PEEK (65)

The Unofficial OSI Users Journal

February 1986
Vol. 7, No. 2

INSIDE

Column One

Welcome to the New PEEK[65]! Actually, about the only thing that's new here is me. The first thing I have to say is a huge "Thank-you" to Eddie and Karin Gieske. They have bent over backwards to make the transition from coast to coast as smooth as possible. And, of course, together they have been keeping this magazine going during the worst period in the history of Ohio Scientific computers. Believe me, we all owe them a lot. Eddie did a good job of introducing me last month. I hope most of you recognize my name from my articles in PEEK and Micro and from the OSI SIG on CompuServe.

As you can see, PEEK hasn't changed much. Forgive the type faces and typos. There wasn't much time to put this issue together and I had to use some of the articles Eddie and Karin had put together. We'll look a lot more solid next month.... I hope.

The question at hand is what changes will I bring to PEEK[65]? The way I see things is more in terms of growth instead of change. PEEK started in response to the derth of information provided by OSI. Users, dealers, and hobbyists had virtually zero documentation back in those days. Things have improved somewhat since then, but not much. PEEK[65] provided a forum for sharing the tips and tricks that users had dug out on their own.

Of course, during the past 10 years, Ohio Scientific has changed quite a bit from the pizza parlor in Hiram, Ohio. Since the demise of the video systems, PEEK[65] and Ohio Scientific have been on largely divergent paths. The company, their dealers, and independent vendors have come to rely solely on the business market for their continued existence, whereas PEEK[65] has remained a bastion for those of us who like to program or play with the hardware. Clearly, PEEK[65] cannot continue on this basis. If the people who read PEEK aren't the audience the advertisers need to reach, they'll stop advertising. By the same token, if PEEK[65] starts publishing solely for the business market the current readership will vanish.

I see only one solution to this problem and that is growth. PEEK[65] must begin to publish articles that will be useful to the people who use OSI systems in their businesses and to actively solicit their subscriptions. That doesn't mean that the current emphasis on personal applications has to suffer. I love to write and given a choice, I'd much rather write about a nifty patch to the operating system than an accounts payable package. No contest. You can count on PEEK[65] to continue to publish articles for us "hackers", both the hardware and software variety. By the same token, I've found that a lot of things that help the business user also help the rest of us in one way or another, so I think everyone will benefit.

PEEK[65] is also going to appear to be a bit more mercinary than it ever has before. From now on, you are going to see a lot of advertisements for products directly from PEEK[65]. There are a lot of reasons for this, not the least of which is that I have to pay for the magazine. That, coupled with the fact that my production costs are higher than the Gieske's had back in Maryland means that the subscription prices are going to go up.

As of April 1, 1986, the one-year subscription price will be going up by $3.00 to $22.00. Foreign subscriptions will go up $4.00 per year. I know that's a lot of money, and that's why I didn't make the price rise effective immediately. Most of you will have ample opportunity to see the changes I'll be making in the magazine before having to re-subscribe. Overall, I think you'll find you're getting more bang for the buck than you ever have before.

One benefit of my purchase of PEEK[65] is that I also run the OSI Special Interest Group Bulletin Board on CompuServe. As soon as possible, I will be making all of the programs that are published in PEEK available for downloading via modem in the OSI SIG. OSI SIG is a fabulous tool for all OSI users. See my article for details.

Enough pontificating. I just want to sum up by saying thanks to all of you who have supported the OSI community in the past and I hope I earn your continued support of PEEK[65]. I'm looking forward to the work that lies ahead for all of us.

Rick

WAZZAT CORNER!

By: L. Z. Jankowski
Otaio Rd 1, Timaru
New Zealand

OSI stuff does not support full printer control. This is very annoying. I got fed up with having to reach over to the printer to turn it off, or on, or to remove the Centronics plug. I wanted to know, from BASIC or WP65Ø2, when the thing was off, off-line, out of paper, or dead. BASIC programs had to be able to use the "TRAP" command to recover from printer error. The solution was found in "PRINTER CONTROL".

I use the remains of a Superboard, with an Epson MX-8Ø printer accessed via a PIA addressed at $CØØ4-7. Data travels from the computer to the printer thru the port at $CØØ6. The software that does the work is in Listings 1 & 2. Listing 1 is best POKEd up from BEXEC*. Listing 2 is the Assembler listing for the code in Listing 1.

HARDWARE

When interfacing printers, it is important to note if an independent printer Line-Feed is required and if the Strobe line should be set HIGH or LOW. My article in the July '84 issue covers these points and the PIA hardware connections that have to be made for a parallel interface to an MX-8Ø. The connections are from the two PIA ports, A and B. From PaØ-Pa7, and from PbØ and Pb1. Four more lines (from Pb2, Pb3, Pb4 and Pb5) need to be taken to Centronics printer plug pins 12, 31, 32 and 13.

THE PRINTER

The four printer lines accessed are PE, INIT, OFF LINE and SLCT. Normally, they are LOW, HIGH, HIGH and HIGH. To the computer they are IN, OUT, IN and IN.

To INIT the printer the signal must be taken LOW then HIGH - see Listing 3. This piece of code can be incorporated into "HOOKS" (see PEEK(65) Dec '83) and called with "I*".

The SLCT line goes HIGH if the printer is in the selected state. It could be read from BASIC to detect if the printer was off or on. This would be an essential part of any BASIC program if the TRAP command was unavailable.

Use, 'IF PEEK(P) AND 32 <> 32 THEN PRINT "Printer error"'.

If the printer runs out of paper, the PE line goes HIGH. If the printer is off-line, then the printer pulls that line LOW.

THE PROGRAM

The three messages in PRINTER CONTROL are longer than the rest of the code! The messages could be replaced by the single word "PRINTER". The word "ERROR" is supplied by BASIC - from the routine at $Ø462. It is this routine that also provides the link to TRAP.

The program begins by saving the contents of the accumulator on the stack and by changing the contents of the output distributor to SCRDEV - the screen. The true contents of OUTDST are temporarily stored in TEMP. TEMP could be made a zero page location, say $D8, to make the code ROM-able. ($D8 is used by the Extended Monitor). The value of OUTDST is changed now to save lengthy code later. Three error conditions are possible. They are tested for, one by one, by testing bits Pb3, Pb2 and Pb4 in PIA port B. On error the appropriate branch is taken to M1, M2, or M3 - the three error messages. If there is no error the correct value is restored to the output distributor and the accumulator. The jump is then taken to PTROUT - send the character to the printer.

Yup, all this happens for every character to be printed! Tedious, but remember that a computer is just a pattern of off-on switches!

OUTPUT DISPATCH TABLE

Now for the cunning part. The code can be switched in or out by POKE 8983,lo: POKE 8984,hi. The "lo" byte must be one less than the actual value. My program begins at $F282 so my two POKEs are: POKE 8983,129: POKE 8984,242.

The POKEs are to $2317 and $2318 and replace $9E (158) and $24 (36). These are the values for parallel printer device 4.

The jump address for device 1 is at $2311 and $2312, (contents $CC, $24). If device 1 is to be used then the POKEs would be 'POKE 8977,lo: POKE 8978,hi.

continued on page 9

```
1 REM LISTING 1
2 :
6 FORX=9375TO9390:READA:POKEX,A:NEXT:PI=49156
7 DATA 141,4,192,206,6,192,238,6,192,169,2,44,6,192,208,251
8 POKEPI,255:POKEPI+1,4:POKEPI+2,9:POKEPI+3,4:POKEPI+2,9
```

```
1 REM Listing 2
2 * = $249F
3 :
10 STA $C004
20 DEC $C006
30 INC $C006
40 LDA #$02
50 there BIT $C006
60 BNE there
70 RTS
```

```
1 REM Listing 3
2 :
10 LDA #1
20 STA $C006
30 LDA #9
40 STA $C006
50 JMP UPDATE
```

published monthly
Editor: Richard L. Trethewey

Subscription Rates	Air	Surface
US		$19
Canada & Mexico (1st class)		$26
Europe	$38	$30
Other Foreign	$43	$30

All subscriptions are for one year and are payable in advance in US dollars. For back issues, subscriptions, or other information, write to:
PEEK[65]
P.O. Box 586
Pacifica, CA 94044 415-993-6029

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

OSI SIG

The OSI Special Interest Group (SIG) on CompuServe provides a national forum for users to exchange messages, information, and programs via modem. The SIG is made up of three disctinct areas.

