PEEK (65)

The Unofficial OSI Users Journal

1819 Bay Ridge Ave., Suite 220
Annapolis, MD 21403

Editor: Al Peabody
Tech Editor: Dick McGuire
Contributing Editor: Corky Kirk
OSIO Editor: Wallace Kendall

Volume I, No. 8
August 15, 1980

Column One

A few minutes ago, Kathy Buck, who does most of the work in this office, came up to me and showed me a list of all the stuff you, our readers and writers, have submitted for this issue of PEEK(65). At the bottom of the list, she had written in, "want to go to 24 pages?" Since we have not yet pasted it all up as I write this, I don't know if it will take 24 pages or not, but my answer is easy. Hell, yes, I want to go to 24 pages. In fact, I want to go to 48 pages, and 5,000 subscribers, and slick paper, and 4 colors!

But, my thoughts continue, size and color and paper and subscribers are not really the most important things. The most important thing is quality of information. And in the area of information quality, the service we provide for OSI users, I believe we are already a giant. Who else will tell you how to time share serial terminals under level I? Where else can you read 3 different fixes for a garbage collection problem you may not even have? In what other publication does every single ad apply directly to your computer? (And your ad land only in the hands of those who might send you money?)

Which brings me back to my old familiar theme: we are in this together. We have received a couple of excellent suggestions this past month for the improvement of PEEK(65), and one offer of a column, devoted to OS-65U. Of course we will accept the offer if it turns out the offerer can write; and the suggestions have mostly already been incorporated. But we need more. We need a column on OS-65D, and a column on medium-sized personal computers, and column featuring hints and programs for the C1P user, and a hardware column, and more than we can fit in 24 pages. Perhaps you are the person to write one of those columns. Want to? It's easy. Just drop me a line and a sample column, and you are on.

SILICON CITY

Some people will argue about anything, and there are some who argue about whether computers have -- or ever will have -- "intelligence", whatever that is.

Let us simply accept the idea, move on to more profitable speculation, and visit Silicon City, where computers and people live in peace together.

As ever, though, "peace" does not necessarily mean convivial harmony or even cordial understanding. There is bigotry. There is prejudice. There are people who become suspicious when five or six computers communicate with one another in a closed circle, and there are computers who suppose people are all right, but wouldn't want their sister to interface with one.

There are occasions when the basis of value used by one group is simply not comprehended by the other.

Human, at a used equipment store: "Wait just a doggone minute! That's more than the list price new!"

Computer: "Of course. This has been running for a year, and you know it works. Any new equipment has recently been subjected to violent handling. How can you be sure it won't malfunction in the first month?"

Human: "But this may be worn out!"

Computer, loftily: "The only moving parts are electrons and maybe a few other itty-bitties. They've been running around in that metal since the Big Bang, and they stay on the right track unless some human fouls it up."

In Silicon City, as elsewhere, more computers work for people than people for computers. Many computers have menial jobs. Some of them work as nightwatchmen. A key element in the job of a nightwatchman is to understand the employer's instructions.

Human: "Got that?"

Computer: "Yes. I check the back door, then the front door, then the side door, then the garage door, then the front window, then . . ." Human: "Okay, okay, don't go all through that again."

Computer: "Right. If the voltage on any line is outside the prescribed limits, I telephone the police, then I telephone you over at that blonde's apartment . . ."

Human: "Never mind the details. Now, how many times do you make that series of checkups between 6 p.m. and 8:00 the next morning?"

Computer: "Using the inefficient decimal system, which for some obscure reason you seem to prefer, and assuming that no phone calls are necessary, I would check 321,432,480 times."

Over at the new equipment store we find humans just about as puzzled as at the second-hand store. Human, admiring a little number priced at $299: "What kind of engine does it have? I'm sorry, I was thinking about new cars for a moment. What kind of CPU?"

Computer: "The 6502, a very powerful chip."

Human: "Speaking of power, I wish I could afford this big, high-powered job with the $2799 price tag on it. That's a real beauty. What kind of engine - er, CPU - does it have?"

Computer: "The 6502, a very versatile chip."

Human: "But isn't that the same one that was in . . ."

Computer, stonily: "Why not?"

I suppose it was inevitable that I should hear charges of laziness. Computers really find it difficult to understand why humans dawdle so. One computer is assigned to a word processor, working with a human who has a reputation (among humans) as an exceptionally fast typist.

"I get an assignment to receive a character and store it. That takes a few microseconds -- maybe up to FF of them if there's a lot of back and forth. You know how long I have to wait for the next assignment? At least 15EFF microseconds. Those people get good pay, too. I tell you, it's just tragic the way they waste .111111111 of their time, or 99.9% in their strange notation."

The computers aren't the only ones who complain; people sometimes think computers insist on doing things the hard way. One example is printing a column of figures so that all the decimal points will be in line. The people think this is a trivial exercise, and have a tendency to be annoyed at the way the computers work out the instructions they give the printers.

The computers have a number of methods, but a common one is to find out what column the decimal is to be in, and when a number is entered, compute the natural logarithm of that number, convert this to the absolute value, compute the natural logarithm of 10, divide the absolute value of the first log by the second log, throw away the fractional part, subtract the quotient from the number of the column that is to contain the decimal point, start counting the columns on the paper from left to right, and start printing the number at the column bearing the number which resulted from this series of computations.

To a human, of course -- even to the rare one who could compute a logarithm in the first place -- this is an incredibly difficult and time-consuming exercise. Computers easily complete all the calculations before the next character comes down the line.

I really hadn't intended to let the discussion get acrimonious, but there is no way to avoid listening to the computers' complaints about humans.

"They simply have no feeling for precision," one computer complains.

"Nor for accuracy," says another.

Others chime in: "They change their minds all the time." "They give ambiguous instructions." "They can't remember what they told us to do." And so on.

And then, as the sun goes down and we take our leave of beautiful Silicon City, the final question: "Do you think humans really have, or ever will have, 'intelligence', whatever that is?"

OSIO

OSIO is a nonprofit, educational organization of OSI users. It is incorporated in the District of Columbia, but is international in scope. It encourages formation of local chapters and presentation of seminars, conducts an exchange of nonproprietary software, occasionally receives and passes along discount offers to members, and publishes a club newsletter. The OSIO Newsletter contains organizational information, consumer reviews and reports, and some short programs. Dues are $15 per year ($30 overseas, by airmail). See the PEEK(65)-OSIO combination offer on the back page.

LONG STRING INPUT FOR OSU SYSTEMS

James H. Sanders

The input buffer in OSU is limited to 72 characters, making several applications difficult to implement in Basic. I am regularly called upon for tasks involving input of very long strings and have developed software which makes life much easier.

Transfer of data from IBM floppys or 9-track tape usually involves blocks of characters either 128 or 256 bytes long, requiring both special input routines and appropriate handshaking hardware. Data entry is often made much easier for the user if the offline features of a smart terminal are used. In this case, 'protected' title fields are displayed on the screen and the data is entered in 'unprotected' areas. The editing capabilities of the terminal are used instead of writing a special editor for each format that comes along. When the record is correct, the 'SEND' key is pressed, transferring several hundred characters to the computer. With minor wiring changes to a 550 board to add ready lines (handshaking), several operators can key data to a single user system at the same time.

The example program is a greatly simplified version of the routines required for a complete package, but illustrates the technique. If interest is expressed, DBMS will offer a complete data entry package tailored to your smart terminal. Be sure to mention your terminal model.

This program listing begins with a display of the assembly program which is provided by data statements and poked in place by Basic. This technique is adequate for small utilities, but you would normally use "LOAD48" to merge the machine language with your Basic program.

The machine language program monitors the console ACIA at address 64512 until a character is ready, then loads it and stuffs it into memory beginning at a buffer address of 28160. Neither of these addresses is special, they must only be in free memory somewhere. When 100 characters (again, arbitrary) have been stored away, it returns to Basic. To easily access this block of characters, we point a dimensioned array (B$) to this block, and set the length and starting address of each element to the appropriate address within the buffer. The way this is done may have to be read twice....

