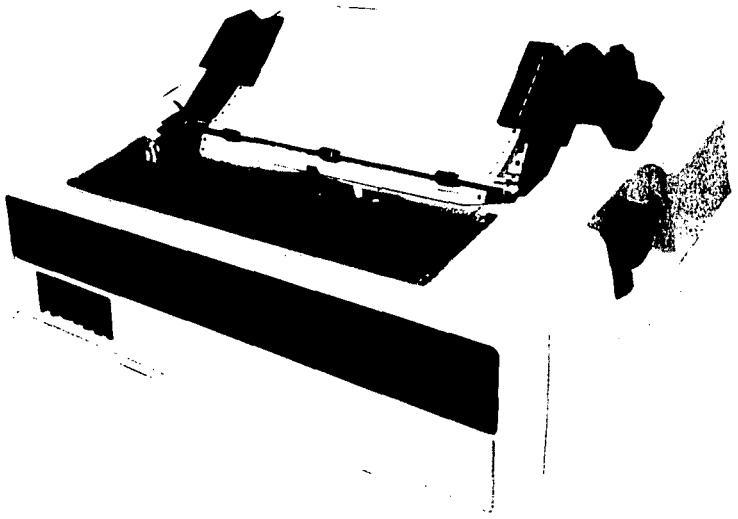
PEEK (65)

The Unofficial OSI Users Journal

Volume 2, No. 1
January 1981

P.O. Box 347
Owings Mills, Md. 21117
(301) 363-3267



INSIDE:

Column One

First, just a little housekeeping. If you move, please do send us your return address before you put the furniture on the truck. Reason is, the US Postal Service will not forward 3rd class mail (PEEK(65) included), so you will not get your issues after you move unless we put the new address on them. We will do our part: we will get your address into the file within a day or two. Do yours, too and we won't have to wonder why you have been "cut off."

The buyout of OSI by MACOM seems to be paying off already. Rumors and facts are flying around the OSI community which bode very well for those of us who must work with the gear. The factory is negotiating to have books written by good micro programmers on how to translate TRS-80 Basic and other popular Basics into OSI Basic. Offers are being tendered to the best programmers to come to Ohio and go to work.

I now have my own C3-D up and running, and will be working on software of interest to me and, I hope, to you in the next few months. Of course, complete reports will be upcoming. One topic of great interest to me is communications. I can hook up my Cat and use the terminal to interface with Micronet and OP (other people's) computers, but as yet have no program to allow me to transfer a disk file from my computer to yours. Nor can I run a program and have the output go into the line, so you can see it. Nor list it to your RAM, so you can save it. Nor, finally, can I send a message which will appear on your terminal ("something isn't working right, take your phone handset out of the modem and let's talk it over.."). What I am talking about starts with terminal emulation, and goes on into message transfer and file sharing. If I can get it running, I'll publish my phone number and hours of operation in PEEK(65), and you can call me up and share stuff with me!

I am also fascinated with all the benefits (real or imagined) of running under OS-CP/M. I have ordered a couple of disks, and will be trying out all the programs I can get my hands on. With results in an OS-CP/M column in PEEK(65), of course!

Naturally, if you have a couple of nifty routines which will do the sort of communicating I am talking about, or if you have been running OS-CP/M for years and would like to tell us all about it, I am no hog. Send in your contribution and we will print it!

TechNotes

by: Dick McGuire

Have you ever used PACKER and lost some of the first couple of files on the disk? Steve Wisely of H/B Computers in Charlottesville told me why. When you type "LOAD "PACKER", 65U reads the DIRectory on the current disk device. One of the things 65U wants to know is "how big is the directory". Typically the operator then switches devices to another drive and types RUN. PACKER assumes the directory size to be the same as 65U remembered it to be. If that's so, OK. However, often it won't be, particularly on a hard disk. It may even be at some other disk address than 25088. The easy fix is to add line 125 OPEN"DIREC*","PASS,1 : CLOSE. This OPEN will cause 65U to pick up the size of the new directory and all should be well.

USER DEFINED INPUT UNDER LEVEL III. Some time ago I wanted to do some fancy input editing in a program. I wrote in the usual manner:

```
100 WAIT 64512,1
110 CH=PEEK(64513) AND 127
```

This really worked neat, then I tried under Level III, and it didn't work at all. I realized that the memory addresses of the Port status and Data Registers were different under Level III. So I tried:

```
100 PS=64512
110 IF PEEK(14948)=76THEN
    PS=52736
120 PA=PS+1
130 WAIT PS,1
140 CH=PEEK(PA)AND127
```

Boy! This worked great (for the console; it didn't work at all for the other terminal). So I did this:

PEEK(65) is published monthly by:
DBMS, Inc., Owings Mills, MD 21117.
Editor: Al Peabody

Subscription rates:

US (surface mail)	$12/year
Foreign (air mail)	$20/year
Back issues	$1.50/ea

For back issues, subscriptions, change of address or other information, write to:

PEEK(65)
PO Box 347
Owings Mills, MD 21117

```
100 PS=64512
110 IF PEEK(14948)<>76
    THEN140
120 INPUT"USER NUMBER";UN
130 PS=52736+(UN-1)*2
140 PA=PS+1
150 WAIT PS,1
160 CH=PEEK(PA)AND127
```

This was ok for me as I usually had a very good idea of my user number. I had this customer who couldn't keep it straight. So I did this:

```
100 PS=64512
110 IF PEEK(14948)<>76
    THEN 140
120 PN=15-(PEEK(63232)AND15)
130 PS=52736+PN*2
140 PA=PS+1
150 WAIT PS,1
160 CH=PEEK(PA)AND127
```

Now this worked just great. It would work for all users and it worked for Level I. The only fly in the ointment was that operating under Level III an INPUT statement wouldn't work at all after I had done that sort of stuff for a while. (I don't know how much of a while, but a while). Tore my hair out. Along came the day when I had to get it running. Larry Hinsley of Software Consultants said something about interrupts and initializing the 6850 ACIA'S and it all clicked.

I reasoned that the following happened. The operator would poke a key on the key board which would generate an interrupt. The Level III executive would see this interrupt and disable the ACIA interrupt and wait for BASIC to come and service it. But BASIC would not come because it was doing something else, so the interrupt would just hang there. Now, BASIC comes along with an INPUT instruction a day late and a dollar short. We have the situation where BASIC is waiting for Level III and Level III is waiting for an interrupt: everyone is waiting including me - for ever, or until someone hit the reset button.

Now the fix is simple, after all the fancy stuff has been done - lines 100-160 above - add another line.

```
170 POKE PS,3: POKE PS,145
```

This gives the port a master reset (3) and an initialization (145).

Eureka! It works. Actually all that PEEK-POKE I/O was done in machine language, not Basic. The Basic worked, but was far too slow and is shown here for clarity only.

What have we learned here? 1) how to interrogate the input port without an INPUT statement. 2) How to switch the port addresses for Level III. 3) How to figure out which partition the program is executing in. 4) How to make it perform USER defined I/O and how to restore the machine when you are done. Go to it folks!

* * * * *

CASSETTE CORNER
by David A. Jones
8902 SW 17 Terrace
Miami, FL.

Originally I had intended to devote this month's column to the techniques of installing an EPROM in place of the ROM Monitor and some of the enhancements that this provides. After reading James Loos' excellent article in the November issue of Peek (65) though, I feel that much of this may now be redundent. However, since we used different approaches I think it may still be of interest to readers to see an alternate method. Also, I used locations $FCD5 to $FCFF for a dumb terminal routine and if anyone intends to implement all of the routines coming up in the following months it will be easier to use the same map. The only hardware change necessary is to isolate pin 21 of U13 from ground and wire it to 5V. Remove the ROM and install the EPROM.

The OSI ROM clears the screen as an inline routine at $FF24 whenever the BREAK key is depressed. There is a second inline screen clear routine at $FE0C executed whenever the MONITOR is entered. Changing one of these to a subroutine allows us to call it from BASIC with the USR(X) function. Since it is a machine language routine it is of course very fast. The first routine uses 17 bytes and the second 30 bytes plus 8 NOPS in front (OSI changed their minds about something I guess). The latter, occupying more space, lends itself to easier modification. To use the routine, POKE 11,14 : POKE 12,254. Thereafter X=USR(X) will clear the screen. Note that the entire screen will be cleared including the area

Software Packages

Evergreen Data Systems
Evergreen Tax Package US1040 Tax return package with selected state schedules **$1000**

Computerized Accounting & Tax Service
Taxman* US 1040 tax return preparation. Can handle 29 schedules. **$3000**

CMC
Legal Billing* Allows attorney to monitor charges based on hourly rates, costs, or flat fees. All AR functions **$2500**

Computer Management Systems
General Ledger & Payroll* Single diskette based, easy to use program. Writes checks, W-2's and 914's. Ideal for small retailer. **$500**

Tek-Aids Industries, Inc.
OS65U fig-FORTH Hard disk multi-user, chains with basic programs, requires OS-65U **$250**
fig-FORTH Stand-alone version of the FORTH Interest Group Model. **$175**
BUS-I Original version, with GL, AR, AP. 6 diskettes with new docs. **$99**
BUS/DMS* Most current version of BUS series. Special $150 discount for P.O.'s submitted with original copies of Digital Technology BUS-II. **$850**
MEMTEST/2 New edition of popular memory test for OSI hardware. 8" and 5" disk. **$50**

DCS Software Products
WP/INT* Interface between WP-2 and any DMS file for form letters. **$80**

Tra-Sta
Amway Distributors Package Order entry/inventory Package for direct Amway buyer. Maintains commission structure. **$995**

BBS
Data Director* Powerful data base manager. Command oriented, very interactive. **$995**

Tri-Comp
System Exerciser Self-prompting test routines for end user troubleshooting. **$60**

Farragher and Assoc.
Med-Bill Single doctor client billing. **$995**

Frisch Computer Systems
Manufacturing Control System* Hard disk based. Standalone inventory with job costing and bill-of-materials. Extensively field tested. **$3500**

DQFLS
DQ Mail* DMS interface for WP6502 form letters Version 1.2 **$75**
DQ Justify right-hand copy justification and incremental spacing program V1.2 **$50**
WP6502 OS65U Version of popular word processor. Version 1.2 **$100** · Version 1.3 **$250**

UCSD System Users Society
USUS Software Exchange Library 6 diskette set of UCSD Pascal programs, USUS membership. **$80**

Abacus Data Systems
Mailer* Text processor plus key file/sort capability. Good mass mailer. **$190**
Payroll* Thorough package for floppy or hard disk. End User Maintenance Service recomm. **$495**
General Ledger* Profit center support, journal based, floppy or hard disk. **$495**

BUS-II Manufacturer's original version **$150**
***DMS Compatible**

Affiliated Dealers

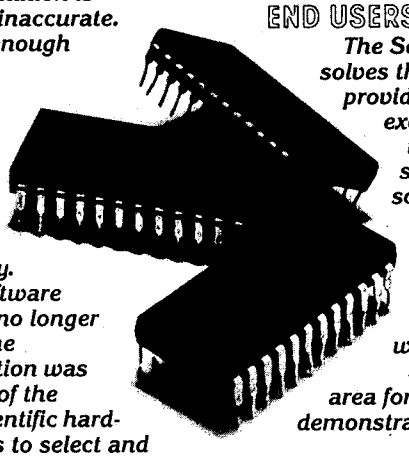
Creative Office Systems Indianapolis, Ind. 46241
Small Business Systems, Inc. Gering, NE 69341
Kansas Computer Liberal, KS 67901
Custom Systems Development Wichita, KS 67214
BIP Murphysboro, IL 62966
Abacus Data Systems Greensburg, PA 15601
Business Computer of Joliet Cresthill, IL 60435
Business Data Systems, Inc. Boulder, CO 80301
Case Computer Bradley, IL 60915
Computer Management Systems Mitchell, SD 57301
CSB Houston, TX 77057
Cybertronics Houston, TX 77084
Data Buss Grayslake, IL 60038
Data Services Computer Corp. Denver, CO 80239
Delta Data Distributors Memphis, TN 38118
Farragher & Assoc. Milwaukee, WI 53213
Frisch Computer St. Paul, MN 55113
International Automation New Kensington, PA 15608
KMH Galesburg, IL 61401
MAP Systems Peoria Hts., IL 61614
Specialized Computer Systems Jackson, MI 49204
Taxman Salt Lake City, UT 84115
Tek-Aids Industries, Inc. Arlington Hts., IL 60004
Tra-Sta Computer Shoppe Pueblo, CO 81005
TriComp Inc. Denver, CO 80221
Total Data Systems Ft. Collins, CO 80525
Whitlock International, Inc. Detroit, MI 48219

EVERYONE WINS

The Ohio Scientific Software Game

Selecting software for your Ohio Scientific computer is a chancy task at best. There are few trustworthy vendors with a national reputation. There are no consistent quality standards and the documentation is often cryptic and inaccurate. If you are lucky enough to find a good package, there's no guarantee of ongoing support. A wrong choice results in months of wasted time, effort, and money.

