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. 7
July 15, 1980

Column One

Several times recently, people have told me that they believe the number of OSI users is approaching "critical mass," that magic number at which a chain reaction becomes self - sustaining. This month's issue of PEEK(65) is further evidence of this. The number of excellent stories and articles we have received has allowed us to expand again, to 20 pages. At the rate subscriptions and letters and stories are coming in, we will soon be at 32 pages, each page filled with important information for OSI users. All of us here at PEEK(65) are delighted with this trend, and invite all of you to write us about your personal experiences with OSI gear, interesting projects you have undertaken, important gripes you have about your computer, and anything else which interests you. We don't publish everything we receive, but we do try to serve as a virtually unedited conduit for the passage of information between OSI users.

As we continue to grow, unfortunately, so do our expenses. It costs more to write (a little), edit, assemble, print and mail a 32-page magazine than an 8-page newsletter. Therefore, like everything else, prices are going up. Effective August 1, 1980, a one-year subscription to PEEK(65) will cost you $12. Effective January 1, 1981, the price will rise again, probably to $15/yr.

But there is good news, too -- I mean in addition to the good news that our circulation and size and quality of material submitted for publication are all increasing much more rapidly than the price. First of all, you can beat the price increase. Even though your subscription has not elapsed, if you renew now and send your money with your renewal, you can have one more year for $8. But only one more. And we must receive your $8 by August 1.

And more good news. The success of PEEK(65) encourages us to boldness: effective with the September issue, we will pay for articles! You will never get rich writing for PEEK(65), but if you have something really worthwhile to say and need some encouragement to take the time to put it on paper (or better yet, on either WP-2 or WP-6502 disk), let's haggle about an honorarium, in addition to the world-wide fame a by-line in PEEK(65) will win for you.

We have received several letters and phone calls recently on the subject of data conversion. Many people need conversion from IBM 3741 floppy disk direct to OSI floppy disk. The bad news is: it can't be done with OSI gear as it now exists. The good news is, it could be done with a modified disk controller and some machine - language programming. What you would wind up with would be a dual- floppy computer with a special copy program. Insert your IBM diskette into drive "A" and your OS-65U disk into drive "B" and run the program and, in about 2 minutes, your OSI disk would be about half full, having emptied the IBM disk of its contents. Question: has anybody already done it? And if we find such a controller and program, how many of you would like to have a set, at about $800 for the controller and software? All in all, it seems like a worthwhile project, one which would save lots of folks lots of money, transform the key - to - disk business and pave the way for gradual conversion of a lot of shops to OSI gear, allowing them to make the transition smoothly without junking existing equipment. Comments invited.

A VISIT TO MONTE CARLO

by Jim Sanders

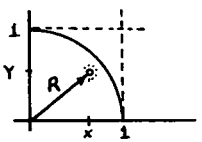
The term 'Monte Carlo' in the computer biz refers to the use of probability to obtain a 'good enough' answer to problems that are too hard to solve in other ways. Two example programs are given here which illustrate the technique.

The first program calculates the value of Pi. Since we have reason to believe that the answer is about 3.14159 this program may also be used to evaluate random number generators. The Microsoft RND function as implemented by OSI is pretty bad, so a modification of the Fibonacci series is used instead.

Given that the area of a quarter circle is:

A=Pi * R * R / 4

we proceed to draw the figure below with a radius (R) of one.




```
100 G=1.61803 :INPUT"SEED (.1 TO .9)";Y: GOTO 240
110 REM GENERATE A TRULY RANDOM NUMBER
120 Y=100*(G^Y): Y=Y-INT(Y):RETURN
130 REM PRINT A GRAPH OF THE RESULTS AS WE GO ALONG
140 J=INT((4*HI/TH-3.14159)*100)+40:IFJ<1THENJ=1
160 IFJ>78THENJ=78
180 IFJ>40THENPRINTTAB(40);"!";TAB(J);"*":RETURN
200 IFJ<40THENPRINTTAB(J);"*";TAB(40);"!":RETURN
220 PRINTTAB(40);"*":RETURN
230 --- PRINT THE STATUS EVERY 100 TRIALS ---
240 FOR TRIAL=1 TO 100
260 GOSUB 120:X=Y:GOSUB120          :REM GET 2 RND NRS IN X AND Y
280 IF X*X + Y*Y <= 1 THEN HIT=HIT+1:REM IF INSIDE THE CIRCLE INCR HIT
300 THROWS= THROWS+1                :REM KEEP COUNT OF ATTEMPTS
320 GOSUB140                        :REM PRETTY GRAPH OF THE STATUS
340 NEXT TRIAL
360 PI= 4*HIT                       :REM EVERY 100 PRINT HOW-GOES-IT
380 PRINT"AFTER"THROWS", PI="PI / THROWS
400 GOTO 240:                       REPEAT THIS PROGRAM FOREVER
```

Noticing that the area of the square around the arc is 1.0, the ratio of the circle segment to the square is:

Ac / As = (Pi / 4) / 1.0

and leap to the conclusion that:

Pi= 4 * (Ac / As).

Now if we just knew the ratio of these areas we would have the answer. If we threw darts at the figure, the probability of hitting in the circle is proportional to their areas. Preventing holes in the wall, we instead pick two random numbers from zero to one, calculate the hypotenuse (R) and conclude a 'HIT' if the answer is in the circle (less or equal to one).

The reader may wish to figure out why the SQR is not used.

The second example requires a story. A thief enters a vault containing three chests. The first has only gold bars, the second has only lead bars, and the third has half gold and half lead bars. She takes two bars from one chest. One turns out to be gold. What is the probability that the other one is also gold?