The primary area is the message base. The message base is a bulletin board, much like those that are available across the country. Unlike local bulletin boards, however, OSI SIG spans the whole country, and in fact, CompuServe is available from virtually any country in the world. Users can address messages to other members individually, or to the group as a whole. This is a great way to ask questions and get feedback very quickly. OSI SIG members are a very helpful bunch, and it's seldom that a question doesn't get answered within 24 hours.

The second area of OSI SIG is the Data Library. This is where we store files in order to make them available to the members. There are currently well over 3 megabytes of program and information files available in the Data Library. Special software in the CompuServe system allows us to transfer programs as either text files or 8-bit object code. Three error-correcting protocols are available, including the popular XMODEM or Christensen protocol, or you can simply have the system send the files "as-is" if your terminal program doesn't support the transfer protocols.

The third area of OSI SIG is the on-line Conference area. The Conference system lets users talk to each other directly. When you type a line in the Conference area, that line is immediately sent to all of the other users who are also in the Conference area. The Conference system is also used by CompuServe's CB Simulator, and the analogy is a good one. OSI SIG holds a regular weekly COnference, during which it is not uncommon for two or three conversations to be going on simultaneously.

In the future, I plan on posting all of the software listings that are published in PEEK[65] available in the OSI SIG Data Library so that members will be able to download them directly into their systems without having to type them in. I also hope that readers will use OSI SIG as a means of submitting articles, letters, and programs for publication in PEEK.

Of course, OSI SIG is a very small part of the CompuServe system. CompuServe provides many other services that can be of tremendous value. For example, there is the electronic mail system, EasyPlex, which lets you exchange private mail with other subscribers.

The Official Airline Guide and other services can help you find the best deals in travelling, and a whole host of companies are doing business in CompuServe's Electronic Mail. You can order merchandise while on-line there.

And information junkies will love the many newspapers, magazines, and columns that are published on the system that are often updated hourly. Specialized information services provide news on the computer industry, as well as hardware and software reviews.

Last, there are hundreds of other Special Interest Groups just like OSI SIG that are dedicated to just about any subject you'd care to name.

So what does this panacea cost? The initial sign-up kits are available in most computer stores. The best kit has a retail price of $39.95 and includes 5 hours of free connect time. After that, you will be billed for the time you spend on the system.

The charges vary depending on the time of day, the baud rate you use to access the system, and any additional charges incurred by the supplimental networks required to access the system from some areas of the country.

The basic connect charge is $6 per hour for 300 baud access during non-prime time (from 6 PM through 8 AM, weekdays). 1200 baud access costs $12.50 per hour during non-prime time. Prime-time access costs roughly $15.00 per hour. TYMNET and TELENET typically add $2 per hour to the above prices. Chances are, that if you live in a major city in the United States, you won't need TYMNET or TELENET. 2400 baud is also available in some areas at a higher cost.

Frankly, I don't think the extra charges are worth the price of 1200 baud access yet. The system doesn't provide a true four-fold increase in transfer rates for most of the things we do in OSI SIG. Further, the users I know who use 1200 baud are always mentioning phone line noise problems, whereas at 300 baud, I see "glitches" once in a blue moon. And I've always found 300 baud to give a very comfortable rate for reading text as it comes in.

I hope I've given you a good idea of what OSI SIG and CompuServe are all about. With the demise of so many local OSI user groups, OSI SIG is one way to keep in touch with the OSI community.

A TALE OF TWO ADS

By: Earl Morris
3200 Washington St.
Midland, MI 68640

Two advertisements appearing in KILOBAUD magazine a number of years ago have had a great infuence on my OSI computer. The first was an ad from an unknown new company called D&N Micro Products appearing Oct. 1978. Offered for sale was an 8K memory board for the OSI bus. At the time this was astounding!! This was the first time anyone else had sold OSI compatible boards. The board held 8K of RAM while the official OSI memory expansion had only 4K. This is not very exciting today when you can purchase an 8K by 8 bit CMOS chip for under $10. However, in 1978 8K of RAM required 64 chips of the 2102 type. I immediately sent off for the new D&N board and as a result tripled the memory size of my OSI machine from 4K to 12K. Since that time I have ordered several other boards from D&N.

The other ad appeared in Feb. 1980 issue of KILOBAUD (by then called MICROCOMPUTING). This was an offer to send in $8 for 12 issues of a magazine with the strange name of PEEK(65). As the radio announcer says "and now you know the rest of the story."

THE DATA SYSTEM

- Stored Report Formats
- Stored Jobs. Formats. Calcs.
- Multiple Condition Reports
- Multiple File Reports
- Calc. Rules Massage Data
- Up to 100 Fields Per Record
- User Designed Entry/Edit Screens
- Powerful Editor
- Merges - Append, Overlay, Match
- Posting - Batch Input
- Nested Sorts - 6 Deep
- Abundant Utilities

HARDWARE REQUIREMENTS: 48K OSI, Hard Disk, serial system, OS-65U 1.42 or Later Space required: 1.3 megabytes for programs and data.

PRICE: $650.00 (User Manual $35.00, credited towards TDS purchase). Michigan residents add 4% sales tax. 30 day free trial, if not satisfied, full refund upon return.

TIME & TASK PLANNER

30 DAY FREE TRIAL — IF NOT SATISFIED, FULL REFUND UPON RETURN

- "Daily Appointment Schedule"
- "Future Planning List" - sorted
- "To Do List" - by rank or date
- Work Sheets for all Aspects
- Year & Month Printed Calendar
- Transfers to Daily Schedule

A SIMPLE BUT POWERFUL TOOL FOR SUCCESS

HARDWARE: 48K OSI, 8" floppy or hard disk, serial terminal system, OS-65U v. 1.3 or later.

PRICE: $300.00 (User Manual, $25.00, credited toward TTP purchase). Michigan residents add 4% sales tax.

FINANCIAL PLANNER

- Loan/Annuity Analysis
- Annuity 'Due' Analysis
- Present/Future Value Analysis
- Sinking Fund Analysis
- Amortization Schedules
- Interest Conversions

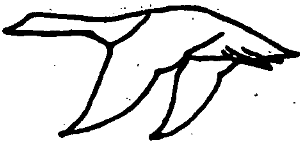
HARDWARE REQUIREMENTS: 48K OSI, 8" floppy or hard disk, serial terminal system, OS-65U v. 1.2 or later.

PRICE: $300.00 (User Manual, $25.00, credited toward Planner purchase). Michigan residents add 4% sales tax.

DEALERS: Your Inquiries Most Welcome

GANDER SOFTWARE, Ltd.

3223 Bross Road
"The Ponds"
Hastings: MI 49058
(616) 945-2821



"It Flies"

FROM THE FOLKS WHO BROUGHT YOU:
All This
THERE IS MORE COMING SOON
Program Generator for TDS
Proposal Planner
Time and Billing A/R

A COMMON 5.25" DRIVE INTERFACE PROBLEM

By: Paul Chidley
Courtesy of TOSIE
Toronto Ohio Scientific Idea
  Exchange
P. O. Box 29
Streetsville, Ont.
Canada L5M 2B7

In the past, I have been asked to repair many different OSI computers. My last adventure reminded me of a problem that has bitten me more than once so I would like to share it with you.

The symptom is a common one, "My system won't boot up." If you examine further, you will find that the disk is indeed accessed and that the first track (track Ø) is put into memory but it doesn't seem to execute. You can determine this by using the 65V monitor to record the memory contents from $22ØØ to $221Ø and from $29FØ to $2A1Ø. Once recorded you can then hit the break key and try to boot from the disk, if you have our problem the machine then appears to go to never-never-land. You can then use the 65V monitor to re-examine the same memory contents where you should find that $22ØØ to $29FF equals the contents of track Ø as shown in Table 1. The Table is taken from a 5.25" 65D V3.2 disk, differences may, of course, be present with different versions. The memory greater than $2AØØ, however, has not changed.

