:up-6502 + C1k ROM p22

PEEK (65)

The Unofficial OSI Users Journal

$1.75

AUGUST 1983
Vol.4, No.8

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

INSIDE

Column One

Our lead story this month is all about the upcoming "Software Issue" we hope to put together in October. Like almost everything else in PEEK(65), it all depends on you. Don't be shy. If your program works, and you like or need it, send us a description. We are offering the equivalent of a free $10 Classified Ad to everyone in the OSI world!

Someone recently asked the origin of the name of our journal. Easy! When we first started working with OSI and learning BASIC, everywhere we looked, we saw 65's. 6502; 65D; 65U; but we didn't understand much about any of those 65's. The manuals in those days certainly didn't help! So we decided to start a users' journal, not because we knew so much and wanted to tell everyone else, but because we knew so LITTLE and needed others to tell us about all those 65's.

Being a beginner at BASIC programming at the time, I thought the PEEK instruction was a perfect name. It would be short and easy to remember, hopefully catchy, clearly computer and BASIC oriented, and was intended to mean that we would PRINT what was in all those mysterious 65's. We keep trying...

This issue, it seems, is a hardware issue. There are more NAND gates and "active low's" in this issue than I remember in a long time. With October's software issue, it should make a good balance.

Speaking of hardware, there are a couple of things we need in the OSI world which perhaps some reader might help us with. One is simple, one more difficult.

The simple one is a small (maybe 256 byte) serial FIFO buffer, which would accept stuff typed in from a serial terminal, store it in the buffer, then send it along to the computer when the computer is ready for it.

The advantages are obvious: you could pre-answer the next five questions you know will come up in a program, or type as fast as you or your terminal screen dump can type without worrying about getting ahead of the computer or word processing program.

The second project is more ambitious: a RAM disk. You know what I mean -- a board with maybe 256K of RAM on it, plus the necessary software modifications to make it look like a 256K floppy disk drive, complete with directory and everything. Every time I listen to my disk drives whirring and clicking away at some disk-intensive process like a sort or an indexing process, I wonder just how many times faster it would all work if it were really happening in RAM instead of on disk. There are two or three such products available for the S-100 bus; how about one for the OSI 48-pin bus?

We print this month an angry letter from Colin Law. We all know that OSI has changed hands twice, and personnel more times than that. We all know that they are loads better at supporting users than they were (how could they be worse?). But when a loyal OSIer like Colin has as much trouble as he has had, the response should be better, and this users' journal feels the duty to let his gripes be aired. Of course, we will feel just as bound to print any response from OSI!

Multiprocessing is the wave of the future in business computing. CPU chips cost just a few bucks apiece, even the powerful new 16- and 32-bit chips. Everyone using a computer will have some RAM of his own anyway; why not give him his own CPU as well?

We are seeing this happening all around us. The new 300 machines from OSI are CP/M multiprocessors. The Denver Board people have brought multiprocessing to those of us with older machines (even C2's). The results are reportedly amazing. I have worked with both timesharing and multiprocessing CP/M systems and can assure you that as the number of users goes up, performance goes down -- except with multiprocessing.

We hope to have an article soon featuring the ins and outs of multiprocessing, how it works and what problems it can create.

Keep those software response sheets coming in. See you next month.

AL

****SPECIAL**** PEEK SOFTWARE ISSUE

One of the most often heard complaints about OSI is the lack of software. We don't believe it! We have talked to and received letters from too many of you who are proud as punch about your little "ditty". But, you don't think it's good enough or professional enough to put your name on it. All the same, it might just be the thing that someone else has been crying for and doesn't know where to find it or even if it exists at all.

If you are in the business arena, you probably have even less exposure than the "hackers" to what's available - besides you haven't had the time to go digging beyond your local dealer.

So PEEK(65) has decided to give every program author (hacker and pro alike) the opportunity to say, "here's what I have". That's right, PEEK is planning to devote a large segment of the October issue to software. That sounds like a long way off, but remember that the deadline for the October issue is September 1st - and that's only days away!

Business, games, utilities, neat little tricks and anything else that can be called a program. And the listing will be entirely FREE.

Listings will be sorted by Basic, Type and Machine to make it easy for you to zero in on the programs that may be of interest to you.

Each listing will have an encoded "head line" that should tell you everything you need to determine if it is interesting and will run on your machine. Next will be the program name and the author's name and address. Lastly, will be the author's description and any special comments. So what will the listing actually look like? Try this little gem!