Basic keeps track of dimensioned string variables by storing a 'descriptor array'. For each element in the array, three bytes are used. The first contains the length of the string, followed by the low 8-bit address, then the high 8-bits of the address of the first character of the string. Since we know where the strings are going to be, we can poke the desired length and address information into the descriptor array for B$. Most of the time we will want the data in some order other than the sequence it comes. We can either write out the array in the order desired, or else point the descriptor array to the desired segment. In this example the pointers are not in sequential order.

Locating the descriptor array is made easier by defining all of the non-dimensioned variables to be used at the start of the program, then defining B$. This will insure that the descriptors are loaded at the start of the space Basic reserved for them. Conveniently, this starting address is available at locations 124 and 125. The first 7 bytes are housekeeping stuff, followed by B$(0) so we begin our pokes ten bytes later. Note that the address 28160 is 0,110 when poked as a 16 bit value. The 6502-code is loaded at 0,108.

When you type in this program, it will be easier to leave out the comments. By all means, begin with line 110. Run the program and key in 100 characters. The reformatted results will be displayed and it will cycle for another block. If you make the first of the 100 characters an 'X', it will stop after the display.

```
3 GOTO 100
4 ---------------
5 HERE IS THE MACHINE LANGUAGE PROGRAM DESCRIBED BY THE DATA
6 STATEMENTS ON LINES 150-160 INTHE PROGRAM BELOW.
7 .
9 .  DECIMAL ASSEMBLY      PROGRAM         REMARKS
10 .  ---------------       ----------      ------------
11 .                        *=27648         STARTING ADDRESS
12 .    160    0            LDY #0          POINT BUFFER START
13 .    173    0  252  GETCH LDA 64512       GET ACIA READY FLAG
14 .    201    3             CMP #3          TEST IF CHAR READY
15 .    208    249           BNE GETCH       WAIT FOR IT
16 .    173    1   252       LDA 64513       GET CHARACTER
17 .    41     127           AND #$7F        MASK OFF PARITY BIT
18 .    145    80            STA (ADR),Y     STORE THE CHAR.
19 .    200                  INY             POINT NEXT BUFFER
20 .    192    100           CPY #100        GOT 100 CHAR YET
21 .    208    237           BNE GETCH       IF NOT, GETANOTHER
22 .    96                   RTS             ELSE RETURN TO BASIC
23 .    ----------------------------------------------------------
100 REM DECLARE ALL VARIABLES TO BE USED
110 I=0:A=0:ML=27648:AB=0:X=0
120 REM DEFINE THE OUTPUT ARRAY
130 DIM B$(5)
140 REM STUFF THE MACHINE LANGUAGE PROGRAM TO 27648
150 DATA 160,0,173,0,252,201,3,208,249,173,1,252,41
160 DATA 127,145,80,200,192,100,208,237,96,999
170 READ A : IF A=999 GOTO 200
180 POKE ML,A : ML=ML+1 : GOTO 170
190 REM PRESET THE B$ DESCRIPTOR ARRAY TO THE BUFFER
200 DATA 10,0,110   : REM B$(1) LEN=10, CHAR 1 TO 10
210 DATA 20,80,110 : REM B$(2) LEN=20, CHAR 81 TO 100
220 DATA 30,10,110 : REM B$(3) LEN=30, CHAR 11 TO 40
230 DATA 5,75,110   : REM B$(4) LEN=5,  CHAR 76 TO 80
240 DATA 35,40,110 : REM B$(5) LEN=35, CHAR 41 TO 75
250 REM GET THE DESCRIPTOR ADDRESS AND LOAD THE PRESETS
260 AB=PEEK(124)+256*PEEK(125)+10
270 FOR I=0 TO 14 : READ A :POKE AB+I,A : NEXT I
280 REM SET THE USR FUNCTION TO 27648 (LO/HI = 0/108)
290 POKE 8778,0: POKE 8779,108
300 REM SET 'ADR' FOR THE MACHINE LANGUAGE TO THE BUFFER START
310 POKE 80,0 : POKE 81,110
320 PRINT"READY"
330 REM ---- HERE WE GO ----
340 X=USR(X) : REM GET 100 CHAR FROM CONSOLE KEYBOARD
350 FOR I=1TO5 :PRINT TAB(10);B$(I): NEXT I: PRINT : REM DISPLAY IT
360 IF LEFT$(B$(1),1) <> "X" GOTO 340: REM CHECK FOR END
370 END
```

ED:

In response to Brian Fearnow's letter in the May issue, check out Sams #21587 The S100 and Other Micro Busses by Poe and Goodwin. They give enough specifics on the 500 board and the KIMSI interface to make the necessary circuitry for that type of scheme clear. As the S100 bus is highly 8080-Z80 oriented and CPU chips and boards are plummeting in price the smartest interface might be an S100 board with 1-4 K of 2114 static memory and circuitry to allow this block to be shared between the S100 and 500-600 system. This obviates the problem of incompatibility of signal timing of some 6502 signals.

Again on Brian Fearnow's letter: See the June 1980 Micro, page 42 where the following Basic locations are named in table 1:

BASIN	FFEB	218,219	FF 88
BASOUT	FFEE	21A,21B	67
CNTRL C	FFF1	21C,21D	99
BLOAD	FFF4	21E,21F	89
BSAVE	FFF7	220,220	94

As discussed in the article, these locations are loaded from ROM at initialization, and may be changed at will to point to your new routines. A Basic input statement looks at BASIN (FFEB) and indirects through 218,219 so any new input routine is called when its entry point is put in 218,219.

Robert Woolery
San Bernardino, CA

ED:

Murphy's garbage collection changes work! There is an obvious typo in line 100. It should read:

```
100 IF I<>A+57 THEN 120
```

[Note: goof made by PEEK(65) -AL]

This program moves a block of code and one subroutine from $B147-$B24C to high memory. This is the garbage collector. As the move is made, certain bytes are changed. Most of these are the calling address of the subroutine. There are three actual changes to the code. At $B189, LDA #$03 is changed to LDA #$04. At $B21E, AND #$04 is changed to AND #$02. Finally, at $B220, LSR A is changed to CLC.

Disk BASIC's garbage collector works all right. It is an altered and lengthened version of the ROM collector. Clearly the writer of Disk BASIC's garbage collector was not as clever as Murphy at fixing the problem.

Tom Berger
Coon Rapids, MN

ED:

Re: Stan Murphy's method of correcting the string handling problem:

I have included a listing of a somewhat more compact version of this program using data statements to specify the locations where internal addresses must be changed. After some searching through the first page of RAM I discovered that the function defined in line eight can be used in place of FRE(X). Thus, it is possible to check this function from time to time and call the modified garbage collector only as needed.

The routine I have tacked onto the end of this one is a BASIC keyboard polling routine. It doesn't produce funny characters like OSI's rather strange decoding routine does. The rubout key is functional and the characters are placed on the screen and in A$ at the top of string memory. (Note: A$ must not be changed.) The input string can be retrieved by setting B$ = LEFT$(A$,LE). The input string can be placed anywhere on the screen by simply specifying A$ in line 10002. The characters in DATA statements at lines 10021 and 10022 should be lower case. It is presently configured to accept 72 characters maximum, but this can be increased to 255 by increasing A$ and changing the limit in 10018. This routine has the advantage that string input from the keyboard can contain commas and colons that are normally interpreted as terminators by BASIC and scrolling can be suppressed if desired.

Robert Badger
Eau Claire, WI

THE MOST POWERFUL MICROCOMPUTER SOFTWARE EVER DEVELOPED IS ALSO THE EASIEST TO USE . . . .

MIMP MICROSYSTEMS INFORMATION MANAGEMENT PACKAGE & BUS-II GENERAL ACCOUNTING PROGRAM

FOR USE ON
OHIO SCIENTIFIC
MICROCOMPUTERS

MICROSYSTEMS INFORMATION MANAGEMENT PACKAGE

PUTS ANY OR ALL YOUR BUSINESS FILES INTO A POWERFUL DATABASE.

CREATE VIRTUALLY ANY RELATIONSHIP YOU CAN THINK OF.