The program below finds the answer and follows the story directly. Try it with the 'better' random number generator and see if your results are closer to the real answer.

```
10 INPUT"SEED";X                    :REM START RANDOM GENERATOR
20 FOR TRIAL=1 TO 100:FOR THIEF=1 TO 100:REM 100 TRIALS OF 100 THIEVES
30 CHEST=INT ( RND(X) * 3 ) +1      :REM WHICH CHEST DID HE PICK?
40 REMIND THIEF OF RULES IF HE'S IN CHEST 2 AND SEND HIM BACK IN.
50 IF CHEST =2 GOTO 30
60 IF CHEST=1 THEN GOLD=GOLD+1:GOTO 90 :WE KNOW THE ANSWER IF CHEST 1.
70 BAR=RND(X):IF BAR<=.5 THEN GOLD=GOLD+1  :REM CHEST 3 REQUIRES MORE.
80 X=GOLD                           :REM CHANGE THE SEED PERIODICALLY
90 NEXT THIEF                       :REM SEND IN ANOTHER
100 PRINT "PROBABILITY OF A GOLD BAR=";
110 PRINT GOLD / (100*TRIAL)  :REM PRINT OUT THE INTERMEDIATE RESULTS
120 NEXT TRIAL                :REM AND THEN GET ANOTHER 100 THIEVES.
130 END
```

A MORAL TALE by Wallace Kendall

When I was shopping for a computer a while back, I suddenly heard the sounds and smelled the smells of long ago.

All at once I remembered the days when my grandfather was a contractor for the War Department. Yes, I told you it was a long time ago.

He sold them mules for the Army. Okay, okay, it was a very long time ago.

A lot of people were pretty mean to mules. Not my grandfather. His mules earned their board and keep, all right, but he saw to it that they had plenty of food and water and their harness fitted properly and they got good veterinary care.

"It's just the moral decent thing to do," he'd say.

After World War I the contracts for Army mules ran out. Tractors were doing more and more of the plowing, and the new cars Henry Ford and others were building killed the sale of mules for wagons and buggies, and my grandfather had lots of time to show a six-year-old his big roll-top desk and the books he kept there. I remember the columns of spidery figures and the Palmer script and the debits and the credits and the red inkwell on one side of the desk and the black inkwell on the other, and how you can read a blotter with a mirror.

That day in the computer store it all came back to me, as clearly as a 70mm wide-screen full color picture with Dolby sound. I could even smell the chewing tobacco nearly all the men used, and certain other long-forgotten odors typical of mule barns.

The store had advertised a spectacular new computer with a foolproof set of programs that would make recordkeeping for any business as easy as ABC.

While I was talking about buying the computer, I saw the books in which the computer store kept its own records. They looked exactly like my grandfather's. The same columns of spidery figures and Palmer script.

There must be a moral there somewhere, but I'm not sure what it is. That was a couple of years ago, and things may have changed. I've wondered how many small businesses even today have computers and use them only for minor jobs, or duplicate everything with a manual system.

I've wondered how many businesses have bought a computer, but (a) won't buy the software to make it run, (b) won't hire or train a competent staff to run it, (c) won't engage consultants, (d) won't waste money on maintenance contracts, and apparently (e) use it largely to impress clients.

Accounting has changed in the last 50 years. My grandfather just wanted a record of what money and other valuables had changed hands, and how. Today's business man - or woman -- wants more. He or she wants to enter a figure once and have it update four or six accounts, to look at figures in different contexts, to project trends, to compare sets of figures in complicated ways, to analyze possible consequences of a course of action, and other chores I don't even understand.

If you're using your computer only to to keep financial records, maybe you should try my grandfather's system. It was cheap, and it worked fine for his purposes.

But a much better course of action would be give your computer the proper care and nourishment and exercise, and let it earn its board and keep.

As my grandfather might have said, "It's just the moral, decent thing to do."

ABOUT 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.

LETTERS

ED:

Can you provide the correct code for the garbagecollector for those who want to burn an EPROM?

Duane Berry
Bryan, TX

ED:

I would like to encourage your readers to copy the following letter and send copies to manufacturers of printers. If enough of us do this, we might see a new, low-cost printer on the market soon.

Dear Printer Manufacturer:

As a potential customer, I challenge you to produce a printer with the following features:

* Fixed line capacity of at least 72 characters

* Fixed spacing of 10 characters per inch

* At least 83 printable ASCII characters

* Maximum print rate of at least 15 CPS

* 8 1/2 inch forms width capacity

* Price less than $326

By eliminating such frills as multiple line lengths, graphics, and high print speeds, I believe you can market a printer that doesn't cost more than the computer that drives it.

Please think it over; buyers are waiting!

Bruce Showalter
Abilene, TX

ED:

I recently received a disk written under OS-65U with some interesting programs on it, written by a friend of mine. He had invited me to try them out on my C2-OEM, but when I tried to run them, I got ERROR 130 which some dredging through the OSU manual indicated meant he had created the files with NO access rights without a password. OK, I'm no dummy, I tried RUN"PORNO","PASS". That didn't work either, so I called my friend to ask him what password he had used, but he was out of town for the weekend, and his friend said she didn't ever mess with that computer, she had no Idea what the

cont'd on page 6

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 RECKIPTS
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
(813) 923-2781

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.

7650 S. TAMIAMI TRAIL SARASOTA, FLORIDA 33581

password was. Well, I didn't want to wait all weekend to have a look at what promised to be a great party game, so I got out good old OS-65U and had a look at CREATE. Right there in the code, I could see how the program transforms the password you give it into some kinda two - digit number and sticks it into the directory. I tried for a while figuring out how to run a number generator to try all possible combinations of 2-digit numbers, but since they are Hex digits there are lots of them, and anyway, I couldn't quite figure out what to do with them. Then it hit me! I could run CHANGE to have a look at the two digit numbers which were stuck in there by CREATE on some of my own programs for which I knew the passwords. Then I just used CHANGE to change the two-digit number on my friend's disk to the same one, and whatayaknow, I could load and run his program with the same password as my own!

Sam Martin
Garland, TX

ED:

I thought you might like to have this subroutine to publish in PEEK(65). Suppose a user has just started an hour-long print-out of an inventory listing and needs to interrupt the listing to do some other task. If the user resets or control 'C's out of the printing, nasty things can happen to your files. A more graceful way to exit would be to allow the user to press a key at the console, have the program poll the console to see if a key had been pressed, and then, if it had been pressed, allow appropriate program action. This works with OS-65U level I and level III. Happy computing.

