PEEK (65)

The Unofficial OSI Users Journal

$1.75

NOVEMBER 1983
Vol.4, No.11

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

INSIDE

Column One

We are continuing this month the OSI Software Listing. The response has been good, but we know there are still some of you out there with programming jewels you have not told us about. Do send us a note about any software you would like to offer to the OSI community.

Several people have asked us whether we will offer the software in the listings for sale directly. The idea appeals to us, but we need to figure out how to do it. Our first thought is that we would make no profit, just offer a central contact point as a service to PEEKers. If you would like to sell/buy software through PEEK(65), let us hear from you, but DO NOT send any orders just yet!

As usual, we have tried to make sure this issue contains something for/about every type and user of OSI equipment. We hear the C2 and C4 users would like to see more articles about ROM machines other than the C1P. Since this magazine is 90% written by its readers, this means those same C2/C4 people must write more stuff about their machines. We pick up the mail daily...

Speaking of writing for PEEK(65), we are still looking for more articles and stories about business users of OSI equipment. Dealers, this is your chance to brag about your best installations, along with the software you have installed there. Between a story and a software listing, you can get lots of free advertising. If you don't, your competitors will.

And of course, we continue to need very basic articles for beginners. As machines are sold, we continue to have readers who are "brand new at it." So never think your particular article is too simple... just send it in, and let us be the judges.

We have heard from a number of people that one of the most difficult parts of using a small computer is figuring out how to hook up the various printers, plotters, modems, etc. which can work with it.

Brian Hartson of the PEEK(65) staff is working on an article all about I/O. Should be ready in a month or two. It will cover printer hookups, theory and practice of getting data into and out of a computer, and lots more. Stay tuned.

With the rather uncertain situation at the factory, some have left the fold, others have joined. We would like to have a good dealer list to help us publicize new developments and products. So even if you subscribe to PEEK(65), we may not know that you are an OSI dealer. Let us hear from you.

Speaking of the uncertain situation at the factory, it is still uncertain. Mid-October has passed, and whatever is going to happen appears to be imminent. We heard that papers are now being signed to restart operations under another new owner; we heard that the new owner or investor is a major Swedish company which will maintain OSI as a computer company; we heard that OSI is looking for new business space; we heard that recalls may go out as early as Nov 1. We don't know whether any of what we heard is true or not. You will probably know by the time you read this issue!

EPROM PROGRAMMER FOR SBII OR C1P

by: Guy Vanderwaeren
Wilgenstraat 73
B-2800 Mechelen, Belgium

For those people who want to put some programs or subroutines in EPROM, or even change their monitor or Basic ROMs, an EPROM programmer is the indicated tool. The only trouble could be the price, as they tend to be rather expensive. Most of them work independently of your microprocessor and thus have their own CPU, together with some ROM monitor, a few "K" of RAM and the programming circuit. A lot of unnecessary chips and money if you already have most of it in your computer. We do have the CPU, a monitor or interpreter and the RAM. Thus the only things we have to do are to write the software and connect a programming circuit. I will not give a "ready-to-run" program here to use with this programmer, because there are different ways of writing it: in assembly language or in Basic (why not?) and furthermore, because this circuit is not made and tested yet. Any remarks about possible errors will be very much appreciated.

Four memory locations are used for this programmer, $F3EC, $F3ED, $F3EE and $F3EF (or 62444, 62445, 62446 and 62447 in decimal).

Let's have a look at the decoding circuit of these locations, which consists of U1, U2, U3, U4, N6, N7, N8, N14 and U7. U1, U3 and N1 give the decodification of A2 to A15. The outputs of U1 and U3 are ORed in N12. This output is then ORed with the R/W signal in N13 to get the enable signal for decoder U2. This decoder decodes A0 and A1 into the 4 different locations (L1 to L4). N10, with N6 and N7, decodes a fifth line, L5 which goes to an input port. The other 4 are output ports. The input port however, has one of the 4 already mentioned addresses (see further on).

N11, N8 and N14 are the decodification of the Data Direction signal (DD), made from an OR of the 4 possible locations, the R/W line and the clock line 02.

The 4 parellel output ports are U9, U10, U11 and U12. They all are latches, connected to the data bus. U10 latches the 3 highest lines which are connected to the EPROM as A10, A9 and A8. U11 latches the 8 data lines, connected as A7 to A0 at the EPROM. U9 latches lines D0 and D1, which indicate the read mode or the program mode (D0) and give the program pulse (D1) when needed. U12 puts the code to be programmed in the location, set by U10 and U11, on the data lines of the EPROM.

The latched program pulse goes to U14, a monostable multivibrator, which will supply the exact pulse time for programming through N16, N17 and T1 or N18 and T2 at pin 18 of the EPROM. There is also a difference in voltage needed between read and program state on pin 21. This is done by the latched D0 and N9, T3, T4, T5 and T6.

Reading the EPROM is possible with the enabling of U13, a tri-state buffer as parallel input port. Its location is the same as that of U12, with the difference of the R/W line. Because it is impossible for the R/W line to be at logic 1 and logic 0 at the same time, we can connect an input port to the same location as an output port to save some decoding difficulties and some memory space.

I've put eight 10K resistances on the 8 data lines of the EPROM, because the 74LS244 is a very sensitive device, to be sure it sees a logic 1 when it should do so.

The circuit also includes a switch (S1), which gives the possibility to switch between a 2708 and a 2716 EPROM. The programming instructions for these two EPROMs are different in the area of the programming pulse. The 2716 is the easiest. It needs a pulse of 50 msec. for every programmed location, with no need to program the locations sequentially. This means that you may program first the 2nd location, next the 256th, next the 10th, etc. Also, every location has to be programmed only once.

The 2708 is a bit more difficult. It needs to be programmed sequentially, starting with the first location and ending with the last. Each location needs 100 consecutive 1msec. pulses for a total of 100 msec pulses. Thus you have to program each location (sequentially) with a pulse of 1 msec and then start all over again 100 times. The 100 times is to get an accumulated time of 100 msec. per location. Don't be disturbed by this. In practice two or three 1 msec. pulses is sufficient to burn your program in it. However, it would be wise to program the full 100 cycles to be sure your program is correct and you have the time to wait for the programming. Otherwise, it might be possible that you could lose your program in a few months.

For these two different requirements, the pulse generator U14 has two timing circuits and two outputs. For the 2708 (1 msec) we have a 15K resistance and a 0.1 mmF condensor which give a pulse to N16, N17 and T1. For the 2716 we have a 33K resistance and a 2.2 mmF condensor which give a 50 msec. pulse to N18, N19 and T2. The reason for N17 and N19 is that the programming pulse connection must be an active device because pin 18 of the EPROM generates a small current when low.

I do not want to leave you completely in the dark on the subject of the software. You can program in assembly language or Basic, with the only difference that the first one will be faster in execution. First thing to do should be reading the complete EPROM to test if it is empty. This is done by setting $01 in location $F3EF, the read mode. Next, set the numbers $000 to $3FF (2708) or $7FF (2716) in locations $F3EC and $F3ED. After the setting of each location, you should read location $F3EE to see if it is empty or not. Empty means that it is full of $FF or 255 decimal. Of course, it can be quite useful at this time, to put already programmed routines in RAM and write them in the EPROM again with a new additional routine.

If this test is done and supposing you have your new routine in RAM, you should set

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

FIG.3

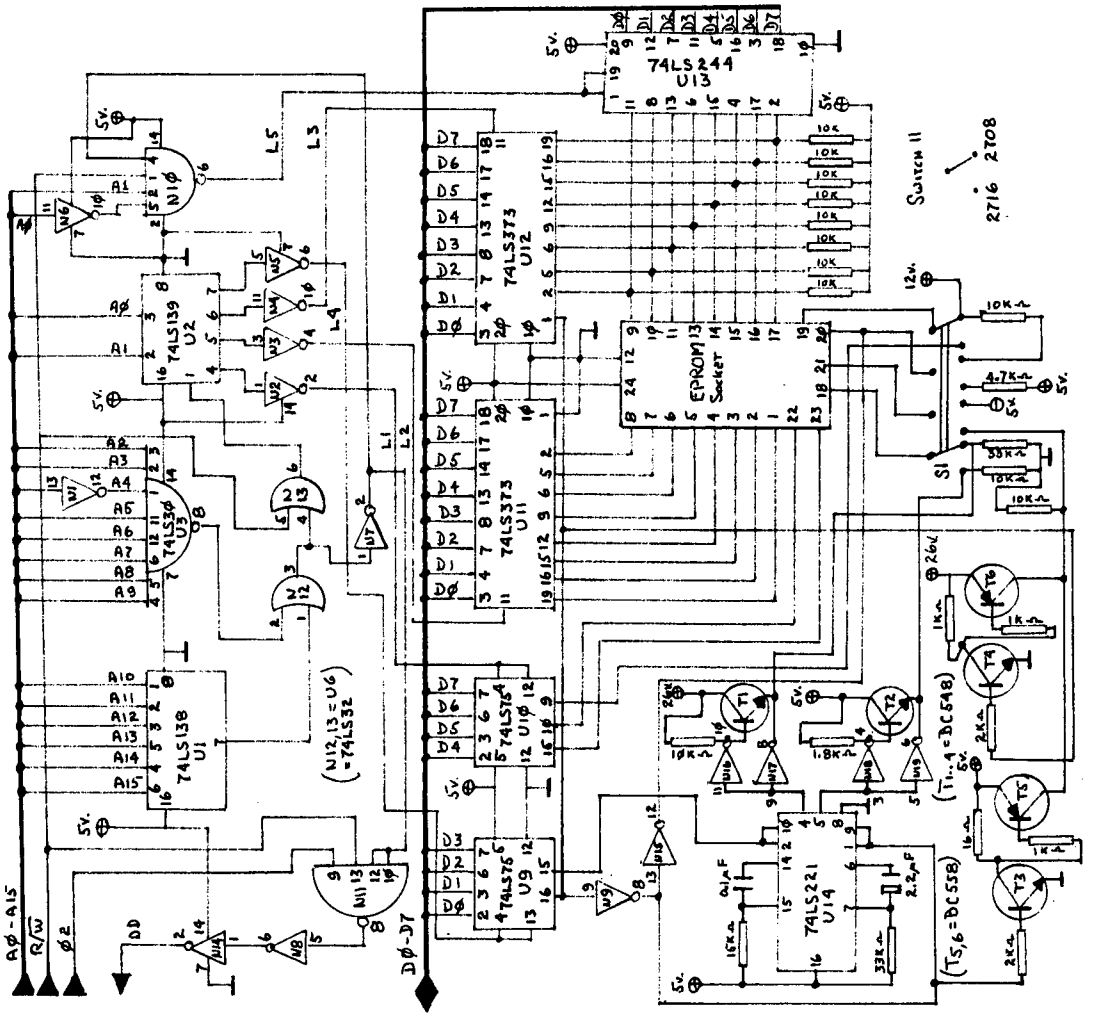
EPROM PROGRAMMER

Address

$F3EC	(62444 D)	A10	A9	A8	X	X	X	X	X
$F3ED	(62445 D)	A7	A6	A5	A4	A3	A2	A1	A0
$F3EE	(62446 D)	D7	D6	D5	D4	D3	D2	D1	D0
$F3EF	(62447 D)	X	X	X	X	X	X	B*	C*

* C: 1 = read
0 = program
B: 1 = program pulse
0 = read

N1...5 = U4 = 74LS04
N6...9 = U5 = 74LS04
N10,11 = U7 = 746520
N14...19 = U8 = 7406



the program mode with a $00 in $F3EF. Next, set the first address (between $000 and $3FF or $7FF) in $F3EC and $F3ED. Then set your data in $F3EE and finally toggle $02 in $F3EF. This last thing is done by setting $02 in $F3EF and immediately setting $F3EF back to $00. Repeat this program cycle for all the EPROM locations once for the 2716 or 100 times for the 2708.