With the Software Federation, you no longer take that risk. The Software Federation was formed by three of the largest Ohio Scientific hardware distributors to select and market quality software through reputable dealers nationwide.



DEALERS

The Software Federation solves the dealer's problems by providing low cost access to high quality software with the sort of demonstration packages, documentation, and support that the dealer needs to successfully sell machines.

AUTHORS

The Software Federation solves the independent vendor's problems by providing a proprietary method of software protection, aggressive enforcement of software licenses, a strong dealer base, primary support, and national advertising.

END USERS

The Software Federation solves the user's problems by providing quality software, exceptional documentation, after-the-sale support, and optional software maintenance services.

Why risk making the wrong choice? With the Software Federation, everyone wins!

See the dealer in your area for a complete turnkey demonstration.

Software Federation Inc.
44 University Drive
Arlington Hts., IL 60004
Phone: 312/259-1355

TM

below the cursor which is not cleared by scrolling.

Also pointed out by James Loos is the starting position of the cursor counter at $FFE0 and the line width constant at $FFE1. Expanding on this, if one has a TV Monitor with so much overscan that he must decrease the line width to 22 or 23 characters (answering 22 or 23 to Terminal Width) changing the value at $FFE1 does it for you and has the added benefit of not requiring it to be reset to 72 when saving a file on tape. In the other direction, if you have a monitor with very little overscan you can increase the number of characters per displayed line by increasing the value stored in $FFE1. To move the display to the left or right adjust $FFE0 accordingly.

While you have your soldering iron hot you might want to install a 10 uF capacitor between the RESET input of the 6502 and ground. Use a 10 or 20 volt electrolytic with the positive end connected to the foil that runs to pin 40 of U8. A good place to put it is adjacent to R12. Now when the power is turned on, the CPU will be held reset until the capacitor charges through R12 and you won't get a screen full of garbage. It looks so much more professional.


```
REPLACE
 0 1 2 3 4 5 6 7 8 9 A B C D E F
FE00 A2 28 9A D8 EA EA EA EA EA EA A2 D4 A9 D0
FE10 85 FF A9 00 85 FE 85 FB A8 A9 20 91 FE C8 D0 FB
FE20 E6 FF E4 FF D0 F5 84 FF F0 19 20 E9 FE C9 2F F0
WITH
FE00 A2 28 9A D8 20 0E FE 84 FF 84 FB 4C 43 FE 48 A2
FE10 04 A0 00 84 FE A9 D0 85 FF A9 20 91 FE C8 D0 FB
FE20 E6 FF CA D0 F6 68 60 FF FF FF 20 E9 FE C9 2F F0
MONITOR ENTRY (RESTART)
FE00 A228          LDX #$28
FE02 9A            TXS
FE03 D8            CLD
FE04 200EFE        JSR CLEAR
FE07 84FF          STY FF       CLEAR $FF
FE09 84FB          STY FLAG     CLEAR CASSETTE FLAG
FE0B 4C43FE        JMP IN       ADDRESS MODE ENTRY POINT
CLEAR SCREEN SUBROUTINE
FE0E 48      CLEAR PHA
FE0F A204          LDX #$04     08 IF 64 CHAR/LINE
FE11 A000          LDY #$00
FE13 84FE          STY LOW      ADDRESS REGISTER LOW
FE15 A9D0          LDA #$D0
FE17 85FF          STA HI       ADDRESS REGISTER HI
FE19 A920          LDA #$20
FE1B 91FE    LOOP  STA (LOW),Y
FE1D C8            INY
FEFE D0FB          BNE LOOP
FE20 E6FF          INC HI
FE22 CA            DEX
FE23 D0F6          BNE LOOP
FE25 68            PLA
FE26 60            RTS

FE27 FFFFFF                     3 UNUSED LOCATIONS
```

* * * * *

THINGS LEARNED THIS MONTH

by Jim Sanders
2338 Riviera Dr.
Vienna, Va. 22180

This year I have resolved to keep a notebook by every one of my terminals and actually write down all the details I learn about the beast within. To help enforce that resolution, I have committed to a monthly column telling you good folk all those goodies. I work with OSU mostly, and do a lot of assembler code. All of my systems are C-3 level one. Thats what this column will be about (mostly).

I have been getting an average of three calls a day from all over the country and am happy to answer your questions when I can, but it does eat into the day. If I get encouragement I will try to set up a hotline. Why is it the problem of the moment is so pressing that none of youall write letters?

OLD AND NEW
Many of the items in my notebook are things I figured out long ago and forgot until someone asked. So I am not claiming that all of this stuff is new...only that it is useful, and not easy to find. For example:

ODDBALL US ERROR
Got a call from a dealer who knew the answer but wasn't thinking right...seems he got a US ERROR IN 350, but that line contained nothing but a PRINT%1,B$. Seems he had been running a program that contained a FLAG9, had not opened the unit 1 file, and this program was looking for line 50000 which did not exist. Thus the 'US' message. Moral: write your software with an exit subroutine. No matter how you end the program, always call this sub first. In it, poke back all the stuff you changed elsewhere. And whenever you change a system location with a FLAG or a POKE, write the normal poke into the exit subroutine. That's called cleaning up your act.

DEVICE 4 USAGE
I got four calls in one week on this, so it must be worth talking about. OSU allows one to INPUT and PRINT to memory, and even assigns device 4 to the task. It is not as easy as it appears, however.

I/O distributers
There are two words of memory in OSU that are used to

control the input and output devices. They are called the Output Distributor (at address 11686) and the Input Distributer (at address 11668).

Output Distributor
Location 11686 is a good friend. You should get to know it well. A basic PRINT statement will cause the output to be printed on EVERY device which is marked by the appropriate bit in this byte. If you POKE 11686,145 for example, a PRINT "Hello" statement will greet you on the 550 board (device 8 since bit 7 of a decimal 145 is a '1'), and on the line printer (device 5 since bit 4 is a '1'), and on the console screen (device 1 since bit 0 is a '1'). A statement such as PRINT #5, "HELLO" will go to device 5 REGARDLESS of the setting of bit 4 in the Output Distributor byte.

Input Distributor
It turns out that this byte (location 11668) is a very different bird. A statement INPUT A$ will go to the LOWEST NUMBERED device whose bit is set to a '1' to get the input. If you POKE 11668,4 you will have to type on device 3 to respond. If you poke a 9 to 11668, device 4 (memory) and device 1 (console) are active, but the input will come from device 1 (least bit set). If you have a '1' in 11668, and you INPUT#4,B$, nothing will happen since that device is not enabled by bit 3 of 11668. Therefore, unlike the Output Distributer, you must enable any device to be used by setting the appropriate bit in the Input Distributer.

Pointers
Moving data in and out of memory with the PRINT#4 and INPUT#4 statements is different from hardware devices in that you must also tell the system where to begin and end in the memory. The locations in OSU that control where stuff goes are the Memory Input Pointer at 11657 and the Memory Output Pointer at 11661. Both of these are two-byte addresses, and the low byte goes first. You must poke the desired starting address in the Input Pointer before you INPUT#4 and of course poke the appropriate address into the Output Pointer before you PRINT#4. (These are usually the same address unless you are being clever.)

Address Calculations
The desired starting address is calculated by:
HI=INT(ADDR/256)
LO=ADDR-256*HI

Exception: Editor
For reasons that I have never understood, if the OSU EDITOR program is enabled, you can not use memory I/O. The stuff appears to go where desired, but the characters stored are always '@'. If you happen to need a good many of those, memory I/O with the Editor enabled is very handy. (I forgot to ask Rick W. about that one last week. Sorry!)

SAMPLE PROGRAM

Here is a goodie you can file away for the next time you want to use memory device 4. A couple of remarks about the program: the ADDR was selected to be on a page boundry out in the middle of the string space used by Basic. Since Basic is not going to use this space, it is ok, but for a working routine, you would set a lower memory top or offset the Basic program (NEW 1023) to guarantee that the memory is not shared. The reason for selecting a page boundry is to avoid dealing with a carry when adding an offset to the low byte of the Input Pointer address on line 85. Note also that the Output pointer is left after the end of C$.

```
10 REM ENABLE DEVICE 4 AND 1
15 POKE 11668,9
20 ADDR=32768: REM WHY?
25 HI=INT(ADDR/256)
30 LO=ADDR-256*HI
35 REM SET OUTPUT POINTER
40 POKE 11661,LO
50 POKE 11662,HI
55 REM GET SOMETHING TO USE
60 INPUT C$: REM FROM CONSOLE
65 PRINT#4,C$: REM STUFF IT
70 REM SET INPUT POINTERS
75 IP=11657 : LS=LEN(C$)-1
80 FOR I=0 TO LS
85 POKE IP,LO+I
90 POKE IP+1,HI
95 INPUT#4,D$
100 PRINT D$
105 NEXT I : END
```

LEVEL 3 HORROR STORY FIXED

For those using level three, there is a 'fix' for the somewhat disturbing tendency of the system to forget what file you are using and write some of your stuff into another partition's file. Thats how some of the data you were sure was entered isn't there. Unfortunately, it IS there, you just haven't found it yet. I have advised all of my customers to stop using level three with more than one user at a time until the 'fix' is installed. I hope to begin testing the new level 3 next week. The odd behavior is so infrequent, it may be a long time before it is safe to say that it is cured.