Gary Hawkins
San Antonio, TX

```
1  REM PROGRAM NAME CONDITIONAL CONTROL C
2  REM REM THE COMPUTER SHOP, SAN ANTONIO,
   TEXAS
10 REM PROGRAM TO DEMO HOW LONG A PRINTOUT
   OR OTHER TASK CAN BE
20 REM INTERRUPTED FROM THE CONSOLE WITHOUT
   HAVING TO PRESS RESET
30 POKE 2073, 96: REM DISABLE CTRL-C
40 FOR I=1 to 1000: REM DUMMY PROGRAM WHICH
   WE WILL INTERRUPT AS A DEMO
50 PRINT I,I*I, SQR(I): REM PRESS CTRL-C
   WHILE IN THIS LOOP
60 IF PEEK (15006) THEN 90
65 REM IF CTRL-C HAS BEEN PRESSED, THEN WE
   WANT TO INTERRUPT THE LOOP
70 NEXT I
80 GOTO 110
90 POKE 15006, 0: REM RESET CTRL-C FLAG
100 UNPUT "ABORT"; A$: IF A$<>"Y" THEN 70:
   PUT IN YOUR OWN DESIRED
105 REM ACTION HERE
110 POKE 2073, 76: REM ENABLE CTRL-C
120 END
```

ED:

I have a C4P Cassette system that I am trying to interface with a Texas Instruments 745 terminal. I want to use the TI as a printer. Thus far, I have been unsuccessful, however. I need some advice about what cable connections are needed and anything else I need to do.

Thomas Blakemore
Menomonie, WI

ED:

To complement E.H. Carlson's letter, I wrote a machine language routine to obtain higher resolution graphics (for the SBII and C1P). What I did, was to break up each character space into four separate squares. The routine then takes the x-y position of the point and, after some juggling, puts the point on the screen using characters I65 to I78 to obtain the doubled resolution. The listing is included in this letter. To use this routine in BASIC:

i) load the program at the top of memory.
ii) break into BASIC and set memory size to 8048
iii) set USR pointer with: POKE I1,I12
POKE I2,3I
iv) in the BASIC program, the x-coordinate is POKED in location 8I(dec.) and the y-coordinate is POKED in location 82(dec.), i.e.:

```
POKE 8I,X
POKE 82,Y
```

To plot the point, execute the command: A=USR(0)

I've also included two extra options: option I is:
i) if the contents of the screen are to remain intact (plotted points "overlay" screen) then do the following before putting the point on the screen:

```
POKE 87,0
```

ii) if contents of screen are to be overwritten, then do the following before plotting the point on the screen:

```
POKE 87,255
```

Option 2 is:
If a point is to be put on the screen then the following should be done before anything else:

```
POKE 85,0
```

if a point is to be removed or 'erased' from the screen then the following should be done before anything else:

```
POKE 85,255
```

NOTE - the range on X and Y are:
$0 \leq X \leq 50$
$0 \leq Y \leq 60$

The best way to understand all this, is to try the routine.

If anyone is interested, I have two BASIC programs that implement this routine. One, that I wrote, draws straight lines, by defining the end points, and the other, courtesy of Mr. Trevor Dee, allows creation of large alphabetic characters for displaying big titles, etc. If a copy is desired, please send $1.00 and your address and I'll get it to you.

Next, I included the listing for a machine language save routine, in response to Mr. B. Showalter. This routine is simple, and is used as following:

i) load the program in memory
ii) place the starting address of the code to be saved in locations F7 and F8 (hex).
iii) place the ending address of the code to be saved in locations F9 and FA (hex).
iv) enter address $0222 in the monitor, while in the address mode, and push "G" for go. But make sure the tape is started first!

The routine will save the code on both the cassette and the screen (good for memory dump). When the routine is done, it will return to the "address" mode of the monitor. Note - this routine will not be destroyed when a COLD start is done into BASIC.

I also have a great real-time clock program (machine lang. and uses about 255 bytes), again, courtesy of Mr. Trevor Dee. The program requires a one wire hardware patch, and displays the time (including AM/PM) at the bottom of the screen. The program does not interfere, in any way, with the BASIC or the monitor. If you'd dike a copy, then please send $1.00 and your address and I'll get a copy to you.

In response to Mr. B. Showalter's letter, the video ram scroll routine is not in the monitor rom, but in the basic rom, starting at location $BF2D.

Finally, as a hardware, man, I'd like to hear from those readers, concerning hardware patches, modifications, etc. Come one guys!! OSI is great stuff and PEEK(65) is the best way we can communicate and make it even better, both hardware and software wise.

Looking forward to hearing from those other readers.