ENGLISH-SPEAKING REPORT GENERATOR.

POWERFUL USER COMMANDS NORMALLY FOUND ONLY IN LARGE COMPUTERS. SELF-EXPLANATORY, EASY TO USE, SUCH AS "INPUT," "OUTPUT," "LIST," "PRINT," "CHANGE," "DELETE," "EXCLUDE," "INCLUDE," AND, PERHAPS MOST IMPORTANT OF ALL, "HELP!"

STRAIGHTFORWARD INPUT AND EDITING ROUTINES— "IDIOT-PROOF."

OPERATOR'S MANUAL WITH FULL USER INSTRUCTIONS IS ON-LINE WITH HELP COMMANDS. PROBLEMS? JUST TYPE "HELP!"

COMMANDS CAN BE USED CONDITIONALLY ON ANY RECORD OR ON ANY FIELD WITHIN A RECORD.

IN-MEMORY SORT, OR, (FOR BIG FILES) DISK SORT.

ALL DATABASE FUNCTIONS ARE SUBROUTINES— CUSTOM PROGRAMMING SIMPLIFIED.

OPTIONAL CASH REGISTER POLLING AND CONTROL PACKAGE AVAILABLE TO QUALIFIED BUYERS (REQUIRES INSTALLATION CONSULTATION).

OVER TWO YEARS' IN-FIELD TESTING. MIMP WORKS— FLAWLESSLY.

ONE YEAR SUBSCRIPTION TO APPLICATIONS NEWSLETTER FREE WITH PURCHASE.

TYPICAL APPLICATIONS: INVENTORY, CUSTOMER FILES, CASH REGISTER CONTROL (OPTIONAL EXTRA), DOCTOR/-DENTIST/LAWYER APPOINTMENTS OR CLIENT/PATIENT FILES, SALES RECORDS, MAILING LISTS.

INTRODUCTORY OFFER

$395

THE KEY TO THIS SOFTWARE IS ITS SIMPLICITY OF OPERATION.

BUS-II

BOOKKEEPING/ACCOUNTING SYSTEM

CPA-APPROVED COMPLETE ACCOUNTING SYSTEM: GENERAL LEDGER, ACCOUNTS PAYABLE, ACCOUNTS RECEIVABLE.

BRANCH ACCOUNTING HANDLES UP TO FIVE DIVISIONS.

COMPLETELY INTERACTIVE. SELF-UPDATING.

BATCH ACCOUNTING INSISTS ON BEING "IN BALANCE."

USER MENU. OPERATOR PROMPTING FOR CORRECT ENTRIES.

HIGHLY FLEXIBLE FORMATTING OF FINANCIAL REPORTS.

NO KNOWLEDGE OF DEBITS AND CREDITS REQUIRED (EXCEPT RECURRING JOURNAL ENTRIES— ASK YOUR ACCOUNTANT).

EXAMPLES OF REPORTS (FROM GENERAL LEDGER MODULE)

GENERAL LEDGER REPORTS

CHART OF ACCOUNTS	CASH RECEIPTS
CASH DISBURSEMENTS	BALANCE SHEET
STATEMENT OF INCOME	DAILY TRANSACTION LISTING
JOURNAL ENTRIES	DETAILED GENERAL LEDGER

ACCOUNTS PAYABLE REPORTS

CHECKS TO BE WRITTEN	CHECK REGISTER
ACCOUNTS PAYABLE	INVOICES NOW DUE
VENDOR LISTING	

ACCOUNTS RECEIVABLE REPORTS

CHECKS TO BE WRITTEN	MONTHLY TRANSACTION LISTING
CURRENT ACCOUNTS RECEIVABLE	DETAILED ACCOUNTS RECEIVABLE
PRINT STATEMENT	LISTING OF PAST DUE ACCOUNTS

IDEAL FOR ACCOUNTING FIRMS AS WELL AS SMALL BUSINESSES.

PROVEN THROUGH YEARS OF USE.

THIS ACCOUNTING PACKAGE REQUIRES ONLY A HIGH SCHOOL-LEVEL UNDERSTANDING OF BOOKKEEPING.

ONLY

$395

DUAL EIGHT-INCH FLOPPY DISK DRIVES REQUIRED.

MIMP & BUS-II OPERATE ON ALL OHIO SCIENTIFIC 8-INCH OR WINCHESTER DISK MICROCOMPUTERS W/48 K RAM.

USER-ORIENTED SYSTEMS — NO PROGRAMMING REQUIRED

CREDIT CARD ORDERS
CALL
714-270-2000

VISA master charge

SEE YOUR OSI DEALER FOR DEMONSTRATION

ALL ORDERS SHIPPED PREPAID WITHIN CONTINENTAL U.S. PLEASE DO NOT SEND CASH.
OVERSEAS ORDERS PLEASE INCLUDE POSTAGE— 5 LBS. SORRY, NO C.O.D.

digital technology inc.
P.O. Box 178590, San Diego, CA 92117

CHEAPIE SUPERBOARD II EXPANSION

by Bruce Showalter

The slogan "You've come a long way, baby" can well be applied to the personal computer. I watched with fascination as solid-state technology crossed one frontier after another through the 1970's. My excitement peaked when Ohio Scientific announced the Superboard II in the fall of '78. Dollar for dollar, it contains more human-level data processing power than any other single board computer I know.

However, it is limited. The economics simply did not permit a large video display capability. That, I promised myself, will be my next research project. Meanwhile, there is the issue of expansion. OSI sells an expansion board for (get this!) MORE than the price of the computer itself. It's worth the money, if one wants a disc interface and a real-time clock as well as the space for 24K more RAM. But if all one has in mind is additional memory, $299 is the hard way to go . . . especially when the actual increase is only 4K! (Additional RAM chips sold separately.)

So, I have come up with a low-cost RAM expansion for the S. II. It requires a new chassis (cheaper than a comparable mother board) and an interface from the S. II's J1 socket to the KIM-4 bus (the simplest and most popular 6502 expansion bus). To expand the memory, all one needs are 16 address lines, 8 data lines, and R/W. My interface board also has room to bring in the 02, READY, and SYNC lines if needed.

At this point, let me reassure prospective S. II buyers. The documentation that comes with the computer is adequate for expansion research. However, schematics of the 600 and 610 boards ($7.95 from OSI dealers) are more helpful...especially in decoding the RAM addresses.

Let's look at the hardware required for the expansion. I chose an aluminum chassis, 5x7x2, in which I've mounted Vector R644 sockets (Radio Shack 276-1551). These should be mounted 1 1/4" apart, center-to-center, to allow room for the wire-wrap pins and the components on each board (see fig. 1). In the first socket, I've plugged my interface board (Radio Shack 276-153 or -154). Since I have used wire-wrap connections, I removed the copper foil traces to avoid possible short circuits. This is easily accomplished by applying a hot soldering iron/gun to the foil. The foil comes "unstuck" when it gets hot enough. Vector sells blank boards (3662-5), but they cost twice as much. As it turns out, though, the Vector boards are required for the RAM chips. The Radio Shack boards are only 4" wide and do not have enough holes to mount 8K of 2114's. (One could settle for 6K per board, if the price or availability of larger boards was too great a hindrance.)

Incidentally, I used the new OK Just Wrap(tm) tool. I recommend buying the kit, since one gets three more rolls of wire (an $8.94 value) and an unwrapping tool (worth $3.49) for only $10 more than the tool itself. (Radio Shack is selling the tool with only one roll of wire for $2 less than OK.) The tool is simple and easy to use, and the connections are good every time. The only problem I had was with the CUT-WRAP slide. It fits pretty loose in its slot, so I glued some paper onto the side of the slide to help it stay in the WRAP position. (Frustration is making a series of daisy-chain wraps and having the slide fall into the CUT position about half-way through.) It sure beats etching, drilling and soldering for saving time.

To get from the IC sockets to the edge connectors, I used Vector T44 pins. I inserted these into the holes connected to the desired edge connectors and soldered them. When I finished, all the pins are sticking out from the "alpha" side of the board, and the sockets are on the "number" side (see fig. 2). Which is which is up to the builder. Just follow the markings on the R644 sockets.

One important factor in the case of etching - drilling - soldering vs. wire-wrapping is the higher cost of IC sockets. Some comparison shopping of mail order ads shows that the 16-pin sockets have the lowest cost/pin. And Quest Electronics has the lowest cost/socket. To get other sizes of sockets, I simply cut up some 16-pin versions and glue them to the board. A razor saw is required for this operation. I mount the socket, pins

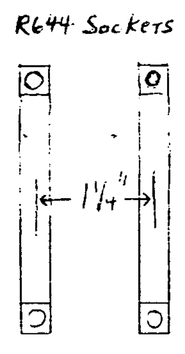


Figure 1

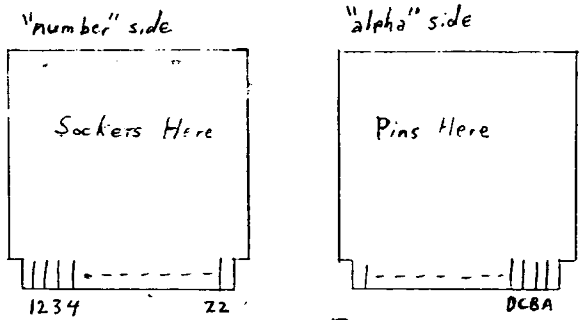


Figure 2

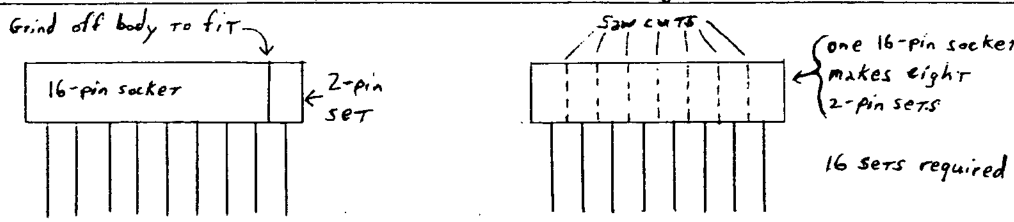


Figure 3

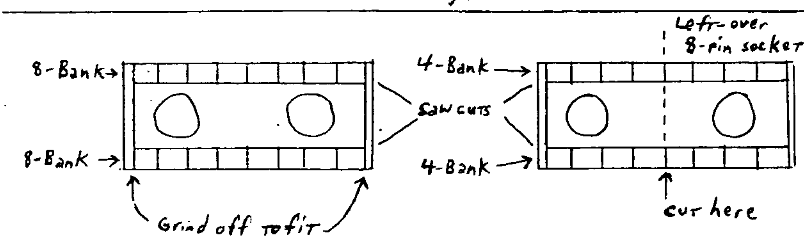
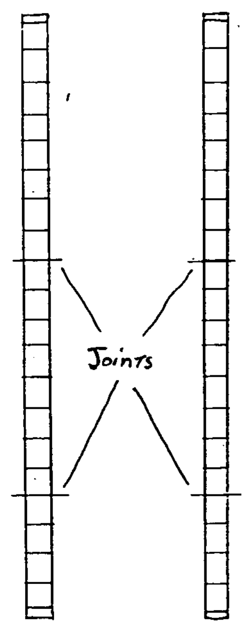
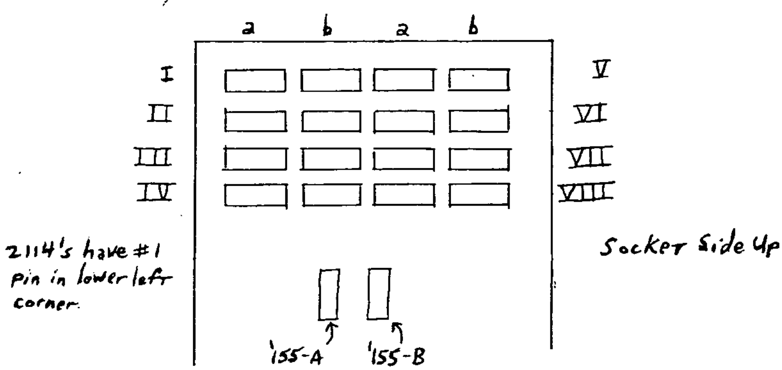


Figure 4





74155's have #1 pin in lower right corner.

Figure 5

down, in a 2" cube of styrofoam to hold it while I'm cutting. For example, to make the 18-pin sockets for the 2114's, I cut up two 16-pin sockets into 2-pin sets. Then I glued a 2-pin set behind a 16-pin socket to make one 18-pin socket. Each 16-pin socket will require shortening of the body on one end to permit the 2-pin set to fit 1/10" away. I accomplished this with a grindstone (see fig. 3). CAUTION: this process can expose pin connections on the cut ends of the socket pieces. BE SURE that exposed connections do not touch one another.

To fasten the sockets to the board, I used Dow Corning(tm) Urethane Bond. It's as strong as epoxy but requires no mixing. Using this cut-up-socket-and-glue technique, I also fashioned a 40-pin socket to accept the ribbon cable which connects the S. II to my interface board. What I did was to cut up two-and-a-half 16-pin sockets into "banks": four 8-pin banks and two 4-pin banks (see fig. 4). This left me with one 8-pin socket, useful for a 555 or whatever, on some other project.

Now we come to the actual circuitry. From the S. II's documentation, we find out the following 40-pin expansion plug pin assignments:

Pin #	Function
4	D0
5	D1
6	D2
7	D3
12	A2
13	A1
14	A0
15	A3
16	A4
17	A5
18	A6
19	A7
20	A8
21	A9
22	A10
23	A11
24	A12
25	A13
26	A14
27	A15
32	R/$\overline{W}$
33	D7
34	D6
35	D5
36	D4

Since my expansion chassis has its own power supply, there is no need to bring the +5 volts or GND connections out. However, both the S. II and the expansion chassis should have a common ground. Keep in mind that all the 6502 pins, except S.O., SYNC, RESET, 01 and READY are available from the S. II 40-pin socket. With a little tinkering, these could be included as well.

OSI located sockets on the S. II for 8T28 buffers for the data lines. So I have not buffered these lines on my interface board. I have buffered the address lines and R/W. This I do with two 81LS95's and a 7408. Digi-Key Corp. and Jade Computer Products are the only firms I know that carry the 81LS95's. I chose those chips because they take less board space by buffering 8 lines instead of just 6. But, they do cost more per chip and require 20-pin sockets.

In order to enable the 8T28's on the S. II, I had to include a decoder circuit on the interface board. This decoder enables the data lines to the expansion chassis whenever the first 8K of memory is exceeded, up to the limit of the RAM space on the memory map. The decoder is a 7445 open-collector 4-to-10 decoder, in wired-OR configuration. In turn, it gates the inverted R/W line to DD (pin 3 of the 40-pin socket) via a 7402. R9 and R74 on the S. II should be disconnected, since DD is not an open-collector signal. (The DD signal from the 610 board is open-collector.) On the next page are the pin connections for wiring the interface board (chip locations are the builder's choice).

On the 7402, connect pin 1 to pin 12, and pin 3 to pin 4. Connect a 2KΩ resistor in series between pin 5 on the 7445 and edge connector 21-Y.

This completes the wiring of the interface board. Insert the IC's, connect the ribbon cable, and plug it into the chassis.