RIGHT JUSTIFICATION

It is sometimes useful in basic to print columns of numbers right justified instead of left justified as does Basic. OSIO is having a contest to see how many ways programmers are getting Basic to do that. For fun, play around with these tidbits: For integers or floating point, first define a function
DEF FNR(X)=11-LEN(STR$(X))
Then if you
PRINT SPC(FNR(J));J
you will get the value of J right justified in an 11 column field. The $R feature of OSU is made more useful if you define the field width by

POKE 9712,23
PRINT $R,3.4
which will right justify '3.40' in a field 23 columns wide starting at the current print position. Remember that the field width for the Basic default columns is in 2720. This is tricky since Basic starts counting from the left margin no matter where you are on the page. For uniform column widths, you can use
POKE 2720, FNR(J): PRINT,J

As usual, here is a program to illustrate the point.

```
10 J=57: POKE 2720,10
15 FOR I= 1 TO 30
20 FOR K= 1 TO I
25 PRINT" ";:NEXT K
30 PRINT,J,J
35 NEXT I: END
```

After the above is understood, change these two lines and run it again.
10 J=57: POKE 9712, 10
30 PRINT $R,J;$R,J

By the way, for those of you who have torn your hair when using the Money-Mode functions for very large or very small numbers, you will be happy to know that the word has reached the factory and a fix is out for OSU 1.2 only. The Technical Newsletter number 28 has the fix, as well as a wealth of other goodies. Get it from OSIO. Fixes the control T function for WP-2 also.

Happy New Year.

★

GOODIES for OSI Users!

( ) C1P Sams Photo-Facts Manual. Complete schematics, scope waveforms and board photos. All you need to be a C1P or SII Wizard, just $7.95 $_____

( ) C4P Sams Photo-Facts Manual. Includes pinouts, photos, schematics for the 502, 505, 527, 540 and 542 boards. A bargain at $15.95 $_____

( ) C2/C3 Sams Photo-Facts Manual. The facts you need to repair the larger OSI computers. Fat with useful information, but just $34.95 $_____

( ) OSI's Small Systems Journals. The complete set, July 1977 through April 1978, bound and reproduced by PEEK (65). Full set only $15.00 $_____

( ) Terminal Extensions Package - - lets you program like the mini-users do, with direct cursor positioning, mnemonics and a number formatting function much more powerful than a mere "print using." Requires 65U $50.00 $_____

( ) RESEQ 5.2 - - BASIC program resequencer plus much more. Global changes, tables of bad references, GOSUBs & GOTOs, variables by line number, resequences parts of programs or entire programs, handles line 50000 trap. Best debug tool I've seen. MACHINE LANGUAGE - - VERY FAST! Requires 65U. Manual & samples only, $5.00 Everything for $50.00 $_____

( ) SANDERS MACHINE LANGUAGE SORT/MERGE FOR OS-65U. Complete disk sort and merge, OS-DMS compatible, handles multiple fields, documentation shows you how to call from any BASIC program, then return to it or any other BASIC program on any disk, floppy or hard. Most versatile and fastest sort/merge yet. It should cost more, but Sanders says sell it for just $89.00 $_____

( ) KYUTIL - - The ultimate OS-DMS key file utility package. Creates, loads and sorts multiple-field, conditionally loaded key files, sorting at over 200 entries per second! Never sort another master $100.00 $_____

( ) The Credit System - - An accounts receivable system that accepts and verifies inputs (charges and payments) for a charge account system, then prints monthly statements, ages receivables, maintains complete disk files, produces aged accounts receivable analysis. Takes inputs in any order, prints statements always in date order of transactions Requires 65U $298.00 $_____

( ) SUPERMAIL - - The last word in mailing list packages. Uses DMS and the fastest label-printing technique known to produce zip-sorted labels, complete ABC circulation reports. Includes programs for input, editing, dupe checking, automatic soundex generation, label and report generation and printing of renewal notices. Requires 65U $1,798.00 $_____

( ) Cash enclosed ( ) Master Charge ( ) VISA

Account No. _____ Expiration date _____

Signature _____

Name _____

Street _____

City _____ State _____ Zip _____

TOTAL $_____

Md Residents add 5% tax $_____

Postage & Handling $ 2.00

TOTAL DUE $_____

master charge
THE INTERBANK CARD

DBMS, INC.
PO Box 347
Owings Mills, MD 21117

VISA

PEEK(65) INDEX FOR 1980

As promised, here is the index for PEEK(65) for all issues in 1980. It is quite straightforward, but perhaps a word or two of explanation is in order:

(?) means you will find a question on the subject mentioned on this page;

(AD) means you will find a classified ad on the subject there.

I decided not to index display ads, and tried to mention only the first occurence of classified ads which ran for more than one month.

Contd. on p. 18

LETTERS

ED:

The 32/64 character video mod by Progressive Computing provides an order of magnitude improvement to the C1P display and is highly recommended to anyone who doesn't mind kludging up his 600 board a bit. As a C1P MF disk system owner, I was, however, a bit dismayed to find that the instructions supplied, which describe the software changes necessary to make the whole thing work applied only to the ROM Basic cassette system. Making the video mod work with a C1P minifloppy system was left as an interesting exercise for the student. Fortunately, the software patches to OS-65D V3.1 turn out to be minimal, and once the system has been patched to reflect the increased screen size, everything works as smoothly as if it had been planned tha way.

The changes are made to the Video Output routine which resides at $2599. To effect a permanent Patch it is necessary to call in the Extended Monitor, read track 0 into upper memory, change the data in a few locations and write the modified system back into track 0. This is actually easier than it sounds and I will go into it in detail for those unfamiliar with such things.
The system normally loads at $2200, so we will make our mods at $4200 to keep track of our locations. Start as indicated below:

Now we should have a new system written on track 0 which will boot up with a 32 character screen width. If it boots up properly, it will only be necessary to copy track 0 (using the Copier routine on track 13) onto any other disk on which you want the wide-screen display. For a 64 character video mod, it should work to put in $E0 wherever $C0 appears in the above changes. I haven't tried a 64 character display yet so I'm not certain that other changes will not also be required.

If you are using Dwo Quong Fok Lok Sow's WP-6502 for word processing, you will want to patch that also to take advantage of your new display width. To make this change, after modifying track 0 as above bring the word processor file into the workspace with a "LOAD WP6502" command. The location which must be changed is at $400A and is originally set to $14, which is the screen width-4 in hex. You can change this using the Monitor by hitting reset, "M" and "400A", which should dsplay "14". Type in "/1C" to change that location, then ".2A51G" to get back to the DOS. The altered file is then saved using "PUT WP6502". Again, for the 64 character screen size, simply substitute "/3C" for the "/1C" above and things should work out properly.

I hope that this information will save some time and exasperation for those who have either made or are contemplating making this otherwise excellent video mod.

Chuck Popenoe
Bethesda, MD.

* * * * *

```
A*EM

EM V2.0

:EXIT

A*CA 0200=13,1

A*GO 0200


- DISKETTE UTILITIES -

SELECT ONE:
1) COPIER
2) TRACK 0 READ/WRITE
? 2


- TRACK ZERO READ/WRITE UTILITY -

COMMANDS:
Rnnnn - READ INTO LOCATION nnnn.
Wnnnn/gggg,p - WRITE FROM nnnn FOR p PAGES
               WITH gggg AS THE LOAD VECTOR
E - EXIT TO OS-65D

COMMAND? R4200


- TRACK ZERO READ/WRITE UTILITY -

COMMANDS:
Rnnnn - READ INTO LOCATION nnnn.
Wnnnn/gggg,p - WRITE FROM nnnn FOR p PAGES
               WITH gggg AS THE LOAD VECTOR
E - EXIT TO OS-65D

COMMAND? E

A*RE EM

EM V2.0

:@45A4

45A4/65 A0

:@45C6

45C6/7D C0

:@45FD

45FD/04 00

:@4602

4602/08 00

:@460F

460F/07 FF

:@4626

4626/65 A0

:@462A

462A/65 A0

:@4637

4637/7D C0

:@4648

4648/20 60

:@464B

464B/00 40

:EXIT

A*GO 0200
```

Contd. on p. 17

digital technology

inc.

P. O. BOX 178590 SAN DIEGO, CA 92117 (714) 270-2000

BUS-II BOOKKEEPING & ACCOUNTING SYSTEM

Turnkey bookkeeping system with interactive G/L, A/R, A/P; stand-alone or interative PAYROLL. Menu-driven, user-oriented, idiot-proof operation. Self-instructing - all documentation is on-line. Demonstration set, model chart of accounts, and detailed reference manual make this one of the simplest bookkeeping systems to install and use. Hundreds in use worldwide.

Ideal for CPA firms, service bureaus, bookkeepers: allows up to 99 clients. Highly flexible formatting of financial reports. Provides branch accounting as well as multiple sales departments. Internal audit trail maintained by each module. An optional CPA Extensions package provides for the special needs of the public accountant.

Inquiry routines in all modules provide means to examine files at any time. Automatic account posting. G/L allows 12-month or 13-period year. A/R allows optional service charges on past due accounts. Both A/P and PAYROLL modules print checks on pin-feed rolls.

Operates on serial C-2 or C-3 systems with dual disks (or Winchester disk) on OS-65U. 48K RAM required.

GENERAL LEDGER REPORTS:

- Chart of Accounts
- Balance Sheet
- Income Statement
- Transaction Listings
- Detailed General Ledger
- Cash Receipts Journal
- Cash Disbursements Journal
- General Journal

ACCOUNTS RECEIVABLE REPORTS

- Daily Transaction Listing
- Monthly Transaction Listing
- Detailed Accounts Receivable
- A/R Status Report
- Statements (with past due notices)
- Open Invoice Listing
- A/R Account Detail
- A/R Ageing (variable period)
- A/R Aged Trial Balance
- A/R Mailing Labels

ACCOUNTS PAYABLE REPORTS

- Vendor Listing
- Transaction Listing
- Payables Listing
- Payables Due or Past Due
- Checks to be Written by Date
- Checks to be Written by Vendor
- Print Checks
- Check Register
- A/P Mailing Labels

PAYROLL REPORTS

- Print Payroll Checks
- W-2 Forms
- Employee History
- Detailed Payroll Report
- Period Payroll Report
- Detailed Quarterly Report
- Summary Quarterly Report
- Payroll Mailing Labels

BUS-II VERSION 3.0	**$ 495**
BUS-II DMS (OS-DMS compatible)	**$ 695**
BUS-II CPA (CPA Extensions Package)	**$ 995**

NOTES

BUS-II V 3.1 will supercede V 3.0 effective 02/01/81; the retail price will increase to $995. Upgrades to new versions always available at nominal charge.

BUS-II CPA requires BUS-II V 3.0 or V 3.1.

OS-DMX (OS-DMS EXTENDED)

Digital Technology's Microsystems Information Management Package superimposed on OS-DMS file structure, resulting in a command-oriented OS-DMS-compatible database management system. OS-DMX can be used in place of (or in addition to) DMS nucleus, query, sort, and many other modules.

OSI users now have the best of both worlds. OS-DMX means synergy, standardization, and support. The synergistic effect of merging the best features of the industry's two leading database management systems results in spectacular performance. And standardization will provide a continually expanding list of business application programs for OSI owners. But perhaps most important of all is support. Digital Technology supports its software 100%.

Control Files allow the operator to create a limitless number of specialized "modules" using the system's English-like command language, and store these operations for later recall as needed.

A Program Sequence Executive (a form of job control language) allows the operator to pre-define a number of BASIC programs, database operations, etc., for "pre-programmed" computer operation. These routines can be stored in executive control files which can, in turn, supervise the operation of DMX control files, BASIC or machine-code programs, etc. The operator can actually instruct the computer to "run itself".

DMX consists of three primary programs for Input, Edit, and Report generation. A number of auxiliary programs support the core operations: Database Create creates DMS-compatible files; Database Mapper displays a detailed view of the database files; two Database Sort programs allow in-memory or disk file sorts. Additional programs will be released in the near future including DMX-MAIL (mailing label generation), DMX-STAT (advanced statistical package), and DMX-COPY (allows modifying database files after-the-fact).

A number of applications packages previously available only to MIMP users will be converted to DMX format, including electronic cash register polling, sales analysis, restaurant inventory and menu explosion, and point-of-sale terminal operation. These and virtually any OS-DMS-based applications programs can be used in conjunction with the DMX package.

DATABASE INPUT OPERATING COMMANDS:

Exclude / Fields / Fixed / Help / Include / Input / Journal / Output / Quit / Reset / Run / Screen / Store / Use

DATABASE EDIT OPERATING COMMANDS:

Add / Again / Change / Count / Clean / Delete / Fields / Help / List / Output / Qlist / Quit / Reclaim / Replace / Run / Store / Sum / Use

Command modifiers: Range, Fieldname(s), Conditions

DATABASE REPORT OPERATING COMMANDS:

Average / Break / Column / Exclude / Fields / Heading / Help / Include / Output / Print / Quit / Reset / Run / Store / Total / Use

Command modifiers: Range, Fieldname(s), Conditions

OS-DMX SPECIAL INTRODUCTORY OFFER **$695**

ED:

I look forward to each issue of PEEK(65) and read with interest Volume I, No. 8, 8-15-80. Mr. Foltz' letter was of special interest, since I was also looking for a method of patching the Port #5 routine to work on a 550 Level I board.

I tried your suggestion, and although it allowed the ctrl D - ctrl W paging features, I feel that Mr. Foltz and I had the form feed paging feature in mind. Since then, I have partially dissassembled 65U and have enclosed a new routine that allows the Port #5 command to be routed to the Level I 550. The routine checks for the 550 device index and should work on any designeted 550 port, although I am only able to check it out on the one installed port of my "homemade" 550 board. Please note that one could write his own code for any port and install it inside the addresses $2E02 to $3E20 and thus get "paging" on any desired port. The only stipulation is that the routine return with the output charactrer in the "A" register. Note that location 15908 (dec) must be poked to 60 (or # line per page) each time a paging reset is desired.

In the process of solving the above problem, I developed a "6502 Dissassembler in Basic", which I have enclosed. The program should be self explanatory with the REM statements and may prove useful to other programmers. It should also be adaptable for output to files so that editing and re-assembly could be easily accomplished. I am a self-taught programmer, so please excuse any errors.

In the BADISS program, note that line 710 contains the printer device variable "DV" and is set to 8 (change as required). Also note that lines 1050, 1080 and 1090 pertain to a "page line counter" which may be removed if desired, I offer the program in the hope that it may benefit other programmers, and thereby benefit me through disclosures of their "discoveries". I would gladly mail out the program to anyone interested for a S.A.E. and copying expense or a "swap" program.

In the September issue Mr. Gibbs mentioned a "problem" that was disclosed by the FDUMP utility. I too thought that data was not being correctly put to files, but discovered that the FDUMP utility was at fault. I am not sure this is Mr. Gibbs problem, but it might be worth checking out.

I ordered the OS65D-3.2 Dissassembly Manual as advertised in PEEK(65) (from Software Consultants) and I highly recommend it for serious programmers. As a final note, OS65U also has self-modifying code. I offer this as a warning to other dissassembler(s) of OS65U.

P.S. I have just received OSI-TNL#28 which gives a similar modification to mine. The TNL also listed a DOS bug which may cure Mr. Gibbs' problem with the FDUMP routine.

Terry L. Wallis
322 Haverford
San Antonio, TX 78217
512/824-3807

Terry:

Many thanks for the fine work. Incidentally, our readers can get the information contained in the Technical News Letter from their nearest OSI dealer.

AL

* * * * *

```
RUN"CHANGE","PASS
DISK CHANGE UTILITY
MODE HEX(H) DEC(D)? H
UNIT? A
ADDRESS OFFSET? C00
ADDRESS? 3E02

3E02  A9  ?  98
3E03  00  ?  48
3E04  8D  ?  AC
3E05  B7  ?  56
3E06  38  ?  4D
3E07  AD  ?  10
3E08  00  ?  02
3E09  F4  ?  A0
3E0A  4A  ?  00
3E0B  90  ?  A9
3E0C  0C  ?  02
3E0D  A9  ?  39
3E0E  0C  ?  00
3E0F  20  ?  CF
3E10  B8  ?  F0
3E11  3E  ?  F9
3E12  CE  ?  AD
3E13  B7  ?  B6
3E14  38  ?  38
3E15  D0  ?  99
3E16  F0  ?  01
3E17  F0  ?  CF
3E18  0C  ?  68
3E19  AD  ?  A8
3E1A  B6  ?  AD
3E1B  38  ?  B6
3E1C  29  ?  38
3E1D  7F  ?  EA
3E1E  8D  ?  EA
3E1F  02  ?  EA
3E20  F4  ?  EA
3E21  4C  ?  X
```

★

OS65U PORT#5 TO PORT#8 MODIFICATION

The following Assembly source listing will modify OS65U so that a Port 5 command sends output to Port 8 (Level I 550 board). The code may be installed by either the CHANGE Utility method, or by the Basic Poking method.

Assembly Listing:

```
3E02  98      TYA           Save Y register
3E03  48      PHA           on stack.
3E04  AC564D  LDY $4D56     Get dev. index
3E07  1002    BPL $3E0B     Positive then proceed
3E09  A000    LDY #$00      Neg so default to 0
3E0B  A902    LDA #$02      Get status reg mask
3E0D  3900CF  AND $CF00,Y   Check status
3E10  F0F9    BEQ $3E0B     Not ready then wait
3E12  ADB638  LDA $38B6     Ready so get char.
3E15  9901CF  STA $CF01,Y   Output char
3E18  68      PLA           Restore Y reg
3E19  A8      TAY           from stack
3E1A  ADB638  LDA $38B6     Leave with char in A
3E1D  EA      NOP           NOP unused memory if
3E1E  EA      NOP           desired or move exit code up
3E1F  EA      NOP
3E20  EA      NOP
3E21  4C2B3B  JMP $3B2B     Original exit code
```

AD$

BIG BAG $25 (BIGGEST BAG (5 1/4" DISK) of software you can buy for the bucks) 1) Single Disk Copier. 2) Terminal Emulator makes a terminal out of your computer. 3) High Memory Disk Buffers. 4) Data Management System (not OSI). 5) Disk Head Load-Unload for all disk I/O. 6) Spool to Disk (better than indirect). 7) UTOC Fixer. 8) Menu of Files at Boot Time. 9) File Transmission Program via phone lines with error checking and recovery. FOR OS65D C1P OR C4P (state which)
Computer Power
3223 Suffolk Lane
Fallston, MD 21047
Phone 301-692-6538

SUPERBOARD - protect your superboard with a handsome pine case. Has room for the power supply, fan and all your extras. Complete plans $1.00. Complete pre-cut kit $20.00. Dee Products, 150 Birchwood, Carpentersville, IL 60110

* * * * *

DATA CONVERSION

Between

IBM FORMAT FLOPPY DISKS

9-TRACK MAGNETIC TAPE

PUNCHED CARDS

&

OSI FLOPPY DISK (OS-65U)

Write for a quote:

DBMS, Inc.
PO Box 347
Owings Mills, MD 21117

ED:

I have installed the AH Systems modification to my C1P and have been using the new system for about 1 week. This modification results in:

1. Software selectable baud rates for cassette or RS232 (300 or 1200)
2. RS232 interface
3. Software control of a cassette 'Remote'
4. 48 character per line video display.

Now that I've got the system I am amazed at the 1200 baud fidelity on cassette. The error rate seems to be as low as it was at 300. The video display is superb. The characters are nice and sharp and how did I ever survive a mere 24 characters per line? (My system is plugged straight into the video vertical deflection amp of the TV so I don't know how well the characters will reproduce if you're using a modulator). I haven't tried the RS232 yet (a printer is next on my list) but on the scope all looks well. The software controlled cassette remote seems silly to me; can anyone think of a real use for this?

If anyone out there is planning on adding this mod to his system then read on: I'll save you some trouble and aggravation.

There are some errors in the AH Systems instructions.

1. In the baud rate test instructions (page 13), the instructions do not tell you that at this point in the mod the system will not save on tape at 300 baud and be able to read it back.
2. In the power supply card test instructions (page 12) there is an error: gnd on J3-9 will cause CTS to go high and 5v on J3-9 causes CTS to go low.
3. In the video modification (page 9) there should be 2 corrections:
   a. the cut called for at U13-21 was located on the solder side of my PWB
   b. the jumpers (page 10) should be added before the cuts are made. The order called for in the instructions will result in inputs to MOS devices being left hanging open and that can destroy the device.

In general, I love the mod. However, this is not an easy mod to install. So if you don't rate yourself an experienced electronics technician do not attempt to install this mod.

On the subject of Peek (65), I wish to add my thanks to the Editor for a job well done and I want to encourage everybody to flood him with letters and/or articles. This is the only magazine I'm aware of that is devoted solely to OSI computers. Seems to me that makes it the very best place to exchange ideas and information.

Well, now that I've got 1200 baud capability I've been giving serious consideration to 2000 baud cassette formats. In theory, there's plenty of bandwidth on the cassette so if we used a 2khz clock and phase-reversal keyed 4khz and added them together, then for recovery we could use a synchronous integrate-and-dump...enough of that. I'll save it for another letter after I get it working.

R. Vun Kannon
Black Canyon City, AZ.

Mr. Vun Kannon:

Better yet, make it an article; with schematics and photos, and we will pay you for it!

AL

* * * * *

ED:

I would like more information on 65D and C2's. I would also like to know of any way to program the character generator - any future hardware mods?
Try this semi fast (half fast) screen clear!

```
10 poke 23,0:
POKE 9770,255:
FORI=1TO6:?:
NEXT:POKE 23,64:
POKE 9770,64
```

I would think OSI would have put in a extended BASIC OP code to do that!

Craig Lombard
Olympia, WA.

* * * * *

ED:

Directions for relocating the C2P-1P version of the OSI Extended Monitor:
The X-Monitor is located at $0800-0FFF, The top of a 4K memory, so it can run in small systems. When you add memory, however, it would be nice to relocate the X-Mon so it would still be at the top, out of the way of other programs such as the Assembler.

You can use the X-Mon itself to do most of the job; just use the 'RELOCATE' command. The X-Mon uses a Jump Table, similar to BASIC's. The addresses in this table must be changed to reflect the new location of the monitor.

JUMP TABLE (STOCK) LOCATED AT $0960-0999

@ - 0B53	N - 0D35
A - 0BB3	O - 0E29
B - 0C9A	P - 0BB0
C - 0CBF	Q - 0D14
D - 0CD2	R - 0DB7
E - 0C57	S - 0EC3
F - 0DA3	T - 0C6E
G - 0BC1	*U - 084C
H - 0E33	V - 0F3B
I - 0C12	W - 0D7E
J - 084C*	X - 0BB2
K - 0BAF	Y - 0BB1
L - 0F43	Z - 0FB7
M - 0D91	

(* = UNIMPLEMENTED COMMANDS)

The addresses are arranged in the Lo Byte, Hi Byte format, so $0960 contains $53, $0961 contains $0B, etc.

When relocating to the top of an 8K memory, you would add $1000 (4K) to all the jump addresses (0B53 TO 1B53, for instance). For 16K memories, add $3000 (0B53 TO 3B53).

To change the "SHIFT P" or @ command to "J" or "U", which are easier to type, you must switch the addresses. Put $084C at locations $0960, $0961 and $0B53 in the "J" or "U" place in the table.

Another change I made was in the "Z" command. If I typed anything but a "LINE FEED" after I'd used the "Z" command, I got a screen full of letters and a "BREAK". Changing the contents of $0FFD to $60 will fix this bug.

For people who have done the 64-character conversion to the C1P, the X-Mon can be patched to your output routine at $0861. Change the JMP $FFEE to the address of your patch.

Kerry Lourash
Decatur, IL

* * * * *

ED:

I would like to take this opportunity to suggest a regular feature that would be of benefit to people like me who are more interested in using what is already there, rather than be clever computerists doing exotically technical things. How about reviewing the software that is already on the market and explain what it is, does, and how to use it. I have a few disks for my C4P MF, and I am finding it time consuming learning how to work with them. A dealer once explained to me that this sort of thing wouldn't be necessary as everything would be explained on the screen. Then he proceeded to get himself tied up in a knot, or a loop, or whatever, and had to read a listing of the program to get himself out of it.

William H. Bodden
Rohnert Park

* * * * *

ED:
I have a C4P and want to add a disk without changing the CPU board and losing the cassette I/O. Is it possible to use the 610 board, the Monitor and Bootstrap of the C1PMF (of course changing the address of the ACIA)?

Dr. Raul A Baragiola
Bariloche, Argentina

* * * * *

ED:

Here for the third time are the fixes for the INSERT and REMOVE programs. I sent the same version to OSIO and PEEK(65). Unfortunately, both publications reprinted these fixes incorrectly. There is only one version of the fixes.

I agree with Mr. Isabella's comments concerning any usefulness this type of information can be for readers. When it is printed it should be correct or it will be of no value.

See PEEK(65), Issue #9, 1980.

Check this line in both programs.

```
485 TT=TT+FL(TF):IF    INDEX
    (K1)<BODF THEN TF=TF+K1:
    GOTO A1
```

Corrections to INSERT

```
605 X1=BODF+((RC-RP)*RL)
606 FORX=1TOTF:INDEX<1>=X1
    +FP(X):INPUT%1,D$(X):NEXT
610 X1=BODF+((RR-RP)*RL)
611 FORX=1TOTF:INDEX<1>=X1
    +FP(X):PRINT%1,RIGHT$
    (S1$+D$(X),FL(X)):NEXT
615 RP=RP+1:IFRP<=C GOTO 605
```

Corrections to REMOVE

```
670 B=K0:X1=BODF+(RE*RL)
671 FORX=K1TO TF:INDEX<1>=X1
    +FP(X):INPUT%1,D$(X)

700 X1=BODF+(WR*RL)
701 FORX=K1TOTF:INDEX<1>=X1
    +FP(X):PRINT%K1,RIGHT$
    (S1$+D$(X),FL(X)):NEXT
710 RE=RE+K1:WR=WR+K1: GOTO A1
```

Radford Compton
Manassas, VA 22110

* * * * *

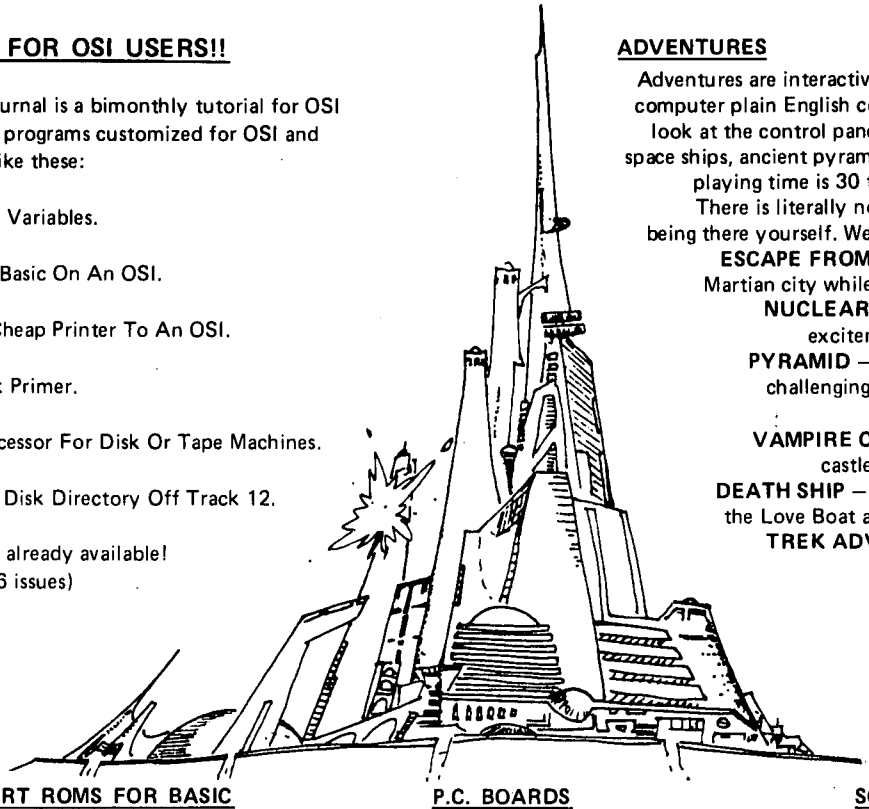
OSI SOFTWARE FOR OSI OSI

A JOURNAL FOR OSI USERS!!

The Aardvark Journal is a bimonthly tutorial for OSI users. It features programs customized for OSI and has run articles like these:

1) Using String Variables.
2) High Speed Basic On An OSI.
3) Hooking a Cheap Printer To An OSI.
4) An OSI Disk Primer.
5) A Word Processor For Disk Or Tape Machines.
6) Moving The Disk Directory Off Track 12.

Four back issues already available!
S9.00 per year (6 issues)



ADVENTURES

Adventures are interactive fantasies where you give the computer plain English commands (i.e. take the sword, look at the control panel.) as you explore alien cities, space ships, ancient pyramids and sunken subs. Average playing time is 30 to 40 hours in several sessions. There is literally nothing else like them – except being there yourself. We have six adventures available.

ESCAPE FROM MARS – Explore an ancient Martian city while you prepare for your escape.
NUCLEAR SUBMARINE – Fast moving excitement at the bottom of the sea.
PYRAMID – Our most advanced and most challenging adventure. Takes place in our own special ancient pyramid.
VAMPIRE CASTLE – A day in old Drac's castle. But it's getting dark outside.
DEATH SHIP – It's a cruise ship – but it ain't the Love Boat and survival is far from certain.
TREK ADVENTURE – Takes place on a familiar starship. Almost as good as being there.

$14.95 each

NEW SUPPORT ROMS FOR BASIC IN ROM MACHINES

C1S – for the C1P only, this ROM adds full screen edit functions (insert, delete, change characters in a basic line.), Software selectable scroll windows, two instant screen clears (scroll window only and full screen.), software choice of OSI or standard keyboard format, Bell support, 600 Baud cassette support, and a few other features. It plugs in in place of the OSI ROM. NOTE: this ROM also supports video conversions for 24, 32, 48, or 64 characters per line. All that and it sells for a measly **$39.95.**

C1E/C2E for C1/C2/C4/C8 Basic in ROM machines.
This ROM adds full screen editing, software selectable scroll windows, keyboard correction (software selectable), and contains an extended machine code monitor. It has breakpoint utilities, machine code load and save, block memory move and hex dump utilities. A must for the machine code programmer replaces OSI support ROM. Specify system **$59.95**

DISK UTILITIES

SUPER COPY – Single Disk Copier
This copy program makes multiple copies, copies track zero, and copies all the tracks that your memory can hold at one time – up to 12 tracks at a pass. It's almost as fast as dual disk copying. – **$15.95**

MAXIPROSS (WORD PROCESSOR) – 65D polled keyboard only - has global and line edit, right and left margin justification, imbedded margin commands, choice of single, double or triple spacing, file access capabilities and all the features of a major word processor – and it's only **$39.95.**

P.C. BOARDS

MEMORY BOARDS!! – for the C1P. – and they contain parallel ports!
Aardvarks new memory board supports 8K of 2114's and has provision for a PIA to give a parallel ports! It sells as a bare board for **$29.95.** When assembled, the board plugs into the expansion connector on the 600 board. Available now!

PROM BURNER FOR THE C1P – Burns single supply 2716's. Bare board – **$24.95.**

MOTHER BOARD – Expand your expansion connector from one to five connectors or use it to adapt our C1P boards to your C4/8P. - **$14.95.**

ARCADE AND VIDEO GAMES

ALIEN INVADERS with machine code moves – for fast action. This is our best invaders yet. The disk version is so fast that we had to add selectable speeds to make it playable.
Tape - $10.95 – Disk - $12.95

TIME TREK (8K) – real time Startrek action. See your torpedoes move across the screen! Real graphics – no more scrolling displays. **$9.95**

STARFIGHTER – a real time space war where you face cruisers, battleships and fighters using a variety of weapons. Your screen contains working instrumentation and a real time display of the alien ships. $6.95 in black and white - $7.95 in color and sound.

MINOS – A game with amazing 3D graphics. You see a maze from the top, the screen blanks, and then you are in the maze at ground level, finding your way through on foot. Realistic enough to cause claustrophobia. – **$12.95**

SCREEN EDITORS

These programs all allow the editing of basic lines. All assume that you are using the standard OSI video display and polled keyboard.

C1P CURSOR CONTROL – A program that uses no RAM normally available to the system. (We hid it in unused space on page 2). It provides real backspace, insert, delete and replace functions and an optional instant screen clear. $11.95

C2/4 CURSOR. This one uses 366 BYTES of RAM to provide a full screen editor. Edit and change lines on any part of the screen. (Basic in ROM systems only.)

FOR DISK SYSTEMS – (65D, polled keyboard and standard video only.)

SUPERDISK. Contains a basic text editor with functions similar to the above programs and also contains a renumberer, variable table maker, search and new BEXEC* programs. The BEXEC* provides a directory, create, delete, and change utilities on one track and is worth having by itself. – $24.95 on 5" disk - $26.95 on 8".

AARDVARK IS NOW AN OSI DEALER!

Now you can buy from people who can support your machine.

–THIS MONTH'S SPECIALS–

Superboard II	$279
C1P Model II	429
C4P	749

. . . and we'll include a free Text Editor Tape with each machine!

Video Modification Plans and P.C. Boards for C1P as low as $4.95

This is only a partial listing of what we have to offer. We now offer over 100 programs, data sheets, ROMS, and boards for OSI systems. Our $1.00 catalog lists it all and contains free program listings and programming hints to boot.