Paul Kourany
63 Westpark Dr.
Ottawa, Ontario K1B-3G4
Canada

```
 10 0000           ; HI-RES. PLOTTER
 20 0000           ; VERSION 1.1 - WITH ERASE/WRITE
 30 0000           ; OPTION : MARCH 1980
 40 0000           ;
 50 0000           ; ADAPTION OF PROGRAM WRITTEN BY
 60 0000           ; J.R. SHERBURNE (MICRO, MARCH 1979).
 70 0000           ;
 80 0000           ; BY PAUL KOURANY
 90 0000           ;
100 1F60           *=$1F60
110 1F60           ;
120 1F60 A900      START LDA #$00 INITIALIZE
130 1F62 8553           STA $0053
140 1F64 8556           STA $0056
150 1F66 38             SEC
160 1F67 A551           LDA $0051
170 1F69 E932           SBC #$32  CHECK FOR VALID X
180 1F6B 3001           BMI CHECK (0<=X=<50)
190 1F6D 60             RTS
200 1F6E           ;
210 1F6E 38        CHECK SEC       CHECK FOR VALID Y
220 1F6F A552           LDA $0052  (0<=Y=<60)
230 1F71 E93C           SBC #$3C
240 1F73 3001           BMI HALF
250 1F75 60             RTS
260 1F76           ;
270 1F76 4651      HALF  LSR $0051
280 1F78 9002           BCC NOCAR
290 1F7A E656           INC $0056
300 1F7C 4652      NOCAR LSR $0052
310 1F7E 9004           BCC NOCRY
320 1F80 E656           INC $0056
330 1F82 E656           INC $0056 DIVIDE X & Y BY 2
340 1F84 A901      NOCRY LDA #$01 DETERMINE QUADRANT OF
350 1F86 A456      LOOP  LDY $0056 NEW POINT AND PLACE
360 1F88 F006           BEQ MATCH QUADRANT NUMBER IN $0056
370 1F8A 0A             ASL A
380 1F8B C656           DEC $0056
390 1F8D 4C861F         JMP LOOP
400 1F90 8556      MATCH STA $0056
410 1F92 0652           ASL $0052 MULTIPLY Y BY DEC. 32
420 1F94 0652           ASL $0052 (# CHARACTERS PER LINE)
430 1F96 0652           ASL $0052
440 1F98 0652           ASL $0052
450 1F9A 9004           BCC NOV1
460 1F9C E653           INC $0053
470 1F9E E653           INC $0053
480 1FA0 18        NOV1  CLC
490 1FA1 0652           ASL $0052
500 1FA3 9002           BCC NOV2
510 1FA5 E653           INC $0053
520 1FA7 18        NOV2  CLC
```

```
 530 1FA8 A552           LDA $0052
 540 1FAA 6943           ADC #$43
 550 1FAC 9002           BCC NOCHG
 560 1FAE E653           INC $0053
 570 1FB0 18       NOCHG CLC
 580 1FB1 6551           ADC $0051 ADD X TO Y*32
 590 1FBE 8552           STA $0052
 600 1FB5 A9D0           LDA #$D0
 610 1FB7 6553           ADC $0053
 620 1FB9 8553           STA $0053
 630 1FBB A010           LDY #$10 LOOK UP CHARACTER IN
 640 1FBD A200           LDX #$00 SCREEN POSITION X+Y*32 IN TABLE
 650 1FBF A152           LDA ($0052,X)
 660 1FC1 88       CHARAC DEY
 670 1FC2 D9F01F         CMP TABLE,Y
 680 1FC5 F009           BEQ FOUND
 690 1FC7 C000           CPY #$00
 700 1FC9 D0F6           BNE CHARAC
 710 1FCB A657           LDX $0057 IF NOT IN TABLE, CHECK
 720 1FCD F001           BEQ FOUND PRESERVE/DESTROY FLAG, $0057
 730 1FCF 60             RTS
 740 1FD0          ;
 750 1FD0 A555     FOUND LDA $0055
 760 1FD2 D00A           BNE ERASE CHECK ERASE/WRITE FLAG
 770 1FD4 98             TYA
 780 1FD5 0556           ORA $0056 COMPUTE NEW CHARACTER
 790 1FD7 A8    49       TAY       WITH ADDED POINT AND
 800 1FD8 B9F01F         LDA TABLE,Y STORE IT ON SCREEN
 810 1FDB 8152           STA ($0052,X)
 820 1FDD 60             RTS
 830 1FDE          ;
 840 1FDE 98       ERASE TYA
 850 1FDF 2556           AND $0056
 860 1FE1 C556           CMP $0056
 870 1FE3 D00A           BNE INVAL  CHECK IF POINT EXISTS
 880 1FE5 38             SEC
 890 1FE6 98             TYA
 900 1FE7 E556           SBC $0056  COMPUTE NEW CHARACTER
 910 1FE9 A8             TAY
 920 1FEA B9F01F         LDA TABLE,Y STORE NEW CHARACTER
 930 1FED 8152           STA ($0052,X) ON SCREEN
 940 1FEF 60       INVAL RTS
 950 1FF0          ;
 960 1FF0 20       TABLE .BYTE $20    SP - 40?
 970 1FF1 A8             .BYTE $A8
 980 1FF2 A6             .BYTE $A6
 990 1FF3 9B             .BYTE $9B
1000 1FF4 A7             .BYTE $A7
1010 1FF5 9C             .BYTE $9C
1020 1FF6 AA             .BYTE $AA
1030 1FF7 AF             .BYTE $AF
1040 1FF8 A5             .BYTE $A5
1050 1FF9 A9             .BYTE $A9
1060 1FFA 9D             .BYTE $9D
1070 1FFB B1             .BYTE $B1
1080 1FFC 9A             .BYTE $9A
1090 1FFD B2             .BYTE $B2
1100 1FFE B0             .BYTE $B0
1110 1FFF A1             .BYTE $A1
1120 2000          .END
```

89

89
72
161

ED:

This program is in response to a reader's letter that got published in your February (2nd) issue. He said that he would like a RTTY program of some sort. I had written this program last year and thought some of your readers would be interested. This program was written on my C4P, and uses the audio output capabilities it has.

Feel free to modify the program in ay way you wish. For instance you might want to add more types of characters that can be sent.

Kevin G. Lew
San Antonio, TX

ED:

My system is composed of a OSI C2/4P with 16K RAM and a Heath WH-14 Line Printer. The printer is actuated via a SAVE command through an RS232 connected to the cassette port. According to Heath the printer is capable of printing 10 inch and 16.5 inch characters via the switch on the printer. This works OK. Also, in addition to the 10 and 16.5 sizes, 12 inch available via software selection. As of this date I have not determined the command for character size change during RUN. Perhaps some other OSI user has solved this problem.