Let me take a few lines to pass on some hints about designing the parts and wiring layout. To see if I had room for my sockets, I used graph paper with 10 squares/inch. This means that each line is spaced exactly like a row of holes on a perf board. However, I can only get this

40-pin Socket To →	81LS95-A	81LS95-B	7408	Edge Connector
4				15
5				14
6				13
7				12
12		12		
13		14		
14		16		
15		18		
16		2		
17		4		
18		6		
19		8		
20	8			
21	6			
22	4			
23	2			
24	18			
25	16			
26	14			
27	12			
32			1	
33				8
34				9
35				10
35				11

Edge Connector To →	81LS95's	7408	7445	7402
21-Y	20	2, 5, 9, 10, 12, 13, 14	16	14
1-A or 22-Z	1, 10	7	8, 12	6, 7, 8, 9, 11

Edge Connector To →	81LS95-A	81LS95-B	7408
B		15	
C		13	
D		11	
E		17	
F		3	
G/H		5	
J		7	
K		9	
L	9		
M	7		
N	5		
P	3		
R	17		
S	15		
T	13		
U	11		
W			3

7402 To →	7445	7408	40-pin Socket
2	2, 3, 4, 5		
5		3	
13			3

81LS95-A To →	7445
11	13
13	14
15	15

paper from an office supply store, which makes it expensive. So I've converted to quadrille paper which has 5 squares/inch. This makes my drawings 2:1 scale. But it's actually better, because it's easier on the eyes. And it's available in most grocery and discount stores. Colored pencils (map colors) are handy for distinguishing power lines, ground lines, control lines, address and data lines. I only wish OK offered more colors of wire for my JW-1. Another neat trick is to lay a piece of carbon paper, carbon side up, underneath the page I'm drawing. This allows me to "draw" the back side of the page at the same time I draw the front. I use this technique when I layout the final socket placement. Then, when I turn the page over, I can spot the #1 pin on each socket to begin drawing my wire connections.

Here's a tip for identifying pins after the sockets are glued to the board. I put a red mark (indelible ink, nail polish, paint) alongside the #1 pin on the wiring side. That way, I don't have to keep flipping the board over to remember which pin is which.

Making the RAM board uses the techniques described so far. The only worry I had was how to decode the address lines for 8K immediately above the existing 8K on the S. II. Trying to find the location on the memory map was confusing, especially when I tried to convert from the hex listing given. This is where that $7.95 OSI schematic paid for itself. It is printed by the Sams Photofact people, and it is <u>excellent</u>. Their memory map is easier to understand than the one that comes with the S. II. And, the 610 expansion board schematic is included. To decode the next 8K, OSI used a pair of 74LS138's. I wanted more drive capability, so I used 74155's. Both perform the same operation: giving one of eight outputs for a specific three-input combination. Only the pinouts are different for the '138 vs. the '155. Later, I realized I could have used 7422's for less money.

The OSI schematic showed me something interesting. The address, chip select, R/W and data lines are buffered only once for all 48 chips! Well, if it's good enough for the pros, it's good enough for me. Actually, the current requirement of 2114 inputs is about 100 times less than that of TTL (10 ua. vs. 1 ma.). As a result, the first 10 address lines, the data lines, and R/W all go direct from the edge connectors to the RAM chips.

A13-15 are decoded by '155-A which enables '155-B. Eight chip select lines come from '155-B which decodes A10-12. Here is the wiring table:

Edge Connector To→	2114's	74155-A	74155-B
1-A or 22-Z	9	14, 8, 2	8
8	11-b		
9	12-b		
10	13-b		
11	14-b		
12	11-a		
13	12-a		
14	13-a		
15	14-a		
21-Y	18	16	16
B	5		
C	6		
D	7		
E	4		
F	3		
G/H	2		
J	1		
K	17		
L	16		
M	15		
N			13
P			3
R			1, 15
S		13	
T		3	
U		1, 15	
W	10		

Notice that I specify "a" and "b" pins on the 2114's. This is because each 8-bit word is split into two 4-bit syllables (nibbles, actually) and stored in a pair of RAM chips. So, on a-b pair store 1,024 8-bit words. Logically, the 2114's are arranged in pairs on the board. Figure 5 shows the arrangement I chose.

The wiring for the address decoders follows:

74155-A	74155-B
10	2, 14

74155-B	2114 Pair, Pin #8
9	I
10	II
11	III
12	IV
7	V
6	VI
5	VII
4	VIII

Well, so much for expansion. Remember, the choice of building I/O ports, D-A interfaces, or whatever is always there via the complete KIM-4 bus. I can install additional R644 sockets and plug in boards as needed. Any 6502-compatible hardware should be adaptable to the S. II this way. I figure the cost of this 8K expansion is less than $125, if you don't count the cost of the wire-wrap tool or the chassis power supply.

AD$

C4MF and C8DF SOFTWARE

Screen Editor-Machine language program that has 11 one key commands that gives the user full cursor control to Delete/Insert characters anywhere in your basic program.
RESEQ-The user inputs which lines to be renumbered and the new starting number and increment. The program properly changes all GOSUB'S, GOTO's, THEN and RUN commands.
COPY 5-Copys up to 5 full tracks in one transfer using a single disk drive (24 K C4MF only).
NAME-This program will name an unnamed program without erasing it. If the input name already exists then the user has the options to Rename, Delete, Input a New Name, Print the free tracks or Quit.
Blockade-Trap game that uses the Dominoes principle. You can play an opponent, the computer or let the computer play against itself at any of the 10 game speeds.
Checkers-Play the computer using the non-moving screen board.
DEEPSUB-Shoot fast and slow moving ships from your submarine.
Pong-The first player to 21 by 2 wins.
Clock-Put idle computer time to work with this large number color alarm clock.
Letter-Get your message across by scrolling large letters in color.
All games have color and sound and use Joysticks or Keyboard. Your choice of 5 programs for $30 then each additional program is only $5 more.
Send check to: Rodney Trugman, 1504 Chestnut St., Bowling Green, KY 42101

BASXR OS65U compatible. Helps with debugging and modification. Lists all variables and/or commands and their line number locations. Locates specific lines on entry of decimal value of Basic commands. Complete program listing and documentation. $45, or send your OS65U disk and we will transfer for $5 extra. No COD's.
EIS, Inc., P.O. Box 5893, Athens, GA 30604

8K-BASIC PROGRAMS ON CASSETTE FOR C1P,-SBII, C4P-High quality programs on quality cassettes with complete documentation. For a complete listing of programs, send SASE to: Universal Software Associates, P.O. Box 308, Syosset, NY 11791, ATTN: Catalog Dept.

SERIOUS SOFTWARE

Smart Terminal Emulator for C1P, C4P, C8P ROM BASIC. Talk full-duplex to other computers, CBBS's or time-shared systems. Transfer files and programs in either direction with a single keystroke. $20 for cassette and complete instructions including implementing RS-232 port! MUST SPECIFY MODEL AND RAM SIZE.
Itel-Dura-Selectric Printer Interface Use KIM as intelligent BUFFERED interface between any serial ASCII computer port and parallel-input Selectrics. No hardware construction. Send SASE for details, $20 for source listing, KIM cassette and complete instructions.
POPENOE ASSOCIATES, 6307 Wiscasset Rd., Bethesda, MD 20016

OSI-65D C1P-SINGLE DISK COPY PROGRAM - Copies 7 tracks at a time on a 20K system. Including 5" disk $20.
Norman E. Bollinger, 3223 Suffolk Lane, Falston, MD 21047

FOR SALE- OSI 6500 Assembler/Editor cassette version. Tape and manual. Never used. $25.00. Jerry Ryan, 9126 Hemphill Dr., Ft. Wayne, IN 46819

FIRST TIME OFFERED!!!

OHIO SCIENTIFIC TEE-SHIRTS

BLACK TEE-SHIRT PRINTED WITH "OHIO SCIENTIFIC" IN VARIED BLUE TONES TO WHITE.

BE THE FIRST! ! ! ! ! ! ! !

ONLY $6.98
+.75 postage
calif. residents add 6% tax

STATE SIZE: sm,med,lg,xlg

COMPUTER BUSINESS SERVICE
P.O. BOX 20384
SAN JOSE, CALIF. 95160

Dealer inquiries invited
club discounts available

GOODIES for OSI Users!