....

B/2/82/S/1.42/MR/D/D/12/1000
WONDER ACCOUNTING SYSTEM
John Q. Smith
123 Flatire St.
Cupcake, NZ 12345
123-456-7890

This system will handle up to three A/R and four R/P accounts at one time. Complete record locking, provided that no more than one user at a time is active. Average storage space required is 8MB. Mammoth overhaul required to run on SSII cassette system. And that makes 10 lines.

....

What's all the "gooble-degok" on the first line? Well, (B) says that its a business package; (2) to run on C3A/B or 200 series; (S) serial terminal; (1.42) basic version no.; (MR) multi-user with record locking; (M) modem supported; (D) sold by dealers; (12) 12 x 10 = at least 120 copies in use; (1000) the bucks at retail. If that doesn't give you enough to tell you if you are interested, we give up!

Got you interested? Just "X" the appropriate boxes and fill in the remaining blanks on the form on the back page of this issue. If you have more than one program to submit (we certainly hope that you do), please feel free to make photo-copies of the form - one for each program!

The hard part will be writing a description that will not exceed PEEK's physical limits (10 lines, each not to exceed 30 characters). We would like to give you more room, but 10 lines of carefully chosen words should be adequate to whet the appetite.

If your software is not directly supported by you, the author, please fill in the DEALER ADDRESS as well as your own address block. If both blocks are filled in, only the dealer address will appear in PEEK. The dealer address may be either the selling dealer or an address where those inquiring may get a list of vendors.

Now it is up to you. Our guess is that our mail box will be full, but don't let us down and make liars out of us.

★

published monthly

Editor - Al Peabody
Technical Editor - Brian Harston
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

SOFTWARE CONSULTANT FIG-FORTH FOR THE OSI OS 65D OPERATING SYSTEM

An Independent Review

By: Newton C. Fawcett
Department of Chemistry
U of Southern Mississippi
Hattiesburg, MS 39406

Software Consultants adaption of William F. Ragsdale's version of fig-FORTH for the 6502 microprocessor is one of at least six low-cost versions of FORTH available for OSI computers. Software Consultants (*), which will hereafter be referred to as simply 'SC', specialize in OSI software as well as software for several minicomputers.

There are two versions of FORTH offered by SC. One runs under OS-65U, the other under OS-65D. The latter is the specific subject of this review, although the two versions are substantially the same. The 65U version is intended for business applications, which may seem somewhat bizarre if you think of FORTH as a language for controlling interfaced gadgets. Apparently, SC has put the major part of their FORTH effort into the 65U version with the intention of taking advantage of FORTH's rapid execution in business software. Indeed, the seven page set of instructions sent with the software was actually intended for use with the 65U version, but the portions not applicable to 65D are quite obvious.

For laboratory applications requiring extensive interface and control functions, the 65D version seems the best choice, because an assembly source listing for OS-65D is available from SC, whereas to the best of our knowledge the source code for OS-65U is not available.

The FORTH system reviewed here has been in daily use for approximately twelve months in a research laboratory. It is installed on an OSI C3-A with eight differential channels of 12-bit A/D conversion, three channels of 16-bit D/A conversion, a Strobe Model 100 drumplotter, a printer and serial terminal. The computer is interfaced to laboratory

instruments and used for data collection and experiment control. The bulk of the laboratory's software is written in FORTH, including the drumplotter software, and a FORTH scientific floating-point package.

Documentation: SC's documentation is concise. Beginning FORTH programers will find it inadequate. The best source of information for the beginner is Leo Brodie's book, "Starting FORTH"(*1). This, along with the fig-FORTH Installation Manual, Glossary, and 6502 Model by William Ragsdale, available through SC or directly from the Forth Interest Group (*2), is sufficient to get started. Complete, well almost complete, assembler source code is available on disk from SC. You can make your own hardcopy. Purchase of the assembler source code is recommended, as FORTH definitions for a few words in the resident dictionary are otherwise not included.