R.D. Bracey
Evansville, IN

```
1       REM MORSE CODE PRACTICE BY KEVIN G. LEW  15 DEC 79
2       INPUT"ENTER CODE SENDING SPEED (1-10)"; T
3       CHSP=30*T : REM CHARACTER SPACE TIME
4       DOT=10*T : REM DOT TIME
5       DASH=30*T : REM DASH TIME
6       WDSP=70*T : REM WORD SPACE TIME
7       ELSP=10*T : REM ELEMENT SPACE TIME :
9       POKE 57089,30 : PRINT
10      PRINT"THIS PROGRAM WILL ENABLE YOU TO SEND MORSE CODE A"
20      PRINT
30      PRINT"LINE AT A TIME. THIS PROGRAM DOES NOT ACCOMMODATE"
32      PRINT
35      PRINT"FOR ANY TYPE OF PUNCTUATION MARKS. ONLY THE FULL"
36      PRINT
37      PRINT"ALPHABET AND THE NUMBERS 0 THRU 9 ARE ALLOWED."
40      PRINT
50      PRINT:PRINT"MESSAGE":INPUT A$
55      PRINT:PRINT"SENDING"
70      L=LEN A$
80      FOR M=1 TO L
90      C$=MID$(A$,M,1)
100     B=ASC(C$)
105     IF B=32 THEN 1998
106     IF B>47 AND B<58 THEN 2001
107     B=B-64
110     H=B+64:H$=CHR$(H):PRINT H$;
111     FOR D=0 TO 100:NEXT D
112     ON B GOTO 3000,3001,3002,3003,3004,3005,3006,3007,3008,3009
113     B=B-10
114     ON B GOTO 3010,3011,3012,3013,3014,3015,3016,3017,3018,3019
115     B=B-10
116     ON B GOTO 3020,3021,3022,3023,3024,3025,3026
120     POKE 56832,3
130     FOR O=0 TO W:NEXT O
140     POKE 56832,1
150     V=V-1:IF V=0 THEN 5000
160     FOR D=0 TO ELSP:NEXT D
170     POKE 56832,3
180     FOR P=0 TO X:NEXT P
```

```
190     POKE 56832,1
200     V=V-1:IF V=0 THEN 5000
210     FOR D=0 TO ELSP:NEXT D
220     POKE 56832,3
230     FOR D=0 TO Y:NEXT D
240     POKE 56832,1
250     V=V-1:IF V=0 THEN 5000
260     FOR D=0 TO ELSP:NEXT D
270     POKE 56832,3
280     FOR R=0 TO Z:NEXT R
290     POKE 56832,1
295     V=V-1:IF V=0 THEN 5000
300     FOR D=0 TO ELSP:NEXT D
310     POKE 56832,3
320     FOR S=0 TO ZZ:NEXT S
330     POKE 56832,1
340     GOTO 5000
1998    PRINT" ";:FOR D=0 TO WDSP:NEXT D
1999    NEXT M
2000    GOTO 50
2001    B=B-47
2002    H=B+47:H$=CHR$(H):PRINT H$;
2003    ON B GOTO 3026,3027,3028,3029,3030,3031,3032,3033,3034,3035
3000    V=2: W=DOT: X=DASH: GOTO 120
3001    V=4: W=DASH: X=DOT: Y=DOT: Z=DOT: GOTO 120
3002    V=4: W=DASH: X=DOT: Y=DASH: Z=DOT: GOTO 120
3003    V=3: W=DASH: X=DOT: Y=DOT: GOTO 120
3004    V=1: W=DOT: GOTO 120
3005    V=4: W=DOT: X=DOT: Y=DASH: Z=DOT: GOTO 120
3006    V=3: W=DASH: X=DASH: Y=DOT: GOTO 120
3007    V=4: W=DOT: X=DOT: Y=DOT: Z=DOT: GOTO 120
3008    V=2: W=DOT: X=DOT: GOTO 120
3009    V=4: W=DOT: X=DASH: Y=DASH: Z=DASH: GOTO 120
3010    V=3: W=DASH: X=DOT: Y=DASH: GOTO 120
3011    V=4: W=DOT: X=DASH: Y=DOT: Z=DOT: GOTO 120
3012    V=2: W=DASH: X=DASH: GOTO 120
3013    V=2: W=DASH: X=DOT: GOTO 120
3014    V=3: W=DASH: X=DASH: Y=DASH: GOTO 120
3015    V=4: W=DOT: X=DASH: Y=DASH: Z=DOT: GOTO 120
3016    V=4: W=DASH: X=DASH: Y=DOT: Z=DASH: GOTO 120
3017    V=3: W=DOT: X=DASH: Y=DOT: GOTO 120
3018    V=3: W=DOT: X=DOT: Y=DOT: GOTO 120
3019    V=1: W=DASH: GOTO 120
3020    V=3: W=DOT: X=DOT: Y=DASH: GOTO 120
3021    V=4: W=DOT: X=DOT: Y=DOT: Z=DASH: GOTO 120
3022    V=3: W=DOT: X=DASH: Y=DASH: GOTO 120
3023    V=4: W=DASH: X=DOT: Y=DOT: Z=DASH: GOTO 120
3024    V=4: W=DASH: X=DOT: Y=DASH: Z=DASH: GOTO 120
3025    V=4: W=DASH: X=DASH: Y=DOT: Z=DOT: GOTO 120
3026    V=4: W=DASH: X=DASH: Y=DASH: Z=DASH: GOTO 120
3027    V=5: W=DOT: X=DASH: Y=DASH: Z=DASH: ZZ=DASH: GOTO 120
3028    V=5: W=DOT: X=DOT: Y=DASH: Z=DASH: ZZ=DASH: GOTO 120
3029    V=5: W=DOT: X=DOT: Y=DOT: Z=DASH: ZZ=DASH: GOTO 120
3030    V=5: W=DOT: X=DOT: Y=DOT: Z=DOT: ZZ=DASH: GOTO 120
3031    V=5: W=DOT: X=DOT: Y=DOT: Z=DOT: ZZ=DOT: GOTO 120
3032    V=5: W=DASH: X=DOT: Y=DOT: Z=DOT: ZZ=DOT: GOTO 120
3033    V=5: W=DASH: X=DASH: Y=DOT: Z=DOT: ZZ=DOT: GOTO 120
3034    V=5: W=DASH: X=DASH: Y=DASH: Z=DOT: ZZ=DOT: GOTO 120
```

```
3035    V=5: W=DASH: X=DASH: Y=DASH: Z=DASH: ZZ=DOT: GOTO 120
5000    FOR D=0 TO CHSP: NEXT D: NEXT M
5001    PRINT:GOTO 50
```

BYTES USED : 2781

ED:

Has anyone done any work on an interface that would allow a C2 to work as a 20 ma terminal or RS232 into a modem? The only mention I saw was for 65D, one is needed for Basic-in ROM machines.