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

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

[ ] OSI's Small Systems Journals. The complete set, July 1977 through April 1978, bound and reproduced by PEEK(65). Full set only $10.00 $________

[ ] RESEQ 5.2 -- BASIC program resequencer plus much more. Global changes, tables of bad references, GOSUBs & GOTOs, variables by line number, resequences parts of programs or entire programs, handles line 50000 trap. Best debug tool I've seen. MACHINE LANGUAGE -- VERY FAST! Requires 65U. Manual & samples only, $5. Everything for $50.00 $________

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

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

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

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

TOTAL $________

Maryland Residents add 5% Tax $________

[ ] Cash enclosed. TOTAL DUE $________

[ ] Bill my ____ Master Charge/____Visa Account No ________________

Expiration date ____________ Signature ________________

Name ________________________________

Street ________________________________

City ____________________ State ______ Zip ____________

master charge THE INTERBANK CARD

DBMS, INC.
1819 Bay Ridge Ave., Suite 220
Annapolis, MD 21403

VISA

LETTERS

ED:

What is the type of connector used between the backplane & boards on OSI buss? I'm in the process of building a 527 board & even the SAM's books don't list a reference for the edge connector. I would guess it's a 'Molex' but am not sure.

What are the 'controls' for the AC interface or how do you access that port? My C2-4P came thru with the AC port but nothing mentioned in documentation.

As a comment, I like PEEK(65) & think it is growing nicely. As a user of a 'Basic-in ROM' machine (someday I'll have a disc) what would be the possibility of maybe next year expanding into running a series on how to upgrade the cheaper machines. I mean like tutorials on adding RS232 capabilities, adding a 2nd cassette, machine controlled, for data or other in that line of goodies.

Neil Dennis
Bliss, NY

Neil:

We would love to publish that sort of "goodie". But first someone has to do it, then second submit it to us for publication. We might even pay a few bucks for a really good article!

AL

ED:

I know that many Apple users use "Bulletin Board" type communications via Acoustic Modem, well all I want to know is if there is a "Bulletin Board" system here in Los Angeles (or some place in California) for the OSI user of a C4P MF. And if there isn't, is it possible I could get one started somehow?

William F. Collier
Agoura, CA

William:

If you have a modem, write a program or two and send us an announcement--we will pass it on.

AL

ED:

Thank you for sending me the May 15, 1980 issue of PEEK(65). I have a lot of questions. I just received a C8P-DF--a really nice machine. It came with a Real Time Clock, however, OSI refuses to tell how I may use it directly. Apparently they want you to buy their Real Time Control Software and Peripherals in order to use it. Surely there are certain POKES to activate my clock!?

Does anyone have a <u>complete</u> description of the OSI altered Microsoft BASIC that came with my C8P-DF?? The manual is deficient on so many things. I would like to know what my BASIC can do.

Does anyone have a BASIC (or FORTRAN) compiler for the C8P-DF? Also is there a CP/M II or equivalent for the C8P-DF? Where do I find software written specifically for the C8P-DF? (How about UNIX version 7?)

Does anyone know how to get any possible volcanic ash out of one's disk drives--particularly off of the read/write heads?

Eugene C. Zinter
Boring, OR

Eugene:

Very carefully, with a vacuum cleaner, lint-free cloth, alcohol, Q-tips--but thoroughly! Any floppies with ash should be copied, then discarded--don't sand down your heads to save $5. For more info on OSI BASIC, see <u>OSI BASIC IN ROM</u> by Edward Carlson, 3872 Raleigh Dr., Okemos, MI 48864, and <u>THE FIRST BOOK OF OSI</u> by Jim Williams and George Dorner, available from AARDVARK, 1690 Bolton, Walled Lake, MI 48088. Both books are reviewed in the May 1980 issue of PEEK(65).

AL

ED:

I have a C1P-MF and am especially interested in information on OS-65D. Unfortunately the majority of stuff in PEEK(65) is devoted to Rom Basic CP/M or OS-65U. I guess what I'm really looking for is a comprehensive tutorial style reference work on OS-65D.

Is there any possibility that OS-65U could be modified to run on the C1PMF? How about CP/M?

Ben T. McGuire
Dickinson, TX

ED:

This CKT will give most any Baud rate needed. It can be wire wrapped onto the 600 board with no trouble. The dip switch can be mounted on the front panel. By setting the switches for 600 baud and adjusting R57 you can tape programs at 600 baud.

By putting a 7404 in blank socket U 68 the RTS line can be used. Poke 61440,64 takes the line low. Poke 61440,128 takes the line high. This gives you a program controlled TTL output to the real world.

Terry Shingara
Harrisburg, PA

ED:

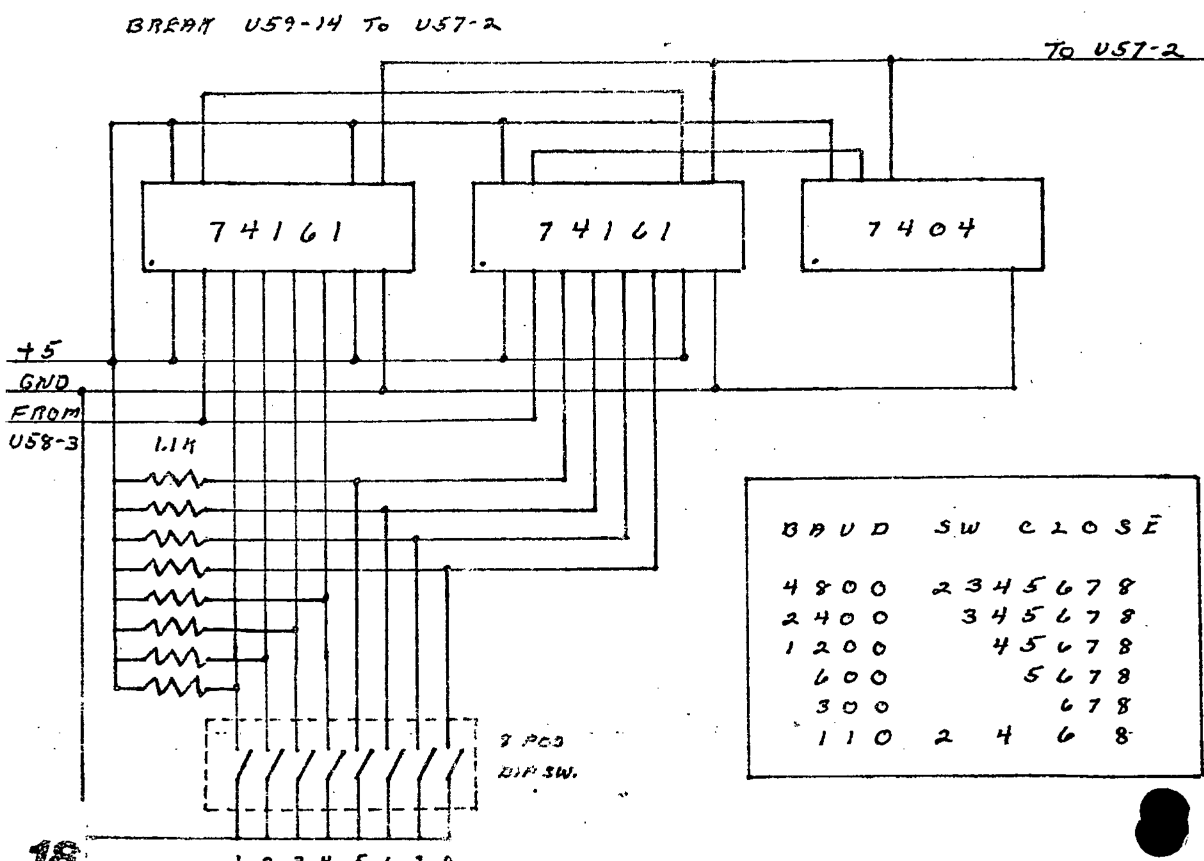
In case you're interested...I am more fortunate than most people in that I am able to write this letter using the text editor of the UNIX operating system (on a PDP 11/70) via my modem. I have written a smart-terminal program for my micro and as a result, have the ability to develop programs on this system and then download them to my OSI. Because of this I hope to be able to develop a small-c compiler for the C4P (now in progress) and am hoping to do the same with a PASCAL compiler. I have also written an improved keyboard scan program for the C4P, which I will submit to PEEK(65).

Phillip Dykstra
Aberdeen, MD

Phillip:

Please do share these programs with us!