Boot-Up: SC fig-FORTH for OS-65D boots up by loading FORTH starting at $200 as if it were BASIC. The normal entry point to BASIC is at $20E4. This is where SC has placed code to configure the operating system specifically for FORTH. The code ends with a jump to $200 where a NOP leads to a jump to COLD which assigns the parameter field of the FORTH vocabulary definition to properly link with the last word in the resident dictionary. Control then passes to WARM which assigns start up parameters to the FORTH USER area. The operating system error return is set to return to the FORTH WARM start, then control passes to the FORTH ABORT routine which leads into QUIT and INTERPRET. When INTERPRET is entered, the ASCII code for EMPTY-BUFFERS 3 LOAD has already been placed in the text input buffer, so this is the first FORTH code to be interpreted. When screen 3 LOADS, it in turn LOADS the user's application and you're off and running.

Memory Map: SC did not provide a memory map, nevertheless, it's fairly easy to find out where things are located and make your own. I do feel that a memory map should be a standard documentation item with any software as comprehensive as the SC fig-FORTH package.

As delivered, SC's software was set-up to run in 31 K ending at $7AFF. The reason for this is not clear, since 48 K was specified on the order. Perhaps SC was saving the space from $7B00 up for indirect files, who knows? The FORTH system has since been expanded to 48 K. This is not difficult to do, but a beginner would probably not succeed, because changing the size of FORTH does require a knowledge of the inner workings, unless you have specific instructions, of course.

The FORTH parameter stack occupies $9D down to $20 in page zero with the stack pointer pointing to $9E when the stack is empty. The return stack top was at $128 as delivered, apparently so as not to interfere with the interrupt vectors which have been inconveniently placed in the middle of page one by OSI. If you're not using interrupts, we suggest you expand the return stack top to $1FF. This can be done after the system is up by storing $1FF into the start-up parameter at $214 and into R0, which is a user variable. Check your assembler source code to find out the address of R0. Executing QUIT makes the changes effective.

Above the return stack, the resident FORTH dictionary begins at $200, just as other OSI language processors do. From $20E4 to $2300 there is some transient OS code that is used to boot FORTH. This can be written over by the non-resident FORTH dictionary once FORTH is booted. The OS 65D operating system is permanently resident from $2300 through $2F78. At $2F79 the FORTH text input buffer or TIB is located. This is where text input from a terminal goes to be interpreted when in the FORTH immediate mode. Above the TIB is one page of space that is reserved as a swap buffer. The main body of the FORTH non-resident dictionary extends from $317E upward. The PAD is located conventionally at $48 bytes above the end of the dictionary. Three 1024 byte disk buffers are placed at the upper end of memory with the last 128 bytes reserved for USER variables.

Dictionary: The dictionary supplied by SC includes virtually all of the words in the Ragsdale Model, mostly verbatim from the fig-FORTH document, plus several words added by SC.

Among the words altered from the Model is FORGET. This is fortunate, since the Model's FORGET will not function properly with imbedded vocabularies.

A word that should have been changed, but was not, is CREATE. As supplied, CREATE tests for the 'dictionary full' error condition based on the Model's memory map, which has the USER area immediately below the disk buffers. Since SC placed the user area above the buffers, which seems preferable, this test should have been altered by SC, but wasn't. The only time this would be important is when you're compiling with an almost full dictionary.

Several words added by SC have to do with storing and retrieving precompiled sections of the dictionary. The ability to do this is absolutely essential in any serious FORTH system. SC's approach to handling precompiled code will be discussed under the heading: Saving Compiled Code.

Also supplied as part of the resident dictionary is MON which allows one to enter the operating system from FORTH.

Disk Buffers: It can hardly be emphasized too strongly how important it is to obtain a FORTH system set-up to use 1024 byte buffers, as in the SC software. There are some hobbyist versions of FORTH available, not just for OSI but for several other computers as well, that come configured for 256 bytes per buffer. Avoid these if you can. This is almost an infallible sign that the FORTH being offered is primitive, really primitive! If the system you buy is blocked for 256 bytes per buffer you'll need four times as many disk accesses for each FORTH screen, and FORTH's disk buffers are virtual with plenty of disk operations to begin with.

Of course, one of FORTH's attributes is the comparative ease with which the block size can be altered. In principle this means it should be easy to reblock a 256 byte system to 1024 bytes. It will be fairly easy if variables and constants have been used, as in the fig-FORTH Model, in defining words that have to do with disk operations. Unfortunately, some of the primitive versions of FORTH being offered have used literals where variables and constants should have been used instead. Reblocking these versions is somewhat tedious.