Another thing that would be nice is a file read idea that works for cassette machines.

Neil Dennis
Bliss, NY

ED:

I have seen several letters in your publication and others from OSI owners that complained about "Garbage Collection" while manipulating strings. If the manipulated string is not going to be used again then all you have to do is poke location 129 (LO Byte) to zero and 130 (HI Byte) to 32 for an 8K machine. But if you have protected high memory for some reason during power up or by poking locations 133 (LO) and 134 (HI) then you will have to adjust this string pointer to below protected memory. This works very well on my OSI 1P machine but I cannot say if it will work on the other OSI machines.

Also if your readers are interested in a -GET- type of routine for the OSI 1P which is also a typewriter routine that makes the keyboard work like a typewriter. It also allows typing ahead for a Morse Code program. It is written in Machine Language and takes only the top 256 bytes of memory. I have also written it in BASIC so it can be incorporated into existing programs and saved. If any of your readers are interested I will send them a listing if they send me a SASE and any original program that they would like to trade on cassette.

Maurice P. Lewton, WA6PHR
1323 Via Del Carmel
Santa Maria, CA 93455

ED:

For Brian Fearnow - The keyboard uses $DF00; the monitor writes to this location then reads it to see what key has been pressed. The subroutine at $FD00 will return the ASCII value in 'A', of the key pressed.

The serial port is at $F001 for the data register and $F000 for the control register. A data sheet for a 6850 ACIA will help or see Basic Microprocessors and the 6800 by Ron Bishop.

The connections are in the Sam's Manual, but watch out for the 'terminal guide'. It's misleading, at least in my copy. Use the photo. Superboard to S-100 interface design is worth a lot more than $20! Try and get the documentation for a 'KIMSI', KIM to S-100 interface to see why. Forethought Products is in Oregon, they make it and are good to deal with

What's a 500 board and why would you want to interface to it?

Bill Gaidzik
Santa Monica, CA

ED:

We have been driving a line printer on a C3 with the 510 board PIS "B" side for several years using a machine code routine. Recently we unsuccessfully attempted to run that line printer off the C2 PIA. On the 500 boards (we have a spare) the port will output only when initialized with a BASIC routine. The machine code one will not work. Obviously because it works in BASIC, the wiring is correct. It is not a PIA chip problem (any of the 6521's we have will work on the 510 but not the 500's).

Any help or suggestions you or your readers could offer on this will be most appreciated.

D. Valentine
New York, NY

ED:

William Hwang wanted RS232 info on the C2-4P.

I have my C2-4P so configured....Simply run a pair of wires from the AUX connector of the CPU board {500B} to a db25 connector which can be mounted in the cut-out already on the rear of the computer. Aux terminal 6 is low and 7 is high. On the 25 pin connector, low is 1 and 7; high is on pin 2 for some printers or 3 for others. He may also have to provide jumpers to tell the printer it's OK to print. No jumpers on the 25 pin connector were necessary on my connector.

This is the output of the 6850; so to output data to the printer, enter the word SAVE. If his printer does not run at 300 baud, it will be necessary to add a switch in place of the jumper J5...in one position selecting C-5 the other C-6 which is selected for the correct baud rate for his printer. I have mine set for several rates with a 5 position switch so as to allow me to feed different types of printers.

If Bill needs a two way RS-232, there are several changes to make on the CPU board or he could go for a CA-10X board from OSI. The newer 502CPU board appears to have the same feature on its AUX connector, but I am only looking at the schematic.

Something else: I am in need of a program [machine code I'd guess] to convert from ASCII to Correspondance code [selectric] and output for my serial printer....or a program to do the same look up and output to my PIA port. Can anyone help?

Robert Groome

Cleveland, OH

ED:

Here is a simple modification to the DMS Nucleus program EDMAFL so that when random records are being modified the operator is not forced to view on the CRT the next record after the previous record has been edited and viewed for correctness.

```
LINE 1284 change GOTO 1290 to GOTO 1400
1400 INPUT"DO NEXT RECORD";A$
1405 IF A$="" OR A$="Y"THEN 1290
1410 CLOSE: OPEN  F$(2),PW$,1:FPTR=1  GOTO
700
```

A response of "N" to the question will transfer control immediately to the search menu of EDMAFL. This has been working fine for 6 months with no problems and saves time getting back to the search menu.

I also have fixes for bugs in the INSERT and REMOVE programs which corrects erroneous reads and writes on the file that is being expanded or purged of extra records. I will be glad to send those in too if you think it would be useful for the journal. (Of Course! -Al)

Radford Compton
Manassas, VA

ED:

I have recently begun working with 65U in a school environment. One of the aspects of 65U that I found attractive was its password protection capability. I have discovered, however, that 65U seems to ignore control characters (except for CTRL-C, CTRL-O and other control characters with reserved uses). Since we are using a hard copy rather than CRT console, the fact that control characters cannot be used as passwords diminishes the usefulness of the password feature. Is the "invisibleness" of control characters an inherent characteristic of 65U or is there some reasonable fix?

Robert Camner
Washington, DC

ED:

A week ago I received the joysticks and programs that I had ordered from Aurora Software Associates (353 south 100 east,#6, Springville, Utah 84663). Although I had never soldered before, I had no trouble making the modifications necessary. The directions were foolproof. However, the programs would not run on my CII 4P/MF. I made this fix, and thought I would share it with others that may have found the same trouble:

In both "Bomber" and "Jet Attack" change dines

200 FOR GH=1TO2:POKE57088,16 (REM JET ATTACK)

200 POKE 2073,96:POKE 57088,16 (REM BOMBER)

And line

410 P=255-PEEK(57088)*8

Thanks for PEEK(65)!!!

Lawrence Huff
Minneapolis, MN

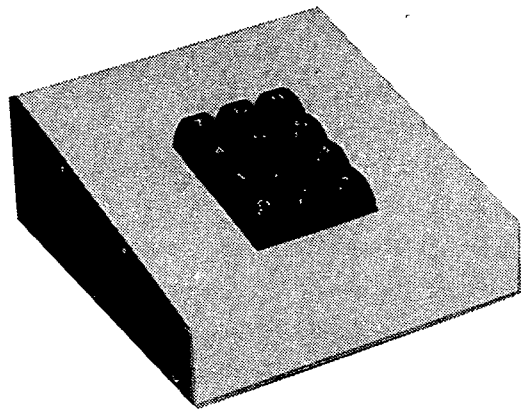
ED:

For a zero hardware Morse receiver, load your auto speed adjusting Morse reader (I got the one from Bob Kurtz, W6PRO in November '78 KILOBAUD to run on my C2), and run the audio from your receiver into the cassette input jack. Now initialize the ACIA for a X1 clock. Instead of expecting serial data to come by at 300 bits/sec (4800 Hz clock and divided by 16 in the ACIA), it checks 4800 times/second. Now the input stuff in the cassette interface is a sort of missing pulse detector, and essentially determines whether there is a signal above about 1600 Hz present. If there is, it sends a 1 (maybe backwards) to the data input of the ACIA. That is what is sampled 4800 times/second. The Morse code is certainly not sent serially with stop and start bits, but if you look at the whole frame as a sample of the data coming in, and expect the data to be either 00 or FF, you can determine whether or not there was a tone present. You're sampling now not at 4800, but more like 480 (with start, 8 bits, and stop making 10 bits/frame) times/second. This is still fast enough for almost all hand-sent Morse. You'll need to interrogate the ACIA with a machine language routine that passes its info back to BASIC for maximum speed. I've used this system to copy stuff at better than 40 WPM. Your receiver has to have a pretty wide-range BFO, though, because you need the audio above 1800 Hz to be solidly detected by the cassette hardware.

Very incidentally, I had a student (I'm a high school physics teacher) hook a key and audio oscillator up to the Morse reader and teach himself to send code with a good hand! The computer was a most strict teacher, and if what he sent wasn't right, the result was garbage on the screen.

Jim Williams, WD9IAF
Calumet City, IL

OSI



BKM'S NUMERIC KEYPAD — $65
FOR OHIO SCIENTIFIC OR LEAR SIEGLER'S ADM-3A

FEATURES

Digits 0-9, Decimal Point, and Return
Terminal Quality Keypad
Ready to use with C4's and C8's
Two Color Beige-Brown Case
Size 7"D x 5"W x 2½"H
Documentation

OPTIONS

Adapter cable kit for
10 minute INSTALLATION — $4.95
SPECIFY

Superboard **C1**
C2-4P **C2-8P**

BKM MICRO SYSTEMS CORPORATION

WE PAY SHIPPING TO CONTINENTAL U.S.
TEXAS RESIDENTS ADD 5%
ORDER DIRECT OR ASK YOUR OSI DEALER
FOREIGN ORDERS ADD 30%
AVAILABLE JUNE 1

VISA

3809 OLD COLLEGE ROAD
BRYAN, TEXAS 77801

713/846-8268
713/846-3817

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

A BETTER RANDOM NUMBER GENERATOR
by David Hille, San Antonio, TX

I have a routine to improve the random number generator for OSI microcomputers. The random number generator in ROM generates 1861 numbers before it repeats its sequence. My routine causes the random number generator to produce millions of random numbers without repeating a sequence.

The way a random number generator works is explained in the book How to Program Microcomputers by William Borden. The book contains a random number generator for a 6502 microcomputer, in machine language. Its random number generator uses two bytes which serve as both a random number and as a seed to generate the next random number. The routine finds a pseudo-random number by multiplying the last pseudo-random number by 5. Only the least significant 16 bits are saved, in effect performing a modulus 64K operation. The routine generates 8K of random numbers before repeating its sequence.

From this description of a random number generator, it is clear that the pattern of numbers generated by a random number generator is affected by the "seeds". So, to fix the OSI random number generator in ROM, you must periodically change the seeds used by it.

The random number generator in ROM starts at memory location 48064. It is not necessary to disassemble it, however, to locate the bytes it uses as "seeds". The bytes affected by the routine can be identified by running a program which, for each byte in page zero (consisting of bytes 0 to 255 in the bottom of RAM), fills up the screen, with each location in video RAM conatining the value of the byte after calling the random number generator routine. For those bytes affected by the random number generator, the screen will be filled with a variety of characters. For those not affected, it will be filled with the same in each location. From this it can be shown that bytes 117, 118, 119, 184, 212, 213, 214, and 215 in page zero are affected by the ROM random number generator. But which of these bytes constitute the "seeds"? This can be learned by POKE'ing the same number into one of these bytes each time before calling the random number generator and then seeing how it affects the other bytes. When you poke the same value into 212, 213, 214, or 215 and the routine for the random number generator, you will notice that the same number or a brief sequence of numbers will appear in each of the other bytes. This indicates that the "seeds" are bytes 212 through 215.

To fix the OSI random number generator it is necessary to run a routine each time the microcomputer generates a random number, to make periodic adjustments to the seeds. For example, the routine could wait until 50 numbers have been generated and then increase the value at Byte 212, generate another 50 and increase the value at Byte 213, and so on.

Using this routine, I ran my microcomputer for 24 hours, constantly generating random numbers. At the end of that time about 1.5 million numbers had been generated and the pattern had not been repeated. It is unclear how many numbers would eventually be generated before the pattern would be repeated but it would probably be in the millions or billions.

And for most purposes that would probably be sufficient to consider the random number generator adequate.

The only remaining consideration that I can see is how to initialize the random number seed. A fairly random set of "seeds" can be generated by using a loop in which, each time through, the routine to generate a random number is called, then the polled keyboard routine is called to detect closure of a designated key. The loop is repeated and a number of random numbers are generated until the designated key is closed. When it is closed, the values in bytes 117, 118, 119, and 184 are transferred to bytes 212, 213, 214, and 215. Since the amount of time before key closure will vary, a fairly random set of "seeds" will be created. Also, this routine may be repeated to get a very random starting point for the random number generator.

This may seem like a lot of trouble for some programmers, but for those that require a true random number generator it should fit the bill.

AD$

C4MF AND C8DF SOFTWARE

SCREEN EDITOR- Delete/Insert characters anywhere in your basic program. RESEQ- Renumber your basic program. COPY5- Copy up to 5 tracks in one transfer (24K C4MF only). NAME- Names an unnamed program or track. BLOCKADE- Challenging one or two player game. CHECKERS- Play computer using screen board. DEEPSUB- Shoot ships from your submarine. PONG- Fast action two player game. CLOCK- Large letter color alarm clock. LETTER- Scroll large letter color messages. All games have color and sound and use Joysticks or Keyboard. Any 5 programs on 5 or 8in. disk for $30.
Send check to: Rodney Trugman, 1504 Chestnut St., Bowling Green, KY 42101

I will pay $25 for 8 of the female 12 pin molex connectors used for the OSI Bus. Dealer and Factory are no help! Call (312) 662-0924 collect. The call will not be accepted if you are not first. Philip Bryan, Park City, IL

OS-65D 3.2- Complete disassembled listing fully commented. Everything you always wanted to know but OSI wouldn't tell you. Over 40 pages. $19.95.
Software Consultants, 7053 Rose Trail, Memphis, TN 38134

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

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

C2-4P USERS- Starting one-for-one program exchange service. Send SASE for details to:
Patricia Dennis, 7940 Cadwell Rd., Bliss, NY 14024

THE USR(X) ROUTINE

by Fred W. Atchley
Mississippi Memory

Anyone who has witnessed a machine coded screen clear, can visualize the graphic realism that is possible with embedded parameters as part of the USR routine call. In the statement, Z=USR(Y), "Y" would be an input from BASIC, and "Z" would be the result returned to BASIC. How do you pass information between BASIC and machine language code with the USR routine? The following is an explanation. (Ref: vol.2-1, O. S. Small Systems Journal (available from PEEK(65))and the 8K BASIC-in-ROM Reference Manual).

Notice, the term "routine", as in a small machine language program (MLP), which performs a specific task. In BASIC, each separate "command" is performed by calling its respective MLP. The USR routine is a unique BASIC command which allows the programmer to "call" a MLP of his own special design.

A similar "calling" technique is used to pass parameters. BASIC uses MLP's to pass parameters (INPUT X, PRINT X, etc.). Likewise, the MLP's we will use to pass parameters in the USR statement are in ROM, ready-to-use. All we have to do is call them. This example is designed for the C1P.

The following MLP example will accept a number, add a one to it, then return the result, n=n+1, to BASIC.

```
Line#   Loc   Code     Label    Mnemonic
 1.     0FD0  20DB0F            JSR  GETNUM
 2.     0FD3  A5AE              LDA  NUMHI
 3.     0FD5  A4AF              LDY  NUMLO
 4.     0FD7  C8                INY
 5.     0FD8  6C0800            JMP  (OUTNUM)
 6.     0FDB  6C0600  GETNUM    JMP  (INNUM)
```

Line#1 is your MLP entry point. The next step is to get your input parameter from BASIC. You tell BASIC that you want this by executing the MLP, "INNUM". The example MLP does this via a JSR, so that once this step is complete, execution returns to Line# 2. In effect, your MLP has called a BASIC MLP within ROM. Note that the MLP "INNUM" is pointed to by the contents of locations 0006 and 0007 (Line#6, 6C0600).

As a result of executing Line#1, the input parameter, which is a two-byte number, has been stored; the most significant byte in location "NUMHI" ($00AE) and the least significant byte in location "NUMLO" ($00AF).

Line#'s 2 and 3 load these inputs into the A-register and the Y-register so we can use them. The example MLP adds one to the least significant byte by incrementing the contents of the Y-register (Line#4). Nothing is done with the byte in the A-register.

To pass a parameter from your MLP back to BASIC, the most significant byte must be in the A-register and the least significant byte in the Y-register. Since this is already the case, the last step is to jump-relative to the MLP "OUTNUM" (Line#5). This BASIC MLP will pass the result back to your BASIC program and you along with it.

The following BASIC program implements the example MLP.

```
 10 K=4048   REM MLP LOAD POINT($0FD0)
 20 FOR J=0 TO 13   REM LOAD MLP
 30 READ X          REM LOAD MLP
 40 POKE K+J,X      REM LOAD MLP
 50 NEXT
 60 POKE 11,208     REM USR LOW ($D0)
 70 POKE 12,15      REM USR HIGH ($0F)
 80 INPUT Y         REM %=USR(Y)
 90 Z=USR(Y)        REM Z=RESULT
100 PRINT Z
110 GOTO 80
120 REM   THE FOLLOWING DECIMAL DATA
130 REM   ARE EQUIVALENT TO THE
140 REM   HEXADECIMAL MACHINE CODE
150 DATA 32, 219, 15  REM  20DB0F
160 DATA 165, 174     REM  A5AE
170 DATA 164, 175     REM  A4AF
180 DATA 200          REM  C8
190 DATA 108,8,0      REM  6C0800
200 DATA 108,6,0      REM  6C0600
```

To use the program, input a number from 0-254. The program will reply with the number plus one. Good luck.

[ ] $8 Enclosed. Send PEEK(65) for one year (12 issues)

Foreign Subscriptions Air Mail - $20.00/yr.

[ ] $15 Enclosed. Enroll me in OSIO

[ ] $20 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.

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: