PEEK (65)

The Unofficial OSI Users Journal

$1.75

DECEMBER 1983
Vol.4, No.12

P.O. Box 347
Owings Mills, Md. 21117
(301) 363-3268

INSIDE

Column One

The deal hinted at here last month has been signed! As of November 4, ISOTRON, Inc. is in the driver's seat. Since it has been just a very few days, details are scarce -- but here is what we know:

ISOTRON, the parent company of which is Investments A. B. Beijer (pronounced Bayer), one of Sweden's largest investment houses with corporate links to Volvo and others, has been involved with OSI for years through its subsidiary ISOTRONIC, the very successful Swedish OSI distributor. ISOTRON has bought the assets, trade marks, etc. of OSI that were held by the Bank of America. Robert (Bob) Lewis has been installed as President and William (Bill) Weisberg as Assistant to the President.

Old OSI employees are beginning to be called back: we hear that Jim Cross (sales) and Eric Davis (manufacturing) are back, that others will be coming in shortly, and that manufacturing will be resumed on November 14. The Bedford, MA operation will close and be moved back to Aurora, OH, thus concentrating everything there except corporate headquarters, which will remain in Trumbull, CT.

It has been reported that the net worth of Investments A. B. Beijer is about $75 million, and we hear they are determined to keep OSI in business, continue to support existing operating systems, and press forward with new products. One spokesman said to PEEK (65), "Our base and strength lie in the machines and users currently in the field. We have to support them. No way are we going to give up OS-U!" Likewise, dealers who have stuck through the past hard times are said to be "our Number One concern."

Concerning new products, work has continued on a new product line (a work station) which should be released in the Spring. We all hope this product will be priced to be a reasonable upgrade for all "P" machine users, as well as a valuable adjunct to the larger machines. Other rumors include a 5 1/4" hard disk and IBM compatibility.

PEEK(65), as always, will do everything we can to keep information flowing both from ISOTRON to you and from you to ISOTRON.

On the subject of information movement: it is easy for those of us who have been working with our machines a while to forget how confusing it all was the first time we turned on the power switch. With the factory back in production, and with used machines changing hands, we must remember that there is a never-ending stream of new users, all just as confused as we were.

We would like to help, by publishing a series of "beginner's instruction" articles in PEEK(65), and by facilitating communication among OSI users' groups.

For the former effort, we need authors. Send us your manuscripts, not just on the latest esoteric modification you have made to your board/system, but also on your experiences starting with opening the box.

For the second effort, we need to: 1) identify OSI dealers and users and 2) help them to communicate with each other. We are continually surprised by letters which tell us that the writer has been using his OSI computer for years, but never heard of PEEK(65), knows of no users' group in his area, etc.

To help remedy this situation, we will continue to publish (with appropriate credit) materials received from users' group newsletters, and would like to have a regular users' group activity column -- just keep us informed as to what your group will do.

We also need to know who OSI's users and dealers are. We have fliers available for distributors, dealers and users' groups, which we will be glad to supply them free -- we just have to know who they are!

Finally, we hope soon to be on CompuServe's OSI SIG (see Rick Tretheway's article on TRM65U last month). We support the effort suggested in the letter from OSI Users/Boston (in this issue).

Computing isn't easy. Working with/against the various incarnations of OSI has not always been easy. It has been our pleasure and our duty to help, and our delight that so many of you have joined us. Now that things are started up again, we need each other at least as much as ever.

al

COMPUTER SERVICE GUIDE

Courtesy of TOSIE
Toronto Ohio Scientific Idea Exchange

Procedures for On-site Software and Hardware Support.

1. Approach the computer in a confident manner. This will undoubtedly intimidate the system and will impress anyone who might be watching. A side benefit; if the system suddenly starts working, you will be credited with solving the problem.

2. Wave the System Reference Manual at the machine. This will cause the monitor to assume that you are at least familiar with the sources of knowledge and are a force to be reckoned with.

3. If the customer asks a question (any question) begin reciting scheduling algorithms, swap-rates, function parameter tables, or anything else technical sounding. He will walk away shaking his head and you will be left alone to proceed to the next step, shaking your head.

4. Ask the user what his operating system parameters are. This will give you time to phone your favorite expert, since most users won't be sure what you want, never mind where to find it.

5. Always make sure that all cables (even power) are plugged in. If possible they should even be plugged into the right sockets. It is amazing the number of problems this has solved in the past. It is also very important that no one sees you, why let them in on your secret problem solving methods.

6. Jar the computer slightly. Results of this operation vary, but every approach should be tried.

7. Ask the computer operator what he thinks. This will win him over. Operators are not used to being treated like thinking beings. Besides, he may know what the problem is.

8. Pray.

★

published monthly

Editor - Al Peabody
Technical Editor - Brian Hartson
Circulation & Advertising Mgr. - Karin Q. Gieske
Production Dept. - A. Fusselbaugh, Ginny Mays

Subscription Rates

US (surface)	$15
Canada & Mexico (1st class)	$23
So. & Cen. America (Air)	$35
Europe (Air)	$35
Other Foreign (Air)	$40

All subscriptions are for 1 year and are payable in advance in US Dollars.

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

PEEK (65)
P.O. Box 347
Owings Mills, MD 21117

WORD PROCESSOR FOR OSI

By: Stanley Harshfield
5758 Fox Bend Ave.
Memphis, TN 38115

Like many OSI owners, I desired a word processor, but could not justify its cost based on the amount of writing that I do. As a compromise, I have been able to combine the best features of the AARDVARK JOURNAL Letter Writer (June '80, pg 12) with the OSI SMALL SYSTEMS JOURNAL Word Processor to make a word processor that is very effective for my use. The result is highly enhanced, and permits use with the least expensive printers (mine is an NCR thermal printer, with an 80 character width). Although the program is written for a C1P-MF, it should work with other OSI computers with sufficient memory (mine is 32K). As listed, the program takes up 6.7K, plus two disk buffers, but the stand-alone REMs may be removed. Even BASIC-in-ROM computers can be utilized, following the changes that I will outline.

This program makes extensive use of OSIs "GET" routine, located at 9547 ($252B) in DOS, and at 64768 ($FD00) in ROM. This routine is located early in the program in line 5, in order that the fastest possible response time is obtained. While expert typists may be able to outrun this routine, it will keep up with my typing. The entry pointers for this routine are located in line 1940. Later on, I'll list the changes needed for BASIC-in-Rom. In my computer, location 9504 holds the ASCII value of the key pressed. I understand that newer versions of OS65D3 may store at 9834, while the C4PMF stores at 9815. Change line 5 to agree with the model you are using.

The keyboard routine (lines 10-22) works in conjunction with the "GET" routine to convert the screen display to upper/lower case (if desired). The keyboard works like a normal typewriter, using either right or left shifts, with the following exceptions. In the upper/lower case mode, the left shift will produce "N" or "O" when these letters are typed. If the right shift is used with these letters, the result will be an "↑" or backspace, respectively. With this routine, a line may be entered at any time with a <return> (see line 11). On the other hand, a large amount of text may be entered with no <return> at all, due to the automatic parsing routine (line 20).

The program is initialized in lines 1900-1960, and the user specifies the format of output in lines 2000-2180. Answer "BOSS" or "boss" to the query "WHO ARE YOU?", and your name and address will automatically be entered for the letterhead (see lines 850-860). Otherwise, enter information as requested. Once the format is established,(upper/lower case, letter, etc.) the program is ready to accept the text. The number "1" will appear, and you may start to type. As the line fills, the number "2" will appear, and a new line will be started, without the need of carriage returns, although the <return> may be used, if desired.

If at the beginning of a line you type the single letter "H" or "h" <return>, the HELP menu will be displayed. The HELP command may be the most important command for beginners until the other commands are committed to memory. Other commands which are available include the first letter of the words List, Verify, Advance, Find, Save, Get, Edit, Print, and New.

Since the text is stored as numbered strings, it may be desired to view these strings with the List command. All or any part of them may be examined. Likewise, the Verify command may be used to examine any single given numbered string. Consecutive strings may then be viewed by using the Advance command. The user may locate a given string (or word) with the Find command. This is helpful in finding unique or mispelled words. The Save and Get commands allow the user to record to or input from disk or tape, any portion of the text that is desired. Separate texts may be combined this way. The Edit command will allow a number of options: Word, In-

sert, Delete, Replace, Global edits. The I, D, & R options refer to whole lines, while the W & G options refer to combinations of characters within lines. The Print option relates to the final edited output, and will be discussed later. Type N <return> to start over.

Also shown in the menu are the control characters that determine the format of the output. When one of these characters is used to begin a line, a special routine is invoked which will indent, skip a space, center a title, etc.. These same characters may be used within a line, with no special effect. The character "I" will indent the text following it (such as "Very truly yours,") by 40 spaces to end a letter. It is important to tell the computer when to stop printing, so use ")" as a final line, and it will signal an end to the job and then return to the menu.

The Print command invokes a very involved routine that is the final outcome of the process. You will first be asked how many characters per line (from 24 to 78) are desired. The computer will then calculate the tab setting needed to center the lines on the page. As it prints, it will count out the number of words that will fit within the number of characters that you specified. If you specified right justification, it will also automatically insert extra spaces between words (inexpensive printers do not allow for proportional spacing) so that the right-hand margin is straight. This section will check for control characters at the beginning of each line, and act accordingly. If at any time, the program stops with an error message (e.g. out of memory), type "GOTO90 <return>". This will allow you to return to the program, and save all the text that you have laboriously entered.

For a disk system, prepare a floppy with a six track file for the word processor. The remaining tracks are used to set up 12 two-track files, called "FILE1", "FILE2",... "FILE12". Tracks 7-11 may be used, since they are not used by DOS by the CIP-MF. Before starting to type, run "CHANGE" and set up two buffers. They will be used to store 4K of text.

Continued

```
1 GOTO1900:***WORD PROCESSOR***
5 X=USR(X):P=PEEK(9504):RETURN:***GET SUBROUTINE***
10 A$="":REM***KEYBOARD SUBROUTINE***
11 GOSUB5:Q=PEEK(57088):IFP=13THEN22
12 IFP=76THEN16
13 IFQ=252ANDP=94THEN19
14 IFQ=250ORQ=252ORQ=255ORQ=218THEN18
15 IFQ=220THENPRINTCHR$(8);:A$=MID$(A$,1,LEN(A$)-1):GOTO11
16 IFT0=2ANDP>64ANDP<91THENP=P+32
17 GOTO19
18 IFP>80ANDP<107THENP=P-16
19 IFP=64THENP=80
20 PRINTCHR$(P);:IFLEN(A$)>55ANDP=32THEN22
21 A$=A$+CHR$(P):GOTO11
22 PRINT:RETURN
89 REM***TEXT ENTRY SECTION***
90 PRINT:PRINTL;:GOSUB10
100 IFLEN(A$)=1THEN130
110 A$(L)=A$:L=L+1:GOTO90
130 A=ASC(A$):IFA=70ORA=102THEN950:FIND
140 IFA=76ORA=108THEN280:LIST
150 IFA=71ORA=103THEN670:GET TEXT
160 IFA=80ORA=112THEN320:PRINT
170 IFA=83ORA=115THEN500:SAVE TEXT
180 IFA=72ORA=104THEN775:HELP
190 IFA=78ORA=110THEN750:NEW DOCUMENT
200 IFA=86ORA=118THEN710:VERIFY
210 IFA=65ORA=97THEN730:ADVANCE LINE
220 IFA=69ORA=101THEN1600:EDIT
230 GOTO110
279 REM***LIST***
280 GOSUB3000:INPUT"FROM (type A for all)";A$
285 IFASC(A$)=65THENX=1:Y=L-1:GOTO300
290 X=VAL(A$):INPUT"TO";Y:PRINT:IFY>LTHENY=L
300 PRINT"HARD COPY?":PRINT:GOSUB5:IFP=89THENGOSUB3010
310 FORI=XTOY:PRINTI;A$(1):NEXT:GOSUB3020:GOTO90
319 REM***PRINT***
320 GOSUB3000
321 R=0:INPUT"HOW MANY CHARACTERS PER LINE (MAXIMUM 78)";WIDTH
322 IFWI>=78THENWI=78:H=WI:T=0:G=T:GOTO335
325 IFWI<24THENWI=24:H=WI
330 WI=INT(WI):H=WI:T=INT((78-WI)/2):G=T
335 PRINT:PRINT"RIGHT JUSTIFICATION?":GOSUB5:IFP=89THENR=1
340 PRINT:PRINT"HARD COPY?":GOSUB5:IFP<>89THENPRINT:GOTO380
350 GOSUB3010
380 PRINT:IFF>1THEN395
390 PRINTTAB(39-(LEN(N$)/2))N$
391 PRINTTAB(39-(LEN(C1$)/2))C1$
392 PRINTTAB(39-(LEN(S1$)/2))S1$:PRINT
395 IFF<>2THEN402
400 PRINTTAB(60-T)C1$:PRINTTAB(60-T)S1$
402 IFF=4THEN420
405 PRINTTAB(60-T)D$:PRINT:PRINT:IFF=3THEN420
410 PRINTTAB(T)B$:PRINTTAB(T)C$:PRINTTAB(T)$$:PRINT:PRINT
420 T$="":FORX=1TOL-1:T$(X)=A$(X):IFASC(T$(X))<48THENGOTO1200
425 IFX=1THENT$=T$(X)
430 IFX>1THENT$=T$+" "+T$(X)
435 IFLEN(T$)<WITHENNEXTX
440 IFLEN(T$)<WITHENFL=1
445 IFLEN(T$)<=WITHENPR$=T$:T$="":GOTO485
450 Q$=LEFT$(T$,WI):FORY=WITO1STEP-1:IFMID$(Q$,Y,1)<>" "THENNEXTY
460 PR$=LEFT$(Q$,Y):T$=RIGHT$(T$LEN(T$)-LEN(PR$))
465 IFRTHENGOSUB1050
485 PRINTTAB(T)PR$:IFFLTHEN491
486 IFLEN(T$)=>WITHEN445
490 NEXTX
491 FL=0:IFF=3ORF=4THEN495
492 PRINT:PRINT:PRINT:PRINTTAB(40)N$
495 GOSUB3020:GOTO90
499 REM***SAVE TEXT***
500 GOSUB3000:INPUT"FROM LINE # (type A for all)";X$
510 IFASC(X$)=65THENX=1:Y=L-1:GOTO530
520 X=VAL(X$):INPUT"TO";Y:PRINT:IFY>L-1THENY=L-1
530 GOSUB3050
540 FORI=XTOY:PRINT#6,A$(I):NEXT:PRINT#6,"&&&&"
550 GOSUB3070:GOTO90
559 REM***REPLACE LINE***
560 INPUT"REPLACE LINE #";1:1F1>L-1THEN90
570 PRINT:PRINT"REPLACE:":PRINT"  "A$(I):PRINT:PRINT"WITH:":PRINT
580 GOSUB10
590 A$(I)=A$:GOTO90
```

Continue

```
629 REM***INSERT LINE***
630 INPUT"INSERT AFTER LINE #";X:PRINT:IFX>LITHEN90
640 PRINT"?";:GOSUB10
650 L=L+1:FORY=LTOX+2STEP-1:A$(Y)=A$(Y-1):NEXT:A$(X+1)=A$:GOTO90
669 REM***GET TEXT***
670 GOSUB3000:GOSUB3030
680 INPUT#6,A$:IFA$="&&&&"THEN90
690 A$(L)=A$:L=L+1:PRINT:PRINTA$:GOTO680
709 REM***VERIFY LINE***
710 GOSUB3000:INPUT"WHICH LINE DO YOU WANT TO VERIFY";I:PRINT
715 IFI=0THENPRINT:PRINT"TOO LOW, TRY AGAIN":GOTO710
720 PRINT:PRINTI;A$(I):GOTO90
729 REM***ADVANCE LINE***
730 GOSUB3000:I=I+1:GOTO720
749 REM***NEW DOCUMENT***
750 FORX=0TOL+1:A$(X)="":NEXT:PRINT:PRINT:L=1
760 GOSUB3000:PRINT"NEW DOCUMENT":PRINT:GOTO2000
774 REM***HELP---LIST COMMANDS***
775 GOSUB3000:PRINT" Help
777 PRINT" Find word
780 PRINT" Edit text
782 PRINT" Verify line
785 PRINT" Advance line
788 PRINT" Save text
790 PRINT" Get text
792 PRINT" List
795 PRINT" Print
797 PRINT" New document
800 PRINT:PRINT
805 PRINT" &:CENTER LINE ON PAGE
810 PRINT" /:NEW LINE
815 PRINT" *:SKIP LINE
820 PRINT" +:INDENT LINE
825 PRINT" #:INDENT NEW PARA.
830 PRINT" %:INDENT SUB-SECTION
835 PRINT" !:INDENT CLOSING
840 PRINT" ):END OF TEXT":PRINT:PRINT:GOTO90
849 REM***ENTER LETTERHEAD***
850 N$="Stanley Barshfield"
860 C1$="5758 Fox Bend Ave.":S1$="Memphis, TN 38115":GOTO2140
869 REM***WORD EDIT***
870 INPUT"EDIT WHAT LINE";I:PRINT:PRINTI;A$(I)
875 PRINT:PRINT"OLD STRING?":GOSUB10:Q$=A$
878 PRINT:PRINT"NEW STRING?":GOSUB10:X$=A$
880 PRINT:PRINT"REPLACE: "Q$:PRINT:PRINT"WITH: "X$
885 PRINT:PRINT"CORRECT?":PRINT
890 GOSUB5:IFP<>89THEN90
900 Y=LEN(Q$):FORX=1TOLEN(A$(I)):IFMID$(A$(I),X,Y)=Q$THEN930
920 NEXTX:GOTO720
930 IFX=1THENA$(I)=X$+MID$(A$(I),X+Y):GOTO720
940 A$(I)=LEFT$(A$(I),X-1)+X$+MID$(A$(I),X+Y):GOTO920
949 REM***FIND STRING***
950 GOSUB3000:PRINT"WHAT STRING ARE YOU LOOKING FOR?":GOSUB10
960 PRINT:INPUT"ON WHICH OCCURRENCE";A:IFA=0GOTO960
970 X=0:B=LEN(A$):FORI=1TOL:FORY=1TOLEN(A$(I))
980 IFMID$(A$(I),Y,B)=A$THEN1000
990 NEXTY,I:PRINT:PRINTA$" NOT FOUND":GOTO90
1000 X=X+1:IFX<ATHEN990
1010 GOTO720
1049 REM***RIGHT JUSTIFICATION***
1050 A=WI-LEN(PR$):IFA=0THENRETURN
1060 Z=1:Y=1:FORV=1TOLEN(PR$)
1070 IFMID$(PR$,V,1)<>" "THENNEXTV
1080 F$(Y)=MID$(PR$,Z,V-Z+1):IFV=LEN(PR$)+1THEN1110
1100 Z=Z+LEN(F$(Y)):Y=Y+1:NEXTV
1110 FORV=1TOY-1:F$(V)=F$(V)+" ":IFV=ATHEN1150
1140 NEXTV:A=A-V+1:GOTO1110
1150 PR$="":FORV=1TOY-1:PR$=PR$+F$(V):NEXTV:IFM=0THENM=1:RETURN
1180 M=0:RETURN
1199 REM***SPECIAL COMMAND ROUTINE***
1200 IFASC(A$(X))=42THEND=1:GOTO1300:42=*
1210 IFASC(A$(X))=47THEND=2:GOTO1300:47=/
1220 IFASC(A$(X))=33THEND=3:GOTO1300:33=!
1230 IFASC(A$(X))=38THEND=4:GOTO1300:38=&
1240 IFASC(A$(X))=37THEND=5:GOTO1300:37=%
1245 IFASC(A$(X))=43THEND=6:GOTO1300:43=+
1250 IFASC(A$(X))=35THEND=7:GOTO1300:35=#
1260 IFASC(A$(X))=41THEN1550:41=)
1270 GOTO425
1300 T$(X)=RIGHT$(T$(X),LEN(T$(X))-1)
1310 ONDGOTO1360,1360,1400,1450,1500,1350,1350
```

Continued on page 6.

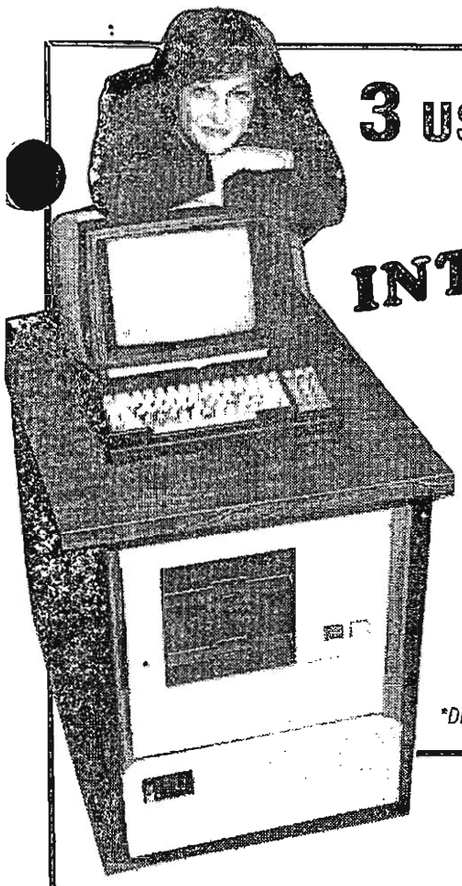
For tape users, make the following changes:

```
5     X=USR(X):P=PEEK(531):
      RETURN
12    IFQ=252ANDP=94THEN20
13    IFQ=252ANDP=95THEN17
14    IFQ=250ORQ=252THEN18
15    IFT0=2ANDP>64ANDP<91THEN
      P=P+32
16    GOTO19
17    PRINTCHR$(95);:A$=MID$
      (A$,1,LEN(A$)-1):GOTO11
540   FORI=XTOY:PRINTA$(I):
      NEXT:PRINT"&&&&"
680   INPUTA$:IFA$="&&&&"THEN
      POKE515,0:GOTO90
690   A$(L)=A$:L=L+1:PRINT:
      GOTO680
1940  POKE11,0:POKE12,253
3000  FORX=1TO32:PRINT:NEXT:
      RETURN
3010  POKE517,1:RETURN
3020  POKE517,0:RETURN
3030  PRINT"TURN ON TAPE PLAYER
3040  POKE515,255:RETURN
3050  INPUT"IS TAPE RECORDER
      ON";A$
3060  POKE517,1:RETURN
3070  POKE517,0:RETURN
```

It will also be necessary to delete line 1960. Does anyone out there know the location of the , & : terminators in BASIC-in-ROM?

I hope that this program will be as much value to you as it has been to me. I find that it is used much more than I expected, since I can now turn out letters and reports with no erasures or obvious corrections.

★



3 USERS-80 Mega Bytes — $9990.00*

INTRODUCTORY SPECIAL

WITH DUAL FLOPPIES
BRAND NEW —
1 YEAR WARRANTY ON HARD DISK!
REGULAR $12,990.00

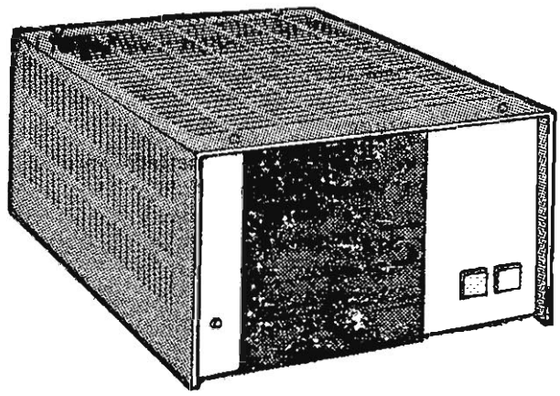
- 90 Days on Power Supply, Floppy Drives — Circuit Boards.
- Configured for Time-Share @ 2 MHZ
- Includes: 2 Serial Printer Ports with Handshake, Improved Cooling, and Ball Bearing Roller Chassis Rails

ALSO AVAILABLE WITH
3 MULTI-PROCESSOR

Denver Boards with 64K each user and Centronics Parallel Printer Port at
$10,990.00

*DEALER DISCOUNTS AVAILABLE

8" HARD DISK SYSTEMS



SINGLE BOX TABLE TOP WITH IMPROVED COOLING 10 M/B HARD DISK AND 8" FLOPPY DISK **2 USERS AND 2 SERIAL PRINTER PORTS**
$5990.00

AS ABOVE WITH 2 MULTI-PROCESSOR 64K DENVER BOARDS
PLUS CENTRONIC PARALLEL INTERFACE **$6990.00**

OR INSTALLED IN CABINET AS ABOVE
WITH DUAL FLOPPIES PLUS 10 M/B.

STD. BOARD TYPE	1 USER w/ Centronics Printer Port	$6490.00
	2 USER w/ 2 Serial Printer Ports	$6990.00
DBI MULTI PROC.	2 USER w/Centronics Printer Port	$7790.00
	3 USER w/Centronics & Serial Printer Ports	$8990.00

MULTI-PROCESSOR DEVELOPMENT SYSTEM SPECIAL

SPECIAL
ONLY $5990.00

- 5 M/B Hard Disk-1 8" Floppy
- 1 Centronics Parallel Printer Port
- 1 Serial Printer Port, 1 Modem Port
- 2 DB-1 Multi-Processors
- Complete Programmer Manual and Software Overlays

DEALERS — We have lots of OSI machines and can build virtually any combination you need. Appropriate dealer discounts.

Please Give Us a Call!

WHERE WE STILL LOVE OS-65U — AND SUPPORT IT!

Space-Com International

22991 LA CADENA DRIVE, LAGUNA HILLS, CALIFORNIA 92653

ORDER TODAY (714) **951-4648** SOME QUANTITIES LIMITED

Listing continued from page 4

```
1350 T$(X)="      "+T$(X)
1360 PRINTTAB(T)T$:T$=T$(X):T=G:WI=H:IFD=2ORD=6THEN435
1365 PRINT:GOTO435
1400 PRINTTAB(T)T$:T=40:PRINT:PRINT:PRINT
1410 PRINTTAB(T)T$(X):PRINT:PRINT:PRINT:PRINTTAB(T)N$:GOTO495
1450 PRINTTAB(T)T$:T$="":T=T+(WI-LEN(T$(X)))/2
1460 PRINT:PRINTTAB(T)T$(X):T=G:GOTO490
1500 PRINTTAB(T)T$:T$=T$(X):PRINT
1510 G=T:H=WI:T=T+5:WI=WI-5:GOTO435
1550 PRINTTAB(T)T$:PRINT:GOTO491
1599 REM***EDIT***
1600 GOSUB3000:PRINT"EDIT HOW?":PRINT:PRINT:PRINT
1605 PRINT" Insert line":PRINT;PRINT" Delete line":PRINT
1610 PRINT" Replace line":PRINT:PRINT" Word edit":PRINT
1615 PRINT" Global edit":PRINT:PRINT:GOSUB5
1620 IFP=73ORP=105THEN630:INSERT LINE
1625 IFP=68ORP=100THEN1800:DELETE LINE
1630 IFP=82ORP=114THEN560:REPLACE LINE
1635 IFP=87ORP=119THEN870:EDIT WORD
1640 IFP=71ORP=103THEN1700:GLOBAL EDIT
1645 GOTO1600
1699 REM***GLOBAL EDIT***
1700 PRINT"REPLACE WHAT WORD?":GOSUB10:Q$=A$
1705 PRINT:PRINT"REPLACE '"Q$"'":PRINT"WITH WHAT?":GOSUB10:X$=A$
1710 B=LEN(Q$):FORI=1TOL:FORY=1TOLEN(A$(I))
1715 IFMID$(A$(I),Y,B)=Q$THEN1725
1720 NEXTY,I:GOTO90
1725 IFY=1THENA$(I)=X$+MID$(A$(I),Y+B):NEXT:GOTO90
1730 A$(I)=LEFT$(A$(I),Y-1)+X$+MID$(A$(I),Y+B):NEXT:GOTO90
1799 REM***DELETE LINE***
1800 PRINT"DO YOU WANT TO DELETE A SERIES?":GOSUB5:IFP=78THEN1835
1805 PRINT:INPUT"LOWEST LINE NUMBER";A
1810 PRINT:INPUT"HIGHEST LINE NUMBER";B
1815 IFA>BTHEN1805
1820 IFA<1ORB>L-1THENPRINT:PRINT"OUT OF RANGE.":GOTO90
1825 C=B-A+1:FORI=ATOL-C:A$(I)=A$(I+C):NEXT
1830 FORI=L-CTOL:A$(I)="":NEXT:L=L-C:GOTO1850
1835 PRINT:INPUT"WHICH ONE THEN";X:IFX<1ORX>LTHEN90
1840 PRINT:PRINTA$(X):PRINT:PRINT"O.K.?":GOSUB5:IFP<>89THEN90
1845 L=L-1:FORY=XTOL:A$(Y)=A$(Y+1):NEXT
1850 PRINT:PRINT"IS THAT ALL?":GOSUB5:IFP=89THEN90
1855 GOTO1835
1899 REM***INITIALIZE****
1900 GOSUB3000:FORX=3TO20:PRINTAB(X)"=";:NEXT:PRINT:PRINTAB(3)"=";
1910 PRINTAB(20)"=":PRINT"   = WORD PROCESSOR ="
1920 PRINTAB(3)"="TAB(20)"=":FORX=3TO20
1930 PRINTAB(X)"=";:NEXT:PRINT:DIMA$(150),T$(150),F$(20)
1940 POKE8955,43:POKE8956,37:REM*KEYBOARD ENTRY
1950 PRINT:PRINT:PRINT:L=1
1960 POKE2976,13:POKE2972,13:REM*DISABLE , & : TERMINATORS
1999 REM***ESTABLISH FORMAT OF OUTPUT***
2000 PRINT"IS THIS TO BE:":PRINT:PRINT:PRINT" 1>UPPER CASE ONLY
2010 PRINT:PRINT" 2>UPPER/LOWER CASE":PRINT:PRINT:GOSUB5
2020 T0=P-48:IFT0>2THENPRINT:PRINT:GOTO2000
2030 GOSUB3000:PRINT"IS THIS A:":PRINT:PRINT
2040 PRINT" 1>LETTER W/LETTERHEAD":PRINT
2050 PRINT" 2>LETTER WO/LETTERHEAD":PRINT
2060 PRINT" 3>REPORT W/DATE":PRINT:PRINT" 4>MANUSCRIPT WO/DATE
2070 PRINT:PRINT:GOSUB5:F=P-48:IFF>4THEN2030
2080 IFF=4THEN775
2090 IFF<3THENPRINT"YOUR NAME?":GOSUB10:N$=A$
2100 IFF>2THEN2140
2110 IFN$="BOSS"ORN$="boss"THEN850
2120 PRINT"YOUR STREET ADDRESS":GOSUB10:C1$=A$
2130 PRINT"YOUR TOWN & STATE":GOSUB10:S1$=A$
2140 PRINT"PRINT"DATE?":GOSUB10:D$=A$
2150 IFF<1ORF>2THEN775
2160 PRINT:PRINT"NAME OF RECIPIENT?":GOSUB10:B$=A$
2170 PRINT:PRINT"STREET ADDRESS?":GOSUB10:C$=A$
2180 PRINT:PRINT"TOWN & STATE?"$GOSUB10:S$=A$:GOTO775
3000 POKE9803,33:PRINT:POKE9803,0:RETURN:***SCREEN CLEAR***
3010 DISK!"IO ,03":NULL6:RETURN:PRINTER ON
3020 DISK!"IO ,02":RETURN:PRINTER OFF
3030 INPUT"GET WHICH MEMORY";A$:A$="FILE"+A$
3040 DISK OPEN,6,A$:RETURN
3050 INPUT"WHICH MEMORY (1-12)";A$:A$="FILE"+A$
3060 DISK OPEN,6,A$:RETURN
3070 DISK PUT:DISK CLOSE,6:RETURN
```

THE 2716, PHASE 2 AND OTHER MYTHS.

By: Paul C. of TOSIE

Almost every hardware question I am asked involves an EPROM, a simple device that continues to confuse a lot of people. It also confused OSI since even they have made design errors using this chip. I don't want to talk about EPROM theory, since it is covered in any electronics book, but I do want to point out what might have been missed. The pinouts are very straight-forward, the 'A' lines are the address lines, the 'O' or 'D' lines are the data lines, GND is ground and Vcc is the supply voltage typ. +5 volts. The pins that cause problems for most beginners are 18 (CE), 20 (OE) and 21 (Vpp). Vpp (program voltage) is used for just that, programming the chip, in normal use (when not programming) this pin should be equal to Vcc (+5V). Grounding this pin will not harm the chip but it is not a logical state and most 2716 will not work if it is. What confuses people is that some makes do work but they shouldn't. In short the Vpp should be +5V and then forget about it. It should also be noted that during application of power, care must be taken to assure that Vcc is applied before or simultaneously with Vpp and Vpp must be removed before or simultaneously with Vcc or you may be buying a new chip.

Pins 18 and 20 are both active low, when the pin is equal to Vcc the function is disabled and enabled when the pin is grounded. Chip Enable (CE) is the power control, and must be used to select the device. Output Enable (OE) controls only the ouput stages and must be used to gate data to the output pins. The common use of these chips with a 6502 system is to activate CE as soon as the address lines indicate the chip is to be selected and use phase 2 to gate the data. The time required for data to be available for output from the time the chip is selected (CE=0) is known as the access time. Typical access times are shown in table 1. The max. access time for a system at 1MHz is approx. 700 to 1000 nanoseconds depending on how the 2716 is used in the circuit. A common mistake involving access times is to use phase 2 in the address decoding that controls the Chip Enable line. This means

that the chip is not even selected till the CPU is halfway through its cycle and the max. allowable access time is therefore divided by two. An example can be seen in the decoding of the BASIC ROMs on the Superboard Rev D.

Even though this is a mistake, the Superboard does still work, the problem is if you try to go to 2MHz the ROMs appear too slow for the system. There are two solutions if you have problems with slow access times, you can buy faster EPROMs which cost more or you can try the following: If CE is active at all times and OE is used to select the ROMs the output enable to output delay time is typically 120 nanoseconds, more than fast enough for most uses. The drawback of this application is that the EPROM will be drawing full power at all times. Power dissipation when selected is 1.OW, and when in standby mode (CE=1) 0.132W.

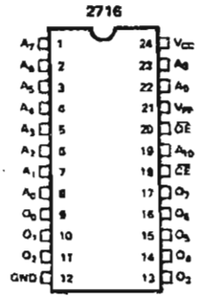
The other common problem arises when substituting 2716s for the OSI BASIC ROMs or monitor ROM. In many models OSI used the 2316; the difference is that this chip is programmed during manufacturing of the chip and that pins 18, 20 and 21 are all programmable chip select lines. In typical OSI fashion, they had these lines programmed differently from the 2716, i.e., they may or may not be active low. Therefore, when replacing these ROMs you may have to invert signals to some of these pins.

After rereading this article, I realize two things: I'm not a technical writer, and no wonder people get confused. I hope this article helps somebody but if you still have questions all you have to do is ask. Values used were taken from the Synertek Data Book.

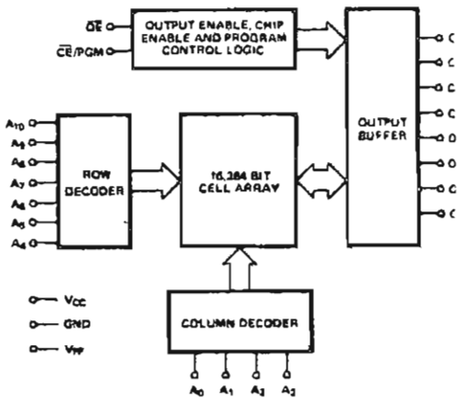
TABLE 1

Parameter	2716	2716-1	2716-2
Address to output delay	450 nS	350 nS	390 nS
Outpnt enable to output delay	120 nS	120 nS	120 nS

PIN CONFIGURATION



BLOCK DIAGRAM



★

RELOCATING WP-6502
Part 2

By: John T. Roecker
5141 Thomas Ave., S.
Minneapolis, MN 55410

I was overjoyed after purchasing an Epson MX80 printer. I had relocated WP-6502 in order to use it with my C1P with a C1S monitor ROM. I knew WP-6502 was working because I could create a tape of an article or letter and take the tape to a friend who had a printer to have it printed. This was inconvenient and also could possibly tax a friendship. I had the RS-232 interface populated so all I had to do was connect the Epson to it. I ran a quick test in BASIC and the printer worked fine. However, when I attempted to use WP-6502, the printer did not! After much head scratching, I remembered that I had to use a different output routine in my C1S monitor ROM in order to output to tapes using WP-6502. I surmised I would have to use this routine to output to the printer also. Those of you who have C1E/C2Es can rest easy because these ROMs appear to use the standard output routines. A quick test checked my idea, WP-6502 would drive the printer with another modification.

My modification to WP-6502 was to add a new command, the Print command, to the WP-6502 repertoire and to disable using the View command to output to the printer. Those of you with standard OSIs or with C1E/C2E monitor ROMs may find this new command useful.

I used the following steps to add the Print command to WP-6502. All address locations mentioned are the original addresses from your nonrelocated tape version of WP-6502. All instructions with a * behind them will have to have their address fields modified to suit your relocation. All instructions with a & behind them are new instructions which have been added.

1. Expand the WP-6502 menu so that the Print command may be added. The menu plus other words outputted to the screen are located at memory locations $070D through $0783. I modified the menu to have it look like this:

```
---WP
   6502
Type
View
Blk View
G/Edit
L/Edit
Move
Print
Zap
R/Tape
W/Tape
```

I used the OSI Extended Monitor to relocate locations $0736 through the end of the cold start code of WP-6502, $0FD0, by 5 bytes. Then I added the new Print command at $0736:

```
$0732  4D6F76E5     Move
$0736  5072696EF4   Print
$073B  5A61F0       Zap
```

2. Contract the View command code to eliminate the Pr? after View. The View command code is located at memory

locations $0795 through $09F9. I dropped the instructions located at $0798, $079A, $079D and $079F by relocating $07A1 to $0798.

3. In the process of performing these two relocations, I managed to destroy two instructions. One of them stopped L/Edit and G/Edit from working from the menu. The instruction which was destroyed for this problem was located at $078F. The address field of this instruction should be modified by the relocations performed in the previous steps.

```
$078F  20940A  JSR  $0A94 *
```

The second instruction which was destroyed caused an insert at the 'End of Text' to operate improperly. This instruction should be:

```
$0C46  4C5A0F  JMP  $0F5A *
```

4. Any references which index into the WP-6502 menu data area may have to be corrected because we added a new command. References for commands and data after the new Print command will have to be modified by adding 5 bytes to the immediate data:

```
$03FA  A052  LDY #$52
$0465  A040  LDY #$40
$06AD  A05B  LDY #$5B
$0787  A055  LDY #$55
$07B4  A043  LDY #$43
$07BB  C04E  CPY #$4E
$07E2  A050  LDY #$50
$09B0  A06D  LDY #$6D
$09F0  A06D  LDY #$6D
$0A5B  A060  LDY #$60
$0B12  A03E  LDY #$3E
$0B4D  A071  LDY #$71
$0D01  A060  LDY #$60
$0D0C  A060  LDY #$60
$0D44  A066  LDY #$66
$0E19  A059  LDY #$59
$0EA9  A06D  LDY #$6D
```

5. The warm start code will have to be modified so it will recognize the Print command. Those of you with C1S/C2S monitor ROMs will have to add this check after the switch to the new output routine which was added in the last article.

This is what the code looked like before the change:

```
$0F8F E057   CPX #$57    Check for
                         W/Tape
$0F91 D003   BNE $0F96
$0F93 20F30E JSR $0EF3
$0F96 4C6504 JMP $0465   Not legal
                         command
```

This is after the change:

```
$0F8F E057   CPX #$57    Check for
                         W/Tape
$0F91 D003   BNE $0F96
$0F93 20F30E JSR $0EF3 *
$0F96 E050   CPX #$50  &  Check for
                         Print
$0F98 D003   BNE $0F9D &
$0F9A 4C9807 JMP $0798 &* Print
                         output
$0F9D 4C6504 JMP $0465 *  Not legal
                         command
$0FA0 5D00
$0FA2 40                 Starting
                         text
                         location
```

6. The cold start code will have to be modified to use the proper data/text starting location. I have indicated this location in step 5 above; in this case $0FA2. The amounts of the relocations will have to modify this address. I have reproduced all the cold start code below for clarity. I have placed my cold start code at $1024.

```
$1024 A94C   LDA #$4C     Store
$1026 8500   STA $00      warm
$1028 A90F   LDA #$0F     start
$102A 8502   STA $02      jump
$102C A90B   LDA #$0B     instruction
$102E 8501   STA $01      ----
$1030 A924   LDA #$24     Store
$1032 8503   STA $03      cold start
$1034 A910   LDA #$10     address
$1036 8536   STA $04      ----
$1038 A90F   LDA #$0F     Store
$103A 8D4202 STA $0242 *  starting
                          text
$103D A9A2   LDA #$A2     Address
$103F 8D4102 STA $0241 *  ----
$1042 A900   LDA #$00
$1044 8546   STA $46      ----
$1046 ADE2FF LDA $FFE2    Test for
                          C1P
$1049 D00A   BNE $1055    Branch
                          if not
$104B A914   LDA #$14
$104D 8D3602 STA $0236 *
$1050 A9FF   LDA #$FF
$1052 8D4002 STA $0240 *
$1055 4C0000 JMP $0000    Jump to
                          warm start
```

The immediate data at $1039 and $103E will have to be modified to point to your starting text address.

Now, after much blood, sweat, and some tears, those of us with nonstandard monitor ROMs installed may use WP-6502. To eliminate all this work, all machine language / Assembler code should start at a suitably high address. The Assembler/Editor starts at location $0240. I feel this would be a good starting address to enable anyone with standard or nonstandard monitor ROMs to use your program.

I have made additions to WP-6502 to utilize some of the features of my C1E monitor ROM. These additions will be the third article in this series. ★

HOOKS INTO BASIC V1.8

by: Rick Trethewey
8 Duran Court
Pacifica, CA 94044

In September 1980, OSI published a method of adding keywords into BASIC under OS-65D. Since that time, I have been expanding on that theme. My first attempts were published in the April 1981 issue of MICRO and a subsequent version was published in the book "MICRO on the OSI". Unfortunately, MICRO sat on the version they put in their book for about a year and even since then I have continued to expand on the code. My feeling has always been that despite the obvious shortcomings of BASIC, it remains the most used language on OSI systems and thus anything that enhances that language is a worthwhile project. I chose OS-65D V3.3 because it is the latest (and perhaps last) "official" release of the operating system and it is virtually identical in both the 8 inch and minifloppy versions. The last point makes it possible for the same code to run on everything from a C4P-MF to a full-blown C-3. However, there is a price to be paid for these additions in lost memory. The object code for "Hooks" requires 10 pages of memory. That's fine for a 48K system and barely acceptable on a 32K system. But if you only have 24K, Hooks will only leave you with an 8K workspace unless you delete some parts of the code.

Hooks' primary latch into BASIC is in the code that handles equations. When BASIC starts to look at a line of code, it checks the first character to see if it's a BASIC token representing a command. If it is a token, the appropriate command is executed. If it is not a token, BASIC assumes that an equation is in progress and begins to handle a variable name. Hooks intercepts the variable name handler and checks to see if the text is one of the new keywords and if it is not, control is passed back to BASIC. If Hooks does recognize a "keyword" it retains control and executes one of the new commands. There are two "gotcha's" involved here. First of all, there is a reduction in the speed of BASIC. The speed overhead isn't that much and since BASIC isn't noted for speed anyway, this should not become a problem. The second is that all of the new Hooks

High Resolution Color Graphics

Our new Color Plus board provides 256 x 192 high-resolution graphics with 15 colors. Two 8-bit resolution joystick interfaces are included. Software extensions to OS65-D BASIC provide a superset of APPLE II® graphics instructions.

Color Plus connects to the standard 48-pin bus or the 16 pin bus.

Pricing:

CP-8 for C8 or C3 computers:	$195
CP-4 for C4 computers (5V only):	$245
CP-bare Bare board with software:	$ 75

Generos Operating System

Generos is a new operating system for OSI computers. Features include descriptive error messages, optimized disk usage, device independence, and:

- ASSEM — Fast disk based assembler generates relocatable code.
- BASIC — Basic Interpreter
- DDT — Machine language debugger with single step, trace, more.
- TECO — Most powerful and widely used text editor ever.

Currently available for 48K 8-inch systems. Call or write about availability for other configurations. Cost: $85

Low Power Memory Board

Our popular MEM + board is ideal for:

- Partitions for multi-user systems
- 64K CP/M systems when combined with the D&N-80 CPU board
- Upgrading systems where backplane space, low power consumption, and/or low heat dissipation is required

Options include:

- OSI compatible floppy disk controller — protects against disk crashes caused by power failures
- Real time clock/calendar — Date and time with battery backup
- Centronics parallel printer interface — Supported by software that automatically patches OS65D and OS65U
- One year warranty

VISA, MasterCard, personal checks and C.O.D.s all accepted. Add $5 per board for shipping and handling.

To order, or for more information, contact:

Fial Computer
5221 S.W. Corbett
Portland, Oregon 97201

(503) 227-7083

MEM + includes the following features:

- Memory chips in quality production sockets; high reliability machine screw sockets optional
- Low power consumption
- Uses 2K x 8-bit memory chips — will accept 2716-type EPROMS.
- Versatile addressing

NEW LOW PRICES!!

Bare — Solder masked and silkscreened $75
Software and documentation provided

16K	$200		
24K	$250	Disk controller	$85
32K	$300		
40K	$350	Real time clock	$65
48K	$390		
52K	$415	Centronics interface	$45
56K	$440		
64K	$490	High-rel sockets add 15%	

07514 51893

Generic Computer Products

5740 S.E. 18th Ave. Portland, OR 97202

Continued from page 8.

keywords are also reserved and cannot be used as variable names so some of your old programs may require editing.

To begin, you'll have to type in the source file for Hooks. Without all the comments, this takes about 9 tracks on an 8" system. Clearly, systems with less than 48K are going to run into trouble using the OSI Assembler/Editor here. For those people, I suggest buying the OSI Assembler/Editor Manual for $5.95. That book describes the method of carrying over the symbol table from one source file to the next, thus allowing the linking of source files. The alternative would be to simply break up the source code into separate files and make sure that each file includes the address labels it needs. Both of these methods are a bit tedious and will make you very aware of the benefits of assemblers that automatically link source files. You will also have to make two changes to the source file if you have less than 48K. The first is the origin address. You should make this $7600 if you have 32K and $5600 if you have 24K. The second change is another origin address that is set just before the code that allows BASIC to understand HEX in expressions. This should be changed to $7E1B on 32K systems and $5E1B on 24K systems. Except for these two changes, both the source code and the BEXEC* program will automatically configure themselves to your system's disk and memory sizes. Before you assemble the Hooks code, you should prepare an OS-65D V3.3 diskette with two files of 1 track each for 8" systems and 2 tracks each on minifloppies. Name the first file "BEXEC*" and the second file "BASIC+". The file "BASIC+" will hold the Hooks object code. When you have assembled and saved the Hooks code to disk, enter and save the BEXEC* program listed here. This BEXEC* will automatically make the necessary changes to BASIC to incorporate Hooks on the initial boot-up and will display the diskette's directory and a menu of options.

Now let's talk about what all that typing will gain for you. The following "keywords" are supported by Hooks:

C*
--
Clears the 540 black and white video.

B*exp

Fills the 540's color background with the value of the expression following the asterisk and sets the cursor backround to the same color.

Q*exp

POKEs 56832 with the value of the expression following the asterisk to set the 540's color, sound, and character size.

R*
--
Enables "NEW", "LIST" , and <CTRL>'C'.

T*
--
Toggles the BASIC line trace on and off just as if the program "TRACE" had been run.

S*a$

Selects the disk drive corresponding to the string (literal or variable) that follows the asterisk. This string must be only one character long and in the range of A-D (upper or lower case) or a SYNTAX ERROR will result.

D*
--
Prints the directory of the currently selected drive. You'll note that the BEXEC* tabs the "* Directory *" message to center it a bit, but that the code will usually left-justify this.

ASM

Invokes the Assembler/Editor.

EM
--
Invokes the Extended Monitor.

LOAD"FNAME"

Loads the file "FNAME" into the workspace.

SAVE"FNAME"

Writes the current workspace contents to the file "FNAME". If "FNAME" already exists, then the current file length is checked to make sure that the workspace contents will fit BEFORE the attempt is made to write out the file. This prevents an "ERR #D ERROR" from trashing the current file contents. If the disk file is not big enough to hold the workspace contents, an error message is displayed saying so. If "FNAME" is not found in the directory, Hooks automatically creates a file of sufficient size to hold the current workspace contents. Again, if there are not enough contiguous free tracks on the disk to make such a file, an error message is displayed. "SAVE" can be used on both BASIC programs or Assembler source files loaded while in BASIC. However, Hooks looks at location $3A7E to determine if the file is a BASIC or Assembler file. This will provide correct results unless a BASIC program with a buffer at the start of the workspace is currently in the workspace. To make sure of proper results, POKE $3A7E,0 (remember, that's legal now!) before using SAVE on these files. Yes, this logic will also fail on Assembler files whose first line number is an even multiple of 256, but I don't know of anyone using such numbering.

MAKE"FNAME",exp

Creates a file named "FNAME" whose length is the value of the expression that follows the comma. Variables can be used for the file name and the desired length can be the result of a calculation.

RENAME "OLDNAM" TO "NEWNAM"

No mystery here, the file "OLDNAM" is renamed "NEWNAM". The code checks for duplicate and legal file names.

KILL"FNAME" (,"F2NAM",...)

Deletes all of the file names following the keyword "KILL" separated by commas.

PACK

Frees up all unused space on the disk by moving all files to the lowest possible track number. This makes all unused disk space available in a contiguous block at the end of the disk. This command requires a disk buffer in the user's workspace the size of one track. PACK tries to find space between where arrays and strings are stored in the workspace. If there is room, the command is executed. If not, an "OM ERROR" is displayed. I suggest that if you incorporate PACK in a program you do a FRE(X) immediately before invoking PACK. This makes the maximum amount of memory available. You shouldn't have any problems with FRE(X) causing a system crash here since utility programs use few string arrays, yet 24K systems may well need the extra RAM. * CAUTION * PACK can cause disastrous results

if a disk error occurs during the packing process. The headaches caused by a bad disk are never worth the price of a new disk.

VIEW

VIEW displays the values of all non-subscripted variables. The variable names are marked for integer or string types. Control characters within strings are displayed with a caret ("^") followed by the letter of the control character. That is "^C" would be <CTRL>'C'.

CALLexp

CALL executes the machine code routine located at the value of the expression following CALL.

WAIT addr.,expl (,exp2)

This is the standard WAIT command from BASIC under OS-65D V3.2.

As you can see, a large portion of Hooks is dedicated to making disk file management commands a part of the language so that they're "on-line" instead of requiring special utility programs. The idea here is to make the system work for you instead of against you. For example, how many times have you, in the midst of developing a program, after adding several lines of code to a program, tried to use DISK!"PU FNAME" only to get the dreaded "ERR #D ERROR"? Unless you were wise enough to maintain a special disk containing a scratch file, you would be really stuck in this situation. You not only have no place to put the latest version of your program, you've also managed to trash the original file as well. Using SAVE, KILL, and PACK will totally alleviate these problems. SAVE also does the track allocation check before attempting to write the file to disk to prevent it from ever trashing the original file.

While we're on the subject of errors, it should be noted that Hooks uses BASIC's error handling wherever possible. This allows use of the TRAP command under OS-65D V3.3. Where there are no proper error messages in BASIC, Hooks provides it's own.

Hooks also makes changes to the language itself. The biggest change is that BASIC will now understand hexadecimal numbers in numeric expressions. Thus "DIRBUF=$2E79" and "POKE $8000,2" are now legal. This does not apply to INPUTs or DATA statements. BASIC can now print in either HEX or decimal as well. To have a value printed in HEX, preceed the expression with "$,". "PRINT $,11897" would display "$2E79". This function is fully compatible with 3.3's cursor addressing, but not PRINT USING since only integer values are printed. There is a difference between the HEX inputs and outputs in accuracy. Only 16-bit inputs are legal, but the output has a 32-bit accuracy.

With Hooks installed, GOTO's and GOSUB's will accept either line numbers or variables. This can make your programs more readable, but you'll have to remember to change the values of the variables if you RSEQ your program. This function is NOT available in ONxGOTO's or ONxGOSUB's.

Finally, BASIC will now allow a limited IF...THEN...ELSE. Hooks does this by putting the keyword "REM" to an extra use. If an IF statement is evaluated as TRUE, all proceeds normally. But if the statement is FALSE, the line containing the IF statement is scanned for a "REM". If a "REM" is found, it is treated as a GOTO and control is passed to the line number (or variable!) following the REM. If no REM is found, control is passed to the next statement. For example, consider "100 IF X=2 THEN Y=3:REM 200". Under Hooks, if X does indeed equal 2, Y is set to 3 and control is passed to the next line in the program. But if X does not equal 2, control would be passed to line number 200. This allows a truer expression of the programmer's intentions sometimes than the normal IF statement would allow. This does mean that you'll have to be careful where you put your REM's though.

I expect that a lot of people will wonder if this code can be run under OS-65D V3.2. The answer is yes. You will have to make 3 changes to the source code for 3.2. The first is to change the label "CASECK" so that it points to an "RTS". CASECK is the routine in 3.3 that converts lower case alphabetic characters to upper case. The second change is to the label "SRCSTR". On mini-floppies this address should be changed

Article continued on page 14

TIME & TASK PLANNER
A REVIEW

PEEK(65)
By: Edward T. Gieske, Jr.

Time & Task Planner's name either tells you a lot or very little - depending upon where you come from. Those who are already in the habit of keeping diaries, appointment books and the like will have a good idea, but I doubt that any would have guessed the completeness of this work. John Huntley (Mr. Gander Software) sure was organized when he put this one together. One of the beauties is that, although the programming is very complex, the user is presented with a very simple and straightforward program to "use". Remember, TTP is out of the same mold as the Financial Planner reviewed in the May issue of PEEK(65).

Very nice, but what's it all about? A better mouse trap? Yes! Start off with the premise that you are not as organized as you ought to be. Run down to the stationers and pick out a desk calendar that comes close to your usual schedule. Put up with its lack of room to make all the entries you need, not to mention, searching the streets for another one like it for next year. Scribble little notes on little scraps of paper about the things you must do - next week, month, or year (forget that - the paper will be gone long before then). Rewrite the list again and again trying to put the important items at the top. Oops! Where did that note go about the estimated taxes due?

Well, you get the picture. Whether its a doctor's daily appointment schedule, Johnny's birthday or the date of your annual check-up, chances are you won't have the information at hand when you need it, much less know that the 18th falls on a Sunday in 1984 when you had planned the meeting for next summer with your stock broker.

Now let's look at it in a positive attitude. TTP can give you the "Winner's Edge". With TTP you will have the opportunity to define and set your goals and thus give you a clear and immediate challenge and the ability to track your performance. Once organized, it is amazing how much more you can get done in a day. In fact, just recording your goals helps to define and clarify them. The priority assignment makes sure that things are done in the proper order. Just try it! You will either get more done or have time left over for things you never had time for before. In any case, it will be worth many times the cost of the software.

That may sound like a dream, but it's fact. If you don't believe it, just take Gander up on their free trial offer. What have you got to lose? Nothing and everything, depending upon how you look at it.

Let's just assume that you have taken Gander up on their offer, and the package has arrived. Installation is routine using INSTAL for hard disks and floppy disk is almost "plug-n-go". For those who suffer compuphobia, there is a 38 page manual that approaches the ideal model. It's clear and concise with all of the user instructions at the beginning and sufficient technical stuff at the back. It is liberally sprinkled with screen printouts and also loaded with practical hints not only for the operation of the system, but also for practical use of its output.

The primary menu is broken down into five major areas.

1. An Appointment Scheduler. This looks just like an appointment book on your screen or paper. The difference is that you set the times (18 slots in all) on the schedule to your convenience - not the publishers. On top of that, your files will keep 60 days worth on line at all times.

2. A "To Do List". This is your reminder; up to 60 items to which you assign a date and a priority number. The list is sorted by either date or priority.

3. A "Future Planning List". Again 60 items, sorted by date. This is primarily for long range planning beyond the "To Do" list. The nice part of this list is that, because both Julian and Georgian calendars are used, it always reports the number of days till the appointment, or for slow pokes, the number of days overdue.

4. A transfer program. Now we are getting to the good stuff! This ditty works its way through either the To Do or Future Lists and allows you to "post" it into a time slot for any day's Appointment Schedule. The screening makes it all so easy and obvious that it is hardly worth reading the manual - but do it anyway.

5. A Calendar Program. Julian calendar to the rescue! A calendar for any month or year from 1910 - 2399. A whole year on a page or one of those nice block planning calendars for one month. Just tell it Screen or Printer.

With the user in mind, there are two means of getting into the above areas of the system. The normal route is through the Master Scheduler which allows one into all phases of a users lists. But, if you want to add something to a group of users, then you may enter either the To Do or Future areas directly from the menu and jump from person to person. Redundancy, yes, but it is convenient.

Lastly, there is a Print Utilities menu selection. This allows direct printing of just about anything in the system, either one day or a range of days and very importantly, prints Work Sheets on which to make notes as things come up.

What makes it all so nice is the beautiful screening - everything is there - all the ESC sequences listed that let you back up a question, get out, ignore last entry, etc. If that feature is not available at that point, it won't appear on the screen. The Edit, Add, Delete like the ESCs are all entered in a command description line at the bottom and are predictable and put the cursor where expected. Full use is made of the required OS65-U v1.43+ operating system. Most non-ANSI terminals are supported and provisions are provided for setting up most others. The printer selected shows up on the screen every time you enter the system. If you change it, it's stored and will be there the next time you enter the system. Just for the record, it is DMS compatible and will keep schedules for five named people. Need more than five, just run a duplicate system.

All the Utilities are here too. The sub-menu for System Maintenance allows simple selection for back-up disk initialization and file creation, down loading and up loading. You can back up just your files or everyone's. Of course, there's an Initialize the System so that you can set

things back to square one when you have finished playing and are ready to get down to business.

That about covers the things that you might expect to find in this kind of package, but there's more - from the programmers point of view, much more. Try a few of these on for size and keep in mind the ease of use. You have lists of 60 items and only 15 to a screen. Hence, forward or backward screening to your choice of screen not to mention automatic switch when you run off the bottom. A (L)ocate function that presents the right page with the cursor on the item. (A)dd an item, tells it the date, and the day of the week appears right next to it. Change the date during an (E)dit and the day of the week changes as does the number of days to the event in the Future list. The (S)ort always keeps things pushed up to the front of the file, so there is no need of a delete and repack routine. It takes a few seconds to do its thing, so a flag is set to prohibit an already sorted file from resorting.

I could go on, but by now you should have the picture. This is indeed a well thought out system. They haven't missed a trick, but most importantly, it is simple to use. Read the manual and within 15 minutes you will be doing meaningful work.

Gripes! There was one question on the screen that I felt should have been prepacked by INP$ and I like my Exits to go back to the base system directory rather than BEXEC*. Programmers seem to prefer BEXEC* but clutz users see that as one more step to get to SYSDIR. Then too, not all machines are set up with sub systems.

When it gets right down to it, this is the kind of program that should be on every machine capable of running OS-U. There isn't one of us who could not profit from TTP. When I look out in the market place and see all of the "bundled" software that comes with machines these days, it seems that someone missed the boat by not having a TTP in the stable. But now OSI dealers can be one up.

I almost forgot! What about support? I don't think that you will need any! The whole idea is to make the package plug and go. I cannot say that I did not talk to Mr. Huntley about the preliminary versions. I did, and found help came fast and to the point. At this point, John's calls are going to be for orders, not help.

★

to $3279 and on 8" systems it's $3179. The last change is that you'll have to delete the line in "BCODE" that says "STX CRSCLR" since there is no cursor color in 3.2. The BEXEC* program that installs Hooks automatically handles the only other change needed for 3.2.

If you want to experiment with adding your own keywords to BASIC, the Hooks source code will adjust automatically. First add the keyword itself to the CMDTBL and then add the label of the code that handles your keyword to the tables JTBL and JTBH like the others are listed. Just make sure that you add your keyword and addresses AFTER the current ones. You may find that you'll have to change the origin address of the source code if you try this, but there is still a small amount of memory in the space currently allotted for Hooks. The main thing is to enjoy!

LISTING 1

by:Rick Trethewey

```
; HOOKS INTO BASIC UNDER OS-65D V3.3
; REV 1.8        9/23/83
;
*=$B600
;
;     BASIC EXTERNALS
;
POKER  =$19    Z-PAGE SCRATCH LOCATION
VARTAB =$7A    START OF VARIABLE TABLE
ARRTAB =$7C    START OF ARRAY TABLE
ENDTAB =$7E    END OF ARRAY & VARIABLES
STRSPA =$80    START OF STRING ARRAYS
BLINE  =$86    CURRENT LINE # BEING EXECUTED
GOTOTK =$88    "GOTO" TOKEN
REMTK  =$8E    "REM" TOKEN
VARNAM =$92    CURRENT VARIABLE NAME
VARPNT =$94    POINTER TO VARIABLE
FORPNT =$96    MULTI-USE POINTER
TOTK   =$9D    "TO" TOKEN
THENTK =$A0    "THEN" TOKEN
MULTK  =$A5    "*" TOKEN
EQLSTK =$AB    "=" TOKEN
VARPTR =$AC    POINTER TO VARIABLE
FACEXP =$AE    F.P. ACCUMULATOR EXPONENT
FACHI  =$AF    F.P. ACCUM. MSB
FACMHI =$B0    F.P. ACCUM. NMSB
FACMLO =$B1    F.P. ACCUM. NLSB
FACLO  =$B2    F.P. ACCUM. LSB
FACSGN =$B3    F.P. ACCUM. SIGN
CHRGET =$C0    FETCH NEXT CHAR. IN PROGRAM
CHRGOT =$C6    RE-FETCH CHAR. AT CURRENT PTR.
TXTPTR =$C7    POINTER TO PROGRAM TEXT
OMERR  =$044C  OUT OF MEMORY ERROR
TYPERR =$0462  DISPLAY ERROR MESSAGE
GOTO   =$08A6  CODE FOR "GOTO"
ADDON  =$08FC  UPDATE TXTPTR TO TXTPTR+Y
REM    =$093C  CODE FOR "REM"
LINGET =$096C  CONVERT ASCII LN# TO BINARY
CRDO   =$0A73  EXECUTE <CR><LF>
BASPRT =$0ACC  OUTPUT STRING
OUTDO  =$0AEE  OUTPUT A CHARACTER
CHKTYP =$0CBC  CHECK FOR NUMERIC VARIABLE
CHKSTR =$0CBE  CHECK FOR STRING VARIABLE
FRMEVL =$0CCD  EVALUATE EXPRESSION
CHKCOM =$0E13  CHECK FOR COMMA
CHKCHR =$0E15  CHECK FOR CHAR. IN ACC.
SNERR  =$0E1E  SYNTAX ERROR CODE
PTRGET =$0F2E  LOOK UP VARIABLE
FCERR  =$10D0  FUNCTION CALL ERROR
GIVAYF =$1218  GIVE A-Y PAIR TO F.P. ACCUM.
FREFAC =$1520  GET POINTER TO STRING
GIEVYC =$1615  DO "CHRGET" & EVAL. EXP. < 256
GETBYT =$1618  EVAL. EXPRESSION < 256
GETVAR =$1A9D  PUT VARIABLE IN FACC.
FLOAT  =$1B44  CONVERT INTERGER TO F.P. TYPE
QUINT  =$1B96  CONVERT F.P. TO INTERGER
ASCFP  =$1BEE  CONVERT ASCII TO F.P.
ASCII  =$1CEC  CONVERT FACC. TO ASCII
PNUMBR =$1CDC  PRINT NUMBER
COPYER =$211C  COPY FILE PARAMS. TO DISK HEADER
NONUMR =$213A  POINT OS-65D TO STRING
;
;     OS-65D EXTERNALS
;
TMP    =$E0    TEMPORARY STORAGE
FNIL   =$E1    Z-PAGE POINTER LSB
FNIH   =$E2    Z-PAGE POINTER MSB
MAXVAL =$E5    HIGH TRACK # FOR PUT/LOAD
ADRL   =$FE    DISK R/W ADDRESS LSB
ADRH   =$FF    DISK R/W ADDRESS MSB
MAXMEM =$2300  HIGHEST AVAILABLE RAM PAGE
SECT   =$265E  DISK R/W SECTOR
PAGES  =$265F  # PAGES READ OR TO WRITE
ADRLX  =$2660  DISK R/W ADDRESS LSB  (NON-
ADRHX  =$2661  DISK R/W ADDRESS MSB   VOLITILE)
TRAKX  =$2662  DISK R/W BINARY TRACK #
HOME0  =$2663  HOME DRIVE TO TRACK 0
SEEKX  =$26A6  MOVE DRIVE TO "TRAKX"
SEEK   =$26BC  MOVE DRIVE TO BCD TRK. # IN ACC.
LOAD   =$2754  LOAD DRIVE HEAD
UNLOAD =$2761  UNLOAD DRIVE HEAD
SAVEX  =$27D7  DISK WRITE FROM ADRLX/ADRHX
FIND   =$28C4  FIND SECTOR ON TRACK
CALLX  =$295D  DISK READ TO ADRLX/ADRHX
DUMRED =$2998  READ DISK - THROW AWAY CONTENTS
SELECT =$29C6  SELECT DISK DRIVE
ERROR  =$2A4B  65D ERROR REPORT ROUTINE
ASMR   =$2ADE  INVOKE ASSEMBLER/EDITOR
EM     =$2B2F  INVOKE EXTENDED MONITOR
LOADER =$2BA7  LOAD SOURCE FILE
FILSAV =$2BE0  SAVE SOURCE FILE
SRCSIZ =$2BE9  SOURCE FILE SIZE IN PAGES
CRLF   =$2D6A  EXECUTE <CR><LF>
SAVEM  =$2C28  SECTOR WRITE ROUTINE
TIN0   =$2CBC  FETCH COMMAND BYTE
SWAP   =$2CF7  PAGE 0/1 SWAP ROUTINE
PRBYTE =$2D92  PRINT ACC. CONTENTS (NUMERIC)
FNDNUM =$2DA6  PUT/LOAD TRACK FINDER
FNDNAM =$2DBE  FILE NAME LOOK-UP
DIRTRK =$2DC4  BCD DIRECTORY TRACK NUMBER
OUTCH  =$2343  PRINT ACC. CONTENTS
STROUT =$2D73  PRINT STRING FOLLOWING "JSR"
TXTBUF =$2E1E  65D COMMAND TEXT BUFFER
DIRBUF =$2E79  DIRECTORY BUFFER
CRSCLR =$32E3  3.3 CURSOR BACKROUND COLOR
CASECK =$3A5F  CONVERT LOWER TO UPPER CASE
SRCSTR =$3A79  SOURCE FILE START ADDRESS
;
       LDY #$01        INIZ POINTER
       LDA (TXTPTR),Y  LOOK 1 CHARACTER AHEAD
       CMP #MULTK      IS IT AN ASTERISK ?
       BNE CHK         NO ==> GO CHECK KEYWORDS TOO
       DEY             YES! BACK UP ONE
       LDA (TXTPTR),Y  REFETCH 1ST CHARACTER
       JSR CASECK      MASK OFF LOWER CASE
       CMP #'C         CLEAR SCREEN ?
       BEQ CCODE
       CMP #'B         SET BACKROUND COLOR ?
       BEQ BCODE
       CMP #'Q         SET SCREEN STATE ?
       BEQ QCODE
       CMP #'R         RESTORE NEW, LIST, & ^C ?
       BEQ RCODE
       CMP #'D         PRINT DIRECTORY ?
       BEQ DCODE
       CMP #'S         SELECT DISK DRIVE ?
       BEQ SCODE
       CMP #'T         TOGGLE TRACE ?
       BEQ TCODE
```

Continued on page 16

D&N MICRO PRODUCTS, INC.

3702 N. Wells St.
Fort Wayne, Ind. 46808
(219) 484-6414

TERMS $3.00 shipping. Foreign orders add 15%. Indiana residents add 5% sales tax.

COMPUTER

MICRO-80 COMPUTER

Z-80A CPU with 4Mhz clock and CP/M 2.2 operating system. 64K low power static memory. Centronics parallel printer port. 3 serial ports. 4" cooling fan. Two 8" single or double sided floppy disk drives. IBM single density 3740 format for 243K or storage, double density format for 604K of storage. Double sided drives allow 1.2 meg on each drive. Satin finish extruded aluminum with vinyl woodgrain decorative finish. 8 slot backplane, 48 pin buss compatible with OSI boards.

MODEL 80-1200 **$2995**
2 8" Single sided drives

MODEL 80-2400 **$3495**
2 8" Double sided drives

MICRO-65 COMPUTER

6502 CPU with 2Mhz clock and DOS-65 operating system. 48K of low power static memory. 2 serial ports and 1 Centronics parallel port. 2 8" single or double sided drives. Satin finish extruded aluminum with vinyl woodgrain finish. 8 slot backplane, 48 pin buss compatible with OSI. Will run OSI 65D and 65U software. Includes Basic E/65 a compiled BASIC for 6502 CPU.

MODEL 65-1 **$2995**
2 8" Single sided drives

MODEL 65-2 **$3495**
2 8" Double sided drives

BP-580 8 Slot Backplane ..... **$ 47**
OSI 48 pin Buss compatible

MEM-CM9 MEMORY/ FLOPPY CONTROLLER

24K memory/floppy controller card uses 2114 memory chips, 1 8K and 1 16K partition. Supports OSI type disk interface

24MEM-CM9 ..... **$325**
16MEM-CM9 ..... **$260**
8MEM-CM9 ..... **$180**
BARE MEM-CM9 ..... **$ 50**
Controller on assembled unit add ..... **$ 90**

BIO-1600 Bare IO card ..... **$ 50**
Supports 8K of memory, 2 16 bit parallel ports, 5 serial ports, with manual and Molex connectors.

PRINTERS

Okidata

ML82A, 120 cps, 10" **$409**
ML83A, 120 cps, 15" **$895**
ML84 Parallel, 200 caps, 15" **$1150**

C. Ioth

8510AP Prowriter, parallel **$419**
120 cps, correspondence quality
8510APD Prowriter, serial **$585**
F10-40PU Starwriter, parallel **$1319**
Letter quality daisy wheel
F10-40RU Starwriter, serial **$1319**
F10-55PU Printmaster **$1610**
parallel, Letter quality daisy wheel
F10-55RU Printmaster, serial **$1610**

DISK DRIVES AND CABLES

8" Shugart SA801 ..... **$385**
single sided
8" Shugart SA851 **$585**
double sided
FLC-6 6 ft cable from D&N **$69**
or OSI disk controller to 8" drive
5¼" MPI B51 disk drive with **$450**
cable, power supply and cabinet. Specify computer type.
FLC-5¼ cable for connection **$75**
to 5¼ drive and D&N or OSI controller, with data separator and disk switch. Specify computer type

HARDWARE

OSI COMPATIBLE

IO-CA10X Serial Printer Port **$125**
Specify Device #3 or #8
IO-CA9 Parallel Printer Port **$150**

CMOS-MEM

64K CMOS static memory board, uses 6116 chips, 3 16K, 1 8K and 2 4K blocks, Partitionable for multi-user, OSI type disk controller, 2 IO mapped serial ports for use with D&N-80 CPU. Ideal way to upgrade from cassette to disk.

64K CMOS-MEM **$490**
48K CMOS-MEM **$390**
24K CMOS-MEM **$250**
16K CMOS-MEM **$200**
BARE CMOS-MEM ..... **$ 50**
Controller add. **$ 90**
2 IO mapped serial ports add. **$125**
on assembled memory board
Z80-IO 2 IO mapped serial **$160**
ports for use with D&N-80 CPU card
FL470 Disk Controller **$155**
Specify 5¼ or 8" drive

STANDARD CP/M FOR OSI

D&N-80 CPU CARD

The D&N-80 CPU allows the owner of an OSI static memory computer to convert to Industrial Standard IBM 3740 single density disk format and CP/M operating system. Double density disk operation is also supported for 608K of storage on an 8" diskette. When used with a 5¼" disk system 200K of storage is provided. Includes parallel printer and real time clock. Also available for polled keyboard and video systems. Compatible with C2, C3, C4 and 200 series OSI computers.

D&N-80-P **$349**

CP/M 2.2 ..... **$150**

64K CMOS-MEM with D&N-80 CPU card **$450**

HARD DISK DRIVER **$140**
Allows D&N-80 CPU board to control OSI 40 or 80 meg hard disk unit. Will not destroy OSI files. Will also allow for a true 56K CP/M system. Specify 40 or 80 meg drive.

BUSS TRANSFER **$135**
Allows for D&N-80 and OSI CPU to be in the computer at the same time. Toggle switch provides for alternate CPU operation.

DISK TRANSFER **$100**
Utility program to transfer OSI CP/M format disk to IBM 3740 single density format. Will also transfer IBM to OSI format.

SYSTEM HARDWARE REQUIREMENTS

D&N-80 CPU, D&N FL470 or OSI 470 controller, 48K memory at 0000-BFFF, 4K memory at D000-DFFF, two disk drive cables.

FORMAT TRANSFER **$15**
You supply software on 8" diskette D&N will transfer OSI CP/M format to IBM 3740 CP/M format. Can also transfer IBM 3740 CP/M format to OSI CP/M format. Original diskette returned.

```
1300 BACK  JSR PTRGET     IT'S A VARIABLE! EXECUTE
1310       STA FORPNT     REPLACED CODE IN BASIC
1320       STY FORPNT+1   GIVE VARIABLE'S ADDR.
1330       RTS            TO BASIC AND GO BACK
1340 ;
1350 CHK   JMP CHKR       SEE IF IT'S A NEW KEYWORD
1360 ;
1370 SCODE JMP SCOD0      JUMP TO SELECT CODE
1380 ;
1390 TCODE JMP TCOD0      JUMP TO TRACE TOGGLE
1400;
1410 ; CLEAR 540 BLACK & WHITE VIDEO
1420;
1430 CCODE LDA #'         LOAD A <SP>
1440       LDX #$08       INIZ PAGE COUNTER
1450       LDY #$00       INIZ PAGE INDEX
1460 C1    STA $D000,Y    CLEAR A CELL
1470       INY            BUMP PAGE INDEX
1480       BNE C1         LOOP TO PAGE END
1490       INC C1+2       BUMP ADDRESS MSB
1500       DEX            DECREMENT PAGE COUNTER
1510       BNE C1         LOOP 'TIL DONE
1520       LDA #$D0       GET ORIGINAL MSB
1530       STA C1+2       RESTORE POINTER
1540UPDATE JSR CHRGET     THROW AWAY ASTERISK
1550       JSR CHRGET     AND THE NEXT CHARACTER TOO
1560 UP1   PLA            CANCEL JSR TO HOOKS
1570       PLA
1580       RTS            AND GO BACK TO BASIC
1590;
1600; SET COLOR BACKROUND & CURSOR COLOR
1610;
1620 BCODE JSR CHRGET     THROW AWAY ASTERISK
1630       JSR GTBYTC     EVALUATE FOLLOWING EXPRESSION
1640       STX CRSCLR     MAKE VALUE NEW CURSOR COLOR
1650       TXA            PUT IN ACC.
1660       LDY #$00       INIZ
1670       LDX #$08       INIZ MEMORY PAGE COUNTER
1680 BC01  STA $E000,Y    CHANGE COLOR
1690       INY            BUMP POINTER
1700       BNE BC01       LOOP TO PAGE END
1710       INC BC01+2     BUMP PAGE ADDRESS ABOVE
1720       DEX            DECREMENT PAGE COUNT
1730       BNE BC01       LOOP 'TIL DONE
1740       LDA #$E0       FETCH ORIGINAL PAGE ADDRESS
1750       STA BC01+2     RESTORE IT ABOVE
1760       BNE UP1        AND EXIT TO BASIC
1770;
1780; SET 540 DISPLAY STATE
1790;
1800 QCODE JSR CHRGET     THROW AWAY ASTERISK
1810       JSR GTBYTC     EVALUATE FOLLOWING EXPRESSION
1820       STX $DE00      SET SCREEN STATE WITH RESULT
1830       JMP UP1        AND GO BACK TO BASIC
1840;
1850; RESTORE "NEW", "LIST", AND <CTRL>'C'
1860;
1870 RCODE LDA #76        THESE NUMBERS SHOULD
1880       STA 741        *VERY* FAMILIAR
1890       LDA #78
1900       STA 750
1910       LDA #173
1920       STA 2073
1930       BNE UPDATE
1940 ;
1950 ; D* CAN BE MADE AVAILABLE FROM OS-65D BY
1960 ; CHANGING 65D AS FOLLOWS:
1970 ; $2E3D = $2A  $2E3E = $B7  $2E3F = $B6
1980 ;
1990 DCODE JSR SWAP       * DOS CONTEXT *
2000       JSR D          PRINT DIRECTORY
2010       JSR SWAP       * LANGUAGE CONTEXT *
2020       JMP UPDATE     AND GO BACK TO BASIC
2030;
2040 DIRIN LDA #DIRBUF    GET DIRBUF LSB
2050       STA ADRLX      GIVE IT TO 65D
2060       LDA #DIRBUF/256
2070       STA ADRHX      HANDLE MSB TOO
2080       LDA DIRTRK     GET BCD DIRECTORY TRACK #
2090       JSR SEEK       MOVE HEAD TO TRACK
2100       JMP READ+3     READ DIRECTORY SECTOR
2110;
2120 D     JSR STROUT     DISPLAY MESSAGE
2130       .BYTE '* Directory *',$D,$A,$A,$0
2140       LDY #$00       INIZ
2150       STY FIFTH      INIZ ENTRY COUNTER
2160       INY            +1 (=1)
2170       STY SECT       SET SECTOR #
2180       JSR DIRIN      READ DIRECTORY SECTOR
2190       JSR D1         DISPLAY CONTENTS
2200       INC SECT       BUMP SECTOR #
2210       JSR DIRIN      READ AND FALL THROUGH
2220 D1    LDY #$00       INIZ DIRBUF INDEX
2230       LDX #$00       INIZ ENTRY INDEX
2240 D2    LDA DIRBUF,Y   FETCH CHARACTER FROM DIRBUF
2250       CMP #'#        EMPTY ENTRY?
2260       BNE D3         NO! DISPLAY NAME! ==> D3
2270       CPX #$00       MAYBE, "#" IS 1ST CHARACTER?
2280       BEQ D4         YES! SKIP ENTRY! ==> D4
2290 D3    JSR OUTCH      PRINT NAME CHARACTER
2300       INY            BUMP DIRBUF INDEX
2310       INX            BUMP ENTRY INDEX
2320       CPX #$06       PRINT ENTIRE NAME YET?
2330       BNE D2         NO! LOOP! ==> D2
2340       JSR TKOUT      YES! DISPLAY TRACK RANGE!
2350 D4    TYA            PUT DIRBUF INDEX IN ACC.
2360       AND #$F8       MASK TO 8'S
2370       CLC
2380       ADC #$08       ADD ENTRY LENGTH
2390       TAY            PUT RESULT BACK IN Y
2400       BNE D2-2       LOOP TO BUFFER END!
2410       LDA SECT       DONE! FETCH CURRENT SECTOR #
2420       CMP #$02       DONE BOTH?
2430       BNE TKOU2-1    NO! QUIT!
2440       JMP CRLF       YES! DO CLEAN UP & QUIT
2450;
2460 TKOUT TYA            PUT DIRBUF INDEX IN ACC.
2470       PHA            SAVE ON STACK
2480       LDA #'         GET A <SP>
2490       JSR OUTCH      PRINT FOR SEPARATION
2500       LDA DIRBUF,Y   FETCH START TRACK #
2510       JSR PRBYTE     PRINT IT
2520       LDA #'-        GET "-"
2530       JSR OUTCH      PRINT IT
2540       PLA            RETRIEVE DIRBUF INDEX
2550       PHA            PUT IT BACK FOR LATER
2560       TAY            PUT INDEX IN Y AGAIN
2570       LDA DIRBUF+1,Y FETCH ENDING TRACK #
2580       JSR PRBYTE     PRINT IT
2590       LDA FIFTH      FETCH # ON THIS LINE
2600       CMP #$03       DONE 4 YET?
2610       BEQ TKOU2      YES! ==> TKOU2
2620       JSR STROUT     NO! PRINT 2 SPACES
2630       .BYTE '  ',$00
2640       INC FIFTH      BUMP # ON THIS LINE
2650 TKOU1 PLA            RETRIEVE DIRBUF INDEX
2660       TAY            PUT IT BACK IN Y
2670       RTS            AND GO BACK
2680 TKOU2 JSR CRLF       DO CLEAN-UP <CR><LF>
2690       LDA #$00       INIZ
2700       STA FIFTH      CLEAR LINE COUNTER
2710       BEQ TKOU1      AND LOOP!
2720 ;
2730 SCOD0 JSR CHRGET     THROW AWAY ASTERISK
2740       JSR CHRGET     FETCH NEXT CHARACTER
2750       JSR FRMEVL     EVALUATE EXPRESSION
2760       JSR CHKSTR     MAKE SURE IT'S A STRING
2770       JSR FREFAC     FIND STRING
2780       STX SCOD1+1
2790       STY SCOD1+2
2800       CMP #$01       CHECK LENGTH
2810       BNE SCOD3      BAD! ERROR!
2820 SCOD1 LDA $FFFF      MODIFIED CODE!
2830       JSR CASECK     MAKE IT ALL-CAPS
2840       CMP #'A        CHECK FOR LEGAL DRIVE #
2850       BCC SCOD3
2860       CMP #'D+1
2870       BCS SCOD3
2880       STA TEMP       SAVE IT
2890       JSR SWAP       * DOS CONTEXT *
2900       LDA TEMP       FETCH DRIVE REQUEST
```

Listing continued

```
2910        AND #$0F        MASK TO LOW NYBBLE
2920        JSR SELECT      SELECT DRIVE
2930        BCS SCOD2       DRIVE NOT READY? ==> SCOD2
2940        JSR HOME0       HOME DRIVE
2950        JSR SWAP        * LANGUAGE CONTEXT *
2960        JMP UP1         AND QUIT
2970 SCOD2  LDA #$06        GET 'DRIVE NOT READY'
2980        JMP ERROR       USE 65D'S ERROR ROUTINE & QUIT
2990 SCOD3  JMP FCERR       ERROR!
3000 ;
3010 ; IN VERSION 1.8, THE KEYWORDS BELOW ARE
3020 ; NOW RESERVED AND *CANNOT* BE USED AS VARIABLES
3030 ;
3040CMDTBL .BYTE 'ASM',0
3050        .BYTE 'EM',0
3060        .BYTE 'SAVE',0
3070        .BYTE 'LOAD',0
3080        .BYTE 'PACK',0
3090        .BYTE 'VIEW',0
3100        .BYTE 'CALL',0
3110        .BYTE 'KILL',0
3120        .BYTE 'MAKE',0
3130        .BYTE 'RENAME',0
3140        .BYTE 'WAIT',0
3150        .BYTE 'FILE',0
3160        .BYTE $00
3170 ;
3180 CHKR   LDX #$00        INIZ CMD. TABLE INDEX
3190        STX COUNT       INIZ CMD. COUNTER
3200        LDY #$00        INIZ TEXT INDEX
3210 CHK1   LDA (TXTPTR),Y  FETCH TEXT CHARACTER
3220        JSR CASECK      MAKE IT ALL CAPS
3230        CMP CMDTBL,X    COMPARE TO KEYWORD
3240        BNE NXTCMD      NO MATCH? ==> NXTCMD
3250        INX             YES! BUMP COMMAND INDEX
3260        INY             BUMP TEXT INDEX
3270        LDA CMDTBL,X    CHECK FOR KEYWORD END
3280        BEQ XCMD        YES! EXECUTE COMMAND! =>
3290        BNE CHK1        NO! LOOP!
3300NXTCMD  INC COUNT       BUMP COMMAND COUNTER
3310        LDA COUNT       FETCH IT
3320        CMP #JTBH-JTBL+2
3330        BNE NXT1        NO! LOOP! ==> NXT1
3340        JMP BACK        YES! RETURN TO BASIC
3350 NXT1   LDA CMDTBL,X    CHECK KEYWORD CHARACTER
3360        BEQ NXT2        AT END OF WORD? ==> NXT2
3370        INX             NO! BUMP COMMAND INDEX
3380        BNE NXT1        AND LOOP!
3390 NXT2   INX             MOVE 1 PAST TERMINATOR
3400        LDY #$00        RESTORE TEXT INDEX
3410        BEQ CHK1        AND LOOP!
3420;
3430 XCMD   LDA COUNT       GET COMMAND #
3440        CMP #$02        ASM OR EM ?
3450        BCS RUNNER      NO! ==> RUNNER
3460        JSR SWAP        YES! * DOS CONTEXT *
3470        LDA #$11        GET TXTBUF LENGTH
3480        STA TIN0+1      RESET FOR TINCH
3490        LDA COUNT       CHECK COMMAND
3500        BEQ GOASM       ASM? ==> GOASM
3510        JMP EM          EM! DO IT!
3520;
3530 GOASM  JMP ASMR        ASM! DO IT!
3540;
3550 RUNNER TAX             PUT COMMAND # IN X
3560        DEX             -1
3570        DEX             -2
3580        LDA JTBL,X      FETCH ADDRESS LSB
3590        STA RUN1+1      SAVE IT
3600        LDA JTBH,X      FETCH ADDRESS MSB
3610        STA RUN1+2      SAVE IT TOO
3620 RUN1   JMP $FFFF       DO COMMAND!
3630 ;
3640 JTBL   .BYTE SAVIT,LODIT,PACKIT,VIEWIT
3650        .BYTE CALR,KILL,MAKER,RENAME,WAIT
3660        .BYTE FILGET
3670;
3680 JTBH   .BYTE SAVIT/256,LODIT/256,PACKIT/256
3690        .BYTE VIEWIT/256,CALR/256,KILL/256
3700        .BYTE MAKER/256,RENAME/256,WAIT/256
3710        .BYTE FILGET/256
3720 ;
```

Continued

```
3730 LODIT JSR ADD4        BUMP TXTPTR PAST "LOAD"
3740       JSR NONUMR      GIVE FILE NAME/# TO 65D
3750       JSR LOADER      LOAD IT INTO WORKSPACE
3760       JSR SWAP        RESTORE LANGUAGE CONTEXT
3770       JSR $2271       COPY OUT FILE SPECS TO BASIC
3780       JMP OUT+3       GO BACK TO BASIC
3790 ;
3800 SORT  LDA #$01        INIZ
3810       STA SECT        START WITH SECTOR 1
3820       JSR DIRIN       READ DIRECTORY SECTOR
3830       JSR SORT0       MARK USED TRACKS
3840       INC SECT        BUMP SECTOR #
3850       JSR DIRIN       READ SECTOR AND FALL THROUGH
3860 SORT0 LDY #$00        INIZ DIRBUF INDEX
3870 SORT1 LDA DIRBUF,Y    FETCH ENTRY 1ST CHARACTER
3880       CMP #'#         EMPTY ENTRY?
3890       BEQ SORT2       YES! ==> SORT2
3900       TYA             NO, PUT INDEX IN ACC.
3910       CLC
3920       ADC #$06        POINT TO STARTING TRACK #
3930       TAY             PUT INDEX BACK IN Y
3940       LDA DIRBUF,Y    FETCH STARTING TRACK #
3950       JSR BCDH        MAKE IT HEX
3960       STA STTK        SAVE AS START TRACK
3970       LDA DIRBUF+1,Y  FETCH ENDING TRACK #
3980       JSR BCDH        MAKE IT HEX
3990       STA ENDTK       SAVE IT TOO
4000       JSR RESR        MARK TRACK RANGE "IN USE"
4010 SORT2 TYA             PUT DIRBUF INDEX IN ACC.
4020       AND #$F8        MASK TO 8'S
4030       CLC
4040       ADC #$08        ADD ENTRY LENGTH
4050       TAY             PUT RESULT BACK IN Y
4060       BNE SORT1       LOOP TO PAGE END
4070 SORT3 RTS             AND QUIT
4080;
4090 RESR  LDX STTK        USE STTK AS ORIGIN
4100 RESR1 INC LIST,X      MARK "LIST" ENTRY
4110       CPX ENDTK       MARKED ALL TRACKS?
4120       BEQ SORT3       YES! QUIT!==> SORT3
4130       INX             NO, BUMP INDEX
4140       BNE RESR1       AND LOOP
4150 ;
4160PACKIT JSR ADD4        MOVE PAST "PACK"
4170       LDA SRCSIZ      GET SOURCE FILE SIZE
4180       CLC
4190       ADC #$02        +2 FOR 8" MAXIMUM
4200       STA STRFLG      SAVE RESULT
4210       LDA STRSPA+1    CHECK AVAILABLE RAM
4220       SEC
4230       SBC STRFLG
4240       CMP ENDTAB+1
4250       BCS PACK1
4260       JMP OMERR       NOT ENOUGH MEMORY!
4270 PACK1 STA STRFLG      SAVE BUFFER ADDR. MSB
4280       JSR SWAP        * DOS CONTEXT *
4290       JSR PAKR        PACK DISKETTE
4300       JMP OUT         RETURN TO BASIC
4310;
4320 PAKR  JSR CLRLST      CLEAR USED TRACK LIST
4330       JSR SORT        MARK USED TRACKS IN LIST
4340 GAP   LDA FNDNUM+1    GET FLOPPY MAX. TRK. #
4350       JSR BCDH        MAKE IT HEX
4360       TAY             PUT IN Y
4370       INY             + 1 !
4380       STY MAXVAL      SAVE AS MAX.
4390       LDY #$00        INIZ
4400 GAP3  LDA LIST,Y      CHECK LIST
4410       BEQ GAP4        CLEAR TRACK ? ==>
4420       INY             NO! BUMP POINTER TO LIST
4430       CPY MAXVAL      AT END OF DISK ?
4440       BNE GAP3        NO! LOOP! ==> GAP3
4450       RTS             YES! QUIT! (DISK FULL)
4460 GAP4  STY STGAP       SAVE 1ST CLEAR TK. #
4470 GAP5  LDA LIST,Y      FIND LENGTH OF "EMPTINESS"
4480       BNE GAP6        USED? => GAP6
4490       INY             NO! BUMP POINTER!
4500       CPY MAXVAL      AT END OF DISK?
4510       BNE GAP5        NO! LOOP! ==> GAP5
4520       RTS             YES! QUIT (DISK CLEAR TO END)
```

Listing continued

```
4530 GAP6  STY STBLK     SAVE END OF GAP + 1
4540       STY POINT     SAVE AGAIN - PROTECTED
4550       TYA           PUT TK. # IN ACC.
4560       SEC
4570       SBC STGAP     SUBTRACT 1ST FREE TK. #
4580       STA GAPLEN    SAVE LENGTH OF GAP
4590 GAP7  INY           BUMP POINTER
4600       LDA LIST,Y    CHECK LIST
4610       BEQ GAP8      EMPTY? ==> GAP8
4620       CPY MAXVAL    NO, END OF DISK?
4630       BNE GAP7      NO! LOOP! ==> GAP7
4640 GAP8  DEY           BACK UP ONE
4650       STY ENBLK     SAVE END OF BLOCK TK. #
4660 MOVE  LDA STBLK     GET TRACK # TO MOVE
4670       STA TRAKX     GIVE IT TO 65D
4680       TAY           PUT IN Y AS INDEX
4690       LDA #$00      INIZ
4700       STA LIST,Y    SHOW TRACK CLEAR NOW
4710       JSR SEEKX     MOVE HEAD TO TRACK
4720       JSR CNTS      COUNT SECTORS ON TRACK
4730       LDA FIFTH     GET SECTOR COUNT
4740       BEQ MOV3      IF 0 ==> MOV3
4750       LDA #$01      INIZ
4760       STA SECT      START AT SECTOR 1
4770 MOV2  JSR READ      READ SECTOR
4780       LDA STGAP     GET OPEN TRACK #
4790       STA TRAKX     GIVE IT TO 65D
4800       JSR SEEKX     MOVE HEAD TO TRACK
4810       JSR WRITE     WRITE SECTOR
4820       LDA SECT      CHECK CURRENT SECTOR #
4830       CMP FIFTH     END OF TRACK ?
4840       BEQ MOV3      YES! ==> MOV3
4850       INC SECT      NO, BUMP SECTOR #
4860       LDA STBLK     GET ORIGIN TRACK #
4870       STA TRAKX     GIVE TO 65D
4880       JSR SEEKX     MOVE HEAD TO TRACK
4890       JMP MOV2      AND LOOP! ==> MOV2
4900 MOV3  LDY STGAP     GET DEST. TK. #
4910       LDA #$01      INIZ
4920       STA LIST,Y    SHOW TRACK IN USE NOW
4930       LDA STBLK     GET ORIGIN TK. # AGAIN
4940       CMP ENBLK     AT END OF BLOCK?
4950       BEQ MOV4      YES! ==> MOV4
4960       INC STGAP     NO! BUMP DEST. TK. #
4970       INC STBLK     BUMP ORIGIN TK. # TOO
4980       BNE MOVE      AND LOOP! ==> MOVE
4990 MOV4  LDA #$01      INIZ
5000       STA SECT      SET SECTOR 1
5010       LDA GAPLEN    GET BLOCK LENGTH
5020       JSR BBCD      MAKE IT BCD!
5030       STA GAPLEN    AND PUT IT BACK
5040 MOV9  JSR DIRIN     READ IN DIRECTORY SECTOR
5050       LDY #$00      INIZ
5060 MOV5  LDA DIRBUF,Y  FETCH DIR. ENTRY
5070       CMP #'$       EMPTY ENTRY?
5080       BEQ MOV6      YES! ==> MOV6
5090       TYA           NO! PUT INDEX IN ACC.
5100       CLC
5110       ADC #$06      POINT TO 1ST TRACK #
5120       TAY           PUT BACK IN Y
5130       LDA DIRBUF,Y  FETCH 1ST TK. OF FILE
5140       JSR BCDH      MAKE IT HEX
5150       CMP POINT     COMPARE TO START OF BLOCK
5160       BEQ MOV11     SAME! ==> MOV11
5170       BCC MOV6      LESS THAN! ==> MOV6
5180       CLC
5190       CMP ENBLK     COMPARE TO END OF BLOCK
5200       BEQ MOV11     SAME! ==> MOV11
5210       BCS MOV6      GREATER THAN! ==> MOV6
5220 MOV11 SED           SET BCD MATH MODE
5230       LDA DIRBUF,Y  REFETCH TRACK #
5240       SEC
5250       SBC GAPLEN    SUBTRACT GAP LENGTH
5260       STA DIRBUF,Y  PUT RESULT BACK IN ENTRY
5270       LDA DIRBUF+1,Y FETCH LAST TRACK #
5280       SEC
5290       SBC GAPLEN    SUBTRACT GAP LENGTH
5300       STA DIRBUF+1,Y PUT BACK
5310       CLD           RESET TO NORMAL MATH
5320 MOV6  TYA           MOVE TO NEXT ENTRY
5330       AND #$F8
5340       CLC
5350       ADC #$08
5360       TAY           AT END OF DIRBUF?
5370       BNE MOV5      NO! LOOP! ==> MOV5
5380 MOV7  JSR WRITE+3   YES! WRITE REVISED DIR.
5390       LDA SECT      CHECK DIRECTORY SECTOR #
5400       CMP #$02      DONE BOTH ?
5410       BEQ MOV8      YES! ==> MOV8
5420       INC SECT      NO! BUMP SECTOR #
5430       BNE MOV9      AND LOOP! ==> MOV9
5440 MOV8  JMP GAP       LOOP TO ALLOCATION CHECKER
5450 ;
```

REMAINDER OF LISTING NEXT MONTH

LETTERS

ED:

This program is for the OSI C1P or by changing addresses 0239 & 023A, it can run on other OSI machines. I have been using this program myself for quite a while and find it very useful.

I know there have been a lot of search programs listed in many magazines, but most were in Basic, so here is mine. This one can be put in through the machine monitor and left in the unused portion of page 2 until it is needed.

The program can be called up by 0222 G then the variable or strings that you wish to find are typed in and a carriage return will produce the line numbers in the Basic program that contains what was typed in.

When you have typed, and after the carriage return, the program looks for the question mark (3F) that is printed out when anything is input without a line number. When the 3F is found, whatever is in the line buffer is stored and then compared with the Basic program until a match is found. At that time a line number is printed and the comparing goes on until the entire basic program is covered.

```
0222 A937     LDA #$37
0224 8504     STA $04
0226 A902     LDA #$02
0228 8505     STA $05
022A A901     LDA #$01
022C 8DB802   STA $02B8
022F A903     LDA #$03
0231 8DB902   STA $02B9
0234 4C74A2   JMP $A274
0237 48       PHA
0238 AD65D3   LDA $D365
023B C93F     CMP #$3F
023D F004     BEQ $0243
023F 68       PLA
0240 4CC3A8   JMP $A8C3
0243 A900     LDA #$00
0245 8D0302   STA $0203
0248 A200     LDX #$00
024A B513     LDA $13,X
024C 9DE702   STA $02E7,X
024F C900     CMP #$00
0251 F005     BEQ $0258
0253 E8       INX
0254 E020     CPX #$20
0256 D0F2     BNE $024A
0258 8EE602   STX $02E6
025B 206CA8   JSR $A86C
025E A200     LDX #$00
0260 A000     LDY #$00
0262 B9E702   LDA $02E7,Y
0265 8DF402   STA $02E4
0268 209E02   JSR $029E
026B 20AB02   JSR $02AB
026E CDE402   CMP $02E4
0271 D0ED     BNE $0260
0273 C8       INY
0274 CCE602   CPY $02E6
0277 D0E9     BNE $0262
0279 E004     CPX #$04
027B 30E5     BMI $0262
027D A5F2     LDA $F2
027F 8587     STA $87
0281 A5F3     LDA $F3
0283 8588     STA $88
0285 8A       TXA
0286 48       PHA
0287 98       TYA
0288 48       PHA
0289 205AB9   JSR $B95A
028C 68       PLA
028D A8       TAY
028E 68       PLA
028F AA       TAX
0290 4C6002   JMP $0260
0293 A9C3     LDA #$C3
0295 8504     STA $04
0297 A9A8     LDA #$A8
0299 8505     STA $05
029B 4C74A2   JMP $A274
029E ADB802   LDA $02B8
02A1 C57B     CMP $7B
02A3 ADB902   LDA $02B9
02A6 E57C     SBC $7C
02A8 B0E9     BCS $0293
02AA 60       RTS
02AB E000     CPX #$00
02AD F008     BEQ $02B7
02AF EEB802   INC $02B8
02B2 D003     BNE $02B7
02B4 EEB902   INC $02B9
02B7 ADFFFF   LDA $FFFF
02BA 8DE502   STA $02E5
02BD E004     CPX #$04
02BF 1003     BPL $02C4
02C1 95F0     STA $F0,X
02C3 E8       INX
02C4 20D102   JSR $02D1
02C7 ADE502   LDA $02E5
02CA 38       SEC
02CB E930     SBC #$30
02CD 38       SEC
02CE E9D0     SBC #$D0
02D0 60       RTS
02D1 ADB802   LDA $02B8
02D4 C5F0     CMP $F0
02D6 F001     BEQ $02D9
02D8 60       RTS
02D9 ADB902   LDA $02B9
02DC C5F1     CMP $F1
02DE F001     BEQ $02E1
02E0 60       RTS
02E1 A200     LDX #$00
02E3 60       RTS
02E4 00       BRK
02E5 00       BRK
02F6 00       BRK
```

Robert Pendt
Poestenkill, NY 12140

* * * * *

ED:

First things first; I have a C4 with a 8 slot backplane and power supply, a D&N CM9 memory (24K) & floppy controller, a 527 board (24K), a D&N IO-1600 and Radio Shack Line Printer 1 (i.e. Centronics 779) and dual MPI minifloppies.

I originally bought a 8K cassette C2-8P so I could learn more about computers by expanding it myself. (How's that line go, "if it wasn't for bad luck I wouldn't have any luck at all").

Adding memory was easy enough, but the disk was trouble. I couldn't read what I had just written to the disk. Eventually, I sent the drive, cable, and D&N controller back to D&N. They added a couple of capacitors to counteract a ringing effect they felt was caused by the metal shield on the drive cable. Finally, everything seemed to be in order. Then I got a deal on a second drive, a D&N IO-1600 board and a RS LP1 Printer. I put up with double line feeds for a long time until after rereading an old PEEK(65) and someone said that the Radio Shack basic interpreter didn't provide a line feed. Therefore, I figured it had to be on the printer (I had thought it was a problem in the OSI DOS). Sure enough, with the help of a service manual, I found a jumper wire that selects an automatic linefeed function. To disable it, change the jumper from E1 & E2 to E1 & E3. Hope this helps somebody.

Now for the problem at hand. I've been working with Planner Plus V1.1 (which runs under OS-65D v 3.2) inputting all my utility bills, etc. for the past seven years (trying to see where all the money goes), and time after time, the thing will start giving me error messages. I've tried new disks (and starting all over), I've tried typing real slow and I don't even dare touch the keyboard when its closing or accessing the files for fear it will get upset. I can hear the disk looking back and forth and then before the error is masked by the trap routine, I see an error 1 (parity error). I've had the drive checked and aligned, I've run memory tests afterwards and read all the old PEEK(65)s. Can anyone help me?

Where can I get a set of the OSI Tech Notes?

Does video Ram have to be as fast as memory Ram to run at 2Mhz?

By the way, I program like I write letters. First, I get an idea, then I write a couple of lines, then I rewrite, then I get a pencil and paper, flowchart it, and start over.

Thank heavens PEEK(65) is independent of OSI, keep up the good work.

Craig S. Borst
Holland, MI 49423

Craig:

Contact your local dealer or OSI for a set of the Tech Notes.

Al

* * * * *

ISOTRON, INC.

PROUDLY ANNOUNCES THE ACQUISITION OF

OHIO SCIENTIFIC, INC.

A MESSAGE FROM THE PRESIDENT

I am happy to report that, even at this very early date, ISOTRON, Inc. is fully committed to: build upon the dedicated base of OSI users and resellers, guarantee a continuing supply of machines, parts and service, and to press confidently toward the release of new products and software.

By next month, we will be in a position to advise you, in a more explicit fashion, of the details which have so excited our team.

We look forward to working with and for you to put OSI back at the technological forefront of the micro computer world.

Merry Christmas and a Happy New Year.

Robert Lewis

ISOTRON, INC.
6515 MAIN STREET
TRUMBULL, CT 06611
(203) 268-3116

ED:

I own an OSI C2-D with 52K memory, 7MB hard disk, single floppy, a Hazeltine 1420 terminal, and a NEC 5500-D letter quality printer all of which runs under 65U.

I would like to have it run CP/M but I don't know where to begin. I realize that I will need to purchase some new boards etc., but I don't know which ones to buy, where to buy them, how much they will cost, etc.. In addition, I would require a version of CP/M that will support the hard disk and the letter quality printer. Where will I get that? Any help you can give me will be appreciated.

John Beamish
Ontario, Canada M6B 4A3

John:

You have several problems:
1) CP/M runs on the 8080/8085/Z80 CPU chip, which the C2-D does not have.
2) CP/M requires static RAM, which most C2-D's don't have.
3) CP/M and OS-65U store information in different disk formats, so files from the two systems can't be intermixed freely on the same disk.

There are several solutions:
If you want to run both CP/M and OS-65D, you must buy a new 510 CPU board and at least supplement your RAM if not replace it completely, then buy the CP/M and OS-U which are compatible with each other from OSI. Dick McGuire uses this system every day.

If you don't want to run OS-U, you can buy the D&N-80 board from D&N Micro Products, which will let you run standard CP/M, read and write IBM 3740 format floppy disks -- but won't let you use your hard disk at all ... yet. D&N has drivers written for the 37 and 74 Mbyte disks for their CP/M, and my guess is they will soon have one for the 7 Mbyte disk. Check with them. I use a D&N-80 board, with a D&N 64K RAM board and a D&N 1600 serial I/O board every day, and they work just fine.

Al

* * * * *

A LETTER TO OSIO & PEEK(65)

As OSI struggles to survive, those of us users who have not switched to other systems must band together for support. Our group in the Boston area has decided to rename ourselves "OSI Users/Boston" to better define our status. We are now proceeding to search out others in the same situation.

We would like to explore the possibility of establishing a national Users Group to expand on the exchange of software and hardware. Preliminary thoughts indicate that it would be a federation of local groups, perhaps using PEEK(65) and Compuserve for information exchange. With gradually decreasing resources, we should not waste efforts supporting extensive local newsletters or other activities, but should pool our capabilities to the maximum extent possible.

In order to increase our ability to keep together, OSI Users/Boston is conducting a campaign to equip all our machines with modems. Our hardware types are working on a club project for an inexpensive home-built modem, while others are purchasing some of the cheap modems now available on the market.

We are not aware of local groups other than OSIO. There must be many who could join in this effort. If you have such a list, we would like your assistance in contacting them. A copy of this letter will go to PEEK(65), which is an essential ingredient of any such plan.

Please respond to this letter, either directly or in PEEK (65). We need discussion, followed by action!

William R. Hutchins
OSI/boston
21 Winthrop Road
Lexington, MA 02173

William:

We agree! Although it will take considerable effort on someone's part to get the ball rolling and the "someone" needs to be found, we concur with the principle.

PEEK(65) can, hopefully, help to make the road easier by being the forum and vehicle of communications. To be more specific, we can provide a User Group Column in which to print the following: 1. The name, address and contact of all groups, 2. List new groups as they occur, 3. Describe the special interests and projects of each group, and 4. Disseminate information from the National Group. Additionally, we will be happy to maintain this list and inform callers of groups in their area.

As regards the dissemination of general information and articles, we already have "swap" arrangements with several User Group News Letters and their material and group are given appropriate credit.

We don't know all of the groups. Others we know of, but not how to contact them. So we make a plea for information. With your help we can develope that list of groups and make it available to both readers and callers. Most important, let us hear from you.

PEEK(65) Staff

ED:

Responding to the letter of Mr. Kent on pg 22 of Sept '83, I bypass the floppy drive disable switch with a 0.1 MF (or MFD), 600 V capacitor. This prevents system crash.

I'm a consultant, and I have two OSI C3-OEM's that I use to develop hardware and software for 6800, 6502, and 280 systems. I also have an IBM PC, Apple II+, Apple IIe, Quasar HHC, and Kim 1.

Rick Miller
Elgin, IL 60120

AD$

FOR SALE: C4PDMF with 48k RAM and 2 - 5 1/4" drives. Lots of OS-65D V3.2 software and utilities. OSI, Sams and V3.2 listing manuals. Very low hours on this like-new system. Asking $700. Excellent 19" color monitor included if you pay for shipping of all. Bob Curran, RD 2, Box 35, Mohawk, NY 13407. Phone (315) 866-7271.

* * * * *

Small eastern Iowa OSI dealer going out of business for personal computers. Everything must go. Best offer takes all or part. Call Steve at (319) 396-2415 after 4 Central Time or all day Saturday.

* * * * *

FOR SALE: OSI C2-OEM computer, 48k, dual 8" floppy. Televideo 920c terminal w/built-in modem. 65D 3.3 and 65U 1.43 operating systems w/manuals and extra system documentation. Misc. programs, extra diskettes w/cases. Asking $1600. Will pay shipping. Phone 703-942-2702.

* * * * *

32K C1P Series 2 Single Disk

AD$ continued on page 23

PEEK(65) INDEX FOR 1983

* * * * * * * * * * * * * * * * *

AD$ continued on page 22

Drive System OS-65D3.3 with extended monitor / assembler. Excellent condition. Full documentation, Sams Manual, best offer. AIS, 3517 Dunedin Dr. #204, Chesapeake, VA 23321. 804-484-8856.

* * * * *

48K, C4P-MF, full documentation, v 3.3 DOS, heavy duty supply, some software, Assembler and Ext Monitor, mint condition, $675, will ship. After 5 PM call 512-681-1983, San Antonio, TX.

* * * * *

Please write or call for free catalog listing of OSI compatible software products. This month's special MUSIC GENERATOR $49.00, includes The Little Fugue by Bach and A Mighty Fortress. Aurora Software Associates, 37 South Mitchell, Arlington Heights, IL 60005, 312-259-3150.

* * * * *

OSI C3 48K Memory with additional 8K on I/O Board CA-18A w/dual 8" Shugart floppies in separate enclosure in Mint condition. Includes Lifeboat 2.2 CP/M, OS-65D/65U operating systems, Microsoft MBasic and Fortran Software with all manuals and hardware documentation. $1500. Brad Miller, 212-490-0535.

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:

19809WLY03000G 3001*0404:0
DR WILEY, JR.
3001 LINDA LN.
SINKING SPRING, PA. 19608

GOODIES for OSI Users!

PEEK (65)
The Unofficial OSI Users Journal

P.O. Box 347 ∘ Owings Mills, Md. 21117 ∘ (301) 363-3268

( ) **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 — $15.00 $ _____

( ) **C2/C3 Sams Photo-Facts Manual.** The facts you need to repair the larger OSI computers. Fat with useful information, but just — $30.00 $ _____

( ) **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** - 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, documentation shows you how to call from any BASIC program on any disk and return it or any other BASIC program on any disk, floppy or hard. Most versatile disk sort yet. Will run under LEVEL I, II, or III. It should cost more but Sanders says, "...sell it for just..." — $89.00 $ _____

( ) **KYUTIL** - The ultimate OS-DMS keyfile utility package. This implementation of Sander's SORT/MERGE creates, loads and sorts multiple-field, conditionally loaded keyfiles. KYUTIL will load and sort a keyfile of over 15000 ZIP codes in under three hours. Never sort another Master File. — $100.00 $ _____

BOOKS AND MANUALS (while quantities last)

( ) **65V Primer.** Introduces machine language programming. — $4.95 $ _____

( ) **C4P Introductory Manual** — $5.95 $ _____

( ) **Basic Reference Manual** — (ROM, 65D and 65U) — $5.95 $ _____

( ) **C1P, C4P, C8P Users Manuals** — ($7.95 each, please specify) — $7.95 $ _____

( ) **How to program Microcomputers.** The C-3 Series — $7.95 $ _____

( ) **Professional Computers Set Up & Operations Manual** — C2-OEM/C2-D/C3-OEM/C3-D/C3-A/C3-B/C3-C/C3-C' — $8.95 $ _____

( ) Cash enclosed ( ) Master Charge ( ) VISA

Account No. _____ Expiration Date _____

Signature _____

Name _____

Street _____

City _____ State _____ Zip _____

TOTAL	$_____
MD Residents add 5% Tax	$_____
C.O.D. orders add $1.65	$_____
Postage & Handling	$ 3.50
TOTAL DUE	$_____

POSTAGE MAY VARY FOR OVERSEAS