The last thing should be a test of the programmed EPROM to see if there are any errors by switching back to the read mode and comparing the data read from the EPROM to the data in RAM. If this is correct you have successfully programmed your EPROM.

If you do not program the entire EPROM, the "don't care" locations should be programmed with $FF (all 1's) and if you update an already partially programmed EPROM, you will have to reprogram the existing contents as well. These are the special requirements of the 2708. The 2716 does not bother with all this, you may program any location in it at any time.

A last word now on another little program. How to get empty EPROMs if they are not. Normally they come empty from the factory, but what if you want to erase an already used one? For this job you will need to buy an EPROM eraser. This is a device with 1 or more special ultraviolet lamps and sometimes a timer. UV-light is the erasing power. This also means you should put a little sticker on the window of each programmed EPROM as the light from fluorescent lamps or sunlight can erase your EPROM in a few weeks. A more predictable and faster method is using a commercial eraser. The time needed for complete erasure depends on two factors. One is the strength of the UV rays, and two is the number of previous

erasures. An eraser always comes with an indication of the strength of the UV light. This is given in microwatts per square cm. Another thing you need to know is the needed dose of UV light required to erase the EPROM. For the 2708 and the 2716, this is 15 Ws/cm2. The time is then calculated with the formula: TIME = STRENGTH UV-LIGHT / NEEDED DOSE OF EPROM.

An example:

-suppose your eraser gives 12000 microW/cm2

-the time for erasure is then:

T = 15 / 0.012
= 1250 seconds
= 1250 / 60 = 21 minutes.

The change of 12000 to 0.012 is to put every factor in microwatts (decimal point 6 places to the left). As mentioned above, it may take a little more if you want to erase an older EPROM, which has been erased many times. For this, a formula does not exist.

All connections from the circuit are made to the 40 pin extention connector on the 600 board. You will have to put two 8T28's in U6 and U7 on the 600 board as data buffers. It would also be wise to put buffers on the address lines, the R/W line and the clock line. A ZIF (zero insertion force) socket for the EPROM can also be a big help.

The circuit consumes approximately 300mA at +5V and about 20mA at 26V (max. 27V!). With a 2708 you need about 65mA at +12V and 45mA at -5V additional.

Good luck. Next time we will describe a circuit for a 6K EPROM.

★

RELOCATING WP-6502
PART 1

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

WP-6502 is probably the best word processor written for OSI video machines. The tape version is assembled to reside in memory locations $0222 through $0F98. Many of the nonstandard monitor ROMs available today use locations $0222 through $0234 disabling the use of WP-6502. I own a C1P which I have modified at various times by adding an Aardvark C1S and C1E monitor ROMs and an expanded video. I wanted to use WP-6502 to write business letters and articles so I had the following choices:

1. Put in the standard monitor ROM.
2. Relocate WP-6502.
3. Write the articles by hand.

I have many routines which utilize the C1E monitor features so the first choice was eliminated. I did not want to write the articles by hand so I chose to attempt to relocate WP-6502. I will describe a procedure that I use to relocate WP-6502 in the three part article of which this is the first part.

The items needed to relocate WP-6502 are:

1. Some program to perform the relocation (OSI Extended Monitor)

2. A listing of WP-6502 (convenient but not necessary)

I used the following steps to relocate WP-6502. All address locations mentioned are the original addresses from your non-relocated tape version of WP-6502. All instructions with a * behind them will have to have their address fields modified to suit your relocation.

1. Relocate WP-6502 to the desired locations. In my case I prelocated it from locations $0222 through $0F98 to locations $0235 through $0FAB.

2. Replace the data which the nonstandard monitor ROM destroyed. If you have an Aardvark C1S/C2S monitor ROM, locations $0222 through $022F will have to be corrected.

```
$0222 = 52    $0223 = 02
$0224 = 01    $0225 = 20
$0226 = 80    $0227 = 40
$0228 = 7F    $0229 = 7C
$022A = 5C    $022B = 5D
$022C = 5E    $022D = 0B
$022E = 23    $022F = 5E
```

If you have an Aardvark C1E/C2E monitor ROM, locations $0230 through $0234 will also have to be corrected.

```
$0230 = 0A    $0231 = 42
$0232 = 3C    $0233 = 0A
$0234 = 00
```

3. Instructions which reference data outside of the relocation will not be modified by the relocation routine. In this case one such instruction exists; the instruction located at $024F. It will have to be modified by adding the amount of the relocation.

```
$024F BD0302 LDA $0203,X *
```

4. Data tables can cause relocation routines to misinterpret data for instructions and to miss the first instruction after the data table. In the case of WP-6502 there are two cases of data tables imbedded in instructions. The instructions at locations $0671 and $0784 will have to be modified by adding the amount of the relocation to them.

```
$0671  201503  JSR $0315 *
$0784  207106  JSR $0671 *
```

5. BIT instructions are sometimes used to provide multiple entry points into subroutines. When a BIT instruction is used in this manner, its address field is really another instruction and not an address. The relocation routine might possibly modify this address field, therefore, modifying the alternate entry point. In the case of WP-6502 there are seven occurrences of BIT instructions. The instructions located at $0307, $0317, $03D0, $03DD and $03EB should all be 2CA204 BIT $024A. The instruction located at $0323 should be 2CA202 BIT $02A2. The instruction located at $03C3 should be 2CA206 BIT $06A2.

6. For the C1E/C2E the cursor positioning will have to be modified. The address field of the following instructions will have to be modified to drop the cursor one row on your screen. The change for a C1P has been applied below:

```
$0638  9D80D3  STA $D380,X
```

The change for a C2P is:

```
$063C  9D80D7  STA $D780,X
```

7. The warm start code will have to be modified if you have a C1S because the C1S will mask out the 'Line Feed' character making it very difficult to edit data using Line Feeds. I modified the warm start code so it will use the old video routines for all commands except for the W/Tape command which must use the new video routines. The following code has worked for me; insert these new instructions, not modifications to the current ones, in the warm start code.

```
$0F3A  A900    LDA #$00
$0F3C  8D2906  STA $0629 *
$0F3F  8D3906  STA $0639 *
$0F42  A92D    LDA #$2D
$0F44  8D1A02  STA $021A
$0F47  A9BF    LDA #$BF
$0F49  8D1B02  STA $021B
$0F8F  A929    LDA #$29
$0F91  8D2906  STA $0629 *
$0F94  A980    LDA #$80
$0F96  8D3906  STA $0639 *
$0F99  A969    LDA #$69
$0F9B  8D1A02  STA $021A
$0F9E  A9FF    LDA #$FF
$0FA0  8D1B02  STA $021B
$0FA3  A920    LDA #$20
$0FA5  8D2A02  STA $022A
```

8. The cold start code will have to be modified to use the proper data/text starting address. This address is $0F9B in the nonrelocated WP-6502. This address will have to be modified by adding the amount of the relocation.

```
$0FAB  A99F    LDA #$9F
$0FB8  A99B    LDA #$9B
```

The warm start address in the cold start code will also have to be modified. The amount of the relocation will have to be added to the immediate data below:

```
$0FA7   A90B    LDA #$0B
```

You now have a relocated WP-6502 which will operate properly in all modes except Print. In the second article of this series, I will discuss adding an independent Print command. In the third part, I will discuss other enhancements I have made to WP-6502.

PRINTED REPORT CHECKPOINTING AND RESTART FOR OS-65U

By: David A. Weigle
108 N. Missouri Ave.
Morton, IL 61550

The prospect of having to reprint a lengthy report or rerun a time consuming report program (due to calculations which must be done before printing a line) as the result of a hardware problem (paper jam, bad printer ribbon, disk data check, etc.) can bring reactions ranging from minor irritation to that of the trauma of having a newly-licensed teenage driver. This is true whether the report is for a business application or personal use. Having the capability to restart the printing of a report from the point of interruption can result in saving both time and money.

This paper discusses a method to permit restarting the printing of:

1. A report at a specific page.

2. The report page currently being printed.

A sample program is provided which is intended to be a prototype to illustrate concepts. The methods described and the program code can be tailored to suit individual needs and/or styles of programming. The code has been adapted from my own report generation programs. It should be pointed out that the sample program uses features contained in releases 1.3 and 1.4 of OS65U (e.g., Extended Input, KILL, and FLAG commands). Code changes will be necessary if some other operating system or prior release level of OS65U is being used.

Restart on a page basis is accomplished through use of a checkpoint file. A checkpoint record is written to the file each time a new page of the report is begun. Each checkpoint record contains data which the program needs in order to resume printing the report beginning with a particular page. Such data could include intermediate totals,

program variable values, etc. For the purpose of our discussion each checkpoint record will contain only the number of the input disk file record associated with the first print line on the page.

Restarting the printing of the report page currently being printed is accomplished through storing (in an array) the data which comprises each print line until the page has been completely printed. The array is purged when the page has been printed, and a new array is allocated for the next page. If the printing of a page is interrupted, the restart routines can be used to either continue printing the page or reprint the page up to the point of interruption. If the page is to be reprinted, it can be done without having to reprocess the input data file records corresponding to the line items on the page.

Before proceeding to the sample program, let us define the criteria around which the program was written:

INPUT DATA FILE

Only one input file is used. It is a DBMS file named "MASTRO"; each record contains 10 fields.

CHECKPOINT RECORD FILE

The checkpoint record file, "CHKPTO" is also a DBMS file. Each record corresponds to a page number and contains one field which is the number of the input data file record corresponding to the first print line on that page of the report. It could be assumed that the checkpoint file is reset (i.e., indicating no checkpoint records exist) by some other program prior to the initial execution of the report print program. However, if the checkpoint file has not been reset, any records which might exist in the file will be overwritten if the program determines that a restart is not being done.