Now that we know track Ø is being loaded, the question is whether or not it is executing, i.e., does the CPU jump to location $22ØØ for its next instruction? This can be tested with a simple program such as the one in Listing 1. This program was intended to be put on data and other such disks that did not have an operating system on them, then when you try to boot it you get the message on your screen. If such a program will boot on your system, you have just proved that track Ø does get loaded and that the CPU does jump to $22ØØ and execute the machine code found there.

The next step is to determine why the drive does not step to track 1. The program in Listing 2 can be merged into a 65D V3.2 disk on track Ø. When this disk is then booted it allows you three commands, H to home the head to track Ø, O to step the head out, and I to step the head in. The command is reflected when entered followed by the track number in decimal followed by the disk's PIA status in hex. If your drive does not behave as expected with this program, you have a different problem than the one I'm building up to. Assuming that the program does behave, we now know that the drive does step properly so let's look at the status word. Broken into binary the meaning of the bits is listed in Table 2. A healthy drive will display a status of $EE or $EC if on track Ø, but let's look at bit number zero. This bit is a left over from the OSI 8" disk interface, with the exception of some very new models, 5.25" drives don't have a drive ready line. If bit Ø is equal to 1 then we have just found our problem.

When you hit "D" to boot the disk your system loads track Ø into memory at $22ØØ and then does a jump to that address. If you examine the code at $22ØØ, you would find that one of the very first things it tries to do is load track 1. It does this by loading the accumulator equal to one (the target track number) and then jumping to the subroutine at $26BC. This subroutine is the standard one used by the operating system. When this routine executes it checks for drive ready, which in this case we don't have, so it then jumps to the error entry point at $2A4B to report ERROR #6 drive not ready. The problem is that the error reporting routines are in memory greater than $2AØØ, i.e., they are on track 1 which hasn't been loaded yet. The result is that the CPU jumped to a location in memory still full of garbage.

The solution to the problem is, therefore, quite simple. Just make sure that the drive Ø ready line (pin 2 of the interface's PIA) is grounded. "So why did we do all those steps above if the answer was so easy?" Simple, now that you know WHY the drive is doing what it is doing you don't have to do all those steps, just make sure the line is grounded.

I hope this helps people further understand, but I especially hope it saves someone a day (or days) of trouble shooting an easy problem.

Table 1

addr	ØØ	Ø1	Ø2	Ø3	Ø4	Ø5	Ø6	Ø7	Ø8	Ø9	ØA	ØB	ØC	ØD	ØE	ØF
22ØØ	A9	Ø1	2Ø	BC	22	2Ø	BC	26	A9	2A	85	FF	2Ø	54	27	86
29FØ	DF	ØØ	DF	AA	FØ	F1	48	2Ø	BC	26	2Ø	73	2D	ØD	ØA	54
2AØØ	52	41	43	4B	2Ø	ØØ	68	2Ø	92	2D	BA	86	FC	2Ø	54	27

continued on page 10

Table 2

Bit	Function
Ø	Drive Ø Ready (Ø if ready)
1	Track Ø (Ø if at track Ø)
2	Fault (Ø if fault, 8" drives only)
3	Not Used (usually = 1)
4	Drive 1 Ready (Ø if ready)
5	Write Protect (Ø if write protected)
6	Drive Select (1 = A or C, 2 = B or D)
7	Index (Ø if at index hole)

OS-65U DATA FILES AND OTHER MYSTERIES: FEAR AND LOATHING GUIDE

PART III

By: Rick Trethewey
8 Duran Court
Pacifica, CA 94Ø44

To a large extent, the things we have discussed are the mere mechanics of data file operations. We need to move on to the more practical applications of the principles we have learned. The critical factor in this discussion is the buffered access techniques employed by OS-65U. The system buffer must be considered at all times or we will waste a lot of time and effort. To demonstrate this, let's consider a simple transfer of information from one Master file to another.

First of all, we have to insure that the source Master file and the destination Master file exists. Further, we must determine if the destination file can hold the information gleaned from the source file. This is not a trivial concern. If the length of a field from the source file is larger than the field length

continued on page 6

continued from page 5

where the information is to be stored in the destination file, we will inevitably run into problems. A simple check of the file headers will tell us if we can proceed. Further, the destination file must be large enough to hold all of the records that will be transferred from the source file. Again, the file headers will tell us what we need to know. These factors are obvious and come to mind quite naturally. It is the actual transfer of records that can lead to cumbersome coding. Consider that transferring each record one by one involves the following operations;

1. Read the track containing the desired record from the source file into the system buffer.

2. Read the track containing the desired record from the destination file into the system buffer.

3. Write the contents of the system buffer back to the destination file/disk.

Each of these operations requires a disk access. Former OS-65D programmers must keep in mind the fact that we are always dealing with a single buffer. Disk accesses take time. A lot of time. To a certain extent, they also involve an indeterminate amount of wear and tear on the disk(s). Obviously, minimizing the number of disk accesses will both speed up the process and improve, however slightly, the reliability of the data media. The solution is as simple as the problem. All we have to do is read in more than one record from the source file before transferring those records to the destination file. The number of records we choose to transfer with each pass is limited by the amount of memory available to the software and by how big each record is. While it is possible to use PEEKs to compute the maximum number of records, your program can handle in a single pass, most programmers choose an arbitrary number that is locked into their programs. It isn't a superior technique, but it has the advantage of simplicity. Fortunately, the BASIC supplied with OS-65U can handle string arrays, and so the code for such multi-record transfers is almost trivial. All 65U programmers must be aware of the system buffer in order to avoid problems. Of special note is that many times, a write to disk requires both a read and a write.

It is in multi-user environments that the system buffer really becomes an overriding factor. In OSI multi-user systems, each user is given a 48K RAM workspace, half of which is devoted to OS-65U and the remaining half is available for programs. In addition, an 8K block of memory from $DØØØ through $EFFF contains software that supervises all users' access to the disk drives, as well as some other functions. This 8K block is shared by all users. The reason is simple enough. It is simply impossible to allow more than one user to control the disk drives. Not only because more than one user could be requesting access to a single disk drive, but also because the operating system must record the position of each disk drive's read/write head. But one of the advantages of multi-user systems is that more than one user may access a single disk file at the same time. Consider a normal business system that uses an inventory file. On the sales floor, cash register/terminals may be used to record the day's sales as they occur. In the office, an auditing program may be recording incoming shipments of new inventory from suppliers. No problem, right? Well... maybe and maybe not. Let's look at what could happen if we're not careful;

1. Register user reads inventory record 1.

2. Auditing user reads inventory record 1.

3. Auditing user writes new record 1.

4. Register user writes new record 1.

Oooppps!! When the register user wrote record number 1, he completely erased the changes made by the auditing user. This situation isn't just due to the fact that both users were changing the same record. If the two users involved had been accessing different records that resided on the same track, the result would have been the same because of the way OS-65U buffers work. Each user under OS-65U has his own system buffer even though the actual access to an individual drive is supervised by the software in the shared memory area. Of course, OSI didn't totally overlook this problem. They implemented a scheme by which each user could gain exclusive access to a given file. They did it with an extension to the BASIC keyword "WAIT". In a multi-user environment, OS-65U maintains a set of 256 user-setable flags which allow each user to tell all of the other users on the system when he is using a particular file. The command;

WAIT FOR xx

checks the flags to see if flag number "xx" is not set, and then sets that flag. As the name "WAIT" implies, there is a time element involved. The "WAIT FOR" command waits a controllable amount of time for the flag to be clear before it continues. While BASIC is WAITing for the flag to be clear, it is decrementing a timing counter byte. When control is returned to the user's program, the user must check the timing counter byte. If the counter is zero, it means that the flag remained set for the duration of the allotted time. If the timing counter is non-zero, the flag was clear, and then set by the WAIT FOR command, indicating the user's program may access the file in question without fear of contention. By the same token, if the timer did go to zero, the user's program should abort the file access. There are a couple of problems with this technique. First of all, except for a few flag numbers reserved by OS-65U, the vast majority of flags are undefined. All programs using the flags must agree on the flag numbers to be associated with each file. This makes it impossible to write generic utilities that can operate in a multi-user environment. On a more practical level, the technique employed by OSI requires a lot of extra disk accesses. Let's say that all of our software assumes that the flag for the inventory file is flag number 1. Then, a typical piece of code to read or write the inventory file might look like this;

```
1000 POKE TIMER,MAXTIM: WAIT FOR 1: IF PEEK(COUNT) >0 THEN 1020

1010 PRINT "FILE #1 IS IN USE.  TRY AGAIN LATER": GOTO ?????

1020 OPEN "INVENO","PASS",1: INDEX<1>=9: INPUT%1, EODF

1030 TN=INT((EODF-BODF)/RL)

1040 RPTR=BODF+(RN*RL)

1050 FOR K = 1 TO NF

1060 INDEX<1>=RPTR+I(K): PRINT%1, A$(K)

1070 NEXT K

1080 CLOSE 1: WAIT CLEAR 1
```

These flags we have been talking about which control access to files are more commonly referred to as "semaphores". Note that in line #1080, "WAIT CLEAR 1" clears flag number 1 that was set by the "WAIT FOR 1" in line #1000. My point is that in the multi-user environment, assuming the flag was clear, each user's software must open the file, which requires reading the directory from the disk and searching it for the file name and checking the file's access rights. It is also worth noting that because the file may have had records added or deleted, every user's program must recheck the end of file INDEX in order to determine the current number of records in the file. Overall, two disk accesses must be done before the file can be touched. And when a write operation is involved, the "CLOSE" command must be executed in order to make sure that the contents of the user's system buffer are actually written to disk so that all users on the system have the most current contents of the file. This system does not really allow more than one user to have simultaneous access to a single data file because the locking is done at the file level. The ideal would be to work out a method for implementing locking at the record level, but that might not be practical using OS-65U. About the best that might be hoped for is locking at the track, or sector level because of 65U's buffering. That final CLOSE is also required because unless the user's local system buffer is cleared, his copy of the operating system considers the contents of the system buffer to be the most recent copy. Without it, a subsequent read or write operation may not do a real fetch from the disk drive, but may well simply use the contents of the local system buffer as they exist in the user's workspace. However, any effort to revise OS-65U from OSI's standards is risky. First, because there is no documentation available for such a low level of programming. More often than not, you're simply stumbling around until you find something that works. Second, because you've left the umbrella of a factory standard, you're totally on your own. Still, these may be prices worth paying.

In order to get your hands dirty with the systems' software, we need to discuss some of the more elementary features of OS-65U's structure. Even if you never intend to program at this level yourself, you may find this information helpful some day when you crash a file or an entire disk.

There is one essential file on all OS-65U diskettes, the directory file. OSI names this file "DIREC*", and the factory deigns it with a type of "OTHER" even though it is really a data file. DIREC* resides at a disk address of 25Ø88, which just happens to work out to track 8 on a floppy, just like our old friend OS-65D. However, each entry in OS-65U's directory uses 16 bytes as opposed to 65D's paltry 8. The structure of the entry is as follows;

Offset	Contents
Ø-5	The first 6 bytes hold the file name in ASCII characters. However, when you delete a file with the utility program "DELETE", the first byte of the deleted file's entry is changed to CHR$(1). Since all files are filled with nulls, CHR$(Ø)'s, when they are created with the "CREATE" utility, directory programs consider any entry beginning with CHR$(Ø) to denote the end of all of the entries in the DIREC* file.
6-7	File Password. Even though 65U applications use 4 characters for password, those four characters are translated into 2 bytes. If you're interested in the coding scheme, check out the program "CREATE". All of the information you'll need is right there in living BASIC. There's nothing wrong with the technique, simple as it is to defeat. Its only purpose is to prevent honest mistakes and to tell others you'd rather they didn't tamper with your file.
8	File Type and Access Rights. Determines if the file is of type BASIC, Data, or Other, and also determines what rights are given to users who access the file without the password - Read, Write, both Read and Write, or None. Again, look at "CREATE", or better yet, "DIR" if you're interested in the bit patterns used for this.
9-11	Starting address of the file. Since no file can be less than 256 bytes long, the least significant byte is not required here. The address is computed with #9*256 + #1Ø*65336 + #11*16777216.
12-14	Length of the file. Same format as the address bytes. Note: current versions of OS-65U require files be an exact multiple of 3584 bytes in length. This is the size of the system buffer and prevents some of the problems we discussed earlier in multi-user environments. No matter which version of OS-65U you are using, I recommend that you adhere to this standard. You'll save yourself some headaches.
15	As far as I know, this byte is undefined.

Unlike OS-65D's directory, the entries of deleted files cannot be simply overwritten when you want to create a new file unless the new file will be exactly the same size as the deleted file. Several OS-65U utilities need to be able to depend upon the principle that there be no totally unassigned areas of the disk between defined files in order to determine how much space is in use on the disk and how much space is left available for new files. In practical terms, this means that for any given directory entry, the sum of the starting disk address of the entry's file and the size of the file will always equal the disk address of the start of the next entry or the start of the free space left on the disk.

All files under OS-65U begin with a 16 byte header. Guess what is stored in those 16 bytes. Think real hard. You guessed it, it's a copy of the directory entry. To my knowledge, no part of 65U uses the header information, but I am not at all sure about this. All I know is that I haven't found any code that uses it. Occasionally, you will need to consider these 16 bytes. For example, if the size of your DIREC* file is 3584 bytes, only 223 entries are really available even though 3584/16 is 224. Even the entry for DIREC* itself begins after this header, even though it is duplicated in the header. The actual contents of all files, no matter what "type", start after these 16 bytes. If you use the BASIC/DOS USR(X) call for your own programs, don't overlook this factor. The program "DIR" is a good illustration of both the effects of the header and how to use the USR(X) call to disk.

What I have tried to present here is an overall view of how OS-65U deals with the disk drives from the operating system's point of view, rather than just in terms of the simple BASIC commands you have at your disposal. If you plan on doing any programming on your own, this information is important, especially if you want your software to be reliable on multi-user systems. All of the information in this article was gathered by close examination of the various utility programs provided by OSI. You can go much further on your own, whether you want to stick to BASIC or move on to machine code programming. If you do get an adventurous urge, I hope you'll share your findings with the rest of us.

This is useful. After POKE 961Ø,2Ø1: POKE 9611,14, output to printer can be toggled on-off with CTRL N.

```
10                  ; PRINTER CONTROL
20                  ; (c) 1985 L. Z. JANKOWSKI
30                  ;
40                  ; From BEXEC*, change
50                  ; @ $2317: from $9E $24
60                  ; to $81 $F2
70                  ;
80 F282                    * = $F282
90                  ;
100 C006=                  PORT = $C006 ; PIA
110 2D73=                STROUT = $2D73
120 249F=                PTROUT = $249F
130 2322=                OUTDST = $2322
140 0462=                PRTERR = $0462
150 0002=                SCRDEV = $2
160                 ;
170 F282 48                PHA
180 F283 AD2223            LDA OUTDST
190 F286 8DF6F2            STA TEMP
200 F289 A902              LDA #SCRDEV
210 F28B 8D2223            STA OUTDST
220 F28E A909              LDA #9
230 F290 4D06C0            EOR PORT
240 F293 F018              BEQ M1
250 F295 A904              LDA #4
260 F297 2C06C0            BIT PORT
270 F29A D025              BNE M2
280 F29C A910              LDA #16
290 F29E 2C06C0            BIT PORT
300 F2A1 F039              BEQ M3
310 F2A3 ADF6F2            LDA TEMP
320 F2A6 8D2223            STA OUTDST
330 F2A9 68                PLA
340 F2AA 4C9F24            JMP PTROUT
350                 ;
360 F2AD 20732D  M1        JSR STROUT
370 F2B0 0A                .BYTE $0A,$0D,'No PRINTER',$0
370 F2B1 0D
370 F2B2 4E
370 F2B3 6F
370 F2B4 20
370 F2B5 50
370 F2B6 52
370 F2B7 49
370 F2B8 4E
370 F2B9 54
370 F2BA 45
370 F2BB 52
370 F2BC 00
380 F2BD 68                PLA
390 F2BE 4C6204            JMP PRTERR
400                 ;
410 F2C1 20732D  M2        JSR STROUT
420 F2C4 0A                .BYTE $0A,$0D,'Printer paper END',$0
420 F2C5 0D
420 F2C6 50
420 F2C7 72
420 F2C8 69
420 F2C9 6E
420 F2CA 74
420 F2CB 65
420 F2CC 72
420 F2CD 20
420 F2CE 70
420 F2CF 61
420 F2D0 70
420 F2D1 65
420 F2D2 72
420 F2D3 20
420 F2D4 45
420 F2D5 4E
420 F2D6 44
420 F2D7 00
430 F2D8 68                PLA
440 F2D9 4C6204            JMP PRTERR
450                 ;
460 F2DC 20732D  M3        JSR STROUT
470 F2DF 0A                .BYTE $0A,$0D,'Printer OFF-LINE',$0
470 F2E0 0D
470 F2E1 50
470 F2E2 72
470 F2E3 69
470 F2E4 6E
470 F2E5 74
470 F2E6 65
470 F2E7 72
470 F2E8 20
470 F2E9 4F
470 F2EA 46
470 F2EB 46
470 F2EC 2D
470 F2ED 4C
470 F2EE 49
470 F2EF 4E
470 F2F0 45
470 F2F1 00
480 F2F2 68      OUT       PLA
490 F2F3 4C6204            JMP PRTERR
500                 ;
510 F2F6 00      TEMP      .BYTE $0
```

OS-65D Revisited

Not so recently, ISOTRON stopped support of OS-65D, effectively freezing development at V3.3. Over the years, many people and groups have written patches, extensions, and other enhancements to this operating system, but without official support from the factory, these efforts had little impact on the vast majority of users.

Over the next few months, I will be working on a new version of OS-65D. I certainly intend to include my Hooks into BASIC, but I want to invite your suggestions as to what feature you have always wanted in OS-65D and BASIC. For example, one of the most attractive features of OS-65D is that it occupies very little memory. Of course, any additions to the operating system will require memory space. What do you think is the maximum size we can allow OS-65D and BASIC to occupy and still leave enough memory for a useable workspace? Is OS-65U's 24K too big?

Another issue is compatibility. We are fortunate that in the 65D world, few packages do POKEs to the operating system. Fewer still do POKEs that would not be made obsolete with the kinds of enhancements I have in mind. However, if we install new keywords in BASIC and do it properly, the new version we produce would no longer be file-compatible with the current versions. This is because the tokens in the new version would have new values. In order to run the BASIC programs you have now under the new version, you would have to transfer the program with indirect files or completely re-type them. It will be a very long time before we have to deal with compatibility, but it's something to keep in mind.

Nothing about this project is etched in stone yet. By the same token, I would welcome any help. Please send in any ideas you might have.

Listing 1

```
.P

  10; TRACK ZERO PROGRAM FOR DATA DISKETTES
  20;
  30; PLACE ON TRACK ZERO OF DISKETTES WHICH
  35; DO NOT HAVE A FULL OPERATING
  40; SYSTEM ON THEM
  50;
  60; By Leroy Erickson, 1981.  *OSMOSUS **
  70;
  80          *=$2200
  90          CLD          ;CLEAR THE DECIMAL FLAG
 100          LDA #$D0     ;CLEAR THE SCREEN
 110          STA $FF      ;
 120          LDA #0       ;
 130          STY $FE      ; $FE,$FF = $D000
 140          LDA #$20     ; GET A BLANK
 150 LOOP1    STA ($FE),Y ; STORE IT
 160          INY          ; INCR INDEX
 170          BNE LOOP1    ; LOOP FOR EACH PAGE
 180          INC $FF      ; INCR PAGE PTR
 190          LDX $FF      ;
 200          CPX #$D8     ; DONE? ; $D4 FOR C1P
 210          BNE LOOP1    ; NO, KEEP GOING
 220          LDA #$D4     ; SCREEN MIDDLE $D2 FOR C1
 230          STA $FF      ;
 240          LDA #$40-MSGLEN/2 ; LEFT MARGIN
 250          STA $FE      ; CENTERED ON LINE
 260          LDY #0       ; ZERO THE INDEX
 270 LOOP2    LDA MESSAG,Y        ; GET CHR
 280          BEQ DONE     ; ZERO IS END OF MESSAGE
 290          STA ($FE),Y ; STORE IT
 300          INY          ; BUMP
 310          BNE LOOP2    ; LOOP TILL END
 320 DONE     JMP DONE     ; STAY HERE FOREVER
 330 MESSAG .BYTE'*** THIS DISK IS NOT BOOTABLE! ***',0
 340 MSGLEN=*-MESSAG
 350          .END         ; THAT'S ALL FOLKS!!!
```

Listing 2

```
  10                   ;*********************************************
  20                   ;*                                           *
  30                   ;* DSTTRO - Disk Stepper Tester on Track 0 *
  40                   ;*                                           *
  50                   ;* by Paul C. - March 10,1984                *
  60                   ;*                                           *
  70                   ;*********************************************
  80                   ;
  90                   ;
 100 2200                      * = $2200
 110                   ;
 120 2683=                     STEPIN = $2683
 130 268A=                     STEPOT = $268A
 140 2663=                     HOME   = $2663
 150 265D=                     TRKNUM = $265D
 160 FD00=                     KEYPOL = $FD00
 170 2343=                     PRINT  = $2343
 180 DE00=                     VIDSIZ = $DE00
 190 2321=                     INDST  = $2321
 200 2322=                     OUTDST = $2322
 210 C000=                     FLOPIN = $C000
 220 29C6=                     SETDRV = $29C6
 230 00E0=                     TS1    = $00E0
 240                   ;
 250 2200 A000                 LDY #$00
 260 2202 8C01C0               STY FLOPIN+1
 270 2205 C8                   INY
 280 2206 8C00DE               STY VIDSIZ
 290 2209 C8                   INY
 300 220A 8C2123               STY INDST
 310 220D 8C2223               STY OUTDST
 320 2210 A040                 LDY #$40
 330 2212 8C00C0               STY FLOPIN
 340 2215 A004                 LDY #4
 350 2217 8C01C0               STY FLOPIN+1
 360 221A A901                 LDA #1
 370 221C 20C629               JSR SETDRV
 380 221F 20D122               JSR SCLEAR
 390 2222 A000                 LDY #$00
 400 2224 B9F522  FP1          LDA MESSAG,Y
 410 2227 F00F                 BEQ S2
 420 2229 204323               JSR PRINT
 430 222C C8                   INY
 440 222D D0F5                 BNE FP1
 450 222F 207822  START        JSR CONVRT
 460 2232 20DC22               JSR STATUS
 470 2235 20AF22               JSR CRLF
 480 2238 2000FD  S2           JSR KEYPOL
 490 223B C949                 CMP #$49
 500 223D D013                 BNE S1
 510 223F 20A422               JSR CPRINT
 520 2242 AE5D26               LDX TRKNUM
 530 2245 8A                   TXA
 540 2246 F0E7                 BEQ START
 550 2248 CA                   DEX
 560 2249 8E5D26               STX TRKNUM
 570 224C 208326               JSR STEPIN
 580 224F 4C2F22               JMP START
 590 2252 C94F    S1           CMP #$4F
 600 2254 D015                 BNE S3
 610 2256 20A422               JSR CPRINT
 620 2259 AE5D26               LDX TRKNUM
 630 225C E8                   INX
 640 225D 8A                   TXA
 650 225E C928                 CMP #40
 660 2260 B0CD                 BCS START
 670 2262 8E5D26               STX TRKNUM
 680 2265 208A26               JSR STEPOT
 690 2268 4C2F22               JMP START
 700 226B C948    S3           CMP #$48
 710 226D D0C9                 BNE S2
 720 226F 20A422               JSR CPRINT
 730 2272 206326               JSR HOME
 740 2275 4C2F22               JMP START
 750                  ;
 760 2278 AD5D26  CONVRT LDA TRKNUM
 770 227B 38                   SEC
 780 227C A2FF                 LDX #$FF
 790 227E E8                   INX
 800 227F E90A                 SBC #10
 810 2281 B0FB                 BCS *-3
 820 2283 690A                 ADC #10
 830 2285 85E0                 STA TS1
 840 2287 8A                   TXA
 850 2288 0A                   ASL A
 860 2289 0A                   ASL A
 870 228A 0A                   ASL A
 880 228B 0A                   ASL A
 890 228C 05E0                 ORA TS1
 900 228E 85E0                 STA TS1
 910 2290 48      PRT2HX PHA
 920 2291 4A                   LSR A
 930 2292 4A                   LSR A
 940 2293 4A                   LSR A
 950 2294 4A                   LSR A
 960 2295 209922               JSR PRTHEX
 970 2298 68                   PLA
 980 2299 290F    PRTHEX AND #$0F
 990 229B C90A                 CMP #$0A
1000 229D F8                   SED
1010 229E 6930                 ADC #$30
1020 22A0 D8                   CLD
1030 22A1 4C4323               JMP PRINT
1040                  ;
1050 22A4 204323  CPRINT JSR PRINT
1060 22A7 48                   PHA
1070 22A8 A920                 LDA #$20
1080 22AA 204323               JSR PRINT
1090 22AD 68                   PLA
1100 22AE 60                   RTS
1110                  ;
1120 22AF A90D    CRLF         LDA #$0D
1130 22B1 204323               JSR PRINT
1140 22B4 A90A                 LDA #$0A
```

```
1150 22B6 4C4323          JMP PRINT
1160                    ;
1170 22B9 A920     SCLSUB LDA #$20
1180 22BB A008            LDY #$08
1190 22BD A200            LDX #$00
1200 22BF 9D00D0   SCL1   STA $D000,X
1210 22C2 E8              INX
1220 22C3 D0FA            BNE SCL1
1230 22C5 EEC122          INC SCL1+2
1240 22C8 88              DEY
1250 22C9 D0F4            BNE SCL1
1260 22CB A9D0            LDA #$D0
1270 22CD 8DC122          STA SCL1+2
1280 22D0 60              RTS
1290                    ;
1300 22D1 A9E0     SCLEAR LDA #$E0
1310 22D3 8DC122          STA SCL1+2
1320 22D6 20B922          JSR SCLSUB
1330 22D9 4CB922          JMP SCLSUB
1340                    ;
1350 22DC A920     STATUS LDA #$20
1360 22DE 204323          JSR PRINT
1370 22E1 AD00C0          LDA FLOPIN
1380 22E4 209022          JSR PRT2HX
1390 22E7 60              RTS
1400                    ;
1410 22E8 208326   PATCH  JSR $2683
1420 22EB E6FD            INC $FD
1430 22ED D005            BNE P1
1440 22EF A906            LDA #$06
1450 22F1 204323          JSR PRINT
1460 22F4 60       P1     RTS
1470                    ;
1480 22F5 48       MESSAG .BYTE 'H/I/O ?',$A,$A,$D,0
1480 22F6 2F
1480 22F7 49
1480 22F8 2F
1480 22F9 4F
1480 22FA 20
1480 22FB 3F
1480 22FC 0A
1480 22FD 0A
1480 22FE 0D
1480 22FF 00
1490
1500 2673                 *=$2673
1510 2673 20E822          JSR PATCH
1520 267A                 *=$267A
1530 267A A062            LDY #$62
```

ATTENTION: DEALERS!

PEEK[65] needs new subscribers and you need new customers, and together we can make it happen with our own Co-op advertising program. This program pays dealers for signing up new subscribers with free ad space in PEEK[65]. Just five paid subscriptions will earn a 1/9th page advertising credit in PEEK[65].

Most dealers sell their own software with the systems they install. By advertising in PEEK, you vastly expand the potential market for your products. And how many sales have you lost because you couldn't find the application your customer wanted? Dealer ads can be our own Yellow Pages. Readers and customers win too by increasing the number of uses for their equipment.

Call or write today for details and your free promotional materials. Making a PEEK[65] subscription a part of every sale is painless and profitable. This time, "Co-op" pays you.

Letters

Editor:

Reguarding my HEXDOS directory utilities article in the Dec. 85 issue; it appears that line 0 in both listings 1 and 3 have lost a character. The line should be;

0 A$ - "(lots of spaces)"

The end quote was missing. I'm not sure of the exact number of spaces, but it is about 60 - whatever fits on a standard 72 character line. I don't think the program will run without the spaces or the last quote.

Jim McConkey
7304 Centennial Road
Rockville, MD 20855

Dear Jim,

Thanks for the correction on your program. Note that BASIC doesn't require the trailing quote on a string that appears at the end of a line of code, but it is good practice to use them so that adding REMs later is a bit easier.

Rick

Editor:

Once again I ask PEEK for help! Isotron Technical Support could not aid me with the following;

(1) About the 630 board, (a) has anyone developed circuitry to convert the RGB signals to composite color? The board has spaces for a chrystal (color burst?) X2 plus some unused IC space. (b) J4 has 16 pins w/digital signals, but no ground return. Is this the so-called 16-pin bus? Must it be used with the A-15 board?
(2) About the 620 board, I've always had the impression that the 620 board allowed one to go from the 40-pin bus to the 48 pin (ie. read/write - send█/receive). The schematics I received with my 620 board show conditions (a) receiver - as is, (b) transmit - requires cuts and jumpers. Can someone explain where each is used? Also, where does J2 (16-pins) connect to?

John A. Turner
1778 Sutter Street
San Diego, CA 92103

Dear John,

The 630 was originally designed to output composite color, much like the 540 board, but that was one of the functions they couldn't get to work reliably (which is why the board was never officially released). I'm afraid I don't have any answers for you. Hopefully one of our readers will write in and let us all know. Readers?

Rick

Editor:

I wonder if you could help me (and some others) with a solution to the following problem. When using WP-2 (65D), the maximum number of characters per line is 127. I would like to utilize the Microline 84's capacity of 231 characters per line. How could one reprogram the WP-2 to attain this?

Stefan Bjorgen
Drottninggatan 71
S-411 07 Goteborg, Sweden

Dear Stefan,

Your problem is not that uncommon, although most people who run into this limit have a 132 character printer. In any event, there is no solution. WP-2 allocates a memory buffer to hold each line it generates for printing. To overcome the size limit problem, you would have to completely disassemble the program, find all of the references to this buffer, and fix them in a way that wouldn't disturb anything else in the program. Hardly a solution. My own Edit-Plus program has a similar size restriction. Perhaps our readers can suggest some alternative sofltware. The only thing I could think of would be to convert your edited WP-2 files into an ASCII file and write a small BASIC utility to print them. If you need help with the format of WP-2 files, just let me know.

Rick

Editor:

S-Forth users who have problems when using a paralell printer may appreciate this. The problem is that the paralell port isn't being properly initialized when 65D boots. The original JSR at address $2214, which used to point to a routine at $2761, now JSR's to $2E79 which is the starting address of a a subroutine added for Forth. At this point, the new routine returns the X register with $03 when the code that initializes the paralell port gets initialized. That initialization code requires X to hold a $00. The fix is to add an LDX #$00 at $2EB1 ($A2 $00) followed by an RTS ($60). On 8" systems, this code resides on sector 1 of track 1. This track has 5 pages of code and is loaded during boot up. To install the change, boot up with standard OS-65D and enter the Extended Monitor. Call the code into high memory at, for example, $5A00 with the command "!CA 5A00=01,1". Change the three bytes indicated above which you will find starting at $5EB1. Then save the code back to disk with the command "!SA 01,1=5A00/5".

John Schneider
R.D. 2 Box 69
Wheeling, WV 26003

Thanks, John. Users should note that this fix writes over some timing bytes required on 1 MHz systems in normal 65D V3.2. In the disassembly John sent in, there appears to be some free space at $2EC3. If you have problems with the above, you might change the JSR at $2214 to point to $2EC3. At $2EC3, install JSR $2E79, LDX #$00, RTS.

Rick

Editor:

In the Oct. 85 issue, Mr. Herb Grassel discussed interfacing a Graffix SEB-1 board to a C1P with a 610 board. I have had the Graffix board for quite a while, and prior to adding disks to my C1P, I used the Graffix board to give me 24K of user RAM, and 6K of either Hi-res or machine code RAM.

With the addition of a D&N board to my system, the Graffix board would no longer work properly. I could write to the VIA, VDG, and video RAM, but the CPU would not read in data. This was verified by setting the VDG for a Hi-res mode and then observing the video display for changes as the data was written to RAM. At first I thought the 8T28 buffers on the Graffix board were the problem, but later discovered that if I disconnected the D&N board, the Graffix board worked fine. In fact, at one point, I had the VIR running as a real time clock by using a program that appeared in PEEK.

What I had to do was set the assembly address of the routine to $8000, and assemble my program. Next, I powered down the D&N board to remove the memory conflict. Then I did a cold start of ROM BASIC, did a BREAK into the Monitor ROM, and then started the clock running.

The VIA on the Graffix board is a very versatile chip, and I would greatly appreciate hearing from any reader that can help me get the Graffix and D&N boards to work together.

C.J. Hipsher
2908 Cardini Place
Virginia Beach, VA 23456

Readers often write to PEEK[65] to ask for help with hardware and software problems. When we can, we try to provide answers in our Letters to the Editor column. Occasionally, the nature of the problem requires contacting the vendor. This article shows how a reliable vendor deals with a customer's problems.

Dear Editor,

I work in a maritime archaeology department in Western Austrailia. In about 1979, I decided that our department needed a computer and we invested in a very early OSI cassette based system. My interest was in being able to do large scale algebra, since I had run out of space on my TI-59. Gradually, the system has been slowly upgraded to dual disk drives and (serial console). I became interested in interfacing our computer with a three dimensional stereophotographic comparator. I had a 3 channel DA converter built which fitted into our C8 system. I ran OS-65U. We then became interested in word processing, and found WP-3 hopeless. I then bought WP6502, which was upgraded as new versions came out. Later on, a data base became a consideration and we tried OS-DMS, but there were a lot of bugs. Then we upgraded to an Overtask Multi User with a 40 MB hard disk, and now operate three users. We finally bought the TDS Gander Data Base system, and have large quantities of data on the machine.

I find TDS a very easy system, once you get used to the instruction book (this seems to be inevitable that software instruction books are totally incomprehensible). TDS has lots of very useful things going for it, and it works remarkably well in our 5 partition system. My only complaints are that it has irritating problems: the global edit has to know the position of the characters in the line; one cannot abort a search and; having gone to the trouble of creating formats and conditions, one cannot get straight on with doing whatever it is you want to. You have to go back and reload the whole thing. However, it does really fast sorts and searches and you can have lots of job files.

Yours sincerely,
Jeremy Green
Head
Department Maritime Archaeology
Western Australian Maritime Museum

Reply from Mr. John Huntley, president of Gander Software, Ltd.

Dear Editor,

Thank-you for the opportunity to reply to the letter sent you by Mr. Jeremy Green. First, such a letter is a reminder of the extraordinary number of places in which one can find an OSI running. We sometimes think that the manufacturer of these wonderful little machines has little idea about the incredible uses to which their equipment is put.

Turning to the substance of Mr. Green's letter, we are extremely pleased that he finds it a "very easy system" that has "lots of very useful things going for it" and that it works "remarkably well". Of course, general purpose software cannot be all things to all people, but we've tried to give TDS real substance together with many "bells and whistles".

The fact that the global edit has to know the position of characters in the line is a function of the DMS file structure, and the fact that to provide "random capability" would, though possible, create an enormous amount of overhead. Mr. Green's is the first request we have received for such capability, indicating to us that such overhead would not be welcomed by the average user.

As to aborting a search, it isn't possible so far as we can determine. The FIND command given the BASIC programmer calls a machine code routine, meaning we cannot sense or check to see if a Control 'C' has been issued. We provide Control 'C' checking in the reports because they are run in BASIC, using no machine code routines, allowing us to check at the end of certain print loops to see if a Control 'C' has been received, and can then give the user a choice of aborting or not under program control.

Mr. Green's final point about not being able to get straight on to running a process on which formats and conditions have been set up relates to the fact that the OSI for which we wrote the TDS has only 48K of memory. These formats and conditions are, after all, set-up routines, not run-time routines. Once they have been set up, the same process can be called over and over again without ever having to specify the format and/or conditions. While it may not be apparent to the average user, much that the TDS accomplishes is by virute of using common variables and "chaining" from program to program. However, there are very real limits on how much we can keep in memory. Many of the processes that look to the user like a single program are, in fact, up to three programs called back and forth as needed.

We have no wish to seem critical, and are not. Mr. Green is one of our valued customers, and we hope this reply will answer his questions.

Sincerely yours,
John Huntley

CALL FOR ARTICLES

PEEK[65] is a user's publication. We all depend on each other to keep the magazine alive and growing. The current library of articles ready for publication is very small, and growing smaller with every issue. So, I am calling on all of you to help.

What are you doing with your system? Do you program for the sheer enjoyment of it? Do you use your OSI in your business? The rest of us want to hear about it. Chances are you've solved a problem that has stumped someone else for a long time. I know it's hard to write about yourself, but remember that you're among friends here. Write it up and send it in.

As I indicating in Column One, PEEK[65] is going to be expanding in its scope. We need articles about the new OSI systems, the 68000 family, and the UNIX environment. I'd also like to see some general business applications. Articles need not be OSI-specific. Many of us own or use several different microcomputers. In fact, one of the most common problems I hear about concerns getting OSI equipment, software, and data to work with other vendors' products. Many of our foreign subscribers use clones of OSI hardware that are virtually unknown in the US. I'd sure like to see a description of some of these systems. The possibilities are endless. Chances are that if you are interested in a subject, the rest of us are too.

And don't forget, PEEK[65] actually pays for articles. You won't get rich, but our rates are reasonable - enough to pay for that utility program you've always meant to buy.

Articles can be submitted in many ways. If at all possible, I would appreciate it if you would upload your article to OSI SIG on CompuServe. Failing that, submissions on disk are best because it saves me the effort of getting your work in machine readable form. Thank-you.

WHOOPS!!

To all subscribers and advertisers:

Please accept my apologies for the delays in getting this issue out to you. Starting up a new business and producing a magazine like PEEK[65] involves a miriad of details, each of which requires attention before you can proceed to the next.

The March issue will be mailed in approximately three weeks. Each subsequent issue will be sent out within three weeks of the prior issue until the normal deadline of the first of the cover month is regained.

Once again, I apologize and wish to extend my sincere thanks to Karin and Eddie Gieske. They did everything they could to help me. How they did this every month is beyond me.

ADVERTISE IN PEEK[65]

Classified ads in PEEK[65] are only 35 cents per word and reach the people who are most likely to be interested in your software and hardware. No charge for "price" words.

HARDWARE

TOSIE-IV PADDLE BOARD FLOPPY DATA SEPARATOR

This popular bare board kit is actually a multiplexer, disk switch, and very stable data separator - sufficient to allow mating most floppy drives with virtually any OSI computer with up to 4 surfaces. The 8 74LS chips required make for economic population, flexibility, and stability. Complete instructions and schematics are included. Although not fancy, it should help resolve many of the problems encountered in matching new drives to OSI machines. Since it is all on the board, minor mods are required on the OSI board to bypass some of the old circuitry. Available from Paul Chidley, RR#2, Ennismore, Ont., CANADA K0L 1T0 for $20.00 total. Non-Canadian orders should be made in US dollars.

SOFTWARE FROM PEEK

With a base of over 40 copies in circulation since its introduction several months ago, OSI-CALC has already proved itself.

It is a full-featured spreadsheet that is interactively on-line with 26 columns by 36 rows. Paul Chidley has written it entirely in BASIC to allow for user modifications, but judging from its speed, you would never know it. It does require 48K on C4/C8P's and we understand that a 32K version for C1P's is in final testing. The program runs under OS-65D V3.3 on either 8" or 5-1/4" floppies, and while it is written for video systems, serial systems should be able to modify it to suit their terminal's control codes. Paul has put the program in the public domain, and PEEK[65] is distributing it for just $10.00 plus our normal $3.70 shipping charge.

Term-Plus

A smart terminal program running under OS-65D V3.3 which allows capturing and transmitting to and from disk. Term-Plus also supports error-free transfers to/from CompuServe. Memory size does not limit the size of files that can be captured or transmitted. Utilities included allow translating captured text files into OSI source format for BASIC and Assembler programs or into WP-2/WP-3 format, translating OSI source files into text files for transmitting to non-OSI systems, and printing captured text files. Runs on all disk systems, mini's or 8", except the C1P-MF. $35.00.

Term-65U

A smart terminal program for OS-65U (all versions) running in the single user mode. Allows capturing text to disk files. Term-65U will transmit text files, or BASIC programs as text. The program will also send WP-3 files as formatted text and can transmit selected fields in records from OS-DMS Master files with sorts. $50.00

Term-32
Same as Term-Plus, but for OS-65D V3.2. $35.00.

ASM-Plus
ASM-Plus is a disk-based assembler that allows linked source files enabling you to write very large programs, regardless of system memory size. ASM-Plus assembles roughly 8 to 10 times faster than the OSI Assembler/Editor. ASM-Plus adds several assembly-time commands (pseudo opcodes) for extra functionality. Includes a file editor for composing files that allows line editing and global searches. Requires OS-65D V3.3. $50.00

Edit-Plus
Styled after WP-3-1, Edit-Plus allows composing and editing WP-3 compatible files and to have those files printed as formatted text. Perfect for correspondence and other small documents, or for editing assembler source files. Very fast on global searching and editing, as well as block moves and copies. Free form text entry - no need to press <RETURN> at the end of each line. Edit-Plus fixes problems in WP-3, including proper pagination control, inputs from the console, and file merging. Requires OS-65D V3.3. $40.00

Data-Plus 65U Mail Merge
A program to insert fields from OS-DMS Master files into WP-3 documents. Output can be routed to a printer or to a disk file for printing later or for transmission via modem using Term-65U. Insertions are fully selectable and are properly formatted into the output. $30.00

AD$

FOR SALE: C3-OEM with 48K RAM-DUAL 8 inch Floppy Drives. OS-65U Operating System, DMS, Busi-Calc, plus more software. ACT-5 Terminal, NEC 5500 letter quality Spinwriter. All in good condition $1500.00. Call (804) 851-0781.

* * * * *

MC-DMS Nucleus
MC-DMS Nucleus is a replacement package to the OS-DMS Nucleus from OSI. All of the programs from the original except SORT have been duplicated and enchanced and new software has been added. The name "MC-DMS" stems from the extensive use of machine code support built into the utilities to replace slower, BASIC code. Features include; (1) MC-DMS Interface code supports up to 8 Master files simultaneously without requiring OPEN/CLOSE commands under Level 3. The only 65U software support needed is for semiphores. This produces a significant increase in speed. READ, WRITE, and FIND commands operate on the field level. FIND skips over embedded garbage and automatically stops on the last record in the file. (2) Machine code DIR utility. Ultra-fast. Automatic paging. ^C interrupt. Can selectively list by file type or can search for file name matches with wildcards. (3) Machine code file manager. Creates, deletes, or renames files in a flash. The file manager is linked to the Master/Key file create utility. (4) Machine code file transfer. Grabs up to 30 records per pass. Single/dual drive. Fully selectable field specifications. (5) Machine code single/dual drive floppy diskette copier. Moves up to 7 tracks per pass. (6) Disk-based mailing label printer. Stores printing format designs on disk. Selectable fields and record range, Key file access, searches, and more. (7) Disk-based report writer. Stores report format designs on disk. Same features as above, but with formatted columns by type and width. (8) Edit-Plus 65U. Most of the same features as the 65D version, but with a significantly smaller workspace. Suitable for correspondence. (9) Data-Plus Mail Merge. CAUTION! Some parts of this code is very young. The MC-DMS interface, DIR, COPIER, and XFER programs are solid. Some of the BASIC utilities and Edit-Plus may well need some work. If you're willing to try this on an "as-is" basis, you can get a 50% discount off the final price of $150.00 if you order before April 30, 1986.

AD$

FOR SALE: C3-OEM with 48K RAM-DUAL 8 inch Floppy Drives. OS-65U Operating System, DMS, Busi-Calc, plus more software. ACT-5 Terminal, NEC 5500 letter quality Spinwriter. All in good condition $1500.00. Call (804) 851-0781.

* * * * *

FOR SALE: OSI C3B SYSTEM (2 MHz) - 3 user including 74 MB hard disk, 2 8" floppy drives, 144K RAM, 555 board, OS65U, 3 Hazeltine 1500 terminals. Sams, hard disk & terminal factory manuals and all other original documentation. $3000 plus shipping or best offer. Challenger II (1 MHz) w/48K, 2 8" floppy drives, 500 CPU board, 570 (RT clock, I/O), 567 (UTI, modem), 470 (Cent. printer), Cent. 779 printer. C2-4P w/8K, 9" BW monitor, Sams manual. C2-4P w/4K. All with original docs. $500 plus shipping. Foreign orders OK. Robert Khlopin, 5627 Lawton Dr., Sarasota, FL 33583, phone (813) 922-8621 8AM-5PM EST.

* * * * *

FOR SALE: 3 OSI C3B/C Computers, many Denver/CM-20 boards, 9 track tape drive, 9 track interface for PC, 3 NEC Spinwriters, 2 Okidata 84S printers, 4 Esprit 6310 CRT, 3 Hazeltine 1420 CRT, 1 Wyse 50 CRT, many reels of tape - 600' and 2400', etc.(612) 252-5007.

* * * * *

FORTH $24.95, Utilities available also, Free Catalog. Aurora Software, 37 S. Mitchell, Arlington Heights, IL 60005.

* * * * *

WATCH THIS SPACE GROW!

PEEK (65)
The Unofficial OSI Users Journal
P.O. Box 586
Pacifica, CA 94044
415-993-6029

DELIVER TO:

GOODIES for OSI Users!

PEEK (65)
The Unofficial OSI Users Journal

( )	**C1P Sams Photo-Facts Manual.** Complete schematics, scope waveforms and board photos. All you need to be a C1P or SII Wizard, just	$7.95	$ ________
( )	**C4P Sams Photo-Facts Manual.** Includes pinouts, photos, schematics for the 502, 505, 527, 540 and 542 boards. A bargain at	$15.00	$ ________
( )	**C2/C3 Sams Photo-Facts Manual.** The facts you need to repair the larger OSI computers. Fat with useful information, but just	$30.00	$ ________
( )	**OSI's Small Systems Journals.** The complete set, July 1977 through April 1978, bound and reproduced by PEEK (65). Full set only	$15.00	$ ________
( )	**Terminal Extensions Package** - lets you program like the mini-users do, with direct cursor positioning, mnemonics and a number formatting function much more powerful than a mere "print using." Requires 65U.	$50.00	$ ________
( )	**RESEQ** - BASIC program resequencer plus much more. Global changes, tables of bad references, **GOSUBs** & GOTOs, variables by line number, resequences parts of programs or entire programs, handles line 50000 trap. Best debug tool I've seen. MACHINE LANGUAGE - VERY FAST! Requires 65U. Manual & samples only, $5.00 Everything for	$50.00	$ ________
( )	**Sanders Machine Language Sort/Merge** for 0S-65U. Complete disk sort and merge, documentation shows you how to call from any BASIC program on any disk and return it or any other BASIC program on any disk, floppy or hard. Most versatile disk sort yet. Will run under LEVEL I, II, or III. It should cost more but Sanders says, "...sell it for just..."	$89.00	$ ________
( )	**KYUTIL** - The ultimate OS-DMS keyfile utility package. This implementation of Sander's SORT/MERGE creates, loads and sorts multiple-field, conditionally loaded keyfiles. KYUTIL will load and sort a keyfile of over 15000 ZIP codes in under three hours. Never sort another Master File.	$100.00	$ ________
( )	**Assembler Editor & Extended Monitor Reference Manual** (C1P, C4P & C8P)	$6.95	$ ________
( )	**65V Primer.** Introduces machine language programming.	$4.95	$ ________
( )	**C1P, C1P MF, C4P, C4P DF, C4P MF, C8P DF Introductory Manuals** ($5.95 each, please specify)	$5.95	$ ________
( )	**Basic Reference Manual** — (ROM, 65D and 65U)	$5.95	$ ________
( )	**C1P, C4P, C8P Users Manuals** — ($7.95 each, please specify)	$7.95	$ ________
( )	**How to program Microcomputers.** The C-3 Series	$7.95	$ ________
( )	**Professional Computers Set Up & Operations Manual** — C2-OEM/C2-D/C3-OEM/C3-D/C3-A/C3-B/C3-C/C3-C'	$8.95	$ ________

TOTAL	$________
CA Residents add 6% Sales Tax	$________
C.O.D. orders add $1.90	$________
Postage & Handling	$ 3.70
TOTAL DUE	$________

POSTAGE MAY VARY FOR OVERSEAS

Name ______________________________

Street ______________________________

City ________________ State __________ Zip __________