These comments regarding 256 byte buffers are not meant to denegrate small-memory systems where FORTH's ability to run in very little space really shines, and 256 byte buffers may be useful. But if you have more than 16 K, why bother with a buffer that can hold only one-fourth of a screen? One feature of the FIG-Ragsdale version of BLOCK used by SC is that only two buffers are checked to see if a block is already in memory. If you use three 1024 byte buffers, as is standard in the SC software, you'll find that repeated calls for the same disk block with a call to another block intervening can result in the same block number being assigned to different buffers.

This problem can be avoided with careful programming, however, I recommend redefining BLOCK so that all three, or any number, of buffers are checked before BLOCK decides that a particular disk block is not already in the buffer area.

Editing: A very nice feature added by SC is the screen editor. Words cannot describe how much better this is than the line editor provided with the fig Model. You may need to reassign the cursor controls, depending on the type of terminal you use. This is easy to do, and SC tells how in their installation instructions.

Basically, the editor allows you to position the cursor anywhere within a FORTH screen, and type in the correction you want without having to re-enter the entire line. There are also control codes for shifting, deleting, erasing, and tabbing.

In my view, the best part of this editor is the ease with which you can modify and extend it to make it do as you wish. For example, as supplied control H places the line on which the cursor rests into the PAD. Then control R causes the line in the PAD to be printed at the current cursor position. We modified this action so that control H would save four consecutive lines in the PAD and control R would retrieve them one line at a time. The result is a very convenient means of transferring portions of one screen to another.

A number of useful words to support editing and screen manipulation are also provided by SC. For example, the word LOC will locate a word in a range of screens, MOVE> and <MOVE, move whole blocks of screens to higher or lower virtual memory. CLEAR and M.CLEAR erase one or more screens, COPY and M.COPY, copy one or more screens, etc.

Terminal and Printer: SC can provide FORTH for either video or serial terminals. The version reviewed here happens to be serial. Provision is made for rudimentary control of printer output. As supplied the word PRN turns a printer on and CRT turns it off. A trivial alteration of the constant OUTDIS causes the output to be distributed in any way desired. Input devices are just as easily specified. Control of printer format, although not specifically provided, is readily accomplished in FORTH.

Saving Compiled Code: Undoubtedly, SC's most important addition to the fig-Model has been a provision for saving the non-resident dictionary on disk.

Fig-FORTH compiles more and more slowly as the dictionary fills. If you have 48 K or more of memory, you'll find that with a nearly full dictionary compilation flows along only nominally faster than molasses in January. (Slow compilation with a nearly full dictionary is not an inherent attribute of FORTH, rather it is mostly a function of the way -FIND functions and of the way the Dictionary is structured in the FIG FORTH version.)

Because of the time required to compile large applications in FIG FORTH, it makes sense to save compiled object code on disk. Then instead of LOADing, <u>i.e.</u>, compiling, an application each time it's needed, the precompiled object code is simply read from disk to memory - a process taking only milliseconds to seconds at worst.

SC has provided the word PUT for saving object code on disk and the word GET for reading it back into memory.

There are two important restrictions in the use of these words. The first is that the dictionary preceeding the object code read from disk with GET should be the same as when the object code was PUT to the disk. The second restriction is somewhat arbitrary. As SC has defined PUT, all words in a given PUT file must belong to the same vocabulary, because only one value for LATEST is saved with the object code. An advanced programmer who understands the FORTH vocabulary linkage will be able to redefine GET and PUT to allow words from any number of vocabularies to be included within a given PUT file. But in any case the restrictions on PUT and GET are not difficult to live with.

Assembler: SC's version of the FORTH 6502 assembler is conventional. The SC assembler loads to high dictionary space. This is a good idea because the assembler is not needed at run time. PUTting and GETting the assembler from and to high dictionary makes it possible to move the assembler in and out without disturbing lower-lying dictionary.

Unfortunately, SC's method of doing this contains one of the few bugs we found in their software. To SC's credit, however, they have noted the bug in their documentation. We purchased our software about a year ago, so undoubtedly SC has made the necessary corrections. Briefly, the fault causes words that are assembled into any vocabulary other than FORTH to be improperly linked at vocabulary intersections. The reason this occurs is that SC included the definition of the ASSEMBLER vocabulary with the assembler itself, so that when the assembler is removed, the ASSEMBLER definition is also removed. Because the link across vocabulary intersections is maintained within the parameter field of the vocabulary definition itself, the proper place for the definition of ASSEMBLER is in the lower dictionary where it will reside whether or not the assembler is loaded. With the definition of ASSEMBLER always resident, the link between vocabularies declared before and after loading the assembler is correctly maintained. Loading and removing the assembler becomes a simple matter of adjusting the 'latest' field in ASSEMBLER to point either to the latest word in the assembler vocabulary, with the assembler loaded, or to the latest word in FORTH with the assembler KILLed. When the assembler is not in memory, the definition of ASSEMBLER can be SMUDGEd to prevent its being invoked.

U<: There was a bug in U< which is a 16-bit unsigned compare. The bug did not become apparent until the 31 K

system shipped was expanded to 48K. This bug was probably a carry over from the original FIG-Ragsdale version. The word U< is used primarily for address comparisons and in the form: n1 n2 U<. A true flag should always be returned if the unsigned value of n1 is less than the unsigned value of n2. Examination of the definition supplied will show that it cannot work correctly in every case where the 16th or sign bit is set. Presumably SC has corrected this bug by now, but if you have already purchased software you may need to correct this definition. The following code is adequate:

```
CODE U<

 CLC, BOT LDA, SEC SBC, BOT
 1+ LDA, SEC 1+ SBC, INX, INX,
 1 # LDA, BOT STA, 0 # LDA,
 BOT 1+ STA, CC IF, 0 # LDA,
 BOT STA, THEN, NEXT JMP,
```

A minor annoyance with the FIG software that SC did not correct is caused by the limited range or the Y register in the word ENCLOSE. As supplied, if ENCLOSE does not find a non-blank character in 256 bytes, execution control becomes lost in a infinite loop within ENCLOSE. This problem can be avoided if one remembers not to leave more than three lines in a row blank within a source screen. The best solution, of course, is to rewrite ENCLOSE.

Conclusions: Despite one or two minor flaws SC FORTH is, in our opinion, the best low cost package currently available for OSI computers. We make this judgment partly on the availability of the virtually complete assembly source code from the vendor.

The Forth Interest Group has made FORTH Models available for several microprocessors. This has apparently lead to a plethora of primitive FORTH systems based more or less directly on the Model. However, the FIG Model is just that, a model. Serious FORTH software requires that the model be considerably extended; Software Consultants have done a good job of this.

(*1) Leo Brodie "Starting FORTH, An Introduction to the FORTH Language and Operating System for Beginners and Professionals," 348+ pages, Prentice Hall, Inc., Englewood Cliffs, NJ (1981).

(*2) Forth Interest Group, P.O. Box 1005, San Carlos, CA 94070

(*) By the time this review appears in print, Software Consultants will have undergone a name change to Applied Computer Technology.

★

BAUD RATE MODIFICATION FOR THE C1P

by: Jim McConkey
7304 Centennial Road
Rockville, MD 20855

In the July Peek(65) letters section, Don Bruechert asked how one could increase the C1P's baud rate to 600 baud. This is a simple mod and here are several ways to do it. I should point out that this mod will also affect the serial RS-232 data port, but this is useful in interfacing with high speed devices such as printers or EPROM programmers. All instructions are given for the rev D 600 board. Owners of revs A, B or C and those with modified boards should check their documentation before attempting these mods.

Figure 1 shows part of the clock circuitry used in the serial interface. S is the switch on the back panel which selects printer, Cassette or modem interfaces which run at 1200, 300 and 300 baud respectively. S selects one of two clock frequencies, 124.8 KHz for 300 baud and 499.2 KHz for 1200 baud. U57 is configured to divide the input frequency by twelve. This divide-by-12 output is not symmetrical so U63 is used to further divide the frequency by two to give two symmetrical complementary 5.2KHz signals (20.8KHz for 1200 baud). One output drives the ACIA Tx clock (Asynchronous Communications Interface Adapter Transmit Clock) and the other drives U64 which is used to derive the two tones that will be recorded on tape to represent ones and zeros.

Figure 2 shows the addition of U57a, a 74LS193 (for you Radio Shack fans, that's part #276-1936, $1.49) which allows several baud rates. Notice that the input to U57 has been changed to a constant 1MHz signal. The trace leading from U57 pin 2 should be cut and a short, direct wire should be used to connect U57 pin 2 to 1 MHz. On a stock 600D board, a good place to pick up 1 MHz is at U29 pin 11.