AL



BAUD	SW CLOSE
4800	2 3 4 5 6 7 8
2400	3 4 5 6 7 8
1200	4 5 6 7 8
600	5 6 7 8
300	6 7 8
110	2 4 6 8

```
1 L=PEEK(134)*256+PEEK(133)-262:GH=INT(L/256):GL=L-256*GH
2 POKE11,GL:POKE133,GL:POKE12,GH:POKE134,GH
3 FORI=0TO261:M=PEEK(I+45383):POKEI+L,M:NEXTI
4 POKEL+67,4:POKEL+216,2:POKEL+217,24:FORI=1TO5:READAD,M:M=M+L
5 AD=AD+L:POKEAD,INT(M/256):POKEAD-1,M-INT(M/256)*256:NEXTI
6 DATA59,140,34,146,84,209,137,146,261,4
7 PRINT"GARBAGE COLLECTOR LOCATED AT"L"GH"GH"GL"GL
8 CLEAR:DEFFNF(I)=PEEK(129)+PEEK(130)*256-PEEK(127)-PEEK(128)*256
9 PRINTFNF(I)"BYTES FREE"
10 A$="AAAAAAAAAAAAAAAAAA":A$=A$+A$+A$+A$:X=USR(X)
11 SA=PEEK(129)+PEEK(130)*256-1:FORI=1TO5:READAD,M:NEXTI
10000 DIM CH$(98):FOR I=1 TO 7:RO(I)=2↑I:NEXT I
10001 FOR I=1 TO 98:READ CH$(I):NEXT I:CH$(30)="/":CH$(97)=CHR$(34)
10002 POKE530,1:LE=0:AD=57088:AS=55040;POKEAS+1,232
10003 FORI=1TO7:POKEAD,RO(I):M=PEEK(AD):IFM<>0GOTO10005
10004 NEXT I:T=0:GOTO 10003
10005 IF M=8 AND I=5 THEN POKE 530,0:POKE AS+LE+1,32:GOTO10020
10006 IF M=16 AND I=5 GOTO 10003
10007 IFM=4ANDI=6THENPOKEAS+LE1,32:POKEAS+LE,232:GOTO10015
10008 POKE AD,1:CA=PEEK(AD)
10009 IF CA<>1 OR CA>1 GOTO 10011
10010 IFI>5OR(I=1AND(M=4ORM=8))OR((I=2ORI=5)AND(M=2ORM=128))THENCA=0
10011 IFCA>16THENCA=30:GOTO10018
10012 IF CA>0 THEN CA=49
10013 FOR Z=1 TO 10:M=M/2:IF M<1 GOTO 10017
10014 NEXTZ
10015 IF LE>1 THEN LE=LE-1:GOTO10003
10016 LE=0:POKEAS,32: POKE AS+1,232:GOTO10003
10017 CA=CA+(I-1)*7+Z-1:IFCA=0ANDT=1GOTO10003
10018 O=CA:T=1:LE=LE+1:POKEAS+LE,ASC(CH$(O)):IFLE=73GOTO10026
10019 POKE AS+LE+1,232:POKE SA+LE,ASC(CH$(O)):GOTO 10003
10020 RETURN
10021 DATA P,;,/," ",Z,A,Q,",",M,N,B,V,C,X,K,J,H,G,F,D,S,I,U,Y
10022 DATA T,R,E,W,,,,,O,L,.,,,-,":","0","9","8","7","6"
10023 DATA "5","4","3","2","1"
10024 DATA P,+,?," ",Z,A,Q,<,M,N,B,V,C,X,K,J,H,G,F,D,S
10025 DATA I,U,Y,T,R,E,W,,,,,O,L,>,,,=,*,↑,),(,',&,%,$,#,,!
10026 POKEAS+LE,32:LE72:PRINT:PRINT"72 CHARACTERS MAX":GOTO10020
OK
```

ED:

A few notes in response to some of the letters in the June issue of PEEK(65).

Bill Devinney asks why only 72 character input strings, and can they be increased. There is no practical way to increase this limit since the basic input buffer resides in page zero and the adjacent memory is used. Since the many routines which access the buffer are all using page zero addressing, there is no simple way to relocate the buffer.

Under OS-65U there is, however, a very simple way to shorten this limit. Just poke the desired length into location 1398 decimal. This is very useful when you wish to limit an entry to a given length, i.e. customer number cannot exceed 6 characters. Just be sure and poke back the 71 when you are through, and do not poke any higher value in this location.

Bill also asks for a way to simulate the "GET" command. Later in the same issue, Ian Morton gives an excellent method for a BASIC-IN-ROM machine. The following subroutine for a terminal based system running OS-65U also simulates the "GET" command. The routine will work under multi-user as well as single user.

```
REM...input single byte...
T=64512:IFPEEK(14948)=76THENT=52736+PEEK
(253)*2
WAIT0+T,1:T=PEEK(T+1)AND127:T$=CHR$(T):
RETURN
```

Terry Pukula asks how to interface a punched card reader and IBM tape drive to his system. Documation makes a card reader/punch with an RS-232 interface. It should be a simple matter to interface one of these to an OSI system. As to the tape drive, Alloy Engineering is presently designing a controller to interface to any tape drive using the Pertec standard interface. This should be available within the next 60 to 90 days.

In response to Tom Westhoff's letter, the memory locations from 23552 to 24560 decimal were originally used by OSI as a trap area for testing purposes. However, if you intend to put your own code in this area, be advised that this area is used by OSI for the editor, resequencer, and the new networking system.

In regards to Tom's question about a "PRINT USING" command, we have available a Terminal I/O Extensions package which includes a numeric masking routine which we consider more useful than the standard "PRINT USING". Many other business oriented features are also included. This package is only available from OSI dealers. Since only a relatively small number of dealers have the package available for sale, possibly DBMS might be persuaded to acquire it and, therefore, make it available to all PEEK(65) readers.

J. Larry Hinsley
Software Consultants
7053 Rose Trail
Memphis, TN 38134

ED:

I've got an urgent need for any info on a "Graphics Tablet" similar to the apple's or a digitizer and subsequent interfacing requirements for a C3 machine. We need to store architectural floor plans on disk and be able to edit them at will, plus compute certain engineering equations from this information.

Also I would like to know if anyone has interfaced a video camera to record above info.(Just the basic floor plan, nothing fancy graphically.)

Really glad you guys are around. You've helped me a lot. If I figure the above out, will write to you about it.

Dale Nimmo
Fessenden Computer Service
k, MO

ED:

Is there a Volume 1, No. 5? I have #4 (May) and #6 (June). [No.]

I've noticed a lot of unanswered questions in past issues and thought I would try to answer those that I know something about.

In the unlabeled issue (#2A?) - [#2], B. Showalter asked about the 'video RAM scroller' and the REPEAT, RUBOUT, and ESC keys. The scrolling routine is an integral part of the video output routine at $BF2D-BFFF. If he will send me an SASE, I'll send back an annotated disassembly listing. The named keys aren't used by OSI (I don't know why), but they use auto-repeat, shift-O, and shift-P instead. The keys are available for the user, though. [In ROM BASIC, 65D, 65U, all three--? AL]

D. Hille had a program to 'shuffle' a deck of cards. I've included the subroutine I use, simulating random permutation without replacement (or, in plain English, taking a member of a scrambled list without putting it back).

In #3, J. Leahy wondered about 256x256 graphics. Sorry, not on the C1/C2; the video section was designed so the user could access characters, not dots. He could burn a new character generator ROM with his own patterns, or maybe design a circuit that used RAM for part of the character generator.

E. Anderson wrote about machine language message routine. One exists in ROM at $A8C3 and was mentioned in MICRO (Nov. 79). The accumulator should have the low byte of the message address and the Y reg the high byte. The message must have a zero byte at the end.

J. Eddington asked about location 57088. This is the scanned keyboard port for the C1/C2 (and maybe the C4). Instructions on its use are in the Graphics Reference Manual.