OSI **Aardvark Technical Services • 1690 Bolton • Walled Lake, MI 48088** OSI
(313) 699-3110 or (313) 624-6316

PRINT ENHANCEMENTS OF 65D V3.0

by:
Kurt Grittner
Milwauki, WI.

```
10 DATA 10,0,0,8,0,42,136,54,136,66,136,78
15 DATA 136,89,134,98,136
20 DATA 109,134,118,136
30 READD:LS$=CHR$(D)
40 FORI=1TOD*2:READX:LS$=LS$+CHR$(X):NEXT
50 REM
60 REM
70 REM
80 DISK!"ME 9000,9080"
90 PRINT#5,A$(31):PRINT#5,A$(1)
100 PRINT#5,CB:PRINT#5,C8%(SI):PRINT#5
110 FORI=1TO5:PRINT#5,TL(I):NEXT
120 DV=PR:GOSUB160
130 REM
140 REM
150 REM
160 DISK!"ME 9000,9000":PRINT#5,LS$
170 POKE574,0:POKE575,8*16:X=USR(DV):RETURN
```

Lines 10 - 40 above show how a string (LS$) is set up which defines a line having ten fields to be printed. The first number in the data statement indicates that there are ten fields in the line. The rest of the numbers in the data statements are pairs of numbers that describe each of the ten fields. The first number in the pair is the print position relative to zero where the field is to start. The second number contains two flags and the length of a numeric field.

The first flag has a value of 128. If this bit is set, then the field is to be treated as a dollar amount. It will cause the field to be translated from exponential representation (if possible) to straight decimal representation. It will cause the amount to be rounded to the nearest penny. (This is designed to prevent the odd things that happen when fractional decimal digits start floating around in numeric variables.) It will assure that there is a decimal point present followed by two digits. (Zeros are assumed for integer values). Furthermore, the amount will be right justified & zero left filled within the given field length.

The second flag has a value of 64. If this bit is set, then the field will have commas inserted in the usual places for amounts of greater than $999.99. The field length can have values of 1 to 15. This length must be specified if the first flag is set since it is used to right justify the field. If the first flag is not set, then the field length and the second flag are ignored.

```
0090=           COMFLG = $90            ;COMMAS  WANTED/NOT-WANTED    = 1/0
0091=           WK1    = $91            ;TEMPORARY ZP STORAGE FOR SUB-RTNS
0092=           FSPEC  = $92            ;HOLD BYTE FOR EACH FIELD DESCRIPTION
0093=           CURSOR = $93            ;OUTPUT LINE CURSOR
0094=           TEMP   = $94            ;INDIRECT POINTER REGISTER FOR STRINGS
0096=           DATAIN = $96            ;POINTS TO EACH DATA FIELD IN TURN
0099=           FLEN   = $99            ;HOLD AREA FOR EACH FIELD OUTPUT LENGTH
009A=           NFLDS  = $9A            ;HOLD BYTE FOR THE NUMBER OF FIELDS TO DO
009B=           FIELD  = $9B            ;CURRENT FIELD NUMBER (COUNTER).
009C=           LSPECS = $9C            ;INDIRECT POINTER TO LINE SPECIFICATIONS
                ;
2322=           OUTFLG = $2322          ;OS65D3 OUTPUT FLAG ADDRESS
2343=           OUTPUT = $2343          ;OS65D3 OUTPUT ROUTINE ADDRESS
2D6A=           CRLF   = $2D6A          ;OS65D3 NEW LINE ROUTINE ADDRESS
238A=           MEMINP = 9098           ;OS65D3 DEV#5 (MEMORY) INPUT POINTER ADDRES
                ;
0020=           SPC    = 32
000D=           CR     = 13
                ;
8000            *=$8000
                ;
                ; PRINT A LINE OF FIELDS WHEN CALLED FROM BASIC.
                ;       INPUT  = MEMINP (POINTS TO LINE SPECS, DATA FILEDS)
                ;                BASIC PARAMETER (HAS DEVICE CODE TO PRINT LINE ON)
                ;                LINE SPECS: NUMBER OF FIELDS,
                ;                      FIELD #1 TAB POSITION, FIELD #1 SPECS...
                ;                      FIELD #N TAB POSITION, FIELD #N SPECS
                ;                DATA FIELDS: FIELD #1 DATA, ... FIELD #N DATA.
                ;
                ;
8000 D8         CLD                     ;SET TO BINARY MATH.
8001 AD2223     LDA OUTFLG              ;SAVE OLD OUTPUT FLAG ON STACK.
8004 48         PHA
8005 20AF80     JSR CALL6               ;INEGERIZE BASIC'S PARAMETER.
8008 A5B2       LDA $B2                 ;GET PARAMETER FROM BASIC.
800A 8D2223     STA OUTFLG              ;THIS IS THE NEW OUTPUT FLAG.
                ;
8200=           TEMPPG = TEMPWK/256*256
800D A982       LDA #TEMPWK/256 ;SET STRING TEMP REGISTER.
800F 8595       STA TEMP+1
8011 A90E       LDA #TEMPWK-TEMPPG
8013 8594       STA TEMP
                ;
8015 AD8A23     LDA MEMINP              ;SET LSPECS REGISTER FROM MEMORY INPUT
8018 859C       STA LSPECS
801A 18         CLC
801B 6980       ADC #128
801D 8596       STA DATAIN              ;DATA IS FOUND 128 BYTES AFTER LSPECS
801F AD8B23     LDA MEMINP+1
8022 859D       STA LSPECS+1
8024 6900       ADC #0
8026 8597       STA DATAIN+1
                ;
8028 A000       LDY #0                  ;NUMBER OF FIELDS IN THIS LINE
802A B19C       LDA (LSPECS),Y           IS THE FIRST BYTE OF LSPECS.
802C 859A       STA NFLDS               ;SAVE NUMBER OF FIELDS.
802E 849B       STY FIELD               ;SET FIELD COUNT, OUTPUT LINE CURSOR
8030 8493       STY CURSOR               TO ZERO.
8032 A59B   LOOP0  LDA FIELD            ;GET FIELD COUNT.
8034 C59A       CMP NFLDS               ;IF ALL FIELDS PROCCESSED,
8036 F06F       BEQ DONE                 THEN DONE.
8038 0A         ASL A                   ;'Y' = FIELD * 2 + 1.
8039 A8         TAY
803A C8         INY
803B B19C       LDA (LSPECS),Y  ;GET DESIRED TAB POSITION, THIS FIELD.
803D AA         TAX                     ;SAVE IN 'X'.
803E C8         INY
803F B19C       LDA (LSPECS),Y  ;GET THIS FIELD'S  SPECIFICATIONS.
8041 08         PHP                     ;SAVE STATUS.
8042 8592       STA FSPEC               ;SAVE FIELD SPECS.
8044 A920       LDA #SPC
8046 E493   LOOP1  CPX CURSOR           ;OUTPUT BLANKS UNTIL 'TAB POSITION'.
8048 F007       BEQ OV2
804A 204323     JSR OUTPUT
804D E693       INC CURSOR
```

CORRECTIONS

In the December Issue, in Phil Hooper's article "Call for OSI Basic," the string mentioned in line 1 must be exactly 23 characters long. Therefore, the string "23 CHARACTERS" won't work. The string Phil submitted was actually "TWENTY-THREE CHARACTERS" which just happens to be 23 characters long!

In November, in the RND(X) article on p. 11, two things:

Should be DISK!"SA 05=4000/8
(not 400018), and
cycle length>500,000
(not 7,500,000)

November, p. 14, top center:
Should be Address C000
(not address 0000)

Lines 80-120 show how a line is printed.

In line 80, the memory output pointer is set to the beginning of the memory area (of at least 256 BYTES) that you are using to pass your data fields to the machine code sub-routine.

In line 90, two string variables are printed into memory. (The memory pointer auto-increments). Notice that the field descriptions for these two variables are zero. They will be printed verbatim, starting at their respective 'tab locations'.

In line 100, two numeric items and a 'null' item are printed into memory. (The use of the 'null' allows you to sometimes print 'nothing' in a field defined as numeric.)

In line 110, an array of five numeric items is printed into memory. This makes ten fields all together.

It is important to note that each field must be seperated from the next with a <CR>. Hence the seperate 'PRINT#5' statements.

In line 120, the desired device number is set up and a subroutine is called that will handle the rest of the interface to the machine code. The value for 'DV' must be combinations of the values: 1,2,4,8,16,32,64,128; not merely the decimal value of the device as is used for 'PRINT#'.

In line 160, the specifications for our output line contained in the 'LS$' string are placed in memory at $9000. The $9000 on the left side of the 'DISK!"ME 9000,9000"' command is used as a convenient method of passing the address of the line-specs to the machine code.

In line 170, the 'USR()' vector is set to $8000 and the machine code is called. The desired output devices for this line are passed to the machine code program thru the basic variable 'DV'.

When our machine code program gets control, it will expect the line-specs to be at $9000 (AS PER'ME' COMMAND), and the field data to be at $9080. (It keys off the memory input ADDRESS+128). It will also save the present state of the output flag and use the output flag that we passed it in 'DV' for printing this line.

```
804F D0F5          BNE LOOP1
8051 AA      OV2   TAX               ;'X' = SPC.
8052 28            PLP               ;IS THIS FIELD A STRING ?
8053 1001          BPL OVC           ;YES, SO KEEP SPACES.
8055 E8            INX               ;'X' = SPC+1  (THROW AWAY SPACES).
8056 8691    OVC   STX WK1           ;SET COMPARE
8058 A200          LDX #0
805A A000          LDY #0
805C B196    LOOP2 LDA (DATAIN),Y    ;MOVE THIS FIELD TO 'TEMPWK',
805E C8            INY               STRIPPING OFF UNWANTED CONTROL
805F C591          CMP WK1           CHARACTERS.  ALSO STRIP SPACES
8061 9004          BCC OVB           IF THIS IS A DOLLAR AMOUNT.
8063 9D0E82        STA TEMPWK,X
8066 E8            INX
8067 C90D    OVB   CMP #13
8069 D0F1          BNE LOOP2
806B 9D0E82        STA TEMPWK,X      ;TERMINATE FIELD WITH 'CR'.
                   ;
806E 98            TYA               ;BUMP 'DATAIN' TO STEP OVER THIS FIELD.
806F 18            CLC
8070 6596          ADC DATAIN
8072 8596          STA DATAIN
8074 9002          BCC OVA
8076 E697          INC DATAIN+1
8078         OVA   *=*
                   ;
8078 8A            TXA               ;IF THIS FIELD IS NULL,
8079 F027          BEQ OV5           THEN PRINT NOTHING.
807B A900          LDA #0            ;SET TO 'NO-COMMAS'.
807D 2492          BIT FSPEC         ;TEST FIELD SPECS.
807F 100F          BPL OV4           ;IF FIELD = STRING THEN SKIP.
8081 5002          BVC OV3           ;IF NO COMMAS WANTED, THEN SKIP.
8083 A901          LDA #1            ;SET TO 'COMMAS'.
8085 8590    OV3   STA COMFLG        ;SAVE COMMA FLAG.
8087 A592          LDA FSPEC         ;SET DOLLAR AMOUNT FIELD LENGTH.
8089 290F          AND #$0F
808B 8599          STA FLEN
808D 20B280        JSR MONY          ;GO EDIT TEMP AS MONEY AMOUNT.
8090 A000    OV4   LDY #0
8092 B90E82  LOOP4 LDA TEMPWK,Y      ;OUTPUT THIS FIELD.
8095 C8            INY
8096 C90D          CMP #13
8098 F008          BEQ OV5
809A 204323        JSR OUTPUT
809D E693          INC CURSOR
809F 4C9280        JMP LOOP4
                   ;
80A2 E69B    OV5   INC FIELD         ;NEXT FIELD.
80A4 4C3280        JMP LOOP0
                   ;
80A7 206A2D  DONE  JSR CRLF          ;FEED A LINE.
80AA 68            PLA               ;RESTORE OLD OUTPUT FLAG.
80AB 8D2223        STA OUTFLG
80AE 60            RTS               ;RETURN TO CALLER.
                   ;
80AF 6C0600  CALL6 JMP (6)           ;JUMP INDIRECT THRU $0006
                   ;
                   ; EDIT A STRING AS A MONEY AMOUNT.
                   ;       INPUT  =  (TEMP)  ADDRESS OF THE STRING.
                   ;                 FLEN    LENGTH OF OUTPUT FIELD.
                   ;                 COMFLG  INDICATES WHETHER COMMAS ARE WANTED.
                   ;
80B2 205D81  MONY  JSR EXPNTL        ;SHIFT DECIMAL FOR EXP FRACTIONS.
80B5 20C380        JSR MONYPT        ;DO ROUNDING AND ADJUST DECIMAL.
80B8 A590          LDA COMFLG        ;DOES HE WANT COMMAS IN ?
80BA F003          BEQ MONY2         ;ZERO MEANS NO.
80BC 202081        JSR COMMAS        ;INSERT COMMAS IN DOLLAR AMOUNT.
80BF 204B81  MONY2 JSR EVEN          ;GO RIGHT JUSTIFY FIELD.
80C2 60            RTS               ;RETURN TO CALLER.
                   ;
                   ; INSURE A DECIMAL POINT AND 2 NUMERALS AFTER IT.
                   ; ROUND THE AMOUNT TO THE NEAREST PENNY.
                   ;       INPUT  = (TEMP) ¼CR½
                   ;
80C3 A92E    MONYPT LDA #'.
80C5 20A781        JSR FINDCH        ;LOOK FOR DEC. PT.
80C8 B00E          BCS MONP0         ;SKIP IF NONE FOUND
                   ;
80CA C8            INY               ;DPT + 1
80CB A90D          LDA #CR
80CD D194          CMP (TEMP),Y      ;WAS DPT FOLLOWED BY ¼CR½?
80CF F00A          BEQ MONP1         ;YES, THEN SKIP
                   ;
80D1 C8            INY               ;DPT + 2
80D2 D194          CMP (TEMP),Y      ;1 DIGIT, THEN ¼CR½ ?
80D4 D014          BNE MONP6         ;NO, THEN GO CHECK ROUNDING.
80D6 F008          BEQ MONP2         ;YES, THEN SKIP.
                   ;
80D8 9194    MONP0 STA (TEMP),Y      ;FORCE DECIMAL POINT.
80DA C8            INY               ;NEXT BYTE.
80DB A930    MONP1 LDA #'0           ;FORCE TWO ZEROS.
80DD 9194          STA (TEMP),Y
80DF C8            INY
80E0 A930    MONP2 LDA #'0           ;FORCE ONE ZERO.
80E2 9194          STA (TEMP),Y
80E4 C8            INY
80E5 A90D          LDA #CR           ;FORCE A ¼CR½.
80E7 9194          STA (TEMP),Y
80E9 60            RTS               ;RETURN TO CALLER.
                   ;
80EA C8      MONP6 INY               ;DPT + 3
80EB B194          LDA (TEMP),Y      ;GET TENTHS OF A CENT.
80ED C90D          CMP #CR           ;IF AT END,
80EF F02E          BEQ MONP9         THEN RETURN.
```

```
 2 INPUT"A,B,C,D ";A,B,C,D
 10 DISK!"ME 9000,9000"
 20 PRINT#5,CHR$(4);
 21 PRINT#5,CHR$(0)CHR$(8);
 22 PRINT#5,CHR$(12)
    CHR$(138);
 23 PRINT#5,CHR$(30)CHR$
    (128+64+11);
 24 PRINT#5,CHR$(50)CHR$(137)
 30 DISK!"ME 9000,9080"
 40 PRINT#5,A:PRINT#5,B:
    PRINT#5,C:PRINT#5,D
 42 POKE574,0:POKE575,8*16
 50 DV=4
 52 X=USR(DV)
 60 GOTO 2
```

You can use this program to verify that everything is working ok. Set line 50 to the code for your console device. Input your numbers and see how they come out.

Lines 21-24 define each of the four fields here. You can also experiment with them.

HINT
If columns are out of alignment, then you probably have a data field larger than you allowed for.

Contd. from p. 8

```
- DISKETTE UTILITIES -

SELECT ONE:
1) COPIER
2) TRACK 0 READ/WRITE
? 2



- TRACK ZERO READ/WRITE UTILITY -

COMMANDS:
Rnnnn - READ INTO LOCATION nnnn.
Wnnnn/gggg,p - WRITE FROM nnnn FOR p PAGES
               WITH gggg AS THE LOAD VECTOR
E - EXIT TO OS-65D

COMMAND? W4200/2200,8



- TRACK ZERO READ/WRITE UTILITY -

COMMANDS:
Rnnnn - READ INTO LOCATION nnnn.
Wnnnn/gggg,p - WRITE FROM nnnn FOR p PAGES
               WITH gggg AS THE LOAD VECTOR
E - EXIT TO OS-65D

COMMAND? E


A*
```

```
80F1 18          CLC             ;ROUND TENTHS OF CENT.
80F2 6905        ADC #5
80F4 AA          TAX
80F5 A90D        LDA #CR         ;FORCE ½CR½ HERE.
80F7 9194        STA (TEMP),Y
80F9 E03A        CPX #'9+1       ;IF NO DECIMAL CARRY,
80FB 9022        BCC MONP9        THEN RETURN.
;
80FD 88     MONP7 DEY            ;NEXT DIGIT.
80FE 3019        BMI MONP8       ;IF AT END, THEN SKIP.
8100 B194        LDA (TEMP),Y    ;GET DIGIT.
8102 C92D        CMP #'-         ;NEGATIVE SIGN ALSO MEANS END OF DIGITS.
8104 F013        BEQ MONP8       ;IF AT END, THEN SKIP.
8106 C92E        CMP #'.         ;SKIP DECIMAL POINT.
8108 F0F3        BEQ MONP7
810A AA          TAX             ;INCREMENT THIS DECIMAL DIGIT
810B E8          INX              AS A RESULT OF CARRY
810C 8A          TXA              FROM ROUNDING.
810D 9194        STA (TEMP),Y    ;PUT INCREMENTED DIGIT BACK.
810F C93A        CMP #'9+1       ;DID IT CARRY OUT AGAIN ?
8111 D00C        BNE MONP9       ;IF NOT, THEN RETURN.
8113 A930        LDA #'0         ;IF SO, THEN MAKE THIS DIGIT A ZERO.
8115 9194        STA (TEMP),Y
8117 D0E4        BNE MONP7       ;NEXT DIGIT.
;
8119 C8     MONP8 INY            ;CARRY EXISTS OUTSIDE EXISTING
811A A931        LDA #'1          DIGITS, SO INSERT A 'ONE' DIGIT.
811C 20BC81      JSR INSTCH
811F 60     MONP9 RTS            ;RETURN TO CALLER.
;  PUT COMMAS IN THE PROPER PLACES IN A NUMERIC FIELD
;
;       INPUT:   (TEMP) ½CR½  = ADDR OF NUMERIC FIELD
;
8120 A92E   COMMAS LDA #'.
8122 20A781      JSR FINDCH      ;SEE IF THERE'S A DPT.
8125 B023        BCS COMMA9      ;IF NOT, THEN RETURN.
;
8127 A200   COMMA0 LDX #0        ;SET DIGIT COUNT TO ZERO.
8129 88     COMMA1 DEY           ;NEXT DIGIT. (INITIALLY = DPT - 1).
812A 3014        BMI COMMA6      ;IF AT END, THEN SKIP.
812C B194        LDA (TEMP),Y    ;GET DIGIT.
812E 20D481      JSR CKNMCH      ;IS IT NUMERIC ?
8131 B00D        BCS COMMA6      ;IF NOT, THEN SKIP.
8133 E8          INX             ;IF SO, THEN COUNT A DIGIT.
8134 E003        CPX #3          ;IS THIS THE THIRD DIGIT ?
8136 D0F1        BNE COMMA1      ;IF NOT, THEN NEXT DIGIT.
8138 A92C        LDA #','        ;IF SO,
813A 20BC81      JSR INSTCH       THEN INSERT A COMMA.
813D 4C2781      JMP COMMA0      ;NEXT GROUP OF THREE DIGITS.
;
8140 C8     COMMA6 INY           ;LOOK AT FIRST DIGIT IN STRING.
8141 B194        LDA (TEMP),Y
8143 C92C        CMP #','        ;IS IT A COMMA ?
8145 D003        BNE COMMA9      ;NO, THEN RETURN.
8147 20DD81      JSR SHFLFT      ;YES, SO ELIMINATE IT.
814A 60     COMMA9 RTS
;
; RIGHT JUSTIFY A FIELD & LEFT BLANK FILL
;
;       INPUT:  (TEMP) ½CR½ = ADDR OF FIELD
;               FLEN        = LEN OF FIELD
;
814B 20EE81 EVEN    JSR FINDCR   ;'Y' = LENGTH OF STRING.
814E C499        CPY FLEN        ;IF LESS THAN DESIRED LENGTH,
8150 B00A        BCS EVEN9
8152 A920        LDA #SPC        ;THEN INSERT A SPACE
8154 A000        LDY #0           AT THE FRONT OF THE STRING,
8156 20BC81      JSR INSTCH       MAKING IT ONE LONGER.
8159 4C4B81      JMP EVEN        ;DO IT AGAIN.
815C 60     EVEN9 RTS
;
;
; TURN EXPONENTIAL FRACTIONS INTO NORMAL NUMBERS
;
815D A945   EXPNTL LDA #'E       ;EXP SYMBOL
815F 20A781      JSR FINDCH
8162 B042        BCS EXPNT9      ;IF NOT FOUND THEN RETURN
;
8164 C8          INY             ;NEXT BYTE IS SIGN OF EXP.
8165 B194        LDA (TEMP),Y    ;GET IT.
8167 C92D        CMP #'-         ;IS THIS NUMBER ½ 1 ?
8169 D03B        BNE EXPNT9      ;NO, SO RETURN.
;
816B C8          INY             ;TENS DIGIT OF EXP
816C B194        LDA (TEMP),Y
816E AA          TAX
816F C8          INY             ;ONES DIGIT OF EXP
8170 B194        LDA (TEMP),Y
8172 20F981      JSR CVB         ;CONVERT TO BINARY.
8175 AA          TAX             ;SAVE IN 'X'.
;
8176 E004        CPX #4          ;IS NUMBER ½ .001 ?
8178 900B        BCC EXPNT5      ;NO, SO SKIP.
;
817A A000        LDY #0          ;YES, SO AFTER ROUNDING THIS WILL
817C B194        LDA (TEMP),Y     BE A ZERO RESULT.
817E C92D        CMP #'-         ;IF IT IS A NEGATIVE NUMBER, THEN
8180 D003        BNE EXPNT5       STRIP THE NEGATIVE SIGN TO PREVENT
8182 20DD81      JSR SHFLFT       NEGATIVE ZERO FROM OCCURING.
8185 A92E   EXPNT5 LDA #'.       ;LOOK FOR DECIMAL POINT IN MANTISSA
8187 20A781      JSR FINDCH
818A B003        BCS EXPNT6      ;IF FOUND
818C 20DD81      JSR SHFLFT       THEN ELIMINATE IT.
818F A000   EXPNT6 LDY #0        ;SET TO INSERT ZEROS BEFORE MANTISSA
8191 B194        LDA (TEMP),Y    ;UNLESS MANTISSA IS NEGATIVE
8193 C92D        CMP #'-
8195 D001        BNE EXPNT4      ;IF IT IS NEGATIVE
8197 C8          INY              THEN STEP OVER THE MINUS SIGN.
8198 A930   EXPNT4 LDA #'0       ;ZERO TO INSERT.
```

```
819A 20BC81  EXPNT7 JSR INSTCH      ;INSERT ZEROS
819D CA             DEX              UNTIL DONE
819E D0FA           BNE EXPNT7
81A0 C8             INY             ;INSERT A DECIMAL POINT AFTER
81A1 A92E           LDA #'.          THE FIRST ZERO
81A3 20BC81         JSR INSTCH
81A6 60      EXPNT9 RTS             ;RETURN TO CALLER.
81A7 A000    FINDCH LDY #0          ;FROM Y=0
81A9 88      FCH0   DEY             ;FROM Y=Y
81AA 8591    FCH1   STA WK1         ;FROM Y=Y+1
81AC C8      FCH2   INY
81AD B194           LDA (TEMP),Y    ;GET NEXT CHAR
81AF C90D           CMP #CR         ;END OF INPUT ?
81B1 F006           BEQ FCH8        ;YES, THEN CHAR NOT FOUND
81B3 C591           CMP WK1         ;IS THIS THE CHAR ?
81B5 D0F5           BNE FCH2        ;NO, SO KEEP LOOKING
81B7 18             CLC             ;YES, FLAG AS FOUND.
81B8 60             RTS
;
81B9 A591    FCH8   LDA WK1         ;RESTORE HIS CHAR
81BB 60             RTS             ;CARRY = SEC. (NOT FOUND)
;
;  INSERT A CHARACTER IN A STRING.
;
;       INPUT: 'Y'     = WHERE IN STRING TO INSERT IT
;              'A'     = CHARACTER TO BE INSERTED
;              (TEMP)  = ADDR OF THE STRING
;
81BC 8491    INSTCH STY WK1         ;WHEN TO STOP
81BE 48             PHA             ;WHAT TO PUT IN.
81BF 20EE81         JSR FINDCR      ;FIND THE END
81C2 B194    INSTC2 LDA (TEMP),Y    ;GET NEXT CHARACTER.
81C4 C8             INY             ;STORE IT AHEAD ONE BYTE.
81C5 9194           STA (TEMP),Y
81C7 C491           CPY WK1         ;IS THIS WHERE TO INSERT ?
81C9 F005           BEQ INSTC6      ;IF SO, THEN SKIP.
81CB 88             DEY             ;IF NOT, THEN DO NEXT BYTE.
81CC 88             DEY
81CD 4CC281         JMP INSTC2
81D0 68      INSTC6 PLA             ;GET THE CHAR TO BE INSERTED.
81D1 9194           STA (TEMP),Y    ;OVERLAY DUPLICATE OF LAST BYTE MOVED.
81D3 60             RTS
;
;  CHECK A CHARACTER IN 'A' FOR NUMERICS
;       INPUT  =  SEC = NOT NUMERIC.
;                 CLC = NUMERIC.
;
81D4 C930    CKNMCH CMP #'0         ;IS IT ≥ 0 ?
81D6 9003           BCC CKNM9       ;YES, THEN SKIP.
81D8 C93A           CMP #'9+1       ;IS IT ≤ 9 ?
81DA 60             RTS             ;IF YES, THEN SEC.  IF NO, THEN CLC.
81DB 38      CKNM9  SEC
81DC 60             RTS
;
;  BACK A FIELD UP ONE BYTE
;
;        INPUT:  'Y'    = DISP OF CHAR TO ELIMINATE
;                (TEMP)= ADDR OF STRING TO SHIFT
;                              ENDED BY ↑CR↓.
;
81DD 8491    SHFLFT STY WK1         ;SAVE THIS 'Y' FOR USE IN RECURSION.
81DF C8             INY
81E0 B194    SHFLF1 LDA (TEMP),Y    ;PUSH A BYTE BACK ONE.
81E2 88             DEY
81E3 9194           STA (TEMP),Y
81E5 C8             INY             ;LOOK AT NEXT BYTE.
81E6 C8             INY
81E7 C90D           CMP #CR         ;IS THIS THE END ?
81E9 D0F5           BNE SHFLF1      ;IF NOT, THEN NEXT.
81EB A491           LDY WK1         ;IF SO, THEN RESTORE 'Y'
81ED 60             RTS              AND RETURN.
;
;  FIND A ↑CR↓
;
;      INPUT:  (TEMP) = WHERE TO START LOOKING
;
;      OUTPUT: 'Y' = DISP. OF ↑CR↓
;
81EE A000    FINDCR LDY #0          ;FROM Y=0.
81F0 88      FINDC0 DEY             ;FROM Y=Y.
81F1 A90D    FINDC1 LDA #CR         ;FROM Y=Y+1.
81F3 C8      FINDC2 INY             ;NEXT BYTE.
81F4 D194           CMP (TEMP),Y    ;IS IT A ↑CR↓ ?
81F6 D0FB           BNE FINDC2      ;IF NOT, THEN NEXT.
81F8 60             RTS             ;IF SO, THEN RETURN.
;
;  CONVERT 2 ASCII DECIMAL DIGITS TO 1 BINARY VALUE
;     INPUT:  X=TENS
;             A=UNITS
;
;     OUTPUT: A=BINARY VALUE
;
81F9 290F    CVB    AND #$0F        ;TURN UNITS INTO BCD.
81FB 8591           STA WK1         ;SAVE UNITS
81FD 8A             TXA             ;GET TENS DIGIT
81FE 290F           AND #$0F        ;TURN TENS INTO BCD
8200 AA             TAX
8201 A900           LDA #0
8203 18             CLC
8204 CA      CVB2   DEX             ;DECREMENT TENS
8205 3004           BMI CVB4        ;IF ALL DONE THEN EXIT
8207 690A           ADC #10         ;ELSE, ADD 10
8209 D0F9           BNE CVB2        ;NEXT
820B 6591    CVB4   ADC WK1         ;ADD UNITS TO VALUE OF TENS
820D 60             RTS
;
824E         TEMPWK *=*+64          ;WORK SPACE FOR STRING TEMPORARIES.
             .END
```

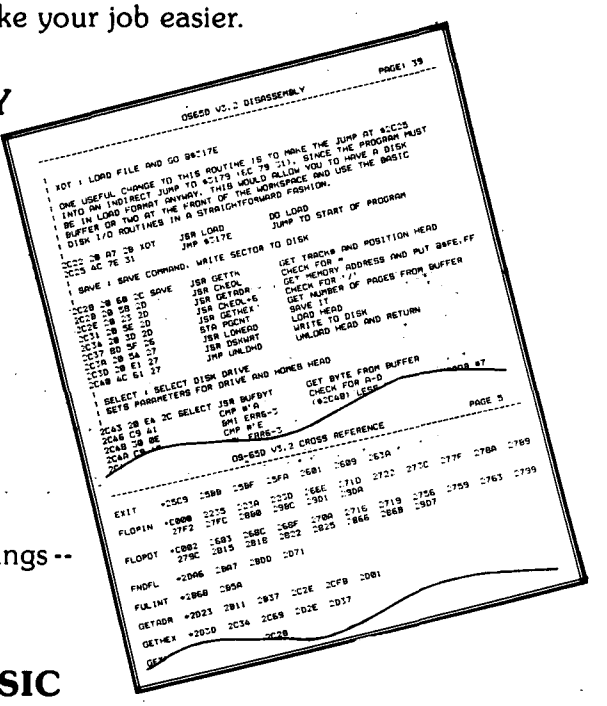
Contd. from p. 7

△

OSI'ers!! **Just for YOU. . .**
from SOFTWARE CONSULTANTS. . .

great products that make your job easier.

1. OS-65D V3.2 DISASSEMBLY MANUAL With CROSS REFERENCE LISTINGS. . .



- 50 pages of Disassembly listings in standard Assembler format.
- FULLY COMMENTED code -- a complete, computer-generated concordance.
- 10 pages of CROSS REFERENCE listings -- not a useless program listing.

2. REF COMMAND UNDER BASIC A Complete CROSS REFERENCE UTILITY. . .

- HIGH SPEED, memory resident utility available on OS-65D or OS-65U.
- LISTS ANY OR ALL occurrences of an individual BASIC variable, line number, or numeric constant.
- GENERATES A SORTED LISTING of all occurrences of any variable, and sends it either to printer or screen.
- REQUIRES 1K of memory. Can either be automatically loaded into the high area of memory (48, 32, or 24K), or under the OS-65U, can be hidden away in the OS.
- AVAILABLE EITHER on 5¼" or 8" floppy disks.

If this is the kind of help you can use, mail your check today for both great programming tools. DISASSEMBLY MANUALS are $24.95 each and CROSS REFERENCE UTILITIES are $29.95 each. Please specify disk size and OS for the CROSS REFERENCE UTILITY. Dealer inquiries are invited. Tennessee residents please add 6% sales tax. Your order will arrive postpaid in 2-3 weeks. Phone us at 901/377-3503 for rush orders.

Mail your check today to: SOFTWARE CONSULTANTS, 7053 Rose Trail, Memphis, TN. 38134 USA.

Here's my order. Please send me ____ DISASSEMBLY MANUAL(S) at $24.95 each and ____ CROSS REFERENCE UTILITIES at $29.95 each. (Check either 5¼" ____ or 8" ____ disk, and either OS-65D ____, or OS-65U ____) I've enclosed a check for the total amount ($____), including tax, if applicable.

NAME _____

ADDRESS _____

FIRM _____

CITY, STATE _____ POSTAL CODE _____

COUNTRY _____

PEEK (65)
The Unofficial OSI Users Journal

P.O. Box 347
Owings Mills, Md. 21117

BULK RATE
U.S.POSTAGE
PAID
Owings Mills,MD
Permit No. 18

Deliver to:

PLEASE SEND PEEK(65) FOR ONE YEAR (12 ISSUES)

( ) $12 Enclosed. (Domestic, USA.)

( ) $20 Enclosed. (Foreign Air Mail)

Maryland Residents add 5% sales tax.

NAME_____

STREET_____CITY_____

STATE_____ZIP_____

Please send the following back issues. I enclose $1.50 ea.
Domestic and overseas surface rate.
Maryland Residents add 5% sales tax.

( ) Jan. #1, "Welcome to the first"
( ) Feb #2, "A month ago in this spot"
( ) Mar. #3, "Peek Continues to grow"
( ) Apr. #4, "We are OSI fans"
( ) May #5, "The continued growth and" (This was labled #4.)
( ) June #6, "This column should"
( ) July. #7, "Several times recently"
( ) Aug. #8, "A few minuttes ago"
( ) Sept. #9, "Of course. It had to"
( ) Oct. #10, "Publishing PEEK(65)"
( ) Nov. #11, "As you can see"
( ) Dec. #12. "This issue marks"