REPORT FORMAT

The first four lines of each page are used for the report title and column headings. Each page of the report will begin with the title centered on the page and the page number justified to the right margin. Following the title is a blank line. The third line is used for column headings. A row of dashes is printed as the fourth line to

```
10000 REM     PROTOTYPE PROGRAM FOR REPORT CHECKPOINTING AND RESTART
10100 :
10200 :
10300 :
10400 :
10500 REM SECTION I -- PROGRAM INITIALIZATION.
10600 :
10700 :
10800 :
10900 REM Routine I. A. -- FLAG commands.
11000 :
11100 :
11200 FLAG 11 : REM space suppression in numeric output to files
11300 FLAG 25 : REM no CONTROL-C program termination
11400 :
11500 :
11600 :
11700 REM Routine I. B. -- define program variables.
11800 :
11900 :
12000 J = 0
12100 DFR = 0 : REM variable used for loop control reading input file
12200 FIR = 1 : REM number of first input file record to be processed
12300 LIR = 0 : REM number of the last input file record
12400 PGN = 0 : REM report page number
12500 EID = 0 : REM end-of-data address -- input data file
12600 BID = 0 : REM beginning-of-data address -- input data file
12700 IRL = 0 : REM input data file record length
12800 ECD = 0 : REM end-of-data address -- checkpoint record file
12900 BCD = 0 : REM beginning-of-data address -- checkpoint record file
13000 CRL = 0 : REM checkpoint record length
13100 NCR = 0 : REM number of checkpoint records currently existing
13200 TCL = 0 : REM report title centering length
13300 RML = 0 : REM right margin length for page number justification
13400 YES = 0 : REM answer for questions (0 for no; -1 for yes)
13500 PRI = 0 : REM current page restart indicator (-1 for in process)
13600 JRI = 0 : REM job restart indicator (-1 if restarting job)
13700 NPY = -1 : REM no printing yet indicator (first page only)
13800 CPC = 0 : REM current page printing continuation return point
13900 CAE = 0 : REM current array entry in print data storage array
14000 AEU = 0 : REM number of array entries used (used by page restart)
14100 TI$ = "" : REM report title
14200 CH$ = "" : REM column headings
14300 CBC$ = "" : REM control break check character
14400 RPLY$ = "" : REM used for operator terminal data entry
14500 MSG$ = "" : REM message work area
14600 :
14700 DIM RECD$(10) : REM assume 10 fields in each input file record
14800 :
14900 :
15000 REM Construct report title and column headings.
15100 :
15200 READ TI$ : REM report title
15300 J = LEN(TI$) : REM length of report title field
15400 TCL = J + INT((80-J)/2) : REM title centering length
15500 RML = 80 - TCL : REM length for right margin justification
15600 :
15700 DATA "...  report title  ..."
15800 :
15900 READ CH$, RPLY$ : REM we assume it takes two DATA statements for
16000 CH$ = CH$ + RPLY$ : REM column headings
16100 :
16200 DATA "...  first half of column headings  ..."
16300 DATA "...  second half of column headings  ..."
16400 :
16500 :
16600 REM Set values for page length and number of lines to be printed.
16700 :
16800 POKE 14387, 66 : REM page is 11 inches long; printing @ 6 lpi
16900 POKE 14457, 60 : REM maximum lines to be used on a page
17000 POKE 15908, 60 : REM lines left to print on the first page
17100 :
17200 :
17300 :
17400 REM Routine I. B. -- Open input data file for processing.
17500 :
17600 :
17700 OPEN "MASTR0","PASS",1 : REM issue open for the file
17800 INDEX(1) = 9  : INPUT%1, EID : REM end-of-data address
17900 INDEX(1) = 20 : INPUT%1, BID : REM beginning-of-data address
18000 INDEX(1) = 31 : INPUT%1, IRL : REM record length
18100 LIR = (EID - BID)/IRL : REM number of records in the file
18200 :
18300 :
18400 :
18500 REM Routine I. C. -- Open checkpoint record file for processing.
18600 :
18700 :
18800 OPEN "CHKPT0","PASS",2 : REM issue open for the file
18900 INDEX(2) = 9  : INPUT%2, ECD : REM end-of-data address
19000 INDEX(2) = 20 : INPUT%2, BCD : REM beginning-of-data address
19100 INDEX(2) = 31 : INPUT%2, CRL : REM record length
19200 NCR = (ECD - BCD)/CRL : REM number of records in the file
19300 IF NCR ) 0 THEN 20400 : REM are there any checkpoint records?
19400 INDEX(1) = BID : REM position at first input data file record
19500 INDEX(2) = BCD : REM position at first checkpoint file record
19600 GOTO 23700 : REM begin report generation
19700 :
19800 :
19900 :
20000 REM Routine I. D. -- Determine if a restart is being done and,
20100 REM                  if so, make preparations for the restart.
20200 :
20300 :
```

continued

separate the body of the report from the title/heading section. The page length is 11 inches, and printing is done at six lines per inch. Six lines are required for top and bottom margins meaning that only 60 of the 66 possible print lines on each page will be used. As the first four lines on each page are reserved for headings, a maximum of 56 lines will be used on each page for the body of the report.

PROCESSING CONSIDERATIONS

The first character of the first field of each input record is tested to determine if a control break has occurred. If this character has changed from the previous record to the current one, a new page will be started.

LINE COUNTING AND FORM CONTROL

The program does not do its own line counting. It relies on the functions provided by OS65U for parallel printers for line counting and advancing the paper to top-of-form. The following memory locations are used:

1. 14387 - page length in terms of print lines.

```
20400 MSG$ = "Is this a restart"
20500 GOSUB 54000 : REM to program/operator dialog subroutine
20600 IF NOT YES THEN 19400 : REM if not a restart
20700 PRINT
20800 RPLY$ = "" : REM clear reply field
20900 PRINT "Enter page number at which to begin report ===) ";
21000 POKE 23721, 255 : REM set overstrike mode for data entry
21100 INPUT [3,"I"] RPLY$ : REM get page number from the operator
21200 IF RPLY$ = "" THEN PRINT CHR$(7); : GOTO 21100 : REM null reply
21300 J = ABS(VAL(RPLY$)) : REM page number (ignore sign, if present)
21400 PRINT
21500 IF J ) 0 AND J (= NCR THEN 21900 : REM checkpoint recd for page?
21600 PRINT "Error -- no checkpoint record for this page"; CHR$(7)
21700 GOTO 20900 : REM repeat request for page number
21800 :
21900 INDEX(2) = BCD + (J - 1) * CRL : REM address of checkpoint record
22000 INPUT#2, FIR : REM get beginning record of input data file
22100 INDEX(1) = BID + (FIR -1) * IRL : REM position input file at first
22200 REM                              record to be processed
22300 IF J = NCR THEN 22500 : REM if this is the last checkpoint record
22400 INDEX(2) = BCD + NCR * CRL : REM position past last checkpt record
22500 PGN = J - 1 : REM decrement page number for control purposes
22600 :
22700 :
22800 :
22900 :
23000 REM SECTION II -- INPUT DATA FILE PROCESSING.
23100 :
23200 :
23300 :
23400 REM Routine II. A. -- Set loop iterations for reading input file.
23500 :
23600 :
23700 FOR DFR = FIR TO LIR : REM loop iterations:  from the first to
23800 :                      REM the last input record
23900 :
24000 :
24100 :
24200 REM Routine II. B. -- Read input data file record.
24300 :
24400 :
24500 FOR J = 1 TO 10 : REM set loop iterations for reading recd fields
24600 INPUT#1, RECD$(J) : REM read an input record field
24700 NEXT J : REM read next field
24800 :
24900 :
25000 :
25100 REM Routine II. C. -- computations on input data file record.
25200 :
25300 :
25400 REM No sample program code is given for this routine.  We will
25500 REM assume that some computations and/or data manipulations are
```

2. 14457 - maximum number of lines to be printed on page (value placed in 14387 less the number of lines reserved for top and bottom margins).

3. 15908 - number of print lines remaining on a page (when the paper is at top-of-form, the value in this location is the same as that in 14457).

Before printing a line, the program interrogates locations 14457 and 15908 to determine if the paper is at top-of-form. If it is, this is considered to be a signal to write a checkpoint record and print page headings prior to printing any more data lines.

INTERRUPT PROCESSING

The interrupt processor section of the program is executed whenever the report print routines detect that the operator has entered CONTROL-C. When this occurs, the operator is given a choice of four courses of action:

1. Terminate the program.

2. Continue printing of the current page.

3. Restart printing of the current page.

4. Restart printing of the report at some other page.

A dialog is carried on between the program and the operator to achieve the desired course of action. The interrupt processor is designed to accept any number of CONTROL-C entries during the course of program execution without re-

```
25600 REM required before the next line of the report can be printed.
25700 REM We will also assume that after all processing of the input
25800 REM data for each record is completed the first eight entries in
25900 REM the input record data array, RECD$(), contain the data which
26000 REM is to be printed on the report and in the same order as they
26100 REM will be printed.
26200 :
26300 :
26400 :
26500 :
26600 REM SECTION III -- REPORT PRINTING PROCESS.
26700 :
26800 :
26900 :
27000 REM Routine III. A. -- Determine status.
27100 :
27200 :
27300 CPC = 1 : REM set current page print continuation return point
27400 GOTO 52100 : REM find out if operator has entered CONTROL-C
27500 IF PEEK(15908) = PEEK(14457) THEN 28700 : REM if now at the top
27600 REM                                        of the page
27700 IF CBC$ = LEFT$(RECD$(1),1) THEN 34100 : REM control break?
27800 :
27900 :
28000 :
28100 REM Routine III. B. -- Write checkpoint record for program
28200 REM                    restart.  The checkpoint record will be
28300 REM                    written only if no previous checkpoint
28400 REM                    record has been written for the page.
28500 :
28600 :
28700 PGN = PGN + 1 : REM increment page number
28800 IF PGN (= NCR THEN 30200 : REM is checkpt record to be written?
28900 PRINT#2, [CRL,"R"] MID$(STR$(DFR),2) : REM write checkpoint recd
29000 REM                                      using number of current
29100 REM                                      input data file record
29200 ECD = INDEX(2) : REM end-of-data address of checkpoint file
29300 INDEX(2) = 9 : REM end-of-data address field in DBMS file header
29400 PRINT#2, ECD : REM update DBMS header end-of-data address field
29500 INDEX(2) = ECD : REM position for next checkpoint record
29600 :
29700 :
29800 :
29900 REM Routine III. C. -- Reallocate print data storage array.
30000 :
30100 :
30200 KILL PDA$() : REM purge print data storage array
30300 J = FRE(X) : REM force garbage collection to clean up string
30400 REM              space.  This will shorten or possibly avoid any
30500 REM              delays caused by garbage collection during
30600 REM              subsequent processing.
30700 DIM PDA$(56,8) : REM reallocate print data storage array
30800 CAE = 0 : REM reset current array entry pointer
30900 NPY = 0 : REM clear nothing printed yet indicator
31000 CBC$ = LEFT$(RECD$(1),1) : REM set control break check character
31100 :
31200 :
31300 :
31400 REM Routine III. D. -- Print report page headings.
31500 :
31600 :
31700 REM Print report title and page number.
31800 :
31900 PRINT #5, [TCL,"R"] TI$; [RML,"R"] "PAGE" + STR$(PN)
32000 :
32100 CPC = 2 : REM set current page print continuation return point
32200 GOTO 52100 : REM find out if operator has entered CONTROL-C
32300 PRINT #5 ; REM one line space between title and column headings
32400 PRINT #5, CH$ : REM print column headings
```

quiring that the program be rerun.

There is one item (not illustrated in the sample program) which might need to be considered when a buffered printer is being used. When processing is interrupted via a CONTROL-C, there is the possibility that unprinted data resides in the printer's buffer that will be printed when the restart begins. It might be advisable to issue a buffer purge command to the printer before performing the restart to avoid printing "garbage". (Refer to the users manual for the printer being used.)

The remainder of this paper is the sample program itself.

★

MACHINE LANGUAGE DIRECTORY

By: Sidney Sosin
1107 Arbor Lane
Glenview, IL 60025

I don't know about you, but I get impatient waiting for the OS65D disk directory utility to print out on the screen, especially when there are 30 or more entries. The remedy I found was to rewrite the utility so that the search and display routines are in machine language. Now, the print out on screen is instantaneous. While I was at it, I made a few additional improvements.

First, I corrected a pet peeve regarding OSI's program: the "....ENTRIES FREE...." message displayed at the end. All it does is tell you how much room there is in the directory, a meaningless piece of information, since the number of slots available in the directory (64) has nothing to do with the number of tracks still available on the disk. The number of unused tracks is computed and printed out in my program.

Second, the standard DIR utility does not allow the choice of disk drive for those having dual disks. The new DIR does, and at the same time senses which drive you ran the program in and returns to that drive after it displays the directory of either disk. This is necessary since the machine language routines are loaded each time the program is run by calling a disk track directly to memory. This requires the track to reside on the disk where the BASIC program is. It's much better,

continued on page 17

```
32500 CPC = 3 : REM set current page print continuation return point
32600 GOTO 52100 : REM find out if operator has entered CONTROL-C
32700 POKE 12098,ASC("-") : REM change print pad character to a dash
32800 PRINT #5, [80,"L"] "-" : REM row of dashes under column headings
32900 POKE 12098,ASC(" ") : REM change pad character back to a blank
33000 CPC = 4 : REM set current page print continuation return point
33100 GOTO 52100 : REM find out if operator has entered CONTROL-C
33200 :
33300 IF PRI THEN 48600 : REM if page restart is in progress
33400 :
33500 :
33600 :
33700 REM Routine III. E. -- Put data for next print line in storage
33800 REM                   array and initiate printing of the line.
33900 :
34000 :
34100 CAE = CAE + 1 : REM advance to next print storage array entry
34200 FOR J = 1 TO 8 : REM set loop for storing print line data items
34300 PDA$(CAE,J) = RECD$(J) : REM store a print line data item
34400 NEXT J : REM store next data field
34500 :
34600 GOSUB 53100 : REM initiate printing of the line
34700 :
34800 :
34900 :
35000 REM Routine III. F. -- Loop control for processing the next input
35100 REM                   data file record.
35200 :
35300 :
35400 NEXT DFR : REM process next record
35500 :
35600 :
35700 :
35800 :
35900 REM SECTION IV. -- END-OF-JOB PROCESSING.
36000 :
36100 :
36200 :
36300 REM No sample program code is given for this program section.  We
36400 REM will assume that all input data file records have been read
36500 REM and the report printed.  This routine would close the files,
36600 REM print any total information on the report, issue messages to
36700 REM the operator and exit to the next program to be run.  Entry
36800 REM to this section could also come from the processing interrupt
36900 REM section if the operator chose to terminate the program.
37000 :
37100 :
37200 REM assume this is the entry point for end-of-job processing
37300 :
37400 :
37500 :
37600 :
37700 REM SECTION V. -- CONTROL-C INTERRUPT PROCESSOR.
37800 :
37900 :
38000 REM This section of the program is executed whenever the operator
38100 REM has interrupted processing by entering CONTROL-C.  There are
38200 REM four courses of action available:
38300 REM     1.  Terminate further processing
38400 REM     2.  Continue printing of the current page
38500 REM     3.  Restart printing of the current page
38600 REM     4.  Restart printing of the report at some other page.
38700 :
38800 :
38900 :
39000 REM Routine V. A. -- Reset CONTROL-C indicator.
39100 :
39200 :
39300 POKE 15006, 0
39400 :
39500 :
39600 :
39700 REM Routine V. B. -- Determine if processing is to be terminated.
39800 :
39900 :
40000 MSG$ = "Terminate the printing of this report"
40100 GOSUB 54000 : REM to program/operator dialog subroutine
40200 IF YES THEN 37200 : REM to EOJ section if terminating the report
40300 IF NPY THEN 28700 : REM if nothing has been printed yet --
40400 REM                   operator has entered CONTROL-C before
40500 REM                   printing of the first line has begun
40600 :
40700 :
40800 :
40900 REM Routine V. C. -- Determine if printing of the current page is
41000 REM                  to be continued.
41100 :
41200 :
41300 MSG$ = "Continue with the printing of this page"
41400 GOSUB 54000 : REM to program/operator dialog subroutine
41500 IF NOT YES THEN 42700 : REM if not continuing with this page
41600 :
41700 REM Resume printing of current page from point of interruption.
41800 :
41900 ON CPC GOTO 27500, 32300, 32700, 33300, 49100
42000 :
42100 :
42200 :
42300 REM Routine V. D. -- Determine if printing of the current page is
42400 REM                  to be restarted.
42500 :
42600 :
42700 MSG$ = "Restart the printing of this page"
42800 GOSUB 54000 : REM to program/operator dialog subroutine
```

Cont. on p.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.

SOFTWARE FOR OSI

EXPLANATION OF LISTING CODES

BASIC Version No./
Minimum computer/
1=SB,SBII,C1P,C2/4P 4=C4P
8=C8P
O=C2/3OEM
D=C2/3-D
2=C200,C3A/B
3=C300

Minimum Storage required/
C=Cassette
5=5 1/4" MF
8=8"FD
7=CD-7
2=CD-23/28/36/74
digit following indicates number of devices required

Systems Supported/
S=Single User
M=Multi-User
H=Hard Disk
R=Record Locking
record lock assumes multi-user. Two may be specified.

Software Support by/
D=Dealer
P=Phone
M=Modem
N=None
O=Other

Sold by/
A=Author
D=Dealer
M=Mail order
O=Other

Copies in Circulation/
No. divided by 10, ie.
1=Less than 11
11=100-110

Price/
Dollars only, no cents, tax, shipping, etc.

* * *

CP/M*BUSINESS*SERIAL

FORECASTER
1.5/O/82/SH/P/A/2/ $400
Author:
F. G. FANTER
P. O. BOX 385
VALLEY CENTER, KS 67147
Seller: SAME

System consists of 5 major programs:
1. Data base mgr-time series
2. Transformation program
3. Regression package
4. Seasonal adjustment
5. Plotting package for HP-7225B.

I will work with any purchaser to help meet your needs.

INCOME TAX PREP
MBAS/O/81/S/D/D/1/ $850
Author:
JIM FRITZ
1413 HARMONY LN.
ANNAPOLIS, MD 21401
Seller:
FINANCIAL ANALYST INC.
DOVER, DE 19901

Income tax processing program for small or large tax office. One office does 3,000 returns. Includes some states. Prints directly on to free IRS forms. Menu driven. Includes 1040, A,B,C,D,G 2210, 6251 plus state.

* * *

OS65-D*BUSINESS*VIDEO

B & W FILE MASTER
3.3/4/51/S/P/A// $105
Author:
BUNIN & WARD
P. O. BOX 895
NEW YORK, NY 10008
Seller: SAME

Create user and/or pre-defined file structures on a formatted screen. Commands allow: Random, sequential, or "find" to recall data; easy editing, screen dumps, updating, and deleting from on screen commands. Print labels and writes comprehensive reports.

TURF MANAGEMENT
/4/5/S/O/A/5/ $150
Author:
BENTON LABS
P. O. BOX 5455
ATHENS, GA 30604
Seller: SAME

Information on the characteristics and cultural practices for 8 common turf species for home and commercial use.

* * *

OS65-D*OTHER*SERIAL & VIDEO

BACKWATER
3.3/8/81/S/D/A/1/ $950
Author:
H & F CONSULTING
P. O. BOX 3568
MEMPHIS, TN 38173

Seller: SAME

Computes Backwater in natural channels. Uses standard step method. Computes critcal depth. Includes editor for entering cross section data.

COMBINE
3.3/8/81/S/D/A/1/ $350
Author:
H & F CONSULTING
P. O. BOX 3568
MEMPHIS, TN 38173

Seller: SAME

Takes stored hydrographs and will combine with another hydrograph of different duration.

EARTHWORK
3.3/8/81/S/D/A/1/ $950
Author:
H & F CONSULTING
P. O. BOX 3568
MEMPHIS, TN 38173

Seller: SAME

Computes earthwork by average end method. Includes editor for data entry. Add or modify cross-sections.

POND
3.3/8/81/S/D/A/1/ $500
Author:
H & F CONSULTING
P. O. BOX 3568
MEMPHIS, TN 38173

Seller: SAME

Takes Hydrograph computed by 'RUNOFF' and routes it through detention basin. User input is elevation vs storage and elevation vs discharge. Include editor for developing data.

RUNOFF
3.3/8/81/S/D/A/1/ $500
Author:
H & F CONSULTING
P. O. BOX 3568
MEMPHIS, TN 38173

Seller: SAME

Uses SCS method of unit hydrograph computation. Input is either basin parameters or time of concentration. Storm durations of 6 to 24 hours. Option to store runoff hydrograph.

WAG
3.3/8/81/S/D/A/1/ $300
Author:
H & F CONSULTING
P. O. BOX 3568
MEMPHIS, TN 38173

Seller: SAME

Takes input hydrograph and will wag it for specified duration.

* * *

OS65-D*OTHER*VIDEO

BASIC WORD PROCESSOR
3.3/4/81/S/O/A/1/ $15
Author:
WILLIS H. COOK
1298 RENEE DR.
LILBURN, GA 30247
Seller: SAME

With 32K RAM it will handle files of about 3 1/3 typed pages useful for address labels, letters or short reports. Line oriented. (You can't out type it). Commands: TYPE, DELETE, FIND LINE NO., REPLACE, INSERT LINE, CHANGE, VIEW, SAVE, LOAD, and PRINT. 65D v3.3. Listing & doc. $8.

BRX-10 HOME CONTROL
3.2/4/5/S/P/A/0/ $39
Author:
F. C. SCHWIERSKE
5014 LAKEFIELD RD.
CEDARBURG, WI 53012
Seller: SAME

Home control operating system allowing a C4PMF to time up to 16 BSRX10 loads. Software is completely menu driven with excellent graphics. Requires BSRX10 transmitter with serial input available from OSI or use plans included to modify a standard Radio Shack unit. For plans only send $5.

HEXDOS
ROM/1/51/S/P/D/17/ $49
Author:
STEVE HENDRIX
ROUTE 8 BOX 81E
NEW BRAUNFELS, TX 78130

Seller:
THE 6502 PROGRAM EXCHANGE
RENO, NV 89509

Full feature disk operating system, with named program & data files, line editing, true backspace &commands for direct disk access. Includes full disk of demo software, including revised disassembler which handles R65C02 opcodes will create a source file for HEXASM.Supports SYNMON,C1E & C1S.

HEXFORTH
HEXD/1/51/S/P/D/0/ $79
Author:
STEVE HENDRIX
ROUTE 8 BOX 81E
NEW BRAUNFELS, TX 78130

Seller:
THE 6502 PROGRAM EXCHANGE
RENO, NV 89509

Just released: a full Forth-79 standard implementation. Also supports floating point, with access to all Basic functions. Full assembler word set included. Additional words for access to HEXDOS files and an editor modeled after the one suggested by the Forth Standards Team incl in source form.

PROGRAMS FOR HOME USE
3.3/4/81/S/O/A/1/ $15
Author:
WILLIS H. COOK
1298 RENEE DR.
LILBURN, GA 30247
Seller: SAME

3 programs, name & address secretary. Holds 48 entries output formatted for labels. Appointment calendar holds 95 appointments, birthdays, etc. If it reminds you of your anniversary once, it is worth the price! Adding machine, great for checkbook balancing 65D v3.3. Listing & doc. $8.

SCREEN EDIT ROM
ROM/8/C/S/D/D/11/ $30
Author:
ROBERT CAMNER
8814 SECOND AVE.
SILVER SPRING, MD 20910
Seller:
COMMUNITY COMPUTERS
ARLINGTON, VA 22201

Powerful full screen editor for all OSI ROM based machines (specify version) cursor L/R/U/D; TAB, line delete char insert/delete; CTRL-S/Q scroll windows; window/screen clear revised keyboard routine with true lower case; disable "OK on null in port." More!

STOCK PORTFOLIO ANALYZER
3.3/4/81/S/O/A/1/ $15
Author:
WILLIS H. COOK
1298 RENEE DR.
LILBURN, GA 30247
Seller: SAME

It won't tell you which stocks to buy, but it will calculate how well yours are doing in addition to individual stock and total portfolio values. It determines the annual rate of appreciation for each one and year end sales records. 65D v 3.3 listing & doc. only $8.

* * *

OS65-D*UTILITY*VIDEO

HEXASM
ROM/1/51/S/P/D/5/ $39
Author:
STEVE HENDRIX
ROUTE 8 BOX 81E
NEW BRAUNFELS, TX 78130

Seller:
THE 6502 PROGRAM EXCHANGE
RENO, NV 89509

Resident assembler with conditional assembly, macros, file chaining, offset assembly, and numerous data types. Optionally creates assembly listing, error list, symbol table, and object code and routes each to screen, printer, tape, or disk. Supports R65C02 extra codes. 24K RAM & 1 disk.

HEXEDIT
HEXD/1/51/S/P/D/1/ $29
Author:
STEVE HENDRIX
ROUTE 8 BOX 81E
NEW BRAUNFELS, TX 78130

Seller:
THE 6502 PROGRAM EXCHANGE
RENO, NV 89509

Just released: screen-oriented text editor which keeps screen fully updated at all times. Includes installation program for customizing to your system & preferences(screen size,ctrl chars,etc).Automatic word wrap dynamic control of justification, numbering & page parameters. Handles any size text.

* * *

OS65-U*BUSINESS*SERIAL

ACCOUNTING SYSTEM (GEN)
1.2/O/82/HR/P/A/2/ $1500
Author:
ELECTRONIC INFO SYSTEMS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

Includes Chart of Accounts, GL, Cash Receipts, Cash Disbursements, Purchases Journal, Sales Journal, General Journal & Aging of Acct. Payable & Receivable. Provides balance sheet, P&L and other reports. Available single or multi-user. Handles multi-state/multi-company.

ACCOUNTS PAYABLE SYSTEM
1.2/2/71/HR/M/D/46/ $800
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

Prgm functions include Automatic Disbursement of inv amts in up to 5 distribution accts w/mthly & yrly acc totals tallied. Other functions include discount & over-ride capabilities, partial & manual pymt of inv & user-controlled payable option which allows user to select inv & vendors to be pd.

ACCOUNTS RECEIVABLE SYSTEM
1.2/2/71/HR/M/D/65/ $800
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

Each transaction entered is monitored by the computer for valid custmr acc no & inv no; current Customer Acc Balance & Inv Amt Due are displayed for user verification prior to posting each trans. Also hard copy audit-trail is prtd with posting process; the audit-trail lists each transaction.

ACCT & ENCUMBERED BUDGET SYS
1.2/2/71/HR/M/D/6/ $3000
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

Allows in-house tracking up to 1000 Projects/Accts in coord with std acct office procedure The BUDGET CONTROL module contains user defined & directed data entry procedures for PO/Requisition paperwk, & other trans. Data Entry provide for

verification & validation of acct nos, dollars amts, etc.

BUSINESS INVESTMENT SYSTEM
1.2/O/81/S/P/A/1/ $29
Author:
ELECTRONIC INFO SYSTEMS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

Includes 15 menu-selected programs: future value of investment/deposit, nominal/effective interest on investment, principal, payment, balance, term of loan, depreciation methods, and declining interest. Quick calculations for common business requirements.

BUSINESS VALUATION
/8/81/S/D/A/1/ $80
Author:
WAYNE R. COLE, CLU
805 CHUMLEIGH RD.
BALTIMORE, MD 21212
Seller: SAME

Illustrate the Value of a Business to establish the needs if any of Life Insurance to be used in a Buy and Sell or Stock Redemption arrangement.

CAPITAL NEEDS ANALYSIS
/8/81/S/D/A/1/ $80
Author:
WAYNE R. COLE, CLU
805 CHUMLEIGH RD.
BALTIMORE, MD 21212
Seller: SAME

Illustrate the need for new Life Insurance by Analyzing the Income potential from the clients other assets. It shows the needs up and through the youngest child's college and for the surviving Spouses life thereafter.

GENERAL ACCOUNTING
1.42/O/82/MH/D/A/5/ $1000
Author:
BROADT COMPUTERS
548 N. DERR DRIVE
LEWISBURG, PA 17837
Seller: SAME

Includes programs for accounts payable, accounts receivable, cash and general journal entries, complete end of month reports, and general ledger entries. Budgeting options are included. Payroll prior month comparison and inventory programs can be integrated with the package.

JOB COST ACCOUNTING
1.42/O/82/MH/D/A/5/ $1200
Author:
BROADT COMPUTERS
548 N. DERR DRIVE
LEWISBURG, PA 17837
Seller: SAME

Each invoice is distributed to one job # but as many accts as desired. Job #'s on payable check stubs. Automatic check writer withholds retainages. Unlimited employee job/hrs/hourly rate input. All phases of construction divided into general/mat'l/equip/labor/ and sub contr.

MAIL LISTS MAINT & LABEL PR SY
1.2/2/71/MH/M/D/41/ $295
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

MAILIST
new 9-digit zip code, comes ready to use by the non-prgmer & inc utility programs allowing user to enter, edit, or delete names & add from files. Labels can be prted either 1, 2,3 or 4 across. Spacing is entrd by user for label size.

MAILING LABEL
1.2/O/81/SH/P/A/2/ $75
Author:
ELECTRONIC INFO SYSTEMS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

Direct cursor aided input/edit feature, automated internal/external file sorting, and packing. All selections from menu and flexible. Permits field codes for selective mail outs.

PAYROLL PACKAGE
1.42/O/82/MH/D/A/5/ $350
Author:
BROADT COMPUTERS
548 N. DERR DRIVE
LEWISBURG, PA 17837
Seller: SAME

The main features include: Hourly rates, commissions, piece rates, tips, prepares 941s, prints W2s, records tax deposits, tax table change program, able to auto post to G ledger.

PAYROLL SYSTEM
1.2/2/71/MH/M/D/37/ $800
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

PAYROLL contains variety options: An OVERRIDE DEDUCTIONS allows operator either enter deductions manually, or bypass deductions from payck. SPECIAL CHECK option allows paymt of wkly, biwkly, or mthly sums (comm., piecewk, bonus, vac, or other non-strd payroll cks) w/deducts taken appropriately.

PAYROLL SYSTEM (PAY)
1.2/O/82/HR/P/A/2/ $800
Author:
ELECTRONIC INFO SYSTEMS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

Prepares payroll for salaried & hourly employees. Writes checks, W-2s, reports, summaries, etc. Handles multi-state payrolls. Can stand alone or integrate fully with EIS Gen. Accounting System. Available single (Level I) or multi user (Level III). Includes job distribution.

PIGGY BACK OR PREMIUM LEVERAGE
/8/81/S/D/A/1/ $240
Author:
WAYNE R. COLE, CLU
805 CHUMLEIGH RD.
BALTIMORE, MD 21212
Seller: SAME

Use up to seven old policies to illustrate the purchase of a new one. Choose the "TOTAL INSURANCE YEARLY OUTLAY" that is comfortable with the client. This outlay can be less than he now pays, equal to what he now pays, or more than what he now pays.

PROFESSIONAL INVESTMENT SYSTEM
1.2/O/82/SH/P/A/1/ $1500
Author:
ELECTRONIC INFO SYSTEMS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

An information management system for use by professional financial investment firms. Free-standing, menu-driven. Provides up-to-date portfolios and many timely reports. Has market classifications, stock/bond information, portfolios, transactions, and system information.

THE DBIS RETAIL PHARMACY SYS
1.2/2/71/HR/M/D/14/ $4000
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

The DBIS Retail Pharmacy Sys enables the pharmacist to handle the clerical tasks involved in prescription filling and 3rd party billing quickly and accurately. Data on families, individuals, patient drug profile, drugs, drug pricing and doctors can be recalled & seen within seconds.

VEHICLE RESERVATION SYSTEM
1.2/2/71/MH/M/D/6/ $800
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

Maintains info bank of all re-

servations; Instant"On Screen" resv info showing name, add, home & bus phone nos, organization, destination, time, chgs, depst, & bal due; Tells avail of vehicles when booking resv; Provides timely & accur info on resv by date or vehic. Warns of over-booking.

WHOLESALE/RETAIL IND DIST SYS
1.2/2/71/HR/M/D/32/ $800
Author:
DBIS
1 MAYFAIR RD.
EASTCHESTER, NY 10707
Seller: SAME

The Wholesale/Retail Industry Distribution System contains 3 modules: Inv., Order Entry/Inv & A/R. All data entry is interactive & fully prompted. Verification of data elements (Product No, Customer Acc No, & Inv No) is computer control w/pricing & price extensions computed & displayed.

WORD PROCESSOR
1.42/O/81/MH/D/A/5/ $250
Author:
BROADT COMPUTERS
548 N. DERR DRIVE
LEWISBURG, PA 17837
Seller: SAME

Special codes used for scrolling margins, underlining, & etc. Allows you to format the page as desired. Documents can be sequenced on a blank page & can be made up of whole or partial pages from the same or different disks, can be used with mailing list data base to produce form letters.

* * *

OS65-U*OTHER*SERIAL & VIDEO

COMCLR
2044/U/8/HR/M/D// $100
Author:
BROWN/COLLINSON ASSOC.
619 "E" AVENUE
LAKE OSWEGO, OR 97034-2244
Seller: SAME

Comes with ELSE, FORMFD. A version of COMKIL that does not require INP$, nor does it disable the DEF FNx()function. Extends the CLEAR statement instead of KILL. Resides in the same space as COMCLR (B000-BFFF).

ELSE
2044/U/8/HR/M/D// $100
Author:
BROWN/COLLINSON ASSOC.
619 "E" Avenue
Lake Oswego, OR 97034-2244
Seller: SAME

Comes with COMCLR, FORMFD. Use (on separate lines!)
If (condition) THEN..:..:...
ELSE..:..:.. or
IF (condition) THEN ..:..:...
ELSE IF (condition) THEN..:..
etc. for a CASE statement. Runs on any machine running U. Flyer available.

NETWK
1.42/O/81/SH/D/D/1/ $500
Author:
ROBERT CAMNER
8814 SECOND AVE.
SILVER SPRING, MD 20910
Seller:
COMMUNITY COMPUTERS
ARLINGTON, VA 22201

NETWK turns your C2/C3 into a host for up to 16 satellite C1P/C4Ps. Features true 2 way communications, dynamic, SAVE & LOAD, PRINT and more. Requires 550 board for host screenedit editor ROM for satellite.

* * *

OS65-U*OTHER*SERIAL

B/CAL4
1.4+/2/2/HR/M/D/1/ $50
Author:
BROWN/COLLINSON ASSOC.
619 "E" AVENUE
LAKE OSWEGO, OR 97034-224
Seller: SAME

This utility is an extension to Level 3 allowing any term to 'connect' to any memory partition or any ACIA to act as the terminal for any part. 3 terminals can talk to one partition or one term may switch to any memory partition by a keyboard sequence without disturbing work in progress.

GRADER
1.2/O/81/S/N/A/1/ $10
Author:
PATRICK CLUSMAN
FOND DU LAC
WI 54935

Seller: SAME

Teacher aid for weighted student grades. Allows for tests and homework.

TIME & TASK PLANNER
1.43/8/81/MH/P/D/1/ $150
Author:
GANDER SOFTWARE
3223 BROSS RD.
HASTINGS, MI 49058
Seller: SAME

A personal success tool! User defined Daily Scheduler, To Do List, Future Planning List, Work Sheets and printed calendars for any mo or yr. Separate files for 5 users (5200 appointmts). Very easy to use but flexible, full support, 30 day FREE trial. You'll never use desk-top calendars again.

* * *

OS65-U*UTILITY*SERIAL

BASIC CROSS REFERENCE (BASXR)
1.2/O/81/SH/P/A/2/ $29
Author:
ELECTRONIC INFO SYS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

BASXR is a programming tool which facilitates modification and debugging. A menu allows listing of (1) All VARIABLES and their line number in order of occurrence (2) twelve disk related OPERATIONS with line numbers of each appearance, and (3) any BASIC COMMAND with complete line printouts.

CSS SCREEN MANAGER
1.2/O/8/SH/P/M/1/ $50
Author:
LEE CONYERS
RESTON, VA 22091

Seller: SAME

Professional screen management system increases productivity with user-friendly data forms for 65U applications. Uses advanced features of many CRTs. Highly customizable for unique situations. Very modular and performance optimized. Commands include: INPUT MODIFY STATS FIND OBTAIN READ WRITE

HARD DISK BACK-UP (HFCOPY)
1.2/D/71/SH/P/A/1/ $75
Author:
ELECTRONIC INFO SYSTEMS
P. O. BOX 5893
ATHENS, GA 30604
Seller: SAME

A series of programs which allow a hard disk to be backed up to floppies by two methods. One allows multiple floppies to be used for large files. The second allows many small files to be backed upon one floppy.

INKEY-LIKE COMMAND
1.43/O/81/HR/D/A/2/ $50
Author:
SIGMA, INC.
CP 5
LEVIS, QUEBEC G6V 6N6

Seller: SAME

Allows any serial terminal to behave like a video keyboard: Program stops and waits for user to press any key, ASCII value of character goes in variable. TT=S.kb(X). X being a number from 0 to 3, allowing to echo character on screen only if of a certain type.

★

3 USERS-80 Mega Bytes — $9990.00*

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

INTRODUCTORY SPECIAL

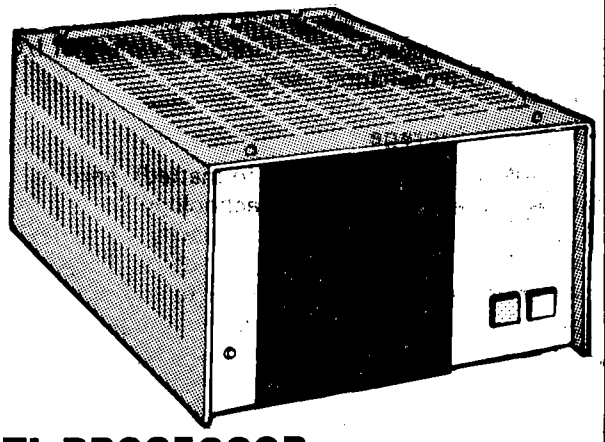
- 90 Days on Power Supply, Floppy Drives — Circuit Boards.
- Configured for Time-Share @ 2 MHZ
- Includes: 2 Serial Printer Ports with Handshake, Improved Cooling, and Ball Bearing Roller Chassis Rails

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 9

```
42900 IF NOT YES THEN 44100 : REM if restart of this page not desired
43000 IF PRI THEN CAE = AEU : REM restore current print data array
43100 REM                     entry number if current page restart
43200 REM                     has been interrupted
43300 GOTO 45600 : REM determine who will reposition the paper
43400 :
43500 :
43600 :
43700 REM Routine V. E. -- Determine if report printing is to restarted
43800 REM                  at some other page.
43900 :
44000 :
44100 MSG$ = "Restart the printing of this report"
44200 GOSUB 54000 : REM to program/operator dialog subroutine
44300 IF NOT YES THEN 40000 : REM repeat conversation if negative reply
44400 PRI = 0 : REM clear current page restart indicator, if set
44500 JRI = -1 : REM set job restart indicator
44600 NPY = -1 : REM set nothing printed yet indicator
44700 IF NCR < PGN THEN NCR = PGN : REM update the count of checkpoint
44800 REM                          record which have been written
44900 :
45000 :
45100 :
45200 REM Routine V. F. -- Determine whether the operator or program is
45300 REM                  to position the paper at top-of-form.
45400 :
45500 :
45600 IF PEEK(15908) = PEEK(14457) THEN 47400 : REM now at top-of-form?
45700 MSG$ = "Have you positioned the paper at the top of the page"
45800 GOSUB 54000 : REM to program/operator dialog subroutine
45900 IF NOT YES THEN 46300 : REM if operator hasn't repositioned paper
46000 POKE 15908, 60 : REM reset number of lines left to print on page
46100 GOTO 47400 : REM go initiate the restart
46200 :
46300 MSG$ = "Should the program position paper at the top of the page"
46400 GOSUB 54000 : REM to program/operator dialog subroutine
46500 IF NOT YES THEN 45700 : REM repeat conversation if negative reply
46600 :
46700 FLAG 101 : REM advance paper to top-of-form
46800 :
46900 :
47000 :
47100 REM Routine V. G. -- Initiate job or current page restart.
47200 :
47300 :
47400 IF JRI THEN JRI = 0 : GOTO 20700 : REM if doing a job restart
47500 :
47600 IF CAE > 0 THEN PRI = -1 : REM set current page restart indicator
47700 REM                        if there are lines to be printed
47800 GOTO 31900 : REM to page heading routine
47900 :
48000 :
48100 :
48200 REM Routine V. H. -- Reprint the body of the report for the
48300 REM                  current page.
48400 :
48500 :
48600 CPC = 5 : REM set current page continuation return point
48700 AEU = CAE : REM store number of print data array entries used
48800 CAE = 1 : REM begin with first print data array entry
48900 GOSUB 53100 : REM reprint a line of the report
49000 GOTO 52100 : REM find out if operator has entered CONTROL-C
49100 IF CAE = AEU THEN 49400 : REM if all lines have been printed
49200 CAE = CAE + 1 : REM advance to next print data array entry
49300 GOTO 48900 : REM print next line
49400 PRI = 0 : REM clear current page restart indicator
49500 GOTO 27500 : REM resume normal processing
51000 :
51100 :
51200 :
51300 :
51400 REM SECTION VI. -- SUBROUTINES.
51500 :
51600 :
51700 :
51800 REM Subroutine VI. A. -- Determine if CONTROL-C was entered.
51900 :
52000 :
52100 PRINT CHR$(0); : REM terminal I/O to permit CONTROL-C checking
52200 IF PEEK(15006) > 0 THEN 39300 : REM to interrupt section if CTL-C
52300 :
52400 ON CPC GOTO 27500, 32300, 32700, 33300, 49100 : REM resume
52500 :
52600 :
52700 :
52800 REM Subroutine VI. B. -- Print report line items.
52900 :
53000 :
53100 PRINT #5, PDA$(CAE,1); TAB(5); ... ; PDA$(CAE,4); : REM 1st half
53200 PRINT #5, TAB(5); PDA$(CAE,5); ... ; PDA$(CAE,8) : REM last half
53300 RETURN : REM return to using routine
53400 :
53500 :
53600 :
53700 REM Subroutine VI. C. -- Program/operator dialog.
53800 :
53900 :
54000 RPLY$ = "" : REM clear reply field
54100 PRINT
54200 PRINT MSG$; "? ==> "; : REM ask the question
54300 POKE 23721, 255 : REM set overstrike mode for data entry
54400 INPUT [1,"A"] RPLY$ : REM get operaor response
54500 IF RPLY$ = "Y" OR RPLY$ = "y"
          THEN YES = -1 : RETURN
54600 IF RPLY$ = "N" OR RPLY$ = "n"
          THEN YES = 0 : RETURN
54700 PRINT CHR$(7); : REM sound term
      inal alarm if invalid response
54800 GOTO 54300 : REM repeat data
                   entry
```

continued from page 9

however, than the usual practice of poking data in one character at a time in the BASIC program. Poking defeats the purpose of having the program run as fast as possible.

Third, the display is in four column format, so all the programs can be shown at once, instead of merrily scrolling up the screen as in the original.

How It Works

BASIC Program: After clearing the screen in line 20 (those of you who have fast clears can substitute them), line 30 saves the drive number which OS65D stores in location 9820 (265C in hex) and pokes 0 into location 2888 (0B48 hex) to permit a null to be entered on return. This prevents an exit to BASIC and makes return into a default input. The default values are enclosed in "<>". The machine language program, which in my version is stored on track 30, sector 1, is called into memory at hex address 9000 using a system command. Line 50 then asks for the drive for which the directory is to be printed and line 60 executes

LISTING 1.

```
10 REM DIRECTORY UTILITY FOR OS-65D
12 REM BY SIDNEY SOSIN
14 REM    GLENVIEW, ILLINOIS
16 JULY, 1983
20 FORX=1TO25:PRINT:NEXT
30 DN=PEEK(9820):POKE2888,0
40 DISK!"CA 9000=30,1"
50 INPUT"WHICH DRIVE (A/B)  <A>";DR$
55 IFDR$=""THENDR$="A"
60 DISK!"SE "+DR$
70 IFDN=1THENDN$="A":GOTO90
80 DN$="B"
90 INPUT"LIST ON PRINTER, TOO (Y/N)  <N>";A$
95 IFA$=""THEN110
100 IFLEFT$(A$,1)="Y"THENDISK!"IO ,0A"
110 PRINT:PRINT
120 POKE574,0:POKE575,144
130 PRINTTAB(24)"DIRECTORY"
140 PRINTTAB(24)"---------"
150 FI$="NAME  TRACKS"
160 LI$="----------  ----------"
170 PRINTFI$SPC(3)FI$SPC(3)FI$SPC(3)FI$
180 PRINTLI$SPC(3)LI$
190 DISK!"CA 2E79=08,1"
200 X=USR(X)
210 NF=PEEK(160)
220 DISK!"CA 2E79=08,2"
230 X=USR(X)
240 NF=NF+PEEK(160)
250 NF=77-NF
260 DISK!"IO ,02":POKE2888,27
270 PRINT:PRINTNF"TRACKS FREE.  YOU ARE IN DRIVE "DN$
280 IFDN$=DR$THENEND
290 DISK!"SE "+DN$
300 END
```

the system command to select that drive. Since the operating system stores drive A and B identities as the numbers 1 and 2, lines 70 and 80 substitute the equivalent letters as strings.

Lines 90 and 100 permit a printer using the OSI line printer interface driver at hex address 249F to provide hard copy of the directory. As set forth in the OS65D manual, the operating system checks an I/O flag and selects one of eight outputs, depending on which of the eight bits in the flag are set. Bit 3 selects the printer, while bit 1 sends output to the screen. More than one output can be used simultaneously, so when line 100 sends the I/O command "0A", it turns on both the screen and printer outputs. This is so because the hex number A is binary number 00001010, which sets bits 1 and 3.

Line 120 sets up the machine language USR call to address $9000. Lines 130 through 180 format the four-column headings, line 190 calls the first sector of the directory into memory at address $2E79 and line 200 executes the machine language routine. Line 210 retrieves the passed parameter -- the number of tracks used -- and stores it in the variable NF. Lines 220 through 240 do the same thing with the second sector of the directory, adding the new passed parameter to NF to indicate the total number of tracks in use on the disk. Line 250 then converts this number to the number of free tracks by subtracting it from the 77-track capacity (tracks 0 to 76). Line 260 puts you back in pure screen I/O mode and restores exit on carriage return. Line 270 prints the total of the free tracks and a reminder of which drive is in use. The remaining lines check to see if it's the drive you started with and, if not, issue a system command to return to that drive.

Assembly Program: Listing 2 sets forth the assembly of the machine language routine which is called from the BASIC USR function. The program is heavily commented and should be fairly self-explanatory, at least to those familiar with assembly language.

About half the program is devoted to converting the binary coded decimal track numbers used by OSI into ASCII for display and into hex for

LISTING 2.

```
 10                ; DIRECTORY DISPLAY AND TRACK COUNTING ROUTINE
 20                ;           (VERSION 2 --  7/23/83)
 30                ;    *** COPYRIGHT 1983 BY SIDNEY SOSIN ***
 40                ;         *** ALL RIGHTS RESERVED ***
 50                ;
 60 9000           *=$9000             ; FOR 48K SYSTEMS
 70 0090=          MEM=$90             ; POINTER TO DIRECTORY
 80 0092=          COUNT=$92           ; COLUMN COUNTER
 90 0093=          TRACKS=$93          ; TEMP TRACK NO. STORAGE
100 00A0=          PASSP=$A0           ; NO.TRACKS-PASSED TO BASIC
110 2343=          VID=$2343           ; DISPLAY ROUTINE
120 2D6A=          CRLF=$2D6A          ; CARRIAGE RETURN/LINE FEED
130 9000 D8              CLD           ; INITIALIZE
140 9001 A979            LDA #$79      ; SET POINTERS TO DIRECTORY
150 9003 8590            STA MEM       ;  STORAGE AREA
160 9005 A92E            LDA #$2E
170 9007 8591            STA MEM+1
180 9009 A904            LDA #4        ; SET 4 COLUMN DIRECTORY
190 900B 8592            STA COUNT
200 900D 206A2D          JSR CRLF      ; START AT MARGIN
210 9010 A000            LDY #0
220 9012 84A0            STY PASSP
230 9014 206690          JSR BLK2      ; CHECK FOR BLANK ENTRY
240 9017 A208            LDX #8        ; 8 CHARS TO EACH ENTRY
250 9019 C8              INY           ; Y=0
260 901A B190    START   LDA (MEM),Y   ; GET CHAR
270 901C E003            CPX #3        ; AT TRACK NO.?
280 901E B00C            BCS NEXT      ; NO
290 9020 48              PHA           ; YES - SAVE CHAR
300 9021 A920            LDA #$20      ; PRINT SPACE
310 9023 204323          JSR VID
320 9026 68              PLA           ; GET CHAR BACK
330 9027 205490          JSR CONVT     ; CONVERT TO ASCII
340 902A 0930            ORA #$30
350 902C 204323  NEXT    JSR VID       ; OUT TO SCREEN
360 902F CA              DEX           ; DONE WITH ENTRY?
370 9030 D01E            BNE INCY      ; NO
380 9032 20A690          JSR TRAKCT
390 9035 206390          JSR BLANK     ; YES-COUNT NEXT ENTRY
400 9038 A208            LDX #8        ; RESET CHAR COUNT
410 903A A920            LDA #$20      ; 3 SPACES
420 903C 204323          JSR VID
430 903F 204323          JSR VID
440 9042 204323          JSR VID
450 9045 C692            DEC COUNT     ; NEXT COLUMN
460 9047 D007            BNE INCY      ; DONE WITH 4 COLS? NO
470 9049 206A2D          JSR CRLF      ; YES - BACK TO MARGIN
480 904C A904            LDA #4        ; AND RESET COLUMN COUNT
490 904E 8592            STA COUNT
500 9050 C8      INCY    INY           ; BUMP POINTER TO MEM
510 9051 D0C7            BNE START
520 9053 60              RTS
530 9054 48      CONVT   PHA           ; SAVE STARTING TRACK NO.
540 9055 4A              LSR A         ; MOVE HI TO LO  NYBBLE
550 9056 4A              LSR A
560 9057 4A              LSR A
570 9058 4A              LSR A
580 9059 18              CLC           ; CONVERT TO ASCII
590 905A 6930            ADC #$30
600 905C 204323          JSR VID       ; AND PRINT IT
610 905F 68              PLA           ; GET WHOLE BYTE BACK
620 9060 290F            AND #$F       ; MASK OFF HI NYBBLE
630 9062 60      DONE    RTS           ; AND RETURN TO PRINT IT
640 9063 C8      BLANK   INY
650 9064 F012            BEQ EXIT      ; IF ZERO, PAGE IS DONE
660 9066 B190    BLK2    LDA (MEM),Y   ; GET CHAR
670 9068 C923            CMP #$23      ; CHECK FOR BLANK ENTRY
680 906A F003            BEQ SKIP
690 906C 88              DEY           ; BACK ONE TO MESH WITH
700 906D D0F3            BNE DONE      ; LINES 250 AND 500
710 906F 98      SKIP    TYA           ; GO TO NEXT ENTRY
720 9070 18              CLC
730 9071 6907            ADC #7
740 9073 F003            BEQ EXIT      ; IF ZERO, PAGE IS DONE
750 9075 A8              TAY           ; SET POINTER
760 9076 D0EB            BNE BLANK     ; AND DO IT AGAIN
770 9078 A5A0    EXIT    LDA PASSP     ; GET NO. OF TRACKS IN USE
780 907A 48              PHA           ; SAVE BCD NO.
```

continued

counting free tracks. These conversions are done in the CONVT, TRAKCT and EXIT subroutines. The counting is done by setting the microprocessor to its decimal mode, which permits the binary coded decimal track numbers to be used as is. Subtracting the starting number from the ending number of each directory entry gives us the number of tracks in the entry. These are cumulated so that we end with a BCD total of the used tracks. The total is then converted in the EXIT routine to a hex number which is passed as a parameter to BASIC.

Changes for Small Systems:

There are two basic changes for systems which use five inch disks and have less than 48K of memory. First, the directory on small disks is maintained on track 12 rather than 8, so lines 190 and 220 of the BASIC program must be changed accordingly. Second, the machine language program must be assembled at a low memory address and a corresponding change made in line 40. Although I don't have access to a small disk system, I believe the important OS65D system routines and USR and other peek and poke addresses should be the same as in the larger versions.

Once you install the instantaneous directory display, you will wonder how you ever did without it. It is particularly valuable to those who have no printer, or to the absent-minded souls like me who forget a name or track number one second after wiping the directory from the screen.

If you have no assembler or are just too lazy to type in the program, I will provide a custom version to your specs on your eight-inch OS65D disk for $6.00 postpaid.

LISTING 2. CONTINUED

```
 790 907B 4A           LSR A         ; MOVE TENS COLUMN TO LOW
 800 907C 4A           LSR A                                  BYTE
 810 907D 4A           LSR A
 820 907E 4A           LSR A
 830 907F F017         BEQ NYB2      ; IF ZERO GET LOW NYBBLE
 840 9081 AA           TAX           ; COUNT THE TENS
 850 9082 CA           DEX           ; USING X REGISTER
 860 9083 D006         BNE EX2       ; IF ZERO WE'RE DONE
 870 9085 A90A         LDA #$A       ; TEN IN HEX
 880 9087 8593         STA TRACKS    ; SAVE IT
 890 9089 D00D         BNE NYB2      ; DO LOW BYTE
 900 908B A90A EX2     LDA #$A       ; SAVE TEN AS MULTIPLE
 910 908D 8593         STA TRACKS
 920 908F D8           CLD           ; GO TO BINARY MODE
 930 9090 18           CLC           ; ADD TENS TIL REGISTER
 940 9091 6593 DECMAL  ADC TRACKS    ;  IS ZERO
 950 9093 CA           DEX
 960 9094 D0FB         BNE DECMAL
 970 9096 8593         STA TRACKS    ; SAVE TOTAL TENS
 980 9098 68   NYB2    PLA           ; NOW DO ONES
 990 9099 290F         AND #$F
1000 909B 18           CLC           ; ADD TO TENS
1010 909C 6593         ADC TRACKS
1020 909E 85A0         STA PASSP     ; AND WE HAVE TOTAL IN HEX
1030 90A0 206A2D       JSR CRLF      ; BACK TO MARGIN
1040 90A3 68           PLA           ; PULL RETURN ADDRESS OFF
1050 90A4 68           PLA                                  STACK
1060 90A5 60           RTS           ; RETURN TO BASIC
1070 90A6 88   TRAKCT  DEY           ; TRACK COUNTING ROUTINE
1080 90A7 B190         LDA (MEM),Y   ; GET STARTING TRACK NO.
1090 90A9 8593         STA TRACKS    ; SAVE TEMPORARILY
1100 90AB C8           INY
1110 90AC F8           SED
1120 90AD B190         LDA (MEM),Y   ; GET ENDING TRACK NO.
1130 90AF 38           SEC           ; SUBTRACT STARTING NO.
1140 90B0 E593         SBC TRACKS
1150 90B2 E6A0         INC PASSP     ; BUMP TRACK COUNTER TO
1160 90B4 18           CLC           ; ADD THE START TRACK
1170 90B5 65A0         ADC PASSP     ; NOW ADD REST OF DIFFERENCE
1180 90B7 85A0         STA PASSP     ; TO KEEP RUNNING TOTAL
1190 90B9 D8           CLD
1200 90BA 60           RTS           ; FOR PASSING TO BASIC
```

```
ERRORS = 0000 <0000>
SYMBOL TABLE
SYMBOL   VALUE
BLANK    9063   BLK2     9066   CONVT    9054   COUNT    0092
CRLF     2D6A   DECMAL   9091   DONE     9062   EX2      908B
EXIT     9078   INCY     9050   MEM      0090   NEXT     902C
NYB2     9098   PASSP    00A0   SKIP     906F   START    901A
TRACKS   0093   TRAKCT   90A6   VID      2343
END OF ASSEMBLY
```

★ ★ ★

READER PROFILE

ED:

You asked that we let you know what we do and what sort of equipment/software we use. I own an OSI C3Sl bluebox with 48K RAM (one low-consumption board) with additional 8K for CP/M. Also, I have a CA-9 parallel interface which feeds into a Compulink (now known as Consolink) 62K Ram spooler, which translates the signal into Parallel (for an OKIDATA ML82A) or Serial RS232C for a C.Itoh Starwriter (25 cps daisywheel printer). I have a Hayes Chronograph and a Racal Vadic Modem (300-1200 baud) but my use of same is limited since my 520 board (CA-10X) gave up the ghost sometime ago and I need RS232C for both. If the need arose, I could use the spooler to feed the Modem but it would mean resetting some dip-switches; also, I would then not be able to use the printers.

Software-wise, I have OS-DMS, 65U, 65D, TMAKERII(CP/M), Dwo Quong's WP6502, DBPACK (CP/M), some CPMUG public domain stuff changed to OSI format for me, with more on the way, Configurable Business System (a data base package also CP/M); CBASICII; Fortran, Cobol, MBASIC, Forth (under 65D), a language that's still beyond my understanding; and PMATE (CP/M text editor which I haven't been able to figure out yet either).

Initially, I got into computing for fun, i.e., more precisely to learn programming.

I succeeded since I'm fairly good at BASIC, less good but learning in CBASIC and very poor (and wishing I could get the hang of it) in Assembly. Then there was a time that I did big namelists and labels for various organizations and actually made a little bit of money with it, but having a very active/ primary job with lots of overtime, the secondary job is going to pot, - so it's back to doing it for the fun and learning experience of it.

I am anticipating a complete change of career sometime in the future, where it may be necessary to leave computering, so I'm not pushing this activity too strenuously at the moment. The original OSI system has served me fairly well since 1979-1980. It saw one transformer replacement (CPU) and two disk drive replacement/repairs, some of which I accomplished myself. One cause of drive failure which occurred with some frequency is that the set-screw on the electro-magnet that controls the height of the head-load position (pressure of the pressure pad on disk) worked itself loose, and each time I adjusted that and put some acetate on it (to keep it in place) the drives worked a while longer. At one time I also replaced the circuit boards on the drives to change the physical (mechanical) B to A drive and vice-versa. These operations should not be attempted by the faint of heart, and I will admit that I really don't know how to repair drives, but I was lucky that it worked out OK.

Fred S. Schaeffer
Jamaica, NY 11435

PROGRAMMING TIPS FOR KEYBASIC

from: OSI

CHAIN COMMAND IN KEYBASIC

The CHAIN command has 3 primary functions:

1. Link from one KeyBASIC program to another preserving all variables or only those variables that have been declared in COMMON command.

2. Merge another program into the currently executing program.

   a) The program as stored on disk is unaltered, the resulting combination of the original calling program and the merged program is executed but not saved.

   b) Preservation of variables is optional as in straight link.

3. Delete lines from the currently executing program, freeing the memory formally used by the deleted code for MERGEing additional code or for freeing memory space for variable storage, etc..

If the optional MERGE parameter is used in the CHAIN command, then the DEF series of commands (DEF, FN, DEFDBL, DEFINT, DEFSNG, and DEFSTR) will still be in effect for the declared variables. If MERGE is not used, the DEF command must be used to redeclare the variables in the CHAINed program. In addition the program that is being merged must have been stored in ASCII format or a file mode error will occur on the CHAIN command file.

COMMON COMMAND:

The COMMON command is used to preserve selected (declared) variables for a CHAINed program when the ALL parameter is not used in the CHAIN command. All variables not declared in a COMMON are lost and the memory they occupied is freed.

   a) COMMON must be redeclared after each CHAIN.

   b) If a variable is declared COMMON, it must have had a variable passed to it in a CHAINed program or be assigned a value in the program in which it was declared or, upon execution of a CHAIN command, an FC error will result with the line reference to the COMMON statement.

The combination of CHAIN MERGE with DELETE and COMMON declarations yields great flexibility in use and recovery of the available memory under keyBASIC. Time spent in experimentation with the power of these commands will enable programmers to write much more complex programs without the lack of memory problems and/or garbage collection often associated with program size. MERGEd programs to delete lines of code no longer required beyond a certain point need only be one or two lines and will occur without the

operator even realizing it happened.

WARNING: WHEN USING CHAIN MERGE/DELETE, BE CAREFUL ABOUT SAVING PROGRAMS WITH CHANGES MADE DURING EXECUTION!!

DEF COMMANDS:

a) If a variable without a type declaration (%, !, #, $) is assigned prior to executing a DEF command, it will default to # (double precision) and future reference must be to V# for the value to be correct.

```
EX: 10 V=12
    20 DEFINT V
    30 PRINTV,V#
    result will be 0 12
    as V%=0, V#=12
```

If we now add:

```
40 V=6:PRINT V, V%, V#
result will be 6  6    12
```

Similar results are obtained with DEFDBL, DEFSNG, and DEFSTR.

LETTERS

ED:

I am writing to see if you or one of your readers has a solution to my problem.

I have a ClP-DF which I use to collect data from a laboratory instrument through the MODEM port, logical unit #1. I then print out the results on an Epson printer with serial interface which requires that I manually select the printer port, also logical unit #1. What I would like to do is install a Centronics type printer port, logical unit #4.

I obtained a Centronics type interface board from D&N Micro Products which plugged into the 610 board, but was unable to get it to work. I am using OS-65D V3.3 operating system software which appeared unable to select logical unit #4. A print #4 command would neither print anything nor cause the computer to stop. After conversing with the people at D&N Micro, I found that this interface would work with a ROM-based system, but they could offer no explanation as to why it wouldn't work with OS-65D V3.3, or V3.1 for that matter.

In talking with my OSI service person, the 620 board was mentioned, but he didn't know what it was used for, so he couldn't say whether or not it was a potential solution to the problem. On page 56 of "The ClP Users Manual" the 620 board is mentioned as required for the addition of the CA-20 board, which permits the addition of several other boards. The CA-9 board, however, is not listed among them.

I would also like the address of someone who can supply me with a copy of the OSI Assembler Editor and Extended Monitor Reference Manual. Any assistance that I receive will be deeply appreciated, and for it I thank you in advance.

Robert C. Vince
Marysville, MI 48040

Robert:

Not being intimately familiar with D&N's interface, it's hard to give an exact fix, however, it would appear that the board is not properly addressed. In other words, the computer is sending out text to a buss address where the interface is expected to be, but it isn't there. The folks at D&N are usually very helpful. Try them again.

The OSI disks that contain AE and EM should have been accompanied with the respective manuals, at least the older versions did. Your dealer should be able to get this for you or go to OSI. Either may require proof of ownership by presentation of an OSI disk containing AE and EM.

PEEK(65) Staff.

* * * * *

ED:

I have a few comments and questions on OS65-D. To add to the notes on disk systems (June '83), it is possible to disable the colon(:) and comma (,) as INPUT string terminators. However, the locations specified in the documentation are reversed. Location $0B9C is the colon and $0BA0 the comma. Location $26F8 also claims to be a colon terminator but changing it to $0D <return> causes the next disk access to crash the system.

In v3.3 there is a new command to dump the screen to an MX-80 printer. Unfortunately, it doesn't work on the original ClP since the command requires inverse character bits which are not present. Video memory pages $D001 and $D003 are printed black.

Now my questions. Are there some pokes which can force the INPUT statement to accept double quotes(") and leading blanks? Is there some information available on using the OS 65D kernel and I/O routines from a machine language program? Also, can the original C1P screen be reformatted to 12x48? And, are there any 64 or 80 column boards which might be adapted?

Frank Glandorf
Columbus, OH 43201

Frank:

Don't know about leading blanks, but (") should be acceptable so long as it is not the first character of the INPUTed string. The I/O kernel is fairly adequately covered in the 3.3 manual. William Barden's "How to Program Microcomputers" (available from PEEK(65) @ $8.95) is certainly a good starting point for ML programming.

PEEK staff

* * * * *

ED:

My system consists of the following: Superboard II, D&N Expansion Interface, 580 Backplane, 24MEM-CM9F by D&N, 1600 I/O card by D&N, ML82A Printer. This is a total of 8K ROM, 32K RAM 1 RS232 on Superboard, 2 RS232 Ports on 1600 I/O Board.

This is a cassette system and after four years I am ready to buy my first disk drive. Everything I have bought from D&N works very well, except the 2 serial ports on the 1600 card. D&N supplied the basic software patch to poke the ports off and on. Believe me it does not work.

After several expensive phone calls to D&N, still no success. D&N wants the board back to check it. I told them that I would see if some of the sharp programmers at Peek(65) (or readers) could help me before returning the board. After four years I have taught myself a little BASIC, but am at a loss with machine programming.

D&N says the location to POKE is 52992 but the short BASIC listing does not change anything at the OUTPUT on the card.

I also have a Dwo Quong Fok Lok Sow word processor which works very well.

Please help me if you can.

Larry G. Horst
Sharpsburg, MD 21782

Larry:

We don't have the answer to your question. Readers please help!

Al.

* * * * *

ED:

I finally decided to add a modem to my cassette-based C1PII and discovered that the modem routine supplied by OSI is not compatible with the 12 x 48 video swap routine (also supplied by OSI). I'm sure that you've answered this one before but I'm relatively new to PEEK. What gives?

Steve McGinnis
Ridgway, PA 15853

Steve:

The problem is that both routines reside at the same memory location, i.e., when you load the second, it over-writes the first. One or the other must be relocated, but that's another story we hope to get into later unless another PEEKer has already done it and drops us a line.

PEEK Staff

* * * * *

ED:

Help! Need hardware and software info to generate analog out/in signals for external control functions from my SB II. Anyone having this info or knowing whereabouts of such, please contact me.

Howard Bard
391 5th Ave.
Chula Vista, CA 92010

Howard:

See software listing this issue. There is a program and kit, look for BSR light controller.

PEEK Staff

* * * * *

ED:

In reference to "Superboard Secrets" in the August 83 issue, I have found that if you want to list a program from a certain line to the end, you only need to write

that line number followed by a dash.

Example: LIST 100-
As opposed to: LIST 100-200

It looks a little queer, but it saves time and frustration in looking for the last line number of the program.

Ralph Philbrick
Hamilton, VA 22068

SOFTWARE LISTING OCTOBER 1983

PEEK(65) inadvertently omitted to print the full addresses of some of the software listings in the October issue. Listed below are the complete names and addresses. Our apologies.

JOHN T. ROECKER
5141 THOMAS AVE. S.
MINNEAPOLIS, MN 55410

RUDY POLACEK
248 PURISIMA
WOODSIDE, CA 94062

JOHN S. FINE
289 ELM ST., #36
MEDFORD, MA 02155

SILEO COMPUTER SYSTEMS
381 SO. BROADWAY
DENVER, CO 80209

MIKE F. PUTNAM
1047 W. CO. RD., B
ROSEVILLE, MN 55113

STEVE DONACHIE
6811 SW 81 TERR.
MIAMI, FL 33143

HENRY KUSKA
289 MALDEN
AKRON, OH 44313

PATRICK CLUSMAN
40 W. MC WILLIAM ST.
FOND DU LAC, WI 54935

LEE CONYERS
2402 FREETOWN DR.
RESTON, VA 22091

STAN SADLER
14885 HATFIELD RD.
RITTMAN, OH 44270

RICHARD LIST
2104 VILLAGE DR.
PITTSBURGH, PA 15221

ARTHUR F. ROSS
5555 S. 152 ST., APT. 6
SEATTLE, WA 98188

RAY LYDON
20 EASTWOOD DR.
GRAFTON, OH 44044

JERRY E. TRAVIS
8533 PACIFIC HWY SE
OLYMPIA, WA 98503

LEON HAVERLY
74 DEBONAIRE DR.
MARIETTA, GA 30064

NORTHEAST FINANCIAL SYSTEMS
16 MAPLE AVE.
WEST NYACK, NY 10994

ROBERT T. KINTZ
104 COUNCIL ROCK AVE.
ROCHESTER, NY 14610

USER GROUPS NOTES

For the reader who asked about a user's club.

Ohio Superboard User Group
146 York Street
Nundah, 4012
Australia

Earl Morris, MI

ERRATA

August '83 PEEK(65), page 13. The Disk RPM Timer, Disk Utility Program contains an error. Line 210 should read:

DISK!"GO 4402":

NOT - DISK!"80 4402":

AD$

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.

* * * * *

FOR SALE: 527 board with all IC's installed or as many as desired of the 2114's at $1.70 ea. $140.00 with all IC's installed. Board is completely socketed. G. Van Horn, 640 S.W. Addison Av., Junction City, OR 97448.

* * * * *

C8P with 48K. Includes Anadex DP-8000 bi-directional printer and interface, many diskettes, joysticks. Well-maintained --- all records. Available immediately. $2600. 860-4915 Reston, VA. Evenings.

* * * * *

ADM-3A Lear Seigler CRT/Modem with acoustic coupler. Hardly used. Over $1300 invested. $800. Reston, VA 860-4915 evenings.

* * * * *

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.

* * * * *

32K C1P Series 2 Single Disk 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.

* * * * *

SALE: Demo OSI computers and parts.

Item	Disk	Cost	Demo Price
230E	10MB	$ 5,610	$3,500
C2D	7MB		2,500
3MHZ Exec +74			
	80MB	12,222	9,900

Memory Boards: 470,550,510,527 Make an offer. Call Zenerex at 612/854-1555, Zenerex Corporation, 1301 E. 79th St., Mpls, MN 55420.

* * * * *

MUST SELL: OSI C3C 48K dual floppies, 16 slot rack, 23 meg hard disc. Features: (2) Hazeltine 1520 CRT's (1) centronics 702 printer (3) CM3 RAM boards, CA9 parallel printer interface, real time clock, CM10 8K static board, CA10K 2 RS-232 ports, line filter, cooling pack, OSDMS, AMCAP upgrade 2-A, OSHDM, TMUM upgrade. System sold for over $22,000 new. Asking $7,500.00. Call Marty Judge at 800-441-9017.

EPROM POWER SUPPLY

12-5=26

Provides 26 volts from available +12 and -5 volts Sufficient to drive programmer for one Eprom

Bare board	**$2.50**
Bag of parts	**$3.50**
Post & Handling	**$3.00**

Md. residents add 5% tax. Send U.S. dollars (drawn on a U.S. Bank to:

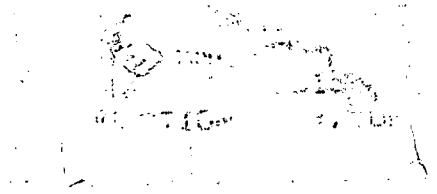
PEEK (65)
P.O. BOX 347
Owings Mills, Md. 21117
(301) 363-3268

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:



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 0S-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