D. Hille wanted to know how to pass value from BASIC to the USR function and back. The BASIC-IN-ROM reference manual has the information. To obtain the value passed to USR, call the routine pointed to by locations 6 & 7; the value is returned at

AE/F. To return a value, load the accumulator and Y reg with the value and call the routine pointed to by 8 & 9. The manual has an error (at least one!): The USR address should be poked into B & C, not 23E/F (for the C1).

In #6 B. Devinney had two questions about input. BASIC allows only 72 characters because that's all that OSI allowed for in their input buffer in page zero. Unfortunately, the input routine is in ROM and can't be changed. He also asked about a 'GET' statement. The unlabled issue has a letter from M. Minasi with one, using USR. I've included the one I use that doesn't require PEEK'ing memory to get the key.

Here's my two cents on the Garbage Collection problem: The problem occurs because OSI miscoded the length of a string array element. They coded it as 3 when it is actually 4. I was astounded when I walked through the code and saw that. [Actually, Microsoft did it.]

For those readers interested in telecommunications, MICRO (July 80) has an article on adding RS232 to a C2; Frank Derfler's column in Kilobaud (July 80) has the info for a C1. Both have dumb terminal software listings.

I enjoy PEEK(65) and eagerly wait for each issue. We need more users to write in and help reduce the backlog of unanswered questions (also to increase the percentage of usable text - #6 was only about 50%). A suggestion: since we all keep a notebook by our machines (don't we?!), each month, after PEEK(65) arrives, we should sit down and write a letter detailing whatever information we've discovered that month. There will be duplication, of course, but also a lot of useful information. Let's wear Al's fingers to the bone from typing!

Mike (Hobbitt) Carroll
P.O. Box 2844
Tulsa, OK 74101

Hobbitt:

Super letter! Please do wear my fingers off, and do keep a notebook.

AL

```
00010 REM CARD SHUFFLING
00020 REM
00030 DIM DK(52)
the card "deck"
00040 REM
00050 REM TO "DRAW" A CARD, GOSUB 9000
00060 REM FIRST TIME THROUGH, CC MUST BE
00070 REM ZERO

09000 REM SHUFFLING/DRAW ROUTINE
09010 REM
09020 IF CC>0 THEN 9090
don't need to shuffle
09030 PRINT "SHUFFLING";
09040 FOR CC=1 TO 52
reset deck
09050 DK(CC)=CC
09060 NEXT CC
09070 CC=52
09080 PRINT TAB(15);"DONE"
09090 XX=INT(RND(1)*CC)+1
get number from 1 to card count draw that
slot's card move last slot to this slot
one less card
09100 CD=DK(XX)
09110 DK(XX)=DK(CC)
09120 CC=CC-1
09130 REM
09140 REM CD HAS THE "DRAWN" CARD
09150 REM
09160 RETURN
```

Card rank is INT((CD-1)/13) 0 - 3
Card suit is CD-RANK*13 1 - 13

E.H. Carlson in the unlabled issue indicates that CHR$(229)-CHR$(232) are the card suits.

```
00010 REM "GET" FUNCTION
00020 REM $0222  $022A
00030 FOR I=546 to 554
load machine language routine
00040 READ J
00050 POKE I,J
00060 NEXT I
00070 POKE 11,34 : POKE 12,2
set USR(X) function address subroutine
data    $ 20 ED FE A8 A9 0 6C 8 0
00080 DATA 32,237,254,168,169,0,108,8,0
00090 REM
00100 REM ASSEMBLY PROGRAM
00110 REM
00120 REM         ORG $0222  --  546
00130 REM GET     JSR $FEED
get a keypress
00140 REM         TAY
set up return arguments
00150 REM         LDA #$00
00160 REM         JMP ($0008)
pass data to basic
```

"GET" a key by executing KEY=USR(X).

ED:

Just a note about the letters in the May issue. Lucas-on-garbage cleared up the problem at last, but I don't have any trouble at all like he describes. PLEASE! Indicate in the letters what system the letter refers to! It sure would make things easier all around.

I see the code R$=RIGHT$(A$,LEN(A$)-1) very often, and so it seems worth mentioning that the statement R$= MID$(A$,2) is a lot easier. In fact, to move the first letter of a line to the end may be coded as A$=MID$(A$,2)+LEFT$(A$,1) much more simply.

Mr. Laborie and his ERROR 17 might try what I did. I think I solved that problem on my C-3 but the fix is no fun. First, the most common time for framing errors seems to be when the head current is changed at track 42. If the actual disk address is around 150532 that is the symptom and your drive head current should be adjusted by a smarter man than I.

Assuming you have a back-up disk, it is frequently possible to make ERROR 17 go away by re-initializing the disk. I have also discovered that if the WRITE-PROTECT slot is not covered before initializing that COPIER will not blow away until after the initialization process. I do not know whether a complete init. job is done when the WRITE-PROTECT is uncovered or not, but you should repeat the process just for good measure.

I found on the 'C' drive that the head lifter did not raise the head from the disk--ever. Look inside with the cover off during a copy files from C to D and verify that the head is actually lifting off the surface and fully dropping back on during the full copy. There is a high probability that the aluminum head lifter is warped and the head either fails to clear or fails to completely seat at one end or the other. If this is the case you must reach inside, with the power off and a disk in place, and bend the head lifter until it seems to be parallel to the disk. There is an adjustment screw with an allen head which is not much use but can be turned to suit. Look at it this way--you can't make it much worse. The biggest problem is getting up your courage to bend it. Be thankful it is not the inaccessible drive!

[illegible] Sanders
Vienna, VA

ED:

1. After trying to get a program in a magazine to run, I wrote Microsoft and received the enclosed letter. This problem showed up on KIM Micro-Z basic, OSI Basic in Rom and on OS-65D ver. 3.0.

2. In regard to Basic Bugs like the one above and mentioned in previous issues of PEEK(65): Is anyone cataloging them and their fixes?

3. There is an error in the "CREATE" utility of OS-65D that prevents a large number of files from being used. Line 580 should read: GOSUB 20000 not 10000.

4. Does anyone know how to get the PAGING feature of output port 5 on OS-65U to work on PORT 3?

Robert H. Foltz
Bronx, NY

Dear Mr. Foltz:

Your problem with FOR loops crashing is a very old bug. The KIM Micro-Z does not have the fix as they never paid us for the original work. Ohio Scientific, on the other hand, was sent a newer version which corrected this but they never installed the changes in their source.

Your real problem lies in lines 2620-2720. Here you have exited an un-terminated FOR..NEXT loop. The "bug" arises only in this instance.

Michael E. Courtney
Technical Support, Microsoft
Bellevue, WA

Robert:

1. Interesting! 2. Why don't you? 3. Does the change to CREATE require a larger directory file? (Careful!) 4. Yes. Just before your program starts printing, insert POKE 11686, 2°(DV-1)+16 where DV = Device number (POKE 11686, 20 for Device #3) then POKE 11686,1 before any printing you want to have come to the console. Then simple PRINT statements will do what a PRINT #3 did before, plus paging. Note--will not work if you have a #5 (CA-9 or other) board in your machine.

AL

[ ] $12 Enclosed. Send PEEK(65) for one year (12 issues)

Foreign Subscriptions Air Mail - $20.00/yr.

[ ] $15 Enclosed. Enroll me in OSIO

[ ] $25 Enclosed. Send PEEK(65) and enroll me in OSIO!

NAME ______________________________

Street ______________ City ______________

State ______________ Zip ______________

Please send the following back issues.
I enclose $1.50 ea.

NOTE: To clear up any confusion about back issues of PEEK(65) and their No.'s, here are there dates, no.'s, and first lines:

Jan. #1, "Welcome to the first issue"
Feb. #2, "A month ago in this spot"
Mar. #3, "Peek continues to grow"
Apr. #4, "We are OSI fans"
May #5, "The continued growth & health" (This was mistakenly labeled #4.)
June #6, "This column should probably be"
July #7, "Several times recently"

PEEK (65)
The Unofficial OSI Users Journal

1819 Bay Ridge Ave., Suite 220
Annapolis, MD 21403

BULK RATE
U.S. POSTAGE
PAID
Annapolis, MD
Permit No. 563

DELIVER TO: