PEEK (65)

The Unofficial OSI Users Journal

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

★★ $1.75 ★★

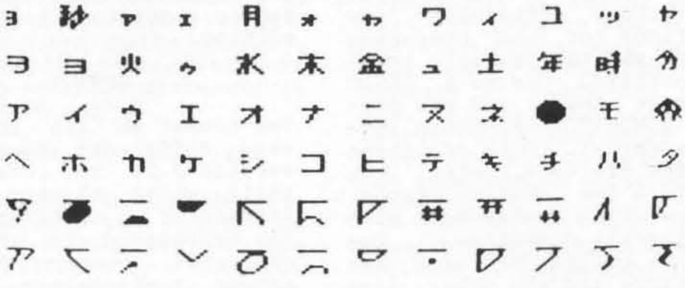
January 1982

Vol.3, No.1

JAPANESE CHARACTER SET

Designed By

Yasuo Morishita



See page 19

Column One

"New things at OSI" has been a repeated theme on this page. This is both because of the natural interest we all feel in what the factory is up to, and because the factory **really** is up to some interesting stuff!

This month is no exception. I have just had an interesting conversation with Chuck Bickoff of -- but that's part of the story. Let me share it with you.

Mr. Bickoff doesn't work for Ohio Scientific; neither does anyone else. He works for M/A-COM OSI, which stands for Office Systems, Inc., at the engineering headquarters in Burlington, Mass. A recent Infoworld article stated that the corporate headquarters would be moving to Burlington, but for the present at least it is just the engineering department; and they are answering the phone "M/A-COM OSI."

So what's new in Massachusetts? Most importantly, OSI is proceeding with all deliberate speed to install a real professional management team, including Chuck, Bill Chalmers the new President and a Vice President of Operations, Phil Collier, formerly with Maganavox.

All these high-powered engineering and management people have already had an effect. OS-65U V 1.3 has been released and is being shipped with all new computers leaving the factory. Manuals are being rewritten as quickly as possible, though this is a major task, particularly considering the evolutionary nature of the software products which they describe.

And the first moves have been made to upgrade and modernize the product line as well. Starting now, these are the machines you can buy from MOSI:

C-100	Polled keyboard, 48K RAM, 1-2 Minifloppies. Much like the old C4P MF, but with a modem, Hi-Res graphics, MDMS, Planner Plus, Music Software, Sargon Chess, 65D 3.3 all included
C-200	Serial I/O, 48K, 2 8" floppies, 65U, much like the C2-OEM
C-210	Serial I/O, 3-chip CPU board, 56K, 2 8" disks, 4 Serial and 1 parallel I/O ports, like a C3-OEM configured for OS-65U or CP/M
C-240	Like the C-210 but in two drawers, with 16-slot backplane, in floor-standing rack; like a 56K C3-A
C-310	Serial I/O, 3-chip CPU board, 56K, 10 Mbyte Winchester hard disk, like the C3-D
C-340	Equivalent to the present C3-C', with 40 Mbyte hard disk
C-380	Equivalent to the present C3-B, with 80 Mbyte hard disk

Lots more exciting things are on the way as well. Look for early release of level 3 time sharing for the C-310, making it a highly competitive multi-user hard-disk machine.

Before long, all the C-300 series machines will feature double-density, IBM compatible floppy disk formats and full 64K user space, with patches to both OS-65U and CP/M to accomodate these changes.

al

OSI OS65D V3.3

by Larry Proteau
1800 Lake Park Drive
Raleigh, NC 27612

Ohio Scientific has released version 3.3 of the OS65D operating system. This new version was designed to make it easier to use the keyboard of BASIC-in-ROM video systems, by emulating the Hazeltine 1410 terminal. The use of random files was also made more efficient. In addition, a number of other useful features were also added. I purchased a copy ($79.00) to run on my C4P/MF system, with one 5 inch disk and 48K of memory.

DOCUMENTATION

The package comes with a three ring binder containing instructions for the operating system, and two properly typeset booklets. One is a BASIC reference manual, and the other is a MONITOR/ASSEMBLER reference manual. Both of these booklets are very useful, but contain a few obvious typos. The operating system book also contains six diskettes. One is blank and may be used for exercises, and the other five are tutorials. These tutorials are accompanied by explanatory text in the notebook, and are designed to teach a beginner all about the system, from the elementary operations through complicated file manipulations. The experienced user is directed to a section explaining the differences between the previous version and this one. The tutorial text is also an excellent reference, and contains many examples. This book is the best documentation I have seen from OSI. It is well written, and accurate, except for some errors in the POKE/PEEK list which has not been properly updated.

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

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

Subscription Rates

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

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

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

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

Mention of products by trade name in editorial material or advertisements contained herein in no way constitutes endorsements of the product or products by this magazine or the publisher.

KEYBOARD

The keyboard driver has been rewritten to emulate a Hazeltine terminal. As a result, the auto-repeat feature has been lost. The REPEAT key will repeat any character when it is held down. The RUBOUT key now works as it should, and the cursor can be positioned anywhere on a line being entered to make character insertions and deletions. The system also includes a small modem program to communicate with time-sharing computers, but having no modem, I could not test it. One of the handy features of the new driver is that lower case and upper case letters can be used as on a typewriter and BASIC does not care which letters are used. This makes the keyboard much easier to handle, especially for text editing.

OUTPUT

The output to the screen is very different than under version 3.2. The cursor is a small white blinking block instead of an underline. It can be changed to any other character, even a blank to remove it altogether. Output always starts at the upper left corner of the screen, and proceeds for 24 lines of 60 columns. Scrolling only starts when the bottom of the screen is reached. Simple print commands or keyboard escape codes can erase the screen instantly, and change from the 32 X 32 to the 32 X 64 format at any time. There is also a "print at" function, similar to that found on unmentionable computers. This allows easy positioning of the cursor at any (x,y) location on the screen within the 24 X 60 boundary. Input can be done wherever the cursor is located. One very interesting feature is the output window. A simple command allows a window of specified size to be created on the screen. All printing and input is automatically done within the window, and POKEs can be used to do graphics on the rest of the screen without affecting the input and output.

OTHER FEATURES

There are several other features and new utilites, some of which can only run under Version 3.2. One of these is a Resequencing command, another is a file copy utility which allows copying files with only one disk drive. I will go into more detail in discussing the new features and utilities at a later date if reader interest is sufficient.

★ ★ ★

AN INTERESTING KEYBOARD ROUTINE

by Martin Ybarra
15856 Ocean Ave.
Whittier, CA 90604

Tired of the long lines of code needed to poll OSI's video keyboard in real time under BASIC? Well, fret no more, because hidden in the DOS extensions at $3180 (hex) and undocumented anywhere (as far as we know) by OSI lies a real time keyboard polling routine. This routine unlike its better known counterpart at $FD00 will scan once and then return if no key press was detected. The routine returns with the ASCII of the key pressed in the accumulator as a null if no key is pressed. It will not, however, auto-repeat. The assembler routine described below takes advantage of this and may be called from BASIC with T=USR(X):

```
START JSR $3180 ; call the
                  keyboard
      TAY       ; routine and
                  store
      LDA #$00  ; the value
                  found at Y
      JMP $1218 ; return to
                  BASIC
```

The value found has been stored at Y. If T=USR(X) is the BASIC program, it is a great deal faster than the equivalent BASIC routine and consumes much less memory. Below is a short BASIC program to demonstrate this:

```
10 REM--POKE IN THE MACHINE
   CODE ROUTINE
20 REM
30 FORX=11984TO11992:READMC:
   POKEX,MC:NEXT
40 REM
50 REM--POKE IN THE USR(X)
   VECTOR
60 REM
70 POKE574,208:POKE575,46
80 PRINT"PRESS ANY KEY FOR
   ASCII VALUE"
90 PRINT"PRESS RETURN TO
   CONTINUE"
100 REM
110 DISK!"GO 252B":REM CHECK
    FOR INPUT FROM KEYBOARD
120 T=USR(X):PRINTT;:GOSUB140:
    GOTO120
130 DATA 32,128,49,168,169,0,
    76,24,18
140 FORS=1TO150:NEXT
150 RETURN
```

★ ★ ★

A MULTIPLE REPORT WRITER IN OS-DMS

By Al Peabody

Many Ohio Scientific computers are now being used by small businesspeople. Many of these users have purchased OS-DMS, Ohio Scientific's Data Base Management System. This article will discuss an undocumented feature of the statistical report writer program in OS-DMS, the ability to store the parameters of many reports, then call them up later quite simply by selecting their titles as desired, rather than answering a long list of questions each time a regularly repeated report is to be run.

More importantly, I hope to present a discussion of the general technique used by this program which will be simple enough to allow computer users who are not accomplished programmers to learn something interesting about how their computer works, while presenting sufficient technical information to allow the slightly more sophisticated user to use similar techniques in his own programs.

If you have a copy of the DMS Nucleus, be brave. First, make a backup copy to preserve your original, then follow me on a trip through the world of report generators. Even if you don't have the nucleus, stick around. I'll try to make the trip worthwhile!

First, a couple of general comments about the way the statistical report writer works. What seems to be one report generator is actually three programs: STAT03, a program which simply asks you a rather long list of questions and stores your answers away on the disk; MFMFRP, a report writer program which reads several DMS Master Files, following your instructions, and produces reports; and STAT3A, a statistical report writer which, like MFMFRP, uses the information which you, the user, have provided as you ran STAT03 to produce informative reports.

We will leave MFMFRP to another article, and discuss the combination of the other two programs, which you use to produce reports concerning the information in a single master file: STAT03 and STAT3A.

Both of these programs are copyrighted by Ohio Scientific, so we will not be able to provide you with listings of the programs. Rather, I will present a listing of a subroutine which I have added to my particular copy of STAT03, which I have no intention of copyrighting, and a discussion of how your computer can use the technique embodied in the routine. If you own a copy of STAT03, you can add the routine directly; if not, you are welcome to use the principles involved.

When you take the menu selection "Generate a Report" on your Nucleus, you are actually running STAT03. As mentioned earlier, all this program does is open a data file (that is, prepare a section of your disk to receive data) and store away your answers to the questions which the program asks you so that the other program, STAT3A, will be able to "find" your answers later, when it is time to print out your report. This is done quite simply. Each question is presented to you with an INPUT statement in BASIC, such as:

```
INPUT"MASTER FILE NAME";MN$
```

This statement in a BASIC program causes the words in quotes to appear on the screen, followed by a question mark, then causes the computer to wait while you type in an answer, which can later be retrieved by the computer by referring to its "name," MN$.

Later on, after other information has been gathered together, the computer can store away the information with the statement

```
PRINT%1,MN$
```

PRINT is the verb in BASIC which causes something to be printed, either on the screen or on the printer, or, if PRINT is followed immediately by the "%" symbol as it is here, on the disk. The "1" following "%" is a shorthand reference to a disk file which has been OPENed previously in the program (as mentioned earlier, OPENing a disk file simply lets the computer know we will be wanting to put some information into it or read information from it). The number is used to keep track of the disk drive, file name and position in the file in case several files are "open" at once. See your OS-65U manual and OSI BASIC manual for a more complete discussion. Finally, the MN$ in this statement is a reference to the very MN$ which you generated by answering the computer's question in the INPUT statement.

So far, quite simple. STAT03 stores information away on a disk file, STAT3A reads it back in again and uses it to decide which file to use, whether you wanted a horizontal or vertical format, what the heading of the report is to be, etc. But here comes the hooker. What STAT3A actually does is to look first of all at INDEX=10 in the disk file (meaning at the 10th character in the file, starting at the very beginning) to get something it calls INDX, then input some more information concerning housekeeping, then go to the position INDX and get the heading and the other information you gave it about the report you wanted.

Let me explain that. The program STAT3A looks into the continuous stream of characters recorded on the disk, to the 10th cha-acter, and sees what number is written there. Suppose it finds the number 600. A little later, it goes to the 600th character in the file, and starts reading out the information about your report.

When I saw this, I was very excited. I write many reports using this program; some, like my monthly list of business transactions, must be run over and over again, and each time I have to answer all the same questions over and over again to generate the report. Even if I remember all the answers, I sometimes make typos and discover that the report didn't run right. Arrgh. But here, it seemed to me, was an answer. All I would have to do would be make sure the information for ALL the reports I run regularly was in the data file, and somehow let the computer know where the information for each report started, then I could just make my selection!

The listing accompanying this article does the trick. It is a subroutine which I wrote and added to STAT03. Have a look at it with me.

Line 5005 starts keeping count of how many reports we have listed for our operator to select from.

Line 5010 OPENs the data file.

Line 5020 lets the computer know that the end of the data file is at character number 10000, so we won't keep looking after we have already looked at all the reports in the file.

Line 5030 tells the computer to look through the data file super quick and find the first time the characters "^^" occur. By simply putting these two characters, not likely to occur in normal prose, at the front of each title we put into the data file (the computer does this automatically for us right after it INPUTs the heading), we can be pretty sure if we find these characters, we have found a report title!

Line 5040 checks to see if any titles were found before we went past the end of the file.

Line 5050 keeps track of how many titles have been found, and where in the file they were found. If you plan to have more than 10 titles in your file, you will have to add the statement DIM KP(20), or however many you think you might eventually have on your list, somewhere near the beginning of the program.

Line 5060 reads the title off the disk, then "throws away" the first two characters, since we know they are "^^" and are meaningless.

Line 5070 prints out the number and the title which we have found, on the screen, so you can have a look at it.

Line 5080 "loops" back to search further into the file, to see if there are more reports for the list.

After line 5200 asks if you want to run one of the old reports, line 5300 determines which one, and line 5310 sticks the position in the file of the beginning of the information on the report you have chosen at the magic spot, position 10, so that STAT3B will find it and run that very report.

Just about the only other part needed is an extra line which asks, after you have input all the information for a new report, whether you want to save that report on the list of reports. If you do, another couple of simple lines look at position 0 in the data file, where the true end of file is kept, write your information on the new report just past that point, then update the end of data file information at position 0.

With these simple additions, the statistical report writer is much easier to use and much less prone to careless errors. However, even if you don't have the Nucleus, or any of the DMS, you can use this same technique. If you have a complicated program to write, one which requires lots of data input by the operator, why not write it as two programs, one to input the data and store it away on disk, another to read the disk data file and actually run the program?

This type of data file, called a SUBMIT file in some systems, works very well for any computer process which requires a great deal of operator input and is frequently re-run.

Add your own refinements, like maybe letting folks take MOST of the information from a submit file and input just what is unique to each situation, and write a report for PEEK (65) about your application.

```
5000 REM RUN PREVIOUS REPORT?
5005 C=0:REM COUNTER
5010 OPEN"STAT3B",1:REM DATA
     FILE
5020 EF=10000:REM END OF FILE
5030 FIND"^^",1:REM ^^ MARKS
     HEADING
5040 IF INDEX(1)>EF THEN 5200
     :REM NO MORE TO SEE
5050 C=C+1:KP(C)=INDEX(1):REM
     POSITION OF A HEADER
5060 INPUT%1,H$:H$=MID$(H$,3)
     :REM SKIP OVER ^^
5070 PRINT C;") ";H$:REM COUNT
     AND HEADING
5080 GOTO 5030:REM GET NEXT
     ONE
5200 INPUT"RUN ONE OF THESE
     AGAIN";AG$
5210 IF LEFT$(AG,1)="Y" THEN
     5300:REM DO ONE AGAIN
5230 RETURN:REM DON'T DO ONE
     AGAIN
5300 INPUT"WHICH ONE";NN
5310 INDEX<1>=10:PRINT%1,KP
     (NN):REM SAVE POSITION OF
     ONE TO REDO
5320 CLOSE:RUN"STAT3A:REM AND
     GO DO IT!
```

DISK COPYING WITH A SINGLE EIGHT-INCH DRIVE

by Willis Cook
1298 Renee Drive
Lilburn, GA 30247

Last month's article described interfacing an 8" floppy drive to a C2-4P or C4P. If you make this modification, you will have to live with the relative inconvenience of copying disks without the benefit of two drives, where you simply copy from drive A to drive B.

With a single drive, copying is done by filling available RAM memory with as much material from a disk as it will hold, then replacing the source disk with the target disk and dumping the first portion, re-inserting the source disk and loading the second portion, and so on. The copy program given here will copy a full disk (73 tracks) in twelve swaps. Making a copy of OS-65D and all utilities (through track 27) requires four swaps. I always put the OS and all utilities on every disk, and this is the only time I use this disk copying program.

The program is written for a 32 kbyte machine. Since the BASIC workspace starts at $317E and ends at $7FFF, approximately 20 kbytes are free. The program itself requires just under 1800 bytes, so some 18 kbytes are available for disk storage without interfering with the program that is telling the computer what to store and read back to disk. This is enough space to hold six 11-page tracks.

The CREATE utility defaults to 12-pages per track, but for some reason only eleven pages are stored on each track under OS-65D. Maybe one page is required as overhead. This is why the SAVE commands in the program only copy up to $B (11 decimal) pages. Some software is commercially available, however, that puts twelve pages on a track, so if you ever have trouble making a back-up copy of a Word Processor or Terminal Simulator you bought, check to see if this is the case. If so, change the "B" in the SAVE commands to "C".

OSI's utilities are all in the 11-page per track format, but the operating system itself, and some of its work files, are in smaller segments. For example, track 8 is divided into four sectors, two for the directory and two BASIC work files.

When copying a track, if it contains more than one sector, each sector must be copied separately. So track 8, for example, requires four copy commands. Also, you cannot copy an empty track. You will notice in the program that tracks 11 and 12 are skipped because they are not used by OSI. (There is nothing to prevent you from using them, however, I use track 11 for a machine-language screen-clear

OSI

AARDVARK NOW MEANS BUSINESS!

OSI

WORD PROCESSING THE EASY WAY– WITH MAXI-PROS

This is a line-oriented word processor designed for the office that doesn't want to send every new girl out for training in how to type a letter.

It has automatic right and left margin justification and lets you vary the width and margins during printing. It has automatic pagination and automatic page numbering. It will print any text single, double or triple spaced and has text centering commands. It will make any number of multiple copies or chain files together to print an entire disk of data at one time.

MAXI-PROS has both global and line edit capability and the polled keyboard versions contain a corrected keyboard routine that make the OSI keyboard decode as a standard typewriter keyboard.

MAXI-PROS also has sophisticated file capabibilities. It can access a file for names and addresses, stop for inputs, and print form letters. It has file merging capabilities so that it can store and combine paragraphs and pages in any order.

Best of all, it is in BASIC (OS65D 51/4" or 8" disk) so that it can be easily adapted to any printer or printing job and so that it can be sold for a measly price.
MAXI-PROS – $39.95

NEW-NEW–NEW TINY COMPILER

The easy way to speed in your programs. The tiny compiler lets you write and debug your program in Basic and then automatically compiles a Machine Code version that runs from 50-150 times faster. The tiny compiler generates relocatable, native, transportable machine code that can be run on any 6502 system.

It does have some limitations. It is memory hungry – 8K is the minimum sized system that can run the Compiler. It also handles only a limited subset of Basic – about 20 keywords including FOR, NEXT, IF THEN, GOSUB, GOTO, RETURN, END, STOP, USR(X), PEEK, POKE, -,=,*,/,(,),<>,Variable names A-Z, and Integer Numbers from 0-64K.

TINY COMPILER is written in Basic. It can be modified and augmented by the user. It comes with a 20 page manual.
TINY COMPILER – $19.95 on tape or disk

THE AARDVARK JOURNAL

FOR OSI USERS – This is a bi-monthly tutorial journal running only articles about OSI systems. Every issue contains programs customized for OSI, tutorials on how to use and modify the system, and reviews of OSI related products. In the last two years we have run articles like these!

1) A tutorial on Machine Code for BASIC programmers.

2) Complete listings of two word processors for BASIC IN ROM machines.

3) Moving the Directory off track 12.

4) Listings for 20 game programs for the OSI.

5) How to write high speed BASIC – and lots more –

Vol. 1 (1980) 6 back issues - $9.00
Vol. 2 (1981) 2 back issues and subscription for 4 additional issues - $9.00.

ACCOUNTS RECEIVABLE – This program will handle up to 420 open accounts. It will age accounts, print invoices (including payment reminders) and give account totals. It can add automatic interest charges and warnings on late accounts, and can automatically provide and calculate volume discounts.

24K and OS65D required, dual disks recommended. Specify system.
Accounts Receivable. $99.95

* * * SPECIAL DEAL – NO LESS! * * *

A complete business package for OSI small systems – (C1, C2, C4 or C8). Includes MAXI-PROS, GENERAL LEDGER, INVENTORY, PAYROLL AND ACCOUNTS RECEIVABLE – ALL THE PROGRAMS THE SMALL BUSINESS MAN NEEDS. $299.95

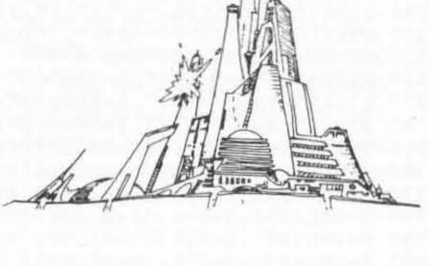
P.S. We're so confident of the quality of these programs that the documentation contains the programmer's home phone number!

SUPERDISK II
This disk contains a new BEXEC* that boots up with a numbered directory and which allows creation, deletion and renaming of files without calling other programs. It also contains a slight modification to BASIC to allow 14 character file names.

The disk contains a disk manager that contains a disk packer, a hex/dec calculator and several other utilities.

It also has a full screen editor (in machine code on C2P/C4)) that makes corrections a snap. We'll also toss in renumbering and program search programs – and sell the whole thing for – SUPERDISK II $29.95 (5 1/4") $34.95 (8").

AND FUN, TOO!



BOOKKEEPING THE EASY WAY – WITH BUSINESS I

Our business package 1 is a set of programs designed for the small businessman who does not have and does not need a full time accountant on his payroll.

This package is built around a **GENERAL LEDGER** program which records all transactions and which provides monthly, quarterly, annual, and year-to-date PROFIT AND LOSS statements. GENERAL LEDGER also provides for cash account balancing, provides a BALANCE SHEET and has modules for DEPRECIATION and LOAN ACCOUNT computation.
GENERAL LEDGER (and MODULES) $129.95.

PAYROLL is designed to interface with the GENERAL LEDGER. It will handle annual records on 30 employees with as many as 6 deductions per employee.
PAYROLL - $49.95.

INVENTORY is also designed to interface with the general ledger. This one will provide instant information on suppliers, initial cost and current value of your inventory. It also keeps track of the order points and date of last shipment.
INVENTORY - $59.95.

GAMES FOR ALL SYSTEMS

GALAXIAN - 4K - One of the fastest and finest arcade games ever written for the OSI, this one features rows of hard-hitting evasive dogfighting aliens thirsty for your blood. For those who loved (and tired of) Alien Invaders. Specify system – A bargain at $9.95

NEW – NEW – NEW

LABYRINTH - 8K - This has a display background similar to MINOS as the action takes place in a realistic maze seen from ground level. This is, however, a real time monster hunt as you track down and shoot mobile monsters on foot. Checking out and testing this one was the most fun I've had in years! – $13.95.

NIGHT RIDER - You've seen similar games in the arcades. You see a winding twisting road ahead as you try to make time and stay on the road. NIGHT RIDER uses machine code to generate excellent high speed graphics - by the same author as MINOS.
NIGHT RIDER – $12.95 cassette only

THIEF - Another machine code goody for the C1P cassette only. You must use mobile cannon to protect the valuable jewels in the middle of the screen from increasingly nasty and trigger happy thiefs. Fast action and fun for one or two players. THIEF $13.95 on C1 cassette only!

SUPPORT ROMS FOR BASIC IN ROM MACHINES – C1S/C2S. This ROM adds line edit functions, software selectable scroll windows, bell support, choice of OSI or standard keyboard routines, two callable screen clears, and software support for 32-64 characters per line video. Has one character command to switch model 2 C1P from 24 to 48 character line. When installed in C2 or C4 (C2S) requires installation of additional chip. C1P requires only a jumper change. – $39.95
C1E/C2E similar to above but with extended machine code monitor. – $59.95

Please specify system on all orders

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

OSI

**AARDVARK TECHNICAL SERVICES, LTD.
2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110**

OSI

routine, so my version of the copy program includes track 11.) Notice, too, that you can copy the tracks in any order. I mixed them up a little so that they would fit my memory space better.

Originally, I wrote the program to copy the 27 tracks used by the operating system and utilities only. But sometimes you may want to copy a disk that has some of your applications programs on it, so the program has been generalized to copy as many tracks as you wish, in multiples of six. This is the code from line 270 to the end.

Since you can't copy empty tracks, a problem develops if you attempt to copy a disk with, say, 30 tracks. After the fourth swap, you will have copied through track 27. On the next swap the program will attempt to copy six more tracks, but after copying three it will come to a blank track and stop. What I do in this case is use the copy program to copy the maximum number of tracks possible without coming to a blank track, end the program by entering a CR without a space, then copy the remaining tracks separately. If you forget, however, there is no harm done. All the valid tracks will have been loaded into RAM and the program will terminate on either line 280, 290 or 300, depending on where the first blank track occured. You can either manually enter the SAVE commands as in lines 320, 330 and 340, or you can simply enter GOTO 310, after putting the target disk into the drive. The program will then copy the last six tracks from RAM to your disk. Since you didn't end up on a proper track number, at least the last track, and maybe more, will be useless (actually a duplicate of a previous track) and you may have to clear it, but this hasn't been a problem for me.

It is important to remember that when you copy track 8 you copy the directory of the source disk onto the target disk. This does NOT mean that all the files named in the directory are present. They aren't unless you have copied the tracks they reside on. This may sound silly, but I've forgotten to copy the last few tracks on a disk, and not discovered it until trying to run something that wasn't there. Also notice that the copy program doesn't copy track 0. This is done manually as follows:

```
With the source disk in the drive, enter
        EXIT                (Enter the OS kernel.)
        A* CALL 0200=01,2   (Load the track 0 copier.)
        A* GO 0200          (Start execution.)
        2                   (Select track 0 copier.)
        R 3200              (Read track 0 to memory starting
                             at address $3200.)

Now insert the target disk and enter
        W 3200/2200,8       (Write 8 pages from memory to disk
                             with addresses starting at $2200.)
```

If you have less than 32 k in your machine, you will have to modify the program to copy less material at each pass. With 24 k, the memory space ends at $5FFF so you would have to stop the first set of CALLs after line 70, dump it to the target disk, then start the second swap with line 80. Memory is so cheap that if you value your time at all, it is more economical to add RAM than to re-write the program.

```
10 REM     ****   DISK   COPIER   ****
20 A$="*** Replace MASTER with NEW disk. ***"
30 B$="* Replace NEW disk with MASTER. *"
40 PRINT"WITH MASTER DISK IN DRIVE, ENTER <SPACE, CR>";
50 INPUT" TO START";C$  :DISK!"CALL 3A00=01,1"
60 DISK!"CALL 3F00=01,2":DISK!"CALL 4400=02,1"
70 DISK!"CALL 4F00-03,1":DISK!"CALL 5A00=04,1"
80 DISK!"CALL 6500=05,1":DISK!"CALL 7000=06,1"
90 DISK!"CALL 7A00=08,1":DISK!"CALL 7B00=08,2"
100 DISK!"CALL 7C00=08,3":DISK!"CALL 7D00=08,4"
110 PRINT:PRINT A$:GOSUB 430
120 DISK!"SAVE 01,1=3A00/5":DISK!:SAVE 01,2=3F00/5"
130 DISK!"SAVE 02,1=4400/B":DISK!"SAVE 03,1=4F00/B"
140 DISK!"SAVE 04,1=5A00/B":DISK!"SAVE 05,1=6500/B"
150 DISK!"SAVE 06,1=7000/A":DISK!"SAVE 08,1=7A00/1"
160 DISK!"SAVE 08,2=7B00/1":DISK!"SAVE 08,3=7C00/1"
170 DISK!"SAVE 08,4=7D00/1"
180 PRINT:PRINT B$:GOSUB 430
190 DISK!"CALL 3A00=07,1":DISK!"CALL 4300=09,1"
200 DISK!"CALL 4E00=10,1":DISK!"CALL 5900=13,1"
210 DISK!"CALL 6400=14,1":DISK!"CALL 6F00=15,1"
220 PRINT:PRINT A$:GOSUB 430
230 DISK!"SAVE 07,1=3A00/9":DISK!:SAVE 09,1=4300/B"
240 DISK!"SAVE 10,1=4E00/B":DISK!"SAVE 13,1=5900/B"
250 DISK!"SAVE 14,1=6400/B":DISK!:SAVE 15,1=6F00/B"
260 DISK!"SAVE 11,1=7F00/1":PRINT:PRINT B$:GOSUB 430
270 D$="16":E$="17":F$="18":G$="19":H$="20":I$="21"
280 DISK!"CALL 3A00="+D$+",1":DISK!"CALL 4500="+E$+",1"
290 DISK!"CALL 5000="+F$+",1":DISK!"CALL 5B00="+G$+",1"
300 DISK!"CALL 6600="+H$+",1":DISK!"CALL 7100="+I$+",1"
310 PRINT:PRINT A$:GOSUB 430
320 DISK!"SAVE "+D$+",1=3A00/B":DISK!"SAVE "+E$+",1=4500/B"
330 DISK!"SAVE "+F$+",1=5000/B":DISK!"SAVE "+G$+",1=5B00/B"
340 DISK!"SAVE "+H$+",1=6600/B":DISK!"SAVE "+I$+",1=7100/B"
350 D=VAL(D$):D=D+6:D$=RIGHT$(STR$(D),2)
360 E=VAL(E$):E=E+6:E$=RIGHT$(STR$(E),2)
370 F=VAL(F$):F=F+6:F$=RIGHT$(STR$(F),2)
380 G=VAL(G$):G=G+6:G$=RIGHT$(STR$(G),2)
390 H=VAL(H$):H=H+6:H$=RIGHT$(STR$(H),2)
400 I=VAL(I$):I=I+6:I$=RIGHT$(STR$(I),2)
410 PRINT"COPIED THRU TRACK";D-1:PRINT:PRINT B$GOSUB 430
420 GOTO 280
430 INPUT"ENTER <SPACE, CR> TO CONTINUE";C$:RETURN
```

★ ★ ★

AN INTERESTING SCREEN WINDOW

by Jim Lin
15856 Ocean Ave.
Whittier, CA 90604

Up to now, there has not been a simple and easy way of manipulating screen sizes with the C4P under OS65D3. In this article, I am going to describe some of the POKEs necessary to construct windows of almost any size. I will also briefly mention a semi-fast screen clear in BASIC.

The screen clear can be found in line 10 in the listing below. The POKE 9770,255 will make each line scroll with about four linefeeds, also the PRINTCHR$(10) will create another linefeed, therefore, it is only necessary to loop it four or five times (rather than 32 with just PRINT statements). The POKE 9770,64 restores it to normal scrolling.

Below is the list of POKEs for manipulating the screen:

POKE 9761, (208-215): This poke will protect the top part of the screen. Pokeing a '208' will be normal scroll and a '215' will scroll only one line at the bottom. (NOTE: POKE 9770,0 is equivalent to POKE 9761,215)

POKE 9736, (0-63): This poke will move the listing to right.

POKE 23, (0-132): This poke determines the screen width. '132' is normally the default value (even though you will never see anything longer than 64 columns).

POKE 9667,208, 9675,209, 9683,210, 9720,211, 9727,212, 9734,913 9746,214, 9781,215 9756,216. This series of pokes will protect the bottom half of the screen. Pokeing a '208' will scroll just the top line, however, if you have overscan, you may not see it at all. Pokeing a '215' will be back to normal scroll. Also, you must poke the same value in all the locations.

Special thanks to Charles Curly (of OSUIN) for supplying these pokes.

Below is a sample program to illustrate this feature:

```
1 REM The following line is a
   quick screen clear for
   OS65D3
2 REM
10 POKE9770,255:FORX=1TO4:
    PRINTCHR$(10):NEXT:
    POKE9770,64
20 A=214
30 B=209
40 C=10
50 D=30
190 REM The following pokes
        protect the bottom of
        the screen.
195 REM
200 POKE9667,A:POKE9675,A:
        POKE9683,A:POKE9720,A:
        POKE9727,A
210 POKE9734,A:POKE9746,A:
        POKE9781,A:POKE9756,A
220 REM
225 REM This poke protects
        the top part of the
        screen.
230 REM
240 POKE9761,B
250 REM
260 REM This poke puts the
        listing to the right
270 REM
280 POKE9736,C
290 REM
300 REM This poke moves the
        list to the left
310 REM
320 POKE23,D
```

HOYT'S TRANSMOGRIFIED SAVE PROGRAM

by Kerry Lourash
1220 North Dennis
Decatur, IL 62522

Another machine language SAVE program, you say? Well, this one is super short, extremely easy to use, and fits right alongside the XMON (cassette version). After developing a ML routine with XMON, save it on tape with a single command. You can even specify the address the computer goes to after it loads the ML program (a good default address is $FE00).

Patch Transmo into the XMON by using the procedure in my article "Notes on the OSI XMON" (Sept. issue). Since I use "U" instead of "@" for the memory view/change command, I changed:

```
$0960 - #$B1
$0961 - #$07
```

The format of the command is : @XXXX = YYYY,ZZZZ . YYYY - Start of memory to be saved. ZZZZ - End of memory to be saved. XXXX - Entry address (after load is completed).

If you want a space after every byte printed on the screen, insert a JSR 0C51 after JSR 0AAC at 07CF. The BPL instruction at 07E5 should be changed so it still goes to the LDX #00 instruction. If you wish to load ML programs and go to BASIC, insert LDA #$4C and STA $00 after the STA 0205 at 07FA. This assures location zero is not changed by noise on the tape. The entry address to BASIC warmstart could be $0000, $A274, or $A27D, depending upon your preference.

XMON Graphic Dump

This is a short routine that helps find text in ML programs. It's similar to the "Z" command, but prints graphics characters instead of hex numbers. Pick one of the unused letter commands in the XMON (see OSI manual) to call this routine. LDX #$20 controls the number of characters per line. C1P users should change #$20 to #$10. Hitting the space bar will cause another line of characters to be printed. Try looking at memory starting at $A080.

Hoyt's Transmogrified SAVE Program

```
0000             *
0000             *
0000             *
07B1                 ORG  $7B1
07B1             *
07B1  20480B     *   JSR  $B48      GET PARAMETERS
07B4  20F7FF         JSR  $FFF7     SET SAVE FLAG
07B7  207AFF         JSR  $FF7A     10 NULLS TO TAPE  C2P = FF78
07BA  192E           LDA  #$2E      "."
07BC  206108         JSR  $861       OUTPUT CHAR
07BF  A5DD           LDA  $DD       OUTPUT START ADDRESS
07C1  A6DC           LDX  $DC
07C3  20780A         JSR  $A78
07C6  A92F           LDA  #$2F      "/"
07C8  206108         JSR  $861      OUTPUT CHAR
07CB  A200           LDX  #0        OUTPUT ONE BYTE
07CD  A1DC           LDA  ($DC,X)
07CF  20AC0A         JSR  $AAC
07D2  A90D           LDA  #$D       OUTPUT CR
07D4  20B1FC         JSR  $FCB1     C2P = BF15
07D7  E6DC           INC  $DC       INCREMENT BYTE COUNTER
07D9  D002           BNE  $7DD
07DB  E6DD           INC  $DD
07DD  A5DE           LDA  $DE       SEE IF DCNE
07DF  C5DC           CMP  $DC
07E1  A5DF           LDA  $DF
07E3  E5DD           SBC  $DD
07E5  10E4           BPL  $7CB      LOOP IF NOT DONE
07E7  A92E           LDA  #$2E      "."
07E9  206108         JSR  $861      OUTPUT CHAR
07EC  A5DB           LDA  $DB       OUTPUT ENTRY ADDRESS
07EE  A6DA           LDX  $DA
07F0  20780A         JSR  $A78
07F3  A947           LDA  #$47      "G"
07F5  206108         JSR  $861      OUTPUT CHAR
07F8  A900           LDA  #0        RESET SAVE FLAG
07FA  8D0502         STA  $205
07FD  4C0908         JMP  $809      RETURN TO XMON
0800                 END
```

XMON Graphic Dump

```
0000             *
0000             *
0000             *
0782                 ORG  $782
0782             *
0782  20110B         JSR  $B11      GET START ADDRESS
0785  20070B         JSR  $B07      LINE FEED, CR
0788  A5DB           LDA  $DB       PRINT ADDRESS
078A  A6DA           LDX  $DA
078C  20780A         JSR  $A78
078F  A220           LDX  #$20      # CHARACTERS PER LINE
0791  20510C         JSR  $C51      PRINT SPACE
0794  A000           LDY  #0        GET BYTE
0796  B1DA           LDA  ($DA),Y
0798  206108         JSR  $861      PRINT IT
079B  E6DA           INC  $DA       INCREMENT BYTE COUNTER
079D  D002           BNE  $7A1
079F  E6DB           INC  $DB
07A1  CA             DEX            DECREMENT CHAR. COUNTER
07A2  D0F0           BNE  $794      LOOP IF NOT DONE
07A4  20070B         JSR  $B07      LINE FEED, CR
07A7  2000FD         JSR  $FD00     GET KEY
07AA  C920           CMP  #$20      IS IT A SPACE?
07AC  F0DA           BEQ  $788      YES, PRINT ANOTHER LINE
07AE  4C3208         JMP  $832      NO, BACK TO XMON
07B1                 END
```

★ ★ ★

DIGITAL DESIGNER'S NOTEBOOK

by: Bruce Showalter
857 Cedar
Abilene, TX 79601

This article discusses a few items which weren't covered in "Getting There With TTL". First, let's examine the use an NAND and NOR gates as inverters.

I've ended up with a surplus of 7402s in my TTL inventory, due to an aborted project which required several. So, I use them in place of 7404s as inverters. To do this, one may tie both inputs of a gate together and use them as a single input. However, this doubles the load to the signal being inverted. I prefer to use only one input and tie the other to ground. If the incoming signal isn't required elsewhere, then I tie the inputs together.

7400s are also useful as inverters. Simply tie one input to +5 volts, and use the other as the inverter input. The 7400 is preferred in this matter, since there is much less power consumed by any input tied to +5 volts rather than to ground. The EXclusive OR becomes an inverter when wired exactly like a NAND gate (one input tied HIGH). Other inverter alternatives are the 7410 and 7420 NAND gates, or the 7427 NOR gate.

Incidentally, there is absolutely no need to use a resistor between +5 volts and the input being tied HIGH. It simply wastes time, resistors, and circuit board space.

I briefly mentioned buffers in the "Getting There" article. A buffer increases the fanout of the input signal. My preference is the AND gate, but OR gates work as well. The OR gate uses more power, since one of its inputs is grounded. Other AND gates are found in the 7411 and 7421. When wired exactly like an OR gate (one input LOW), the EXclusive OR gate becomes a buffer, too. Tri-state buffers can be permanently turned on by tying the Enable line HIGH or LOW as required. Although they cost more, they can be real space-savers on circuit boards.

It is preferable to employ gates with the desired output, rather than to tack on an inverter. This is because of the time it takes for an input signal to produce an output. A typical inverter takes 15 nanoseconds (billionths of a second) to react. This time is called propagation delay. It's practically instantaneous to you and me, but it becomes significant when you're clocking a microcomputer at one megahertz or more.

At one megahertz, the clock will be HIGH for 500 nanoseconds, then LOW for 500 ns. During this small piece of time, the 6502 CPU will send out an address and read or write some data. This is why RAMs and ROMs are rated in access time. They must respond to the CPU in the time allowed, or the data will not be communicated. If there are many propagation delays between the CPU and the memory ICs, a communications gap results. One solution is to either decrease the clock frequency or employ faster RAMs and ROMs. But it is better to carefully design the circuit so that propagation delays are minimized.

The C1/Superboard has an example of deliberate propagation delay. A signal called T3 is derived by sending the clock through 3 AND gates. Each gate delays the signal about 20 ns, for a total delay of about 60 ns. This allows other data to arrive at its destination in sync with the clock.

So far, I've talked only about TTL. This "family" of devices is the largest and most-used. However, the amount of circuitry that can be built into one IC (called scale of integration) is limited. Other families were developed to overcome this. The first family of large-scale integration was PMOS. Next came NMOS, developed primarily to circumvent the negative voltages that PMOS requires. The newest and fastest growing, is CMOS.

The rapid growth in the popularity of CMOS is due to a number of factors. Mainly, it is compatible with both NMOS and TTL. Like NMOS, CMOS requires very little power to drive it. Fanout with itself or NMOS is at least 50. And, like TTL, CMOS works on a single, positive voltage. A 5-volt TTL supply will power CMOS nicely.

To drive TTL, though, CMOS must be buffered. The easiest way to do this is to tie two or more inputs together on one CMOS gate, then use its output to drive one TTL input (one LS

cont. on p. 13

cont. from p. 10

TTL input would be even better). Two or more CMOS gates could be paralleled for still greater output. There are special CMOS buffers as well.

CMOS has its own version of tri-state logic, called transmission gates. One of these is basically two parallel CMOS transistors with a common control input. When the control input is in the disable mode, the two transistors present a high impedance to the circuit, much like a switch turned off.

The main flaw of the MOS families is in the area of speed and propagation delays. CMOS speed is highly voltage-dependent. At 10 volts, CMOS is about as fast as TTL. But a 5-volt supply seriously retards CMOS action. Propagation delays almost triple. This is why the MOS families have not yet superceded TTL in digital logic applications.

CMOS adapts well to analog or linear applications, more readily than TTL. This is due in part to its ability to operate from a voltage as high as 18. Add to this its low current drain, and you see why battery-powered circuits employ CMOS. Thus, CMOS has found a home in mobile and automotive uses.

If you plan to employ CMOS circuits in the future, I urge you to read a databook and/or applications guidebook for CMOS.

★

WORD PROCESSOR - 3.2
BETTER, BUT STILL HAS PROBLEMS

by Linda G. Christie
5536 E. 11th
Tulsa, OK 74112

When I first got WP-3 for my C3 OEM, I was elated. So many of the time consuming problems of WP-2 had been corrected, it was a pleasure to use the new system. No more worries about line numbers. No more slow cursor moving for editing. And, tabs at last!

First impressions are deceiving, I found out. I'd acquired some very nice features and some new problems as well.

Let's examine the good things first.

CURSOR CONTROL - EDIT MODE

The WP-3 FIND command puts the cursor right on the word to be changed. WP-2 just presented the line on which the word occured. If it appeared on 5 different lines it listed all 5 lines and then I selected the line I wished to edit.

The cursor now can be moved immediately to the right margin with CONTROL-R and to the left margin with CONTROL-F. This makes a significant difference in efficiency. WP-2 required the cursor to travel space by space across the screen.

SCROLLING TEXT - EDIT MODE

Using the UP ARROW and the DOWN ARROW to scroll the text forward and backward is also helpful. However, if the cursor is not on a line of text to be edited when the UP ARROW is activated, the text will scroll down instead of up.

Also, scrolling is a misnomer for the UP ARROW function. The whole text doesn't move on the screen. The next previous line is written below the last line presented, i.e., the screen will present line E, D, C, B, A from top to bottom on the screen. This reverse order is very difficult to proof read so what I do is run it back to the line where I want to start proofing and then run it forward A, B, C... so I can read the text in a normal top to bottom direction.

PERSONALIZING TEXT

The ^A (ESC A) code stops the printer at a given point in the text so data can be entered from the terminal. This allows me to personalize letters in the inside address and in the body of the letter. A nice touch for this application.

TAB, NEW LINE, AND AUTOMATIC RETURN

Thanks for the tab! I write fiction and every line might be a new paragraph of dialogue. Having to indicate end of line (^#), spacing 5, and the quotes to begin the dialogue nearly broke my hand on WP-2. Now my only problem is the ^# (ESC #). Why does it require two characters that are difficult to reach on the keyboard to indicate a new line? (I never learned my numbers and symbols by touch anyway.) I've gotten around this difficulty by using an odd character ([) instead and then changing it to ^# when I've finished. (CHANGE"[","^#".) This works fine except for when it appears at the beginning of the text. For some reason, it won't change the symbol. Knowing this, I can manually edit line 10 or 20 where it appears.

The automatic RETURN at the end of the screen is very convenient for entering text. Returning to a conventional typewriter is like using stone axes.

SOME CUMBERSOME PROBLEMS

I've run into a couple of them already while typing this manuscript. At times, I wish the programmers had left out some of the fancier aspects and polished the everyday routines.

FIND COMMAND

It's nice that the cursor goes straight to the word to be edited. However, I miss the display that WP-2 gave me of every line where the word occurs. With WP-3, if I want to FIND "display" and it appears five times in my manuscript (some editors might say I was over using the word), I must push RETURN, F, RETURN to see the next occurance - step by step until I find the fifth one so I can change it to please my editor.

For some strange reason, which I've not quite figured out, sometimes the FIND command will present the END OF TEXT message. Then I must try again.

EDITING WHILE CREATING TEXT

If I make a mistake while entering new text, I can't immediately scroll back with the UP ARROW to the previous line to correct the mistake. First I must hit return twice to get the dot (.). Then I must type z (zen) to have the monitor display the last line of the text in the edit mode. Then I enter SHIFT UP ARROW to move back to the previous line for editing - 5 entries counting returns just to back up one line to correct a mistake.

If I try using the FIND command, I must hit return twice, enter FIND, and then correct the error - 4 steps. If UP ARROW worked while new text is being entered, the process certainly would be simpler.

ADDING TEXT

Adding text in the middle of a line is a real hassle. If the text added is more than what will fill the line, I must first use CONTROL - B to drop the last part of the line down. So far so good. I then add part of the new text to the rest of that line and push RETURN. At that point I get several dots (.) on the screen. I'm not sure what's happening, but if I try to type NEW or N to add more text, I get more dots. Finally, it will accept my command to let me enter new text. In the beginning I was foolish enough to do this and then on reviewing my manuscript, I found that the portion of the line I had dropped off appeared before this new text instead of after.

Not to be defeated by a mere machine, I devised a new method. I let it give me all of the dots it cares to until it shows me the next line of text (By the way, it's not the text which I dropped down, but the next line! I can't be sure what happened to the portion of the line I dropped down at this point.) Then I enter SHIFT UP ARROW. This will display the dropped off portion and next, the line which I originally added text to. Then I push RETURN to get the dot immediately after the line to which I added text. At this point, I enter N to allow new text to be entered. Whew! If I can remember what I wanted to add, I continue. I'm not even going to count the steps for this one. It's too depressing.

Another problem with adding new text is that it will not accept it very fast so I tend to override the system particularly at the AUTOMATIC RETURN operation. I lose 2-3 letters here if I'm not careful. A "beep" to let me know we're going to the next line might be helpful. Or better still, why can't it accept the text at the same speed as usual? Aren't computers supposed to be fast? I'm not even an accomplished typist for heaven's sake! I just type because I have to.

USING DELETE KEY

Deleting complete lines or blocks of text by using the DELETE key is time consuming. Also, the line number still exists in the text. The main problem with this is that if I'm stepping through the text line by line to edit, when the cursor comes to and "empty" line number, it goes crazy.

If this empty line occurs at the end of the text, I'm in real trouble. There's no way to resume entering text at the END OF TEXT. The z (zen) command moves the cursor to the empty line and it bounces like a ball to some remote place on the screen. No amount of coaxing will make the cursor behave. I must ask for a printed display (p700-) to determine the last line number (the empty one) and having done so, DELETE the line number (D830). Then I can type Z or ZEN to continue. Using D CONTROL - D will eliminate this problem.

MOVING TEXT/TRANSFERRING TEXT/ DELETING TEXT

I lump all of these processes together because they all use the same procedure which has BIG problems. DOCUMENTATION. The documentation doesn't tell me how to move, transfer, or delete text with the CONTROL - D code.

For those of you who have been pulling your hair out, after many hours of discussion and experimentation, we have the solution. You may want to get your manual out and write in these simple instructions:

ENTERING A CONTROL - D: To enter a CONTROL - D, press the CONTROL KEY DOWN and with it depressed, strike the key "d" or "D". A flashing colon will appear on the screen at the point the cursor was located.

DELETING TEXT WITH CONTROL - D:

1) Place a CONTROL - D at the beginning of the text you wish to delete. A flashing colon will mark the spot.

2) Go to the end of the text you want to delete and place another CONTROL - D. Again the flashing colon will mark this spot.

3) Exit from the edit mode by pressing RETURN. A DOT (.) will appear.

4) Type "D CONTROL - D". That is, enter capital "D" and without spacing press the CONTROL KEY DOWN. Holding the CONTROL key down, press "d". The screen will show a capital D with a colon flashing after it. (PLEASE NOTE: THE USE OF QUOTE MARKS IN THIS EXPLANATION DOES NOT MEAN TO ENTER QUOTES ON THE KEYBOARD.)

5) Hit RETURN and wait until the dot appears. At this point the text that appeared between the first two flashing markers will disappear.

MOVING OR TRANSFERRING WITH CONTROL - D:

1) Place a CONTROL - D at the beginning of the text you wish to move or transfer. (Transferring moves the text without erasing the original text from it's original location. That is, it duplicates the text.)

2) Place a CONTROL - D at the end of the text you wish to move or transfer.

3) Exit from the edit mode by pressing RETURN. CAUTION: PRESS RETURN A COUPLE OF TIMES SINCE THE AUTOMATIC RESEQUENCE PROCESS MAY NOT ALLOW YOU TO BE IN THE COMMAND MODE IMMEDIATELY.

4) Move the cursor to the spot in the text where you wish to move or transfer the text to and enter CONTROL - B. The text to the right will drop down a line.

5) Press RETURN. Once the DOT (.) has appeared, enter "M CONTROL - D" for MOVE or "T CONTROL - D" for TRANSFER. NOTE: MAKE SURE THAT YOU ARE POSITIONED RIGHT AFTER THE TEXT YOU WISH TO FOLLOW. YOU MAY HAVE TO SCROLL BACKWARD AND FORWARD TO INSURE THE CORRECT PLACEMENT.

6) To get rid of the blinking cursors, enter C":"," "... Translated this is capital C; quote, CONTROL - D, comma, quote, space, quote. This will change the blinking colon to a space. To shorten this you can omit the last space and quote.

One problem we haven't solved is how to move text to the END OF TEXT. When we use the above procedure, the last line of the original form will always appear at the end of the text. If anyone has a solution, perhaps they will let us know.

The question I have is, when is OSI going to take that extra time to write user manuals in the manner I've just done? Not until they do, will they be able to satisfy the great demand for word processor systems. Or, for small business systems. The average user cannot and will not spend hours deciphering the manuals.

WHY BOTHER AT ALL?

At this point you might be saying, why doesn't she buy an Apple or a Radio Shack computer? They won't do the business applications we need. Also, I don't have another $5,000.

Basically, I feel that OSI has a good system with poor user documentation and a few troublesome bugs that they can correct if they want. I do know one thing, I wouldn't think of returning to my typewriter.

★ ★ ★

COUNTER POINT - WP 3.2

Virtually everything you read here in PEEK (65) is produced by WP 3.2. If anyone gives it the acid test, we do! Like Linda, we wouldn't trade down to WP-2 for anything. Hopefully, we can share a few thoughts with Linda, and maybe even help explain a few of the strange doings.

Find Command

We too, like the cursor on the word to be edited, but it stands to reason that it can not be on the word, ready for editing, in each of 5 occurences within the text at the same time. If we need to see every occurence, we use WP-2. In the mean time "F <return>" is not too great a price to pay. If you are bugged by having to do this 5 times to get to the word you are looking for, just try adding a space and the first letter of the next word within the quotes of the FIND command - or the last letter of the preceeding word, a space and the word. This should give the FIND something unique to find and thus jump directly to the one you want. A caution - if the two word parts appeared on two different lines as you typed them (not in the formatted output) then FIND won't find. FIND looks at just one input line at a time.

^# is awkward! That's why many people have changed this three key command to a single character, a tilde or some other "odd" character.

Editing While Creating

Both WP-2 and 3.2 as well as WP6502 and even Word Star go through somewhat similar routines. Word Star does eliminate the necessity to scroll first back and then forward, but it still takes 5 or 6 key strokes to get the job done. The problem here is that we are dealing with serial terminals that only print out, serially, what they are sent.

Adding Text

The question here is where do all those added characters go? Basically, in a serial machine, the remainder of the text must be moved down one character for every character that is added. This could be done on a character by character basis, but that would really slow things down. So

WP-3.2 waits until a entire line has been typed and then counts the number of characters added and then moves the back-end down by that much. The more text that has to be moved, the longer it takes. While WP is moving the text, you are merrily typing, but WP is not ready to accept it. That's why it skips those 2 or 3 letters.

See the cautioning note on page 33 of your manual and keep your files smaller, or if you have a large chunk of text to add, type it after a ZEN and later Transfer the whole thing back where you want it.

General

We too wonder about the quality of the documentation, but we understand that it is in the process of being made into a "real" manual. Just for the record, can you imagine how long it would take to go back and insert all those little spaces on a typewriter to make all these lines come out right justified? If someone goes back to the typewriter, it won't be us!

Demarcators

These neat little marks are great for some things, but it is frequently easier when deleting, moving or transferring whole paragraphs to use the old WP-2 line orientated command, even if you have to use a CTRL B to get a clean line break.

by Production staff
PEEK (65)

LETTERS

ED:

My comments in PEEK (65), criticizing the legibility of FORTH programs previously drew the wrath of a FORTH fanatic - so I write reluctantly, fearing the abuse which may be heaped upon me by CP/M "freaks."

I concur with most of your observations about CP/M in a previous issue, especially with respect to CP/M's SLOW disk access!! Let me draw a few comparisons to 65U and CP/M as implemented for us by Lifeboat Associates.

First, a word about Lifeboat's support. I must have spent at least $30.00 in phone calls to their technical department trying to learn how to automatically run an application upon booting up. (On one occassion, I spent four minutes on "hold" while the Lifeboat receptionist waited to connect me with a technician. This is unpleasant anytime, but especially so when calling 2,000 miles during "prime time".) A promised returned phone call was never received and Lifeboat's conclusion was that with OSI CP/M, it couldn't be done.

Next followed phone calls to the "hotline" run by Digital Research. They sent an application note promptly which detailed, (if you can call Digital Research documentation detailed), how to auto-execute. They gave me advice over the phone on how to "DDT" CP/M so that it would execute a SUBMIT file, which is very easy to prepare or modify. Wouldn't work. More phone calls. Final conclusion: when Lifeboat made the BIOS changes necessary to run CP/M on OSI, they made the cold boot and warm boot re-entry points the same which kept the system going in a loop when using SUBMIT. But I still learned how to auto-execute using "DDT", which I detail as shown below. In any event, auto execution is very difficult in CP/M in comparison to the ease of running your program choice from BEXEC* in 65U. For the "freaks" that can't see why this is important, I can only say that I believe that programs written for small businesses ought to be as simple and uncomplicated as possible and auto execution into a menu is an important aspect of this simplicity.

A major problem with OSI and CP/M is that the floppy step rate problem relating to Siemens drives has not been resolved. Despite the fact that there are problems, OSI will not "fix" this problem under normal warranty, and Lifeboat says OSI told them there shouldn't be any problem and they are unwilling to modify their CP/M. Running CP/M at 2 megahertz, is impossible with my Seimans and though much better at 1 megahertz, is still not totally reliable. Conclusion: don't buy CP/M without testing it first if you have Seimens drives.

I have always considered OSI documentation as marginal at best, but that was before I tried reading the Digital Research manuals on CP/M. There were seven different manuals supplied from Lifeboat. The Lifeboat manual was the easiest to understand, but there were errors in how to select the CA101 printer board for output. I was also informed by Lifeboat that you cannot use any but the first port on the CA10x board. So forget any idea of running two serial printers, which I like to do (e.g. a Diablo on the first port and a TI on the 2nd). In any event, the proper LST device for the CA101 is CRT:, not UL1. The Digital Research manuals are more confusing than anything I have seen from OSI. Despite the fact that we are buying version 2.2 of CP/M, most of the manuals relate to version 1.4 and you are somehow supposed to read the one 2.2 manual and figure out what changes apply to the other manuals.

CP/M utilities are another learning experience. Despite the flexibility of PIP, for example, it certainly is not as easy for a beginner as is COPYFI in 65U, for example. Another advantage of 65U is that these utilities are written in BASIC and can be easily modified or customized for end-user applications. Not so with CP/M.

If CP/M is so bad, why am I using it??? Because, as our Editor noted previously, there are some great CP/M applications. Where else can I get a combination word processor/ analytical tool like TMAKER? Only CP/M supports the great new data base management tool, DBASE II. (Incidentally, OSI dealers can get a demo package

cont. on p. 18

cont. from p. 16

of DBASE II at no risk.) Since it supports a FIND command, somewhat similar to 65U, and has an extremely powerful programming language (intelligible, unlike some languages!), I am tempted to adopt it as my "standard".

CP/M may be the "software buss", but it is not for the uninitiated.

Modifying CP/M to Auto-Execute

(1) First, create a CP/M system and save as a data file:

```
  A> OSI GEN
     Source Drive Name A
     Destination Drive Name RETURN

  A> SAVE 50 CPM.SYS
```

(2) Next, "DDT" CPM (This is like an editor into the operating system.)

```
  A> DDT CPM.SYS
     - S 987
       0987 00 - Type in the number of characters in your .com
                 file (TMAKER = 6)
        988 20 -
               - } Enter the hex equivalents to the characters
               -   in the .com file

               .  When finished, type a period then type
        GO        GO
```

(3) Finally, save the patched system on drive A:

```
  A>    OSI GEN
        Source Drive Name RETURN
        Destination Drive Name A
```

Tom Stover
Gering, NE 68341

* * * * *

Tom:

1) Our experiences have differed considerably on Lifeboat's support. Our experiences have all been good (phone calls returned promptly, etc).

2) Our Siemens drives work fine, but we have had trouble transferring disks from one machine to another, perhaps sensitive to alignment. Doesn't WFBIOS.COM help you?

3) OS-CP/M 2.2 from Lifeboat works fine on any CA-10 port. Use DDT to change $20F6 to 02 for port 2, 04 for port 3, 06 for port 4, etc. Or, POKE this location in any program before PRINTING.

4) I modified one disk to auto-execute and got the message FILE A : COPYRIGHT not found. When I inserted the auto execute characters, it just read the next characters in the buffer (Copyright Lifeboat Assoc. 1980) and added them to the command line!

5) CP/M seems to have been written by and for hackers and enthusiasts, not businessmen. Like you, I await an OS which will be good for both. OASIS, maybe?

Al

* * * * *

ED:

After that terrible letter I sent you a few weeks ago (an exercise in self-pity and frustration vented toward OSI), I want to make up for it by writing something useful. I recently purchased Jim Sander's Terminal program and it merits some laudatory comments.

TERMINAL SIMULATOR FOR OS 65-U

The above title is the name of a fairly new product, sold by J.H. Sanders (J & T Assoc., 2338 Riviera Dr., Vienna, VA 22180), Version 1.21 dtd August 1, 1981.

Before I review this product, I must advise my fellow readers that I have another terminal simulator, sold by another outfit, which up to now still doesn't work properly, and which was more expensive to boot! (No pun intended.)

Jim Sanders' program is a gem, not only because it works well, but because it is simple to use. I tested it by dialing up 'The Source' (to which I subscribe), and following Sanders' very complete documentation, I was able to temporarily store and then print from memory storage, on the printer, something which heretofore wasn't possible (unless, I assume, one is using a terminal to which a printer is directly connected, or, one uses a KSR).

My hookup isn't complete yet (the handshaking lines aren't on my 550 board yet to the modem), but despite that, the Sanders program worked with only slight dropouts (caused by lack of proper timing, etc).

Concisely, the Sanders' product is designed to make your terminal 'semi-smart', i.e. you can access your modem via your 550 board (or 430, or 555 board), and at some given time in the program, by means of an ESC sequence (which doesn't transmit to the distant computer (in the case of 'The Source', PRIME, via Telenet or Tymnet), you can command your computer to store incoming data. The data can then be saved on disk (if you so desire), and it can subsequently be printed out on your printer (now there, it would be very handy to have despooler, which I haven't got, because the only one I know of is written for hard disk (I have a C3S1)).

The Terminal program is part BASIC and part ASSEMBLER. Basic controls the menu functions; assembler does the work after you've connected to the distant computer, and it is very fast.

There are a lot of important features:

1. Display: you can display what has been accumulated in memory, on screen.

2. Print: you can print the accumulated data, from memory or from disk storage; Sanders' program includes commands for condensed or bold printing for MX-80; I changed these without too much effort for the coding for my Centronics 737-1 (parallel/incidentally, the BEXEC* includes settings (POKEs) for Serial as well as Parallel).

3. You can selectively place an EOF indicated at the end of the accumulated data.

4. You can 'ADD' text to the beginning of the stored array. This is particularly handy to place identifying remarks ahead of the data you're receiving, or adding program lines, or whatever. Thus far, I've used it to make labels

(when too lazy to fire up a DMS disk), i.e. the disk has OSU Editor program on it, and as such, if you want to type an address on one envelope, you can do it using the ADD command. I'm sure that's not what it was intended for, but it works very well in that respect.

5. You can store the incoming data (or send data from) any of your drives. The program asks you to specify UNIT. There are two large data files on the disk supplied by Sanders, but you can specify FILENAME on any UNIT.

6. There's a method to deal with long lines (those over 71 characters).

7. You may return to BASIC at any time.

8. You can begin or quit saving data when you want to.

9. You can quit saving data including or excluding the current (last) line.

10. You can change the full/ half duplex flag.

11. You can abort a send operation.

12. The data file STAYS OPEN until you tell it to close.

13. You can override the BUSY flag.

14. There's a null count switch (particularly handy when you have a slow printer).

15. You can change the modem echo.

16. You may linefeed the remote computer, or linefeed only YOUR console.

There's lots more, and above all, the documentation is very COMPLETE (Something most of us consider a rarity!). I recommend that if you do lots of data transfer or work with SOURCE or other similar services, between yourself and distant or remote computers, this is one program you need to get and well worth the price of $27.50.

Frederick S. Schaeffer
Jamaica, NY 11435

* * * * *

ED:

I have recently added a serial interface to my C4-P and can send material to my Epson printer at 4800 baud even though my machine is cassette based and this aspect still works at 300. Would you be interested in details?

Why is Yasuo Morishita so hung up in machine screen clear when the Cegmon monitor chip has a very fast clear in its program? Also along those lines, I did not buy this chip until much later than did Yasuo, and after he bought his, the ads changed and stated that the monitor chip did not completely fix the collection problem and offered the basic 3 chip at a third of what I paid for mine. The monitor also allows a rapid change in the screen color without for... next loops by poking each color cell. This changes values in $224-226. Does anyone know where to poke to change the color when doing this?

I have installed the D&N Micro CM9-F memory and disc controller and have a Shugart SA-400 5 1/4" drive. Does anyone know how to make or where to buy the data separator board to make this work or even if it will work?

Finally, I need to buy a transistor (2N5226). This is needed for the serial port. I have something close but OSI calls for this one and perhaps if I had this one, it would fix my difficulties in getting my printer to accept output when first using it. I think the cassette could work at 600 if I knew what to poke in dec. 64512. Does anyone know what values cause this to run at different baud rates?

Stephen Rydgig
Collinsville, IL 62234

Steve:

An article on a data seperator appeared in the November 1981 issue of PEEK (65). A more detailed article appears in BYTE, Feb. 78.

The problem wth your C4P and the serial printer port is that most RS-232 receivers like to see what is called zero crossing. Look at your C4P schematic and you will see that the emitter is at +5 volts while the collector is at -9 volts.

To solve your problem, you need only add -9 volts to your C4P. This is done simply by connecting a 9 volt battery in the following manner. Hook the plus side (+) to ground (+) and the negative side (-) to minus 9 volt point. The 2N5226 is also a Sylvania ECG 159 which you can get from most any TV parts place or a close substitute would be Radio Shack part number 276-2023.

Please do send us an article on adding a serial interface to your C4-P.

Brian

* * * * *

ED:

In a previous letter I mentioned that the stock OSI character generator chip can be replaced by a 2716 EPROM to give custom characters. The following photo shows an example of what can be done. This may be the only OSI computer in the world which speaks Japanese!

Note that one character is not Japanese, but is really an alien invader!

E. Morris
Y. Morishita
Midland, MI

* * * * *

Editor's Note:

For those who need to write in Japanese, there is a thimble for the Nec Spinwriter which will produce Japanese characters.

* * * * *

ED:

First, in reference to the letter by M.J. Petyak (Wilkes-Barre, PA). I am a member of the Philadelphia Area Computer Society's (PACS) OSI User's Group. We have members with a wide range of experience in hardware and software. Regular members represent one superboard (highly modified), two C4PMF, and (through my company) a C3C, and one of the old Challengers (400 CPU board, etc). I think we could provide the kind of help needed, both hardware and software. I myself, have: interfaced a Centronics printer using the 6820 PIA, interfaced a modem using the 6850 ACIA (on a populated PERF-board), interfaced 2708 and 2716 proms (interlaced into the Challenger's upper 16K memory). I own a single trace oscilloscope that does o.k. setting the pots for video and cassette. I'll bring it to a meeting if requested. The meetings are 11:00 on the 3rd Saturday of the month. They are held at Lasalle College (20th St. &

Z-FORTH IN ROM by Tom Zimmer
5 to 10 times faster than Basic. Once you use it, you'll
never go back to BASIC! $ 75.00
source listing add $ 20.00

OSI FIG-FORTH True fig FORTH model for 0S65D with fig editor
named files, string package & much more $ 45.00

TINY PASCAL Operates in fig-FORTH, an exceptional value
when purchased with forth.
TINY PASCAL & documentation $ 45.00
FORTH & TINY PASCAL $ 65.00

SPACE INVADERS 100% machine code for all systems with
64 chr. video. Full color & sound on C2, 4P & 8P systems. The
fastest arcade program available. $ 14.95

PROGRAMMABLE CHARACTER GENERATOR $ 99.95
Use OSI's graphics or make a complete set of your own! Easy
to use, comes assembled & tested.
2 Mhz. boards $109.95

PROGRAMMABLE SOUND BOARD $ 74.95
Complete sound system featuring the AY-3-8910 sound chip.
Bare boards available. $ 29.95

32/64 CHARACTER VIDEO MODIFICATION $ 39.95
Oldest and most popular video mod. True 32 chr. C1P, or 32/64
chr. C4P video display. Also adds many other options.

ROMS!!!
Augment Video Mod with our Roms. Full screen editing, print
at,selectable scroll, disk support and many more features.
Basic 4 & Monitor $ 49.95
Basic 3 $ 18.95
All 3 for $ 65.00

65D DISASSEMBLY MANUAL. by Software Consultants
First class throughout. A must for any 65D user. $ 24.95

NUMEROUS BASIC PROGRAMS, UTILITY PROGRAMS AND GAMES ALONG WITH HARDWARE PROJECTS. ALL PRICES ARE U S FUNDS. Send for our $1.50 catalogue with free program (hardcopy) Memory Map and Auto Load Routine.

CHARGEX VISA

master charge

OSI Software & Hardware
3336 Avondale Court
Windsor, Ontario, Canada N9E 1X6
(519) 969-2500

3281 Countryside Circle
Pontiac Township, Michigan 48057
(313) 373-0468

progressive computing

Olney Ave., 12 blocks North of US 1 on Broad St.) in the Science Building. The PACS OSI user Group's official address is: PACS OSI User Group, c/o Al McCann, 30 S. Elmwood Ave., Glenolden, PA 19036.

Second, in reference to the letter by Don Colin (Defuniak Springs, FL). I wrote a program to set up, in memory, the binary encoded "words" for all 256 OSI ASCII and graphics characters. Note that they can be edited, etc., before sending them to a 2716 Eprom. I also have the routines to drive an Optimal Technology prom programmer. I upgraded my 440 video board to the full 256 character set. (I still have some problems with the keyboard accessing interference. I intend to change the keyboard to the PIA on board the 400 CPU board.) The program and encoding reside on 8" disks under OS65D V3.2. Given some encouragement, I'll try and get it on 5 1/4" floppy. The best interface is for the requestor to set up the new characters on a disk and send it to me (or have someone else burn it onto the 2716). (I burn intel 2716's, I believe that's what fits into the modified ROM socket.)

Third, in reference to my letter earlier this year, I'm surprised no one picked up on the 470 board timing change I mentioned . We've run over 6 months with very infrequent disk errors after this change.

John Christiansen
Glenolden, PA 19036

* * * * *

ED:

In reference to Mr. M.J. Petyak's letter, (PEEK (65) Vol 2, #10, page 12).

1) RS232 Mod.

If he has an early version of C4P, he can find an RS232 modification space on his 502 board. But this circuit will be driven by the 6850 A.C.I.A. which is used for cassette port communication in the standard machine. If he can forget a cassette port, it is not so hard to modify it for RS232 use.

If he wants to have both cassette and RS232, he has to add an extra circuit. One solution would be to use a mechanical switch such as a rotary switch or relay, the other may be to use a few extra parts and have a software selectable switch. The latter one can be seen in PEEK (65) Vol. 2, #2, page 3-4. This modification is what I'm using.

2) Cassette LOADing problem

If your C4P is out of warranty already, you may open the C4P up and adjust R75 10K ohm pot to make the DL period 555 microseconds. You have to watch the wave form of pin #5 of U22 74123 while LOADing some program. Sam's manual says set it at 500-645 microseconds. You may try several different cassette LOADs and find the best value. (With a brand new C4P, I had a problem LOADing the Assembler/Editor (OSI) until I did the adjustment, DL was 625 microseconds in the new C4P).

Another solution for critical volume setting would be to add a 741 OP-amp prior to the cassette speaker input. In this case you are required to have ±5V power supply. You may use single voltage OP-amp to eliminate the use of -5V power supply.

Please note that to do the above, you should have an oscilloscope and some electronics experience, especially with ICs.

Good luck!

Yasuo Morishita
Elk Grove Village, IL 60007

* * * * *

ED:

I recently was blessed with an OSI C3C by my company. Unfortunately there are very few personal games that can be played on it. I would like to know if it could be modified to play some games, how to do it, and what would it cost?

Wayne Teres
Clark, NJ 07066

Wayne:

There are a few games that were written for your C3 that are available. Contact your local OSI dealer for details.

As you know the C3 is a serial terminal machine and as such will not support the fancy video games that are available for the video machines, i.e. OSI C1 and C4, Apple, Atari, Radio Shack, without the addition of either a graphics terminal or OSI's 540 series video boards.

Unfortunately all of OSI's software support for their 540 series video boards address the board in the timeshare work space and hard disc buffer areas. (D000 - EFFF). This means you cannot use a video board and hard disk drive in the same machine. To do so would damage your hard disk.

Brian and Al

* * * * *

ED:

In the September 1981 issue of Compute, Jim Butterfield points out a bug in the PET garbage collector. The same bug exists in OSI BASIC in ROM. The problem with collecting subscripted string variables has been well discussed previously in PEEK. The program below demonstrates another problem with the OSI garbage collection. This bug does not cause a system crash and screen flickering, but changes the content of stored strings. The program will eventually cause an out of memory error. However, before this happens, you can see the "=" gets lost from the building string.

```
100 X=FRE(8)/4-10
110 DIM A(X)
120 Z$=Z$+"X"+"="
130 PRINTZ$:GOTO120
```

```
RUN
X=
X=X=
X=X=X=
X=X=X=X=
XXXXXXXXX=
XXXXXXXXXXX=

?OM ERROR IN 120
```

Lines 100 and 110 consume all of memory except 30 bytes, thus forcing a garbage collection during string concatenation. This occurs both with the OSI original ROMs and with the corrected code I presented in the June 1981 PEEK. The bug occurs in this contrived example when a garbage collection occurs during concatenation of three strings. This would happen extremely rarely in a real program.

Earl Morris
Midland, MI 48640

* * * * *

ED:

I wrote you recently with comment on the PEEK (65) operator D+7*(D>9). I have found another use in a one line DEC/HEX converter:

```
15 A=INT(D/16):B=D-A*16:H$=
   CHR$(B-7*(B>9)+48)+H$:D=A:
   IFD<>0THEN15
```

Well, almost, if you want to use it twice, H$ must be nulled. So how about a three line program with input/output?

```
10 H$="":INPUT"DEC";D
15 (AS ABOVE)
20 ?"HEX"SPC(2)H$:?:GOTO10
```

Actually if you needed conversion internal to a program H$="" could be stuck in the line before or after conversion. Also, note there is no program limit to numerical value.

Ray! for the PEEK operator. I think it's neat. Actually an IF statement without routing or with parallel output. (Analog will show.)

Harry Hawkins
Burton, SC 29902

Harry:

Thanks. I have used your 3-line convertor in the middle of a program already and it works fine, fast and neat!

Al

* * * * *

Editor's Note:

I am really embarrassed, it appears that my response to the question posed by a PEEK (65) reader concerning OSI's parallel printer interface was not totally correct. My thanks to Joe Linden of Four State Microcomputers. It appears that Rev A and Rev B boards do have all eight data bits interfaced to the printer. However, OSI's printer driver routine masks the eighth bit. So even if you have a Rev A or Rev B board you still need to change the printer driver routine. Again our thanks go out to Joe, who it appears has solved this problem. The solution is to Poke 15900 and 15901 with a 234. For those of you who like machine code/assembly language insert EA (NOP) into $3E1C and $3E1D. This NOP's the masking of the eighth bit. You should now be able to use all the nice graphics supplied with the printers.

Brian

* * * * *

FROM THE PEEK [65] CBBS

ED:

S.O.S. Help the hearing impaired - read on... SYS-OP and users, I would recommend trying out the Youngstown area computer club CBBS. Jimmy Isabella did a fantastic job of enhancing the CBBS framework which runs this system. We are grateful to the mother CBBS for helping our club.

Does anyone have any ideas on how to convert this CBBS to both BAUDOT/ASCII capabilities?? We are trying to make our CBBS accessible to the hearing impaired with TTY's. If you have any ideas, leave me a message on this CBBS, on our CBBS (216)757-3711, or on hex (301)593-7033.

Thanks a lot, hope to hear from you.

Bob Barth

* * * * *

ED:

DMX V1.1 works fine but I am having trouble using it with multi users. I know of the need for the PRINT #5!. But this does not solve the problem of hanging up the current user after a report has finished. I need any help anyone has. Chuck Sulka of Digital Technology is aware of this problem but has no solution as of yet.

Dick

* * * * *

ED:

Does anyone out there have any idea how to rewrite the Centronics 737 right margin justification instructions that comes with the Atari 825 documentation (the Atari 825 is the same thing as a C 737)...I have this machine, and to my knowledge no OSI user has as yet been able to do right margin justification... It does everything else! Like selection of 3 different fonts-proportional, condensed and default, and underline, backspace and heavy printer. If you are (hopefully) a former Atari owner, please let me hear from you!

Fred Schaeffer
Jamaica, NY 11435

* * * * *

HAPPY NEW YEAR!