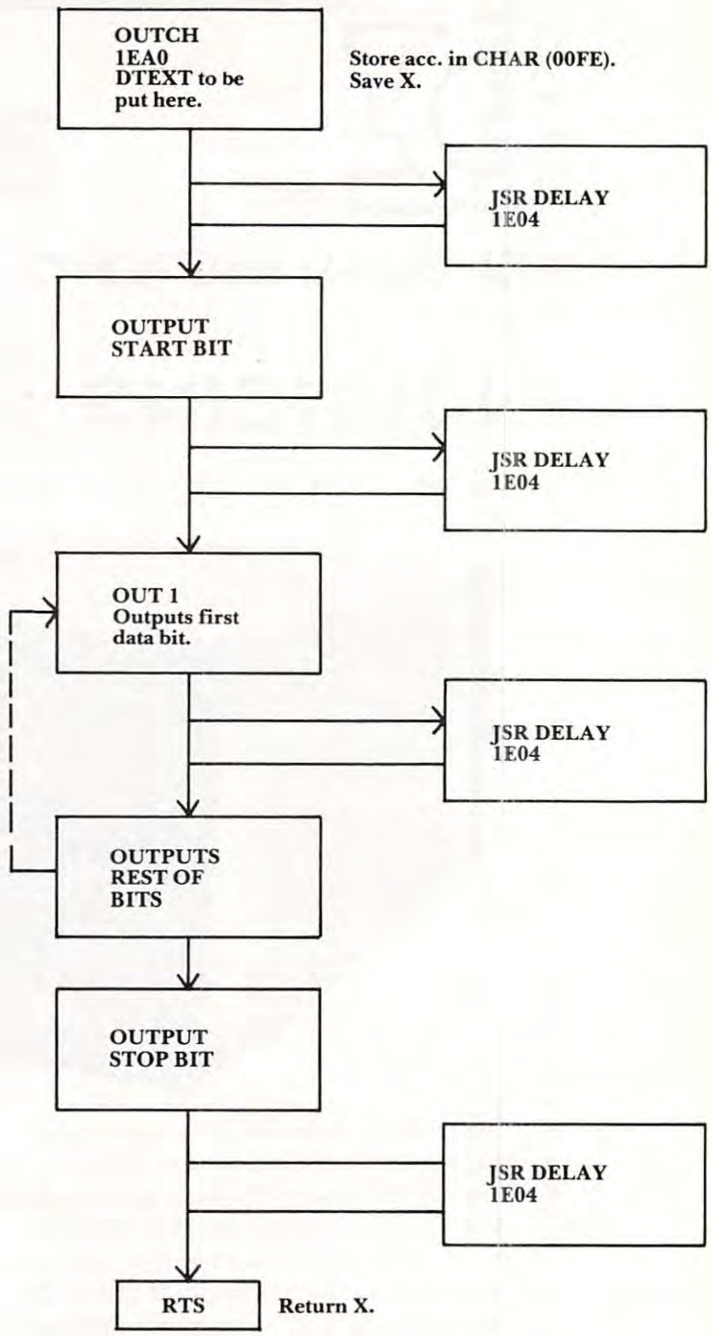


Figure 3. OUTCH Subroutine

prints a carriage return, then a line feed, then JSR PRTST prints "KIM", then jumps to SHOW1 (1DAF), Fig. 4, and then back to CLEAR, Fig. 7.

8. CLEAR. The accumulator gets loaded with 00 and is stored in INL & INH. The program tests for a character in GETCH, Fig. 8. In GETCH it stays in a loop waiting for a start bit. After the start bit, the rest of the character is retreived and loaded into the accumulator, the program then comes back, and we test for KB mode again. If no KB the character is changed into a hex number in PACK, Fig. 9, and then in SCAN, Fig. 10, the program determines if the hex number is an execute key. If not, it will get another character.

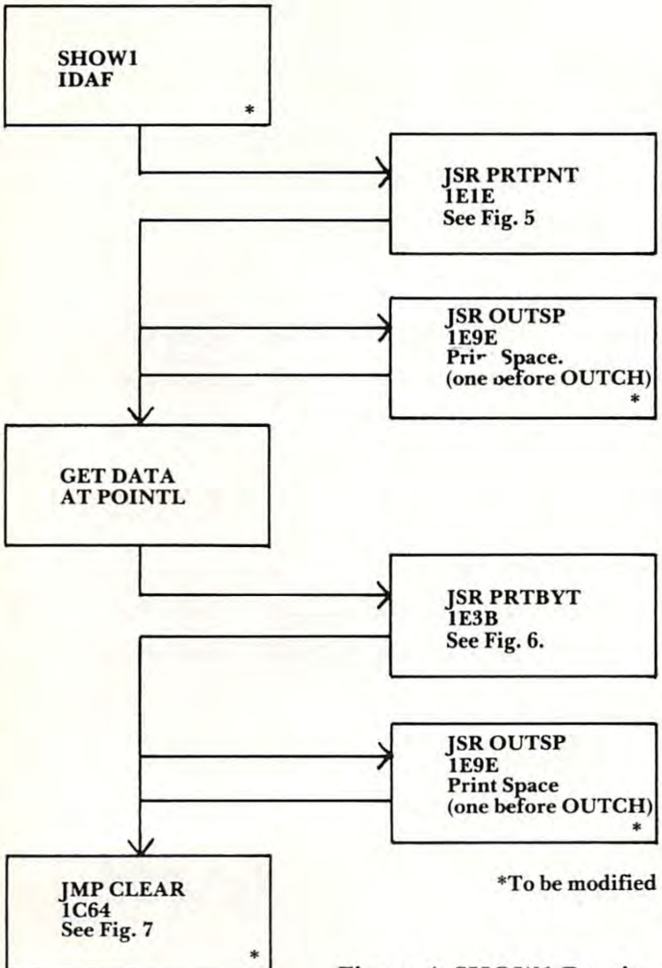


Figure 4. SHOW1 Routine

So this is the program I need to simulate a teletype. The problem now becomes, what are the subroutines I have to rewrite and which ones of the KIM's ROM subroutines can I use. Obviously, any part of the program that refers to a ROM address has to be rewritten, such as in a JMP. Also when the accumulator gets the ASCII character that is to be displayed, the program that does the displaying, in this case called DTEXT (the Microtechnology software routine), has to be addressed at the right point, and thus any subroutines involved here have to be rewritten. So definitely the subroutine OUTCH has to be changed to add DTEXT. We get to OUTCH from CRLF so that has to be rewritten. CRLF is addressed from START which is part of the whole RST routine. As you can see it starts to get involved. So if you go this route table I lists all the KIM ROM routines that must be rewritten. Of course in this rewriting, some branches have to be changed as well as addresses. (A SASE sent to me will get you a list of the changed addresses.)

Now my keyboard acts just as a teletype, and I can display all the teletype outputs from the KIM on the CRT. First I go to the RST program address, the one I rewrote, on the KIM display, switch to teletype mode, hit RS on the KIM, then press the rubout key on the keyboard. The SWTP keyboard doesn't have an actual rubout key, but there are two spare keys, one of which can be wired as rubout. Then I press the G key which puts me into the RST program, (rewritten). When the rubout key is pressed again the CRT will display "KIM" and also the address of the RST program; now we are as a teletype with all its functions. Simple, wasn't it.

Next time I'll go into the actual file program that creates a music file, and then can search it for any of a number of subjects.

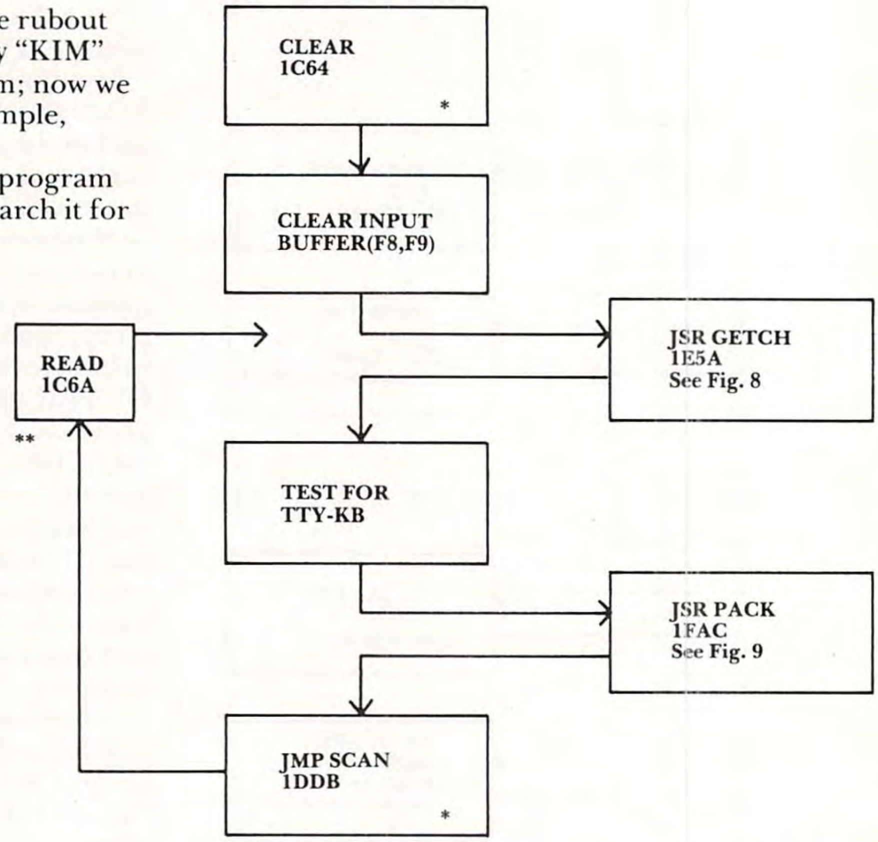


Figure 7. CLEAR Routine

*To be modified
**Re-entrance from SCAN

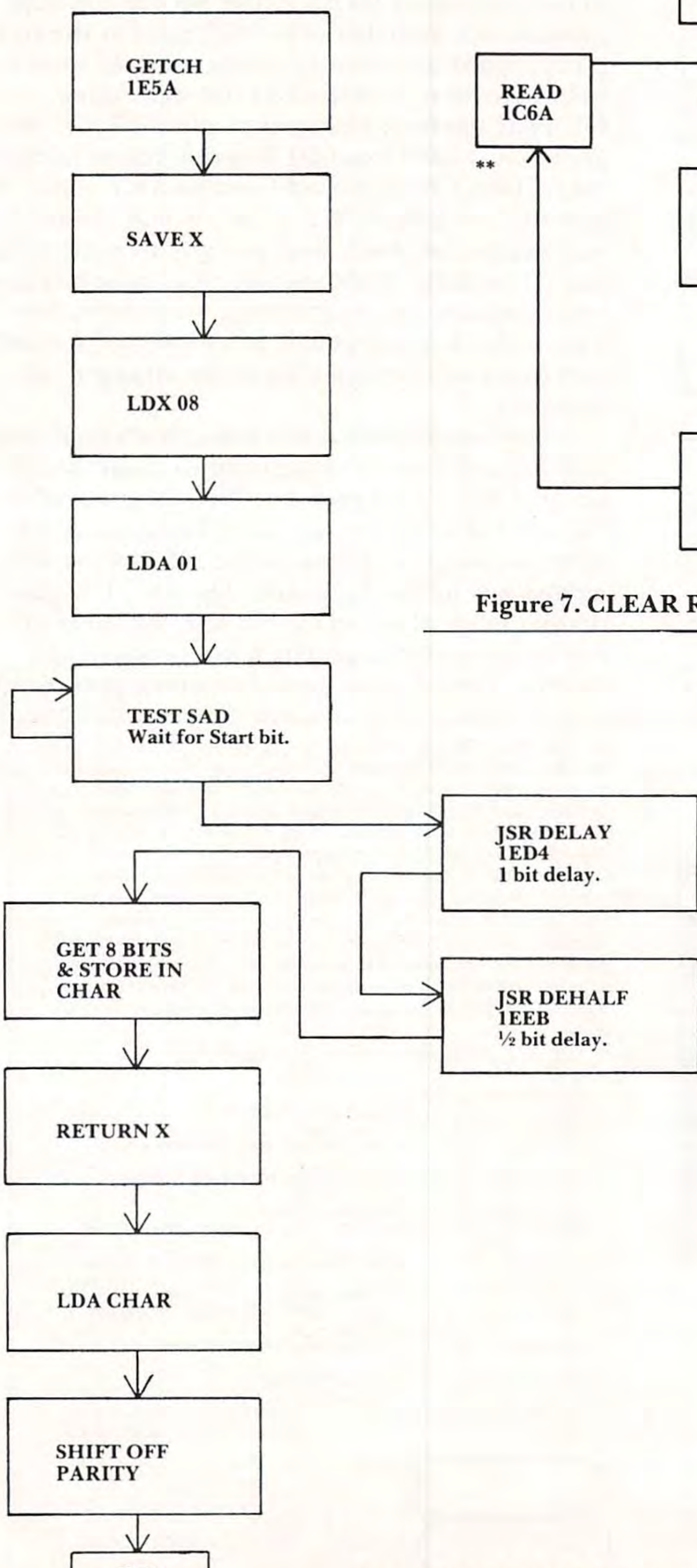


Figure 8. GETCH Subroutine

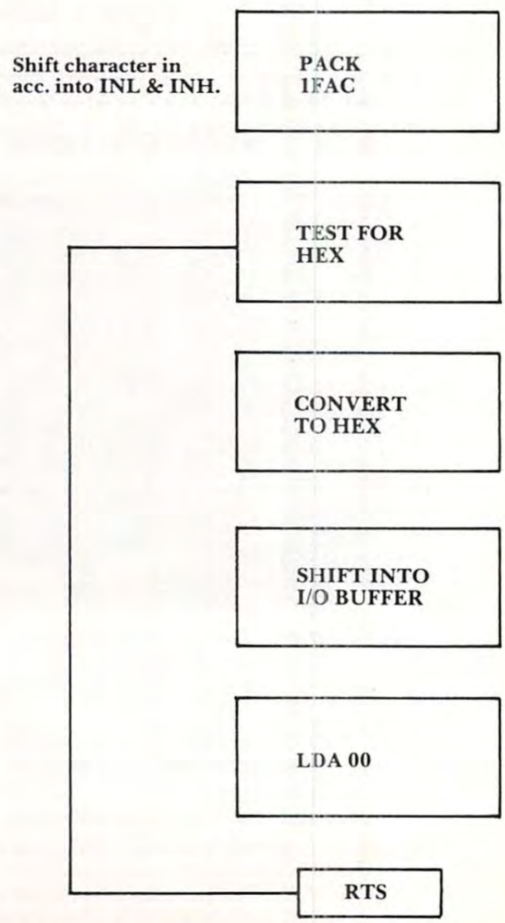


Figure 9. PACK Subroutine

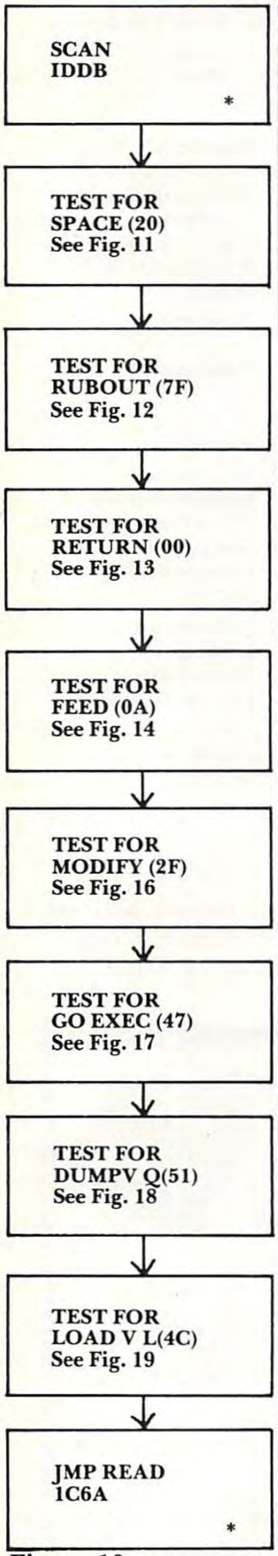


Figure 10. SCAN Routine

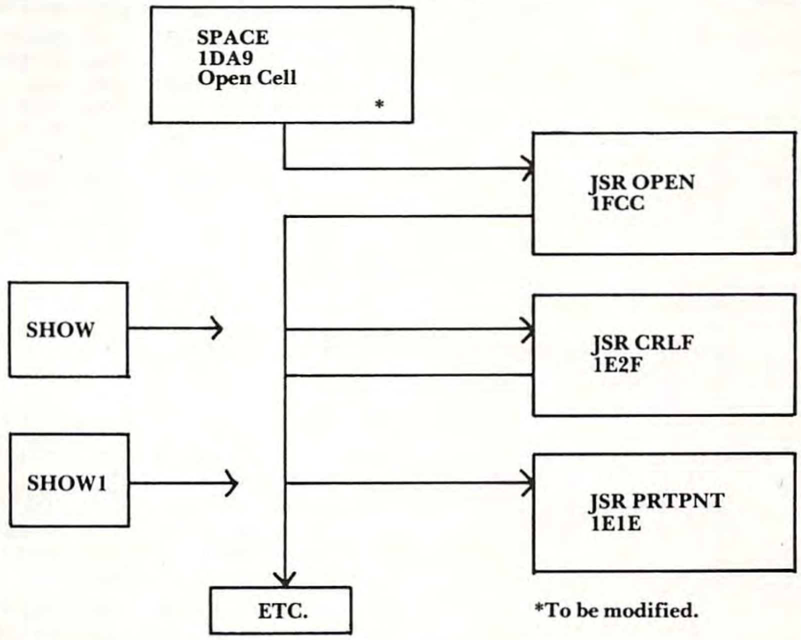


Figure 11. SPACE Routine

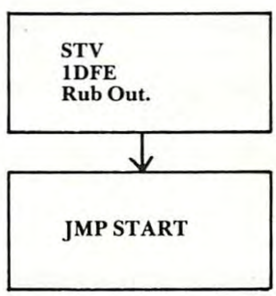


Figure 12. STV Routine

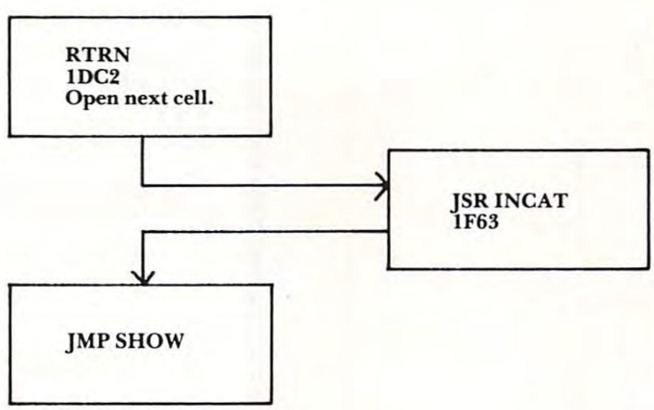


Figure 13. RTRN Routine

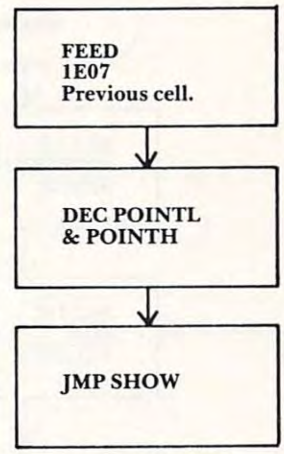


Figure 14. FEED Routine

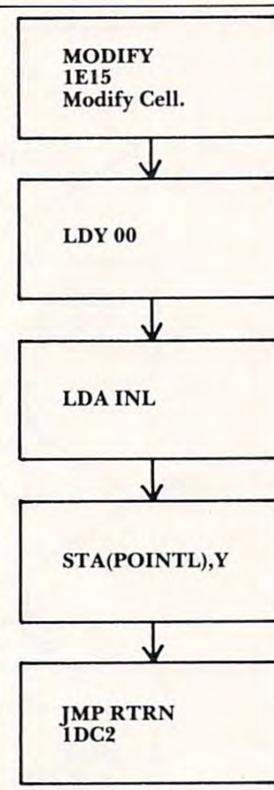


Figure 16. MODIFY Routine

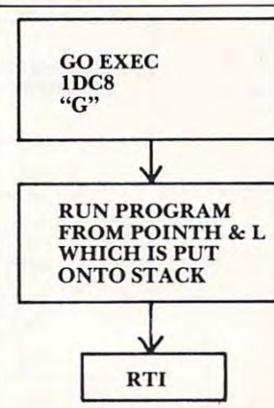
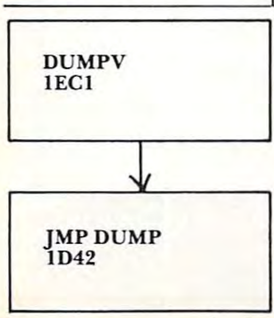


Figure 17. GOEXEC Routine



Dump from open cell to hi limit. Prints out memory.

Figure 18. Dump "V" Routine

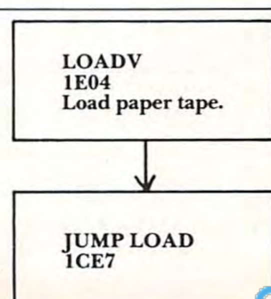


Figure 19. Load "V" Routine

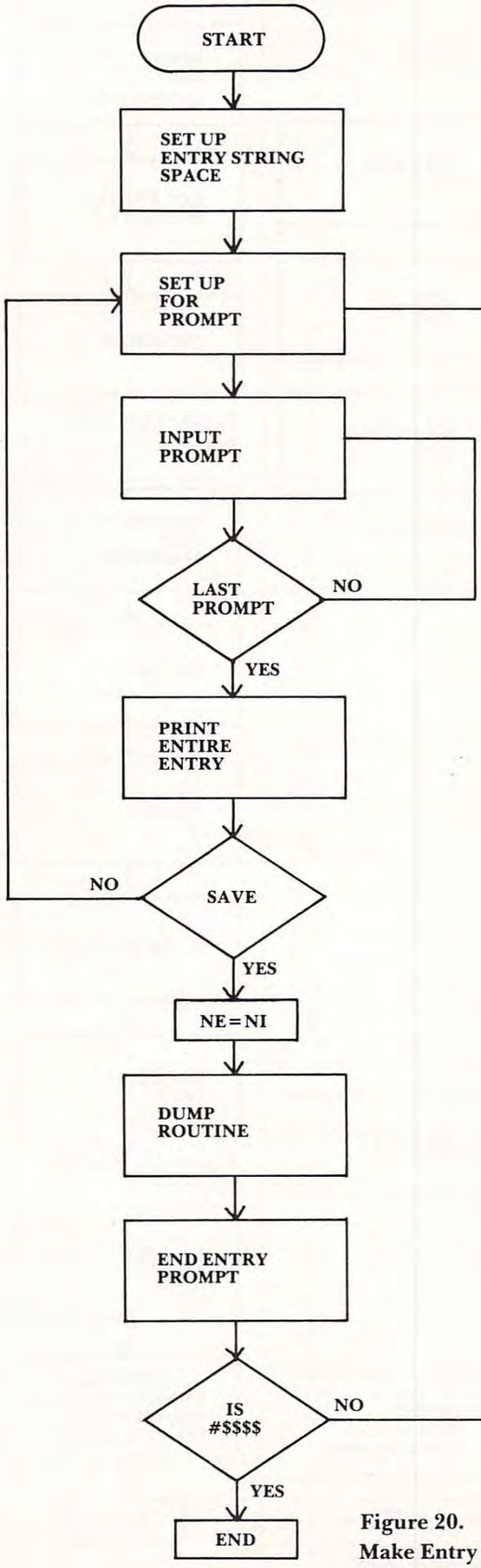


Figure 20.
Make Entry Routine

Assembly Language Program for Cassette DUMP & LOAD

```
0300 A5 01     DELAY  LDA DELTIM Load delay
  02 85 EE            STA TIMEA    value.
  04 A9 30     DECB   LDA 30     Load
  06 8D 04 17         STA 1704     timer.
  09 2C 07 17  TEST   BIT 1707   Test timer.
  0C 10 FB            BPL TEST   Branch if not run out.
  0E C6 ED            DEC TIMEB  Reduce time value.
  10 D0 F2            BNE DELB   Start again.
  12 C6 EE            DEC TIMEA  Reduce delay value.
  14 D0 EE            BNE DELA   Branch if not done.
  16 60               RTS        Return.

0317 A9 02     TWRITE LDA #02    Turn tape on.
  19 10 02            BPL TAPE
  1B A9 01     TREAD  LDA #01    Turn tape off.
  1D 4D 03 17  TAPE   EOR 1703
  20 8D 03 17         STA 1703
  23 60               RTS        Return.

0324 20 17 03  WRITE  JSR TWRITE Turn tape on.
  27 20 00 03         JSR DELAY  Delay for tape speed.
  2A 20 00 02         JSR HYPER  Record in hypertape.
  2D 20 17 03         JSR TWRITE Turn tape off.
  30 20 8C 1E         JSR INITI  Open ports again.
  33 60               RTS        Return.

03D5 20 1B 03  READ   JSR TREAD  Turn on tape.
  D8 20 36 03         JSR LOADT  Load tape.
  DB 20 1B 03         JSR TREAD  Turn off tape.
  DE 20 8C 1E         JSR INITI  Open ports again.
  E1 60               RTS        Return.
```

Note: HYPER is taken from The First Book of KIM page 119 relocated to address 0200.

LOADT is taken from the KIM-I User Manual Program listing page 6 relocated from address 1871-1931 to 0334-03D4.

If you wish to use the same routines in the same addresses as I did, send a SASE and I'll let you know what locations have to be changed in those listings to get it to run right. ©

CAPUTE!

Corrections/Clarifications

From Raymond Diedrichs ("Pet File I/O In Machine Language", April, 1981, Issue 11, pp. 144-145):

"In the machine language open statement, the following lines are missing:

LDA #DEVICE-NUMBER
STA $D4
LDA #SECONDARY-ADDRESS
STA $D3.

They should appear directly below the line which reads:

STA $D2.

And here's the missing listing from Charles Brannon's "String Arrays in Atari BASIC," April, 1981, Issue 11, p. 103.

```
100 REM SIMPLE BAR GRAPH PROGRAM
110 GRAPHICS 0
120 PRINT "NUMBER OF COMPANIES";
130 INPUT NC
140 MAXLEN=20:DIM A$(MAXLEN*NC),L(NC),
    ¬A(NC),T$(MAXLEN)
150 FOR I=1 TO NC
160 T$="                    ":REM 20 ¬
    ¬SPACES
170 E=I:GOSUB 20000
180 PRINT "ENTER THE NAME OF COMPANY ";I
190 INPUT T$:GOSUB 20000
200 PRINT "AMOUNT FOR ";T$;
210 INPUT A:A(I)=A:IF A>HI THEN HI=A
220 PRINT:NEXT I
230 GRAPHICS 0
240 FOR I=1 TO NC
250 E=I:GOSUB 30000
260 PRINT:PRINT T$
270 FOR J=1 TO (A(I)/HI)*30
280 PRINT CHR$(160);
290 NEXT J
300 NEXT I
310 END
20000 L=LEN(T$):IF L>MAXLEN THEN ¬
    ¬L=MAXLEN
20010 L(E)=L:START=(E-1)*MAXLEN+1
20020 A$(START,START+L-1)=T$:RETURN
30000 START=(E-1)*MAXLEN+1
30010 T$=A$(START,START+L(E)-1):RETURN
READY.
```

Program Listings for COMPUTE

Cursor control characters will appear in source listings as shown below:

```
h=HOME           , ĥ=CLEAR SCREEN
↓=DOWN CURSOR    , ↑=UP CURSOR
→=RIGHT CURSOR,  ←=LEFT CURSOR
r=REVERSE        , r̂=REVERSE OFF
```

Graphics (i.e. shifted) characters will appear as the unshifted alphanumeric character with an underline. This does not apply to the cursor control characters. The Spinwriter thimble doesn't have a backarrow symbol, so a "~" is used instead.

The "¬" is used to indicate the beginning of a continuation line. It is also used to indicate the end of a line which ends with a space. This prevents any spaces from being hidden.

New Products

THE CARVERY

The Carvery
156 Front Street West
Toronto, Ontario M5J 2L6
Telephone 977-7770

New Toronto Restaurant

Why does a computer pour the drinks at the Carvery, a downtown Toronto restaurant? Jim Butterfield, who is a small shareholder in the establishment, has no comment. Neither does he explain why the machine always pours him doubles.

Atari Launches Major Software Acquisition Program

Sunnyvale, California — April 3, 1981 — A major new effort to expand the library of consumer-oriented software for its personal computer systems is being launched by Atari, Inc. Atari is looking for high quality programs that can be used immediately, and easily, by people with little or no training in the use of computers.

"We want to acquire software in the areas of personal finance, self-improvement, education and home entertainment. We are encouraging the creation and marketing of software by vendors and developers, and want to help market appropriate materials from outside authors," Bruce W. Irvine, vice president of software for Atari's Computer Division said. "To start things off, we are sponsoring a $100,000 contest for software authors."

The acquisition program involves the creation of Atari Software Acquisition Program regional centers where qualified developers can work with Atari equipment and receive technical assistance, and Atari Program Exchange, a free quarterly catalog of user-written software to be distributed to Atari computer owners. In addition, Atari will offer periodic technical seminars for qualified software authors to familiarize them with the inner workings of Atari computer products and enable them to write programs that take advantage of all the advanced features of the ATARI 400™ and 800™ computers.

"We recognize that a broad selection of readily available software is a critical key to the ultimate consumer market. No one company can create the amount of material needed to properly address the market, so we are going to do our best to encourage our users and software vendors to create programs compatible with Atari computers. Often, a user or developer is an expert in a field we don't know much about; with our assistance, that person can make his or her programs available to the wide audience they deserve," Irvine added.

Acquisition Centers

Beginning with an initial installation in the Sunnyvale area which will open in mid-May, Atari will develop software acquisition centers in geographical areas where there are high concentrations of programmers and users, such as metropolitan areas with technical universities. No timetable has been announced for the opening of these additional facilities.

Qualified developers will be able to use the centers on an appointment only basis. Design of the centers will help insure the privacy of material under development. Each center will be equipped with Atari computers annd peripherals, all necessary reference materials and technical manuals. Center staff will help answer technical questions and review and evaluate completed software.

Once a program is completed, Atari may be interested in marketing it under the company name, or accept it for listing in the Atari Program Exchange catalog. Or, developers may wish to market the program on their own.

Pet User Group Celebrates Third Birthday

As shown in the birthday cake picture above, SPHINX celebrated their third year with a full sized Pet cake (complete with keyboard and message on the screen) at their meeting March 14, 1981 at the Lawrence Hall of

Science, Berkeley, California.

Originally formed by Niel Busey and Milt Lee, SPHINX, (Society For Pet Handlers Information Exchange), cooperated with Lawrence Hall of Science in putting out a newsletter which contained basic information about the Pet when there was little from the manufacturer.

Although the newsletter has been discontinued, they are still active in exchanging programs. At the sixth West Coast Computer Fair, April 5, 1981, a proposal was made that librarians from user groups across the United States trade programs on a disk basis. To this end SPHINX would like to receive 2040 or 4040 format disks from other groups and will return the diskette(s) with programs from our library (currently 13 diskettes and growing.)

Other current SPHINX projects are a nationwide Pet/CBM telephone network for Pet users with modems. Some interest in sponsoring this has been shown by Commodore. SPHINX also plans to start a library for the VICcolor computer because of the tape and software compatability. Many of their programs will run on a VIC with minor or no modifications.

For further information, please write to SPHINX C/O their sponsor:

PC Computers
10166 San Pablo Ave.
El Cerrito, CA 94530

Meetings in the Bay Area are the only way SPHINX currently exchanges individual programs — the second Thursday of the month at Lawrence Hall of Science, Chem. Lab, Berkeley, CA at 7:00 p.m.

New Low-cost 80-Column Dot Matrix Printer

MICROTEK, Inc. has announced a new low-cost (under $300) 80-column dot matrix printer. Dubbed the "BYTEWRITER-1", the printer accepts single sheet or roll paper up to 8½ inches wide and prints at 60 lines per minute using a 7 x 7 dot matrix.

The BYTEWRITER-1 interface is similar but not identical to a Centronics parallel interface, and has been designed specifically to operate with the Apple II, the Atari 400/800, and all models of the TRS-80. Using a print mechanism and logic board designed and manufactured in the U.S., the unit is priced at $299 (interface cable slightly extra). MICROTEK is directing its marketing efforts towards the personal computing and hobbyist segments of the market, and will sell the printer direct only. The BYTEWRITER-1 carries a 90-day limited warranty. Delivery is from stock to 60 days.

For further information, contact Diane Barney-Laukat at MICROTEK, INC., 9514 Chesapeake Drive., San Diego, California 92123. (714)-278-0633.

High Performance Data Communications System

Norcross, Georgia — Hayes Microcomputer Products, Inc., announces the Hayes Stack Smartmodem high performance data communications system for small computers.

The Smartmodem, an FCC-approved direct-connect device, is designed for use with RS-232C compatible computers or terminals to communicate via the telephone system with other computers or time sharing systems.

A unique feature is that the Smartmodem can be program controlled in any language by ASCII character strings.

This intelligent datacomm system analyzes and executes commands and in response sends result codes which, at the user's discretion, can be English words or decimal digits. The Smartmodem has auto dial and auto answer capabilities. A special design feature is that all circuitry required for auto dial and auto answer is installed within the Smartmodem. This eliminates the need for any auxiliary equipment and makes the Smartmodem a stand-alone system.

The Hayes Stack Smartmodem can be connected to any telephone system in the U.S. since dialing can be either Touch-Tone* or pulse. Furthermore, both dialing modes can be combined within a command with pulse being used, for example, to access a PBX board and Touch-Tone for dialing an outside number after the second dial tone is received.

An audio monitor permits the user to follow the progress of the call and be alerted to wrong numbers and busy signals. If a busy signal is encountered, by entering a repeat command, the Smartmodem will automatically redial the number at any time.

Operation can be in full or half duplex with a data rate of 0 – 300 baud. Power-on default options are controlled by the

DOWN-TO-EARTH PRICES ON OUT-OF-THIS-WORLD PERSONAL COMPUTERS AND COMPONENTS.

Look at this!

Ohio Scientific Superboard II $299

- It's the first complete computer system on a board.
- Superboard II uses the ultra powerful 6502 Microprocessor
- 8K Microsoft BASIC-in-ROM
- 4K static RAM on board, expandable to 8K
- Full 53-key keyboard, with upper and lower case. Plus user expandability.
- Video interface and audio cassette interface.

The Ohio Scientific Superboard II at $299 — in today's economy — has got to be the best buy by far. It will entertain you with spectacular graphics made possible by its ultra high resolution graphics and super fast BASIC. It will help you in school or industry, as an ultra powerful scientific calculator. Advanced scientific functions and a built-in "immediate" mode allow you to solve complex problems without programming.

The Superboard II can be expanded economically, for business uses, or to remotely control your home appliances and security. Even communicate with other computers.

Read what's been written about Superboard II:

"We heartily recommend Superboard II for the beginner who wants to get into microcomputers with a minimum cost. A real computer with full expandability."
—POPULAR ELECTRONICS, MARCH 1979

"The Superboard II is an excellent choice for the personal computer enthusiast on a budget."
—BYTE, MAY 1979

Look at these easy hardware prices:

610 Board For use with Superboard II and Challenger 1P. 8K static RAM. Expandable to 24K or 32K system total. Accepts up to two mini-floppy disk drives. Requires +5V @4.5 amps.	$ 298
Mini-Floppy Disk Drive Includes Ohio Scientific's PICO DOS software and connector cable. Compatible with 610 expander board. Requires +12V @1.5 amps and +5V @ 0.7 amps. [Power supply & cabinet not included.]	299
630 Board Contact us for important details.	229
AC-3P 12" combination black and white TV/video monitor.	159
4KP 4K RAM chip set.	79
PS-005 5V 4.5 amp power supply for Superboard II.	45
PS-003 12V power supply for mini-floppies.	45
RF Modulator Battery powered UHF Unit.	35
CS-900B Metal case for single floppy disk drive and power supply. [While stock lasts.]	49
AC-12P Wireless remote control system. Includes control console, two lamp modules and two appliance modules, for use with 630 board.	175
AC-17P Home security system. Includes console, fire detector, window protection devices and door unit for use with 630 board.	249
C1P Sams C1P Service manual	8
C4P Sams C4P Service manual	16
C3 Sams Challenger III manual	40

Ohio Scientific and independent suppliers offer hundreds of programs for the Superboard II, in cassette and mini-floppy form.

Freight Policies All orders of $100 or more are shipped freight prepaid. Orders of less than $100 please add $4.00 to cover shipping costs. Ohio residents add 5.5% Sales Tax.

VISA / Master Charge

Hours: Call Monday thru Friday.
8:00 AM to 5:00 PM E.D.T.
TOLL FREE: 1-800-321-5805

Guaranteed Shipment
Cleveland Consumer Computers & Components guarantees shipment of computer systems within 48 hours upon receipt of your order. **Our failure to ship within 48 hours entitles you to $35 of software, FREE.**

To Order: Or to get our free catalog **CALL 1-800-321-5805 TOLL FREE.** Charge your order to your **VISA** or **MASTER CHARGE** account. Ohio residents call: (216) 464-8047. Or write, including your check or money order, to the address listed below.

CLEVELAND CONSUMER COMPUTERS & COMPONENTS
P.O. Box 46627
Cleveland, Ohio 44146

Order Form: CLEVELAND CONSUMER COMPUTERS & COMPONENTS P.O. Box 46627 Cleveland, Ohio 44146

- ☐ Superboard II $299.
- ☐ 610 Board $298.
- ☐ Mini-Floppy Disk Drive $299.
- ☐ RF Modulator $35.
- ☐ AC-3P 12" B/W Monitor $159.
- ☐ C1P Sams Manual $8.

[Attach separate sheet for other items.]

NAME ______

ADDRESS: ______

CITY: ______ STATE: ______ ZIP: ______

PHONE: ______

Payment by: VISA ______ MASTER CHARGE ______ MONEY ORDER ______

Credit Card Account # ______

Expires ______ Interbank #[Master Charge] ______

TOTAL CHARGED OR ENCLOSED $______ [Ohio Residents add 6.5% Sales Tax]

Orders of less than $100, please add $4.00 to cover shipping costs. Orders will be accepted from U.S. and Canada only. All prices quoted are U.S., date of publication, standard UPS shipping FOB the factory. C

EAST COAST

ΩMEGA Sales Co.
12 Meeting St.
Cumberland, RI 02864

1-800-556-7586
1-401-722-1027

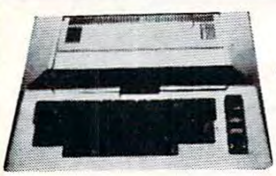
ΩMEGA SALES CO.

SPECIAL OF THE MONTH!!

INTERTEC SUPERBRAIN 64K RAM $2849

OR TOGETHER FOR ONLY $5195

NEC 5510 SPINWRITER $2395



16K ATARI 800 $759

DIABLO 630 $1995
Tractor option $200

OKIDATA MICROLINE-83	$949
OKIDATA MICROLINE-80	$469
APPLE II PLUS 48K	$1150
APPLE DISK w/ 3.3 DOS Controller	$545
APPLE DISK w/o Controller	$435
BASE II PRINTER	$649
HAZELTINE 1420	$799
NORTHSTAR HORIZON II 32K QD	$2975
ATARI 400 8K	$399
ATARI 400 16K	$499

- ***NO SURCHARGE FOR CREDIT CARDS***
- ***WE ACCEPT C.O.D.'S***
- ***ALL EQUIPMENT FACTORY FRESH W/MANUFACTURER'S WARRANTY***
- ***STOCK SHIPMENTS USUALLY SAME DAY OR NEXT DAY***
- ***NO HIDDEN CHARGES - WE LIVE BY OUR PUBLISHED PRICES***

ΩMEGA SALES COMPANY

positioning of seven option switches. Four of these options can be overridden by software command. LED status indicators on the front panel of the unit provide a visual check of the Smartmodem's operational status.

In addition, the unique "Set" commands allow the user to select (and change) various operational parameters such as dialing speed, escape code character and number of rings to answer on.

In announcing the release of the Smartmodem, Glenn Sirkis, Hayes Vice President, stated, "The Smartmodem, offers all the classic modem functions *plus* some special features — e.g., pulse and Touch-Tone dialing — that are available only with a limited number of modems. Add to this the features that are unique to the Smartmodem — e.g., programmable in any language and Set commands for customized operation — and you'll know why we believe the Smartmodem is everything you could ever want in a 300 baud modem."

The Smartmodem has a Two Year Limited Warranty. The suggested retail price for the Hayes Stack Smartmodem system is $279.00. Included in this price are the Smartmodem unit, a power pack, one modular telephone cable to connect the unit to the telephone line and an owner's manual.

The Smartmodem is the first product in a new series that features the exclusive Hayes Stack design. This compact design permits other Hayes components to be stacked on top of the Smartmodem, thereby eliminating clutter.

TM Trademark of Hayes Microcomputer Products, Inc.
* Trademark of American Telephone and Telegraph.

New Professional Applications Package For The Medical Profession

Charles Mann & Associates, Micro Software Division, has announced the release of a new professional applications package for the Medical Profession called "Medirec". The Medirec system is a total Medical History and Report Preparation System. The professional using the system can prepare office input forms, enter patient and family histories, record patient visit symptoms, diagnosis, and treatments, prepare referral requests, prepare patient history summaries, and prepare referral reports. The program compliments the firm's existing Medical Billing Package.

Medirec is designed with today's professional practice liability in mind. The system allows the diskette recording of up to 550 professional visits per diskette. Individual patient records can be recalled, linked together and printed either in whole or in part. The system allows the practitioner to search past history files for common symptoms, diagnosis or the administration of conflicting drug treatments.

The system can recall records for past due follow treatment, prepare reminder notices, prepare liability release forms and print file folder labels. The system comes with a full featured address data base system and a programmable form letter writing element. The system can be programmed to prepare referral report letters, and requests for specialist treatment.

The Medirec system requires a 48K Apple II, Apple II +, or Apple III, an 80 column printer, and two disk drives. A special Corvus Systems hard disk version is also available for system configurations up to 40MB of on line storage.

The system is available from over 700 CMA dealers worldwide for an introductory price of $199.95 (Corvus version $249.95). Preview Documentation is available for $25.00. Additional information and dealer location information can be obtained from Charles Mann & Associates, Micro Software Division, 7594 San Remo Trail, Yucca Valley, CA 92284. Phone (714) 365-9718.

NYSAEDS Conference

On October 18, 19 and 20, 1981, The New York State Association for Educational Data Systems (NYSAEDS) will hold its annual conference in Syracuse, NY. NYSAEDS, an affiliate of AEDS, is composed of people who have a common interest in computers and education.

The theme of this year's conference is "Software". The keynote speaker is Marge Kosel from MECC and the banquet speaker is Dr. Earl Joseph (Futurist) from Sperry Rand. A variety of workshops will be held concerning the uses of microcomputer software in education.

For further information, please contact Don Ross, Ardsley High School, Ardsley, NY 10502.

New Power Line Filter

Pilgrim Electric Company, Plainview, New York, introduces new, compact "Plug-In" style VOLTECTOR®. It provides the most cost-effective protection against voltage spikes, surges, transients and high frequency interference for Word-Processors, Microcomputers and other Microprocessor-based equipment.

The unique heavy duty design meets the latest industry Surge Voltage Standards and provides the highest level of commercially available two-way protection. It assures electronic equipment of clean incoming AC power ... and it prevents disturbances generated by equipment from being conducted back into the power line. The VOLTECTOR® Series 6 Plug-In Power Conditioner supplies the positive surge-protection and filtering that most manufacturers have left out of their equipment.

Available with current ratings of 1, 2, 5, 7 and 10 amps.

Prices: Begin at $79.50
Delivery: Stock

Manhattan Software Announces Four Atari Game Programs

Manhattan Software, long a publisher of programs for the TRS-80, has begun issuing a series of programs for the Atari Computer. The first four releases are:

Gin Rummy 3.0, with color card graphics and sound, which plays a full regulation game of Gin, and can hold its own against even skilled Gin players. Prices at $19.95, the program requires 32K memory and one joystick.

Casino Blackjack/Counter, a realistic simulation of playing at a casino table — card graphics show five hands dealt, and the user plays the center hand while the computer plays the rest. A major purpose of the program is to teach card-counting, a method which is claimed to give the player a statistical advantage over the house in some situations. Priced at $19.95, for 24K and one joystick.

Labyrinth Run, a test of skill and coordination, using the joystick to guide a fast-moving runner through twists, turns, reverses and slaloms, with thunderous crashes when the runner hits a wall. Three skill levels. $14.95, the game requires two joysticks.

These programs are available at dealers, and direct from Manhattan Software, P.O. Box 35, Pacific Palisades, CA 90272. Telephone (213) 454-8290.

Atari Adds Missile Command To Its Video Computer System Game Library

Missile Command™, a popular coin operated video game currently in arcades, is now available in a home video game version, it was announced today by Atari, Inc., creator and manufacturer of both products.

Largely due to Missile Command's success as an arcade game and in response to considerable consumer demand, Atari designed the game cartridge for its Video Computer System™ programmable TV game.

The Missile Command game cartridge is a one or two player game that uses joysticks and offers 34 game variations.

According to Michael J. Moone, president of the Consumer Electronics Division, "Missile Command is one of the most challenging skill and action video games ever created. We believe its popularity will be as pervasive as that of its predecessors, Space Invaders and Asteroids."

The game begins with wave after wave of enemy missiles raining down on an earth missile base and 6 surrounding cities. The player, as base commander, is responsible for protecting and defending the territory from enemy attack. To combat each wave of enemy missiles, the base commander is given 30 guided defense missiles which when exploded in the path of attacking missiles destroys them. Each successive wave of attacking missiles comes faster than the previous one and the game continues until all cities and the missile base are lost.

Additional features include game difficulty adjustment to correspond to player skill levels, slow game variations designed for young children and screen color changes as game progresses to reduce eyestrain during extended game play.

Suggested retail price is $31.95 and cartridges will be available nationwide by April.

New Product releases are selected from submissions for reasons of timeliness, available space, and general interest to our readers. We regret that we are unable to select all new product submissions for publication. Readers should be aware that we present here some edited version of material submitted by vendors and are unable to vouch for its accuracy at time of publication.

Big Savings On Atari & PET!

No Risk - No Deposit On Phone Orders - Shipped Same Day You Call° - C.O.D. or Credit Card

* On all in stock units

Please Call Between 11AM & 6PM (Eastern Standard Time)

(800) 233-8950

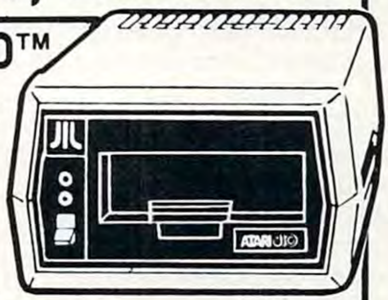
ATARI® 800™ PERSONAL COMPUTER

List $1080

$759

ATARI®

ATARI® 810™ DISC DRIVE



List $599.95

New Low Price only $489.00

Maxell Disks	10 for $34
Sycom Disks	10 for 29
Atari Disks	5 for 22

400 8K	$419
400 16K	449
410 Recorder	62
815 Disk	1199
822 Printer	359
825 Printer	779
830 Modem	159
850 Interface Module	179
CX852 8K RAM	94
CX853 16K RAM	149
CX70 Light Pen	64
CX30 Paddle	18
CX40 Joystick	18
CX86 Printer Cable	42
CO16345 822 Thermal Printer Paper	5
CAO16087 825 80-col. Printer Ribbon (3/box)	17
CX4119 Conversational French	45
CX4118 Conversational German	45
CX4120 Conversational Spanish	45
CX4125 Conversational Italian	45
CXL4009 Chess	30
CXL4011 Star Raiders™	45
CXL4004 Basketball	30
CXL4006 Super Breakout™	30
CXL4010 3-D Tic-Tac-Toe	30
CXL4005 Video Easel™	30
CXL4007 Music Composer	45
CXL4015 TeleLink™	20
CXL4002 BASIC Computing Language	46
CXL4001 Education System Master	21
CXL4003 Assembler Editor	45

CX4115 Mortgage & Loan Analysis	$13
CX4101 An Invitation to Programming 1	17
CX4106 An Invitation to Programming 2	20
CX4117 An Invitation to Programming 3	20
CX4107 Biorhythm	13
CX4103 Statistics I	17
CX4121 Energy Czar	13
CX4108 Hangman	13
CX4102 Kingdom	13
CX4112 States & Capitals	13
CX4114 European Countries & Captials	13
CX4105 Blackjack	13
CX4111 Space Invaders	18
CX8106 Bond Analysis	20
CX8107 Stock Analysis	20
CX8108 Stock Charting	20
CX4104 Mailing List	17
CX4110 Touch Typing	20
CX8102 Calculator	24
CX4109 Graph It	17
CX4120 Conversational Spanish	45
Talk & Teach Courseware; CX6001 thru CX6017	23

Combination Special!

825 Printer & 850 Interface

825 Lists for $1000
Regular Mail Order Price: $779
850 Lists for $220.
Regular Mail Order Price: $179

Save $20.00

Buy Both For Only $938

CX8104 Atari 810 Master Diskette II

New DOS 2 Operating System Master

only $21.00

Microtek RAM 16K or 32K

- Full 1 year warranty
- Compatible with 400 or 800
- Assembled and Tested

16K	$ 99
32K	189

Atari Specialists . . .

We Carry It All

commodore

Commodore Computers:

4032 N	$1090
4032 B	1090
8032	1499

Commodore Peripherals:

CBM 4022 Printer	675
CBM 4040 Dual Drive Floppy Disk	1090
CBM 8050 Dual Drive Floppy Disk	1420
CBM C2N Cassette Drive	87
Tally 8024 Printer	1679

EBS Accounts Receivable Inventory System	$695
Dr. Daley Mailing List	129
Dr. Daley Inventory	89
OZZ Information System	329
BPI General Ledger	329
Tax Package	399
Dow Jones Portfolio Management	129
Pascal	239
PET to IEEE Cable	37
IEEE to IEEE Cable	46

NEW -

8096	1890
VIC - 20	299

Software

WordPro 3 (40 col.)	$186
WordPro 4 (80 col.)	279
WordPro 4 Plus (80 col.)	339
Visicalc - Apple	$122
Atari	163
PET	163

Printers

NEC 5530	$2495
Diablo 630	2195
Trendcom 100	299
Trendcom 200	489
Paper Tiger 445G	769
Paper Tiger 460G	1219
Epson MX-80	539

To Order:

Phone orders invited (800 number is for order desk only). Or send check or money order. Equipment Shipped UPS collect. Pennsylvania residents add 6% sales tax. Add 3% for Visa or MC. Equipment is subject to price change and availability without notice.

Computer Mail Order
501 E. Third St.
Williamsport, PA 17701
(717) 323-7921

Advertiser's Index

The 6502 Resource Magazine
COMPUTE! PET•ATARI•APPLE OSI•KIM•SYM•AIM

My computer is:
☐ **PET** ☐ **APPLE** ☐ **ATARI** ☐ **OSI**
☐ **KIM** ☐ **SYM** ☐ **AIM** ☐ **OTHER**______________
☐ **Don't yet have one...**
During the next year I expect to buy:
☐ **computer** ☐ **printer**
☐ **disk drive** ☐ **other peripherals**

☐ Please enter my 1 year (12 issue) subscription to COMPUTE!
☐ New subscription ☐ Renewal subscription
☐ $20.00 U.S. subscription
☐ $25.00 Canada and international surface mail: Payment in U.S. funds, drawn on U.S. bank or money order.
☐ Bill me (U.S. only). $1.00 billing fee.

Name, Address: ____
____
____
____

Charge my: ☐ Visa ☐ MC
Number______________________________ Expires__________/__________/
5 6 7 8 9 10 11 12

The Editor's Feedback:

My computer is:
☐ **PET** ☐ **APPLE** ☐ **ATARI** ☐ **OSI**
☐ **KIM** ☐ **SYM** ☐ **AIM** ☐ **OTHER**______________
☐ **Don't yet have one...**
During the next year I expect to buy:
☐ **computer** ☐ **printer**
☐ **disk drive** ☐ **other peripherals**

Content:
Best Article This Issue (page #, title)

____

Other suggestions:

5 6 7 8 9 10 11 12

The 6502 Resource Magazine
COMPUTE! PET•ATARI•APPLE OSI•KIM•SYM•AIM

My computer is:
☐ **PET** ☐ **APPLE** ☐ **ATARI** ☐ **OSI**
☐ **KIM** ☐ **SYM** ☐ **AIM** ☐ **OTHER**______________
☐ **Don't yet have one...**
During the next year I expect to buy:
☐ **computer** ☐ **printer**
☐ **disk drive** ☐ **other peripherals**

☐ Please enter my 1 year (12 issue) subscription to COMPUTE!
☐ New subscription ☐ Renewal subscription
☐ $20.00 U.S. subscription
☐ $25.00 Canada and international surface mail: Payment in U.S. funds, drawn on U.S. bank or money order.
☐ Bill me (U.S. only). $1.00 billing fee.

Name, Address: ____
____
____
____

Charge my: ☐ Visa ☐ MC
Number______________________________ Expires
5 6 7 8 9 10 11 12

Place
Stamp
Here

COMPUTE.
Post Office Box 5406
Greensboro, NC 27403

Place
Stamp
Here

COMPUTE!
Post Office Box 5406
Greensboro, NC 27403

Place
Stamp
Here

COMPUTE!
Post Office Box 5406
Greensboro, NC 27403

The Great American Solution Machine.

Meet Commodore. The business computer that's providing solutions for more than 100,000 people all over the world. Built by one of the pioneers in office machines. With a reputation for quality that can only come from vertical integration of the total manufacturing process. Commodore builds, not assembles.

Compare Commodore's word and data processing capabilities with computers costing twice or even three times as much. You'll see why so many small businesses are turning to Commodore for solutions to problems as varied as these:

☐ A car leasing company's customers were terminating too early for account profitability. Solution: A 16K Commodore. It analyzes cash flow on over 1200 accounts, identifies those for early penalties, and even writes up lease contracts. Commodore paid for itself within weeks.

☐ A law firm needed a high quality, easy-to-use, affordable word processing system. Solution: Commodore plus its WordPro software package. At a $6,000 savings.

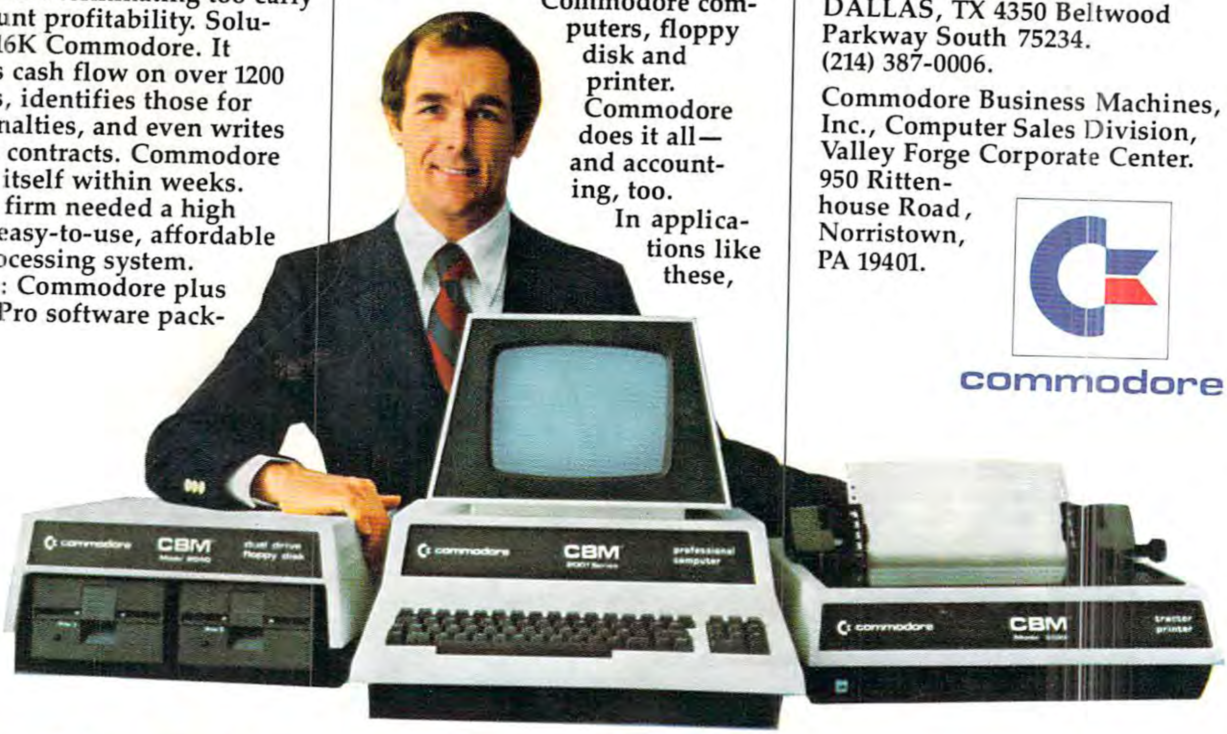
☐ A gasoline retailer needed to inventory, order and set prices; determine Federal and state income taxes; and comply with Federal pricing and allocation regulations. All done daily, weekly, monthly and yearly. Solution: Commodore. It keeps his business on track—and Uncle Sam off his back.

☐ A paint and wallpaper store had to inventory over 600 expensive wallpaper lines for profitability, monitor distributor sales, set and track salesmen's goals, and help the customer select the right size, pattern and quantity. Solution: Two 32K Commodore computers, floppy disk and printer. Commodore does it all—and accounting, too.

In applications like these, and many more, Commodore solves the problems that stand in the way of increased profitability. Commodore can provide the solution in your Great American business, too. Find out more by calling or writing any of Commodore's District Sales Offices.

COSTA MESA, CA 2955 N. Airway Avenue 92626. (714) 979-6307.
SANTA CLARA, CA 3330 Scott Boulevard 95051. (408) 727-1130.
DECATUR, GA 5360 Snapfinger Woods Drive 30035. (404) 987-3311.
BENSENVILLE, IL 790 Maple Lane 60106. (312) 595-5990.
NORRISTOWN, PA 950 Rittenhouse Road 19401. (215) 666-7950.
DALLAS, TX 4350 Beltwood Parkway South 75234. (214) 387-0006.

Commodore Business Machines, Inc., Computer Sales Division, Valley Forge Corporate Center. 950 Rittenhouse Road, Norristown, PA 19401.



commodore

COMMODORE