The trace connecting U57 pin 11 and U63 pin 11 should be cut. Be careful though, not to sever the U57 pin 11 to U58 pin 5 connection. I chose to mount U57a piggyback on top of U57. A short wire should now be used to connect U57 pin 11 to U57a pin 5. This now provides the correct clock for binary divider U57a. Connect the severed line from U63 pin 11 to U57a pin 6. Also, sever the trace coming from U63 pin 9. Connect the remaining leads of U57a to +5 or ground as shown.

All that remains is to connect the proper baud rate line to the severed trace leading to the ACIA Tx/Rx input, U14 pins 3 and 4. Figure 3 shows three ways to do this. Method 1 is a brute force method using a switch. The big disadvantage of this method is that it requires sending clock signals over long wires to and from the switch. The extra radiated noise might not agree with your TV or radio (or worse yet, your neighbors!).

Methods 2 and 3 use software controlled multiplexers to select the proper baud rate. Method 2 can also be controlled by a switch providing a static control signal of either +5 or ground. Method 3 allows baud rates of 300, 600, 1200 and 2400 baud while method 2 allows any two of these, usually 300 and one other. The software control occurs through the command register at 55296 ($D800), which is normally used to enable color and sound. The output corresponding to the data line D5 is unused and is a perfect candidate to control the two input multiplexer of method 2 which uses a 74LS00 quad NAND gate (RS #276-1900, $.79) to select the proper clock. A convenient place to mount the multiplexer is piggyback on top of U58, right next to U57/U57a.

Method 3 is similar but uses a 74LS153 (sorry, but the Shack doesn't carry this one) 4 to 1 multiplexer. This requires two address lines. A careful check of the 600 D schematic shows that BK0 and BK1 go to two unconnected NOR gates. What the intended purpose of these two gates was is unclear. In any case, they are not used, and since they occupy two adjacent bit positions, it is convenient to use these two lines to address the 'LS153.

All outputs of U72 are reset to zero on power-up and any time the break key is pressed. With this in mind, both multiplexers are set up for 300 baud on power-up. This can be

changed by swapping inputs, if desired.

Figure 3 also shows the addition of a U72a. This 74LS244 (RS #276-1941, $1.99) tri-state buffer allows the reading of the command latch status. On a stock 600 D board this latch is write-only. The address decoding is already done by U20. The Y1 output goes low when a read occurs from 55296, enabling the tri-state buffer which puts the command latch status on the data bus.

The operation of the switch-based method 1 is self-explanatory. The operation of method 2 without U72a requires poking 55296 with zero for 300 baud and 32 for the other baud rate if no other options are selected. Add 2 to enable color, 16 to enable DAC, etc. With U72a, POKE 55296, PEEK (55296) AND 31 for 300 baud and POKE 55296, PEEK(55296) OR 32 for the other baud rate. Operation of method 3 is analagous:

rate	w/o U72a
300	POKE 55296,0+extras
600	POKE 55296,4+
1200	POKE 55296,8+
2400	POKE 55296,12+

rate	with U72a
300	POKE 55296,PEEK(55296) AND 243
600	POKE 55296,(PEEK(55296) AND 247) OR 4
1200	POKE 55296,(PEEK(55296) AND 253) OR 8
2400	POKE 55296,PEEK(55296) OR 12

The readable command register is a handy addition. You can now enable and disable various features by proper ANDing and ORing without worrying which other features are already selected.

The cassette interface seems to be just as reliable at 600 baud as at 300 baud and the extra speed will be immediately appreciated. With decent quality tapes, I have had no problem running the cassette at 1200 baud either, although very long lines might cause problems.

This mod seems to be very popular. I hope I have given enough detail to allow even the most inexperienced hackers to get it up and running. Since everybody seems to have it or want it, perhaps we should propose to use 600 baud as a standard rate for inter-user tape exchanges. Is anyone interested?

Figure 1
OSI 600 D Baud Clock

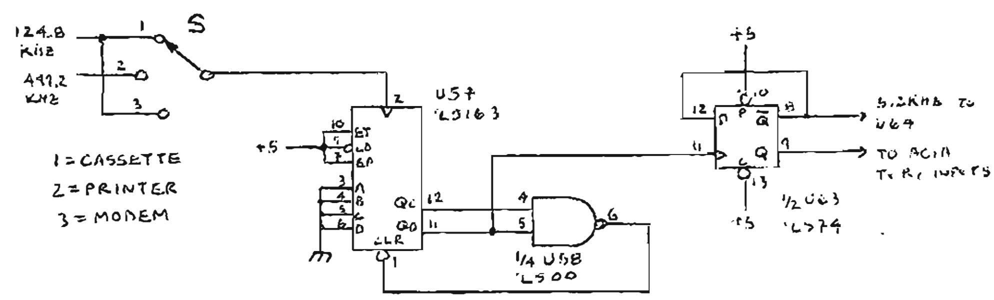


Figure 2
Modified Multi-baud Clock

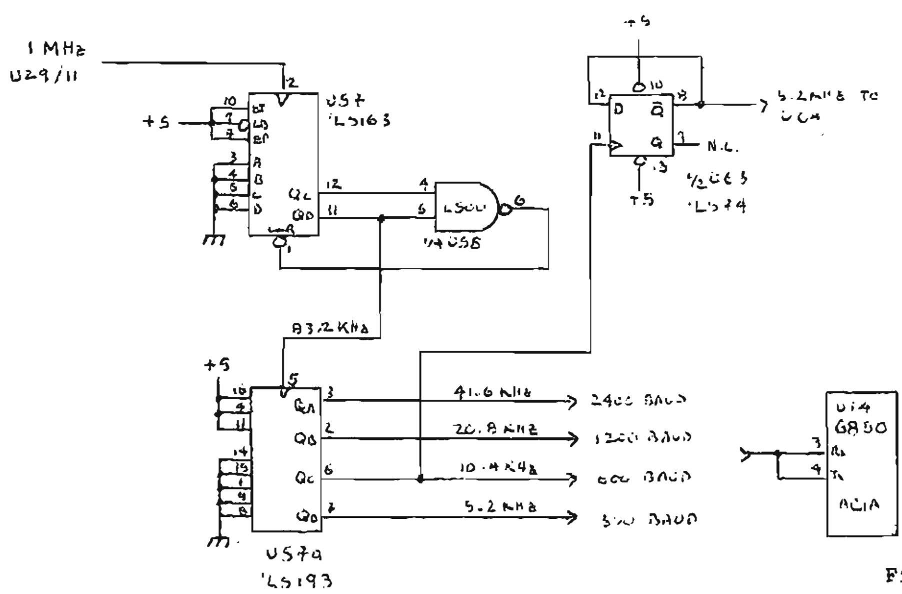
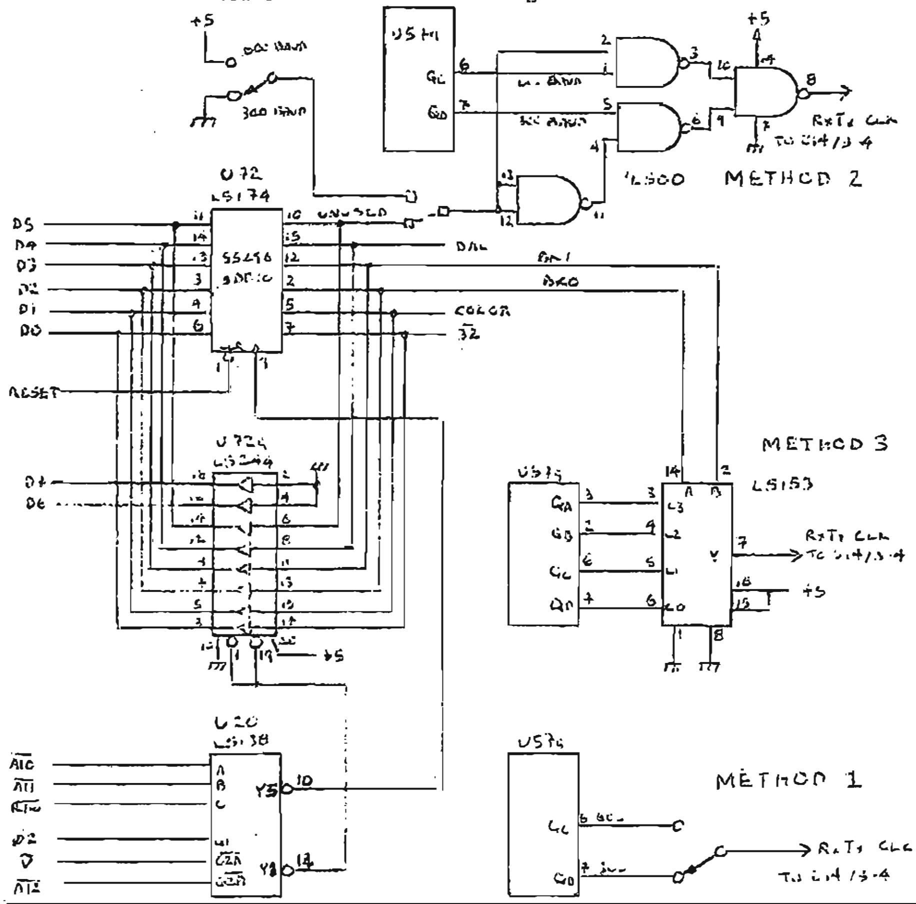


Fig 3 on page 8

Figure 3
Various Methods of Selecting Baud Clock



SUPERBOARD SECRETS

by: Frederick W. Chesson
144 Fiske Street
Waterbury, CT 06710

The OSI "Superboard II" has much to commend it, being one of the first one-board micro computers, containing not only BASIC-in-ROM, but a keyboard as well. Unfortunately, documentation has been one of the weak links in the "Chain of Command", and various individuals have attempted to close this gap with such publications as the Aardvark OSI software series and the "First Book of OSI" set by Eltron.

As a new Superboard owner, I was faced with a number of informational gaps, some of which have been closed, while others are yet to be fully bridged. The following items are passed along in the hope that they will help make the computer more familiar and more useful and interesting.

One of the most intriquing mysteries I first encountered was the 256 unit character set, with its large assortment of exotic and often puzzling graphics. The Graphics Application booklet, accompanying the computer, gave a clear picture of each graphic, but was silent as to description and application. After a detailed study and consultation with OSI and other sources, the list shown in Table One was compiled, although its descriptions may be expanded in many cases.

NUMBER 0 (zero) is the graphic seen most commonly, as it fills up the entire video display screen, whenever the set is first powered up. It and its companion, Number 1, can be used as racing cars, as seen from above, or also in less active roles as game pieces for chess and checkers.

Most of the game piece graphics appear oriented towards war games, especially those involving outer-space encounters. The sample game on page 13 of the manual pits the good ship Enterprise(in two parts, headed right or left...Numbers 9-10 and 11-12) against a battery of ground-based anti-aircraft guns. This is an encounter which Captain Kirk seldom had to face, although the gun crews in this simulation are apparently too awed by their target's reputation to fire back. To give them this option, along with scoring and two- player competition, is an interesting lesson in video game programming for the newcomer.

Number 30 is a strange-looking graphic, resembling a shovel handle, which OSI describes as a "Death Ray", though admitting that some programmers refer to it as the "birdbath symbol". It could be combined with other elements to depict the head of a monster or a robot, but when placed over Number 229, it forms an acceptable shovel for a hidden-treasure game. For constructing mazes or dungeons, elements 203-210 and 216-219 are especially useful.

Less strenuous games can be engaged in by means of Numbers 229 through 232, representing the Heart, Club, Spade, and Diamond, Number 239, by the way, makes an acceptable jet aircraft. Ground combat concludes the graphics roster, with elements 248 through 255 providing all possible positions of a tank, seen from above (or from beneath, if one is crouched in a drafty foxhole). Combat of a more personal nature is found in graphics 240 and 241, representing a Gunslinger at the ready, and with drawn gun, respectively. With the computer behind the gunman's

SUPERBOAD GRAPHICS

Number	Description	Function
0	geometric design	game piece, racing car
1	'' ''	'' '' '' ''
4	UFO Target in OSI cassette	space games
5-6	two-part, submarine.pointed left	war games
7-8	two-part. submarine,pointed right	war games
9-10	two-part, Enterprise,pointed right	space games
11-12	two-part, Enterprise,pointed left	space games
13	tree	war game, obstacle course
14	small house	war game, obstacle course
15	large house	war game, obstacle course
16-23	directional arrows	graphics, missiles, planes
24-28	proof-readers' marks	editing
29,31	wave motions, up and down	animations, games
30	"death ray" or monster head	space games
32	blank	clearing spaces
33-127	Alphameric upper and lower case figures.	titles, messages, etc.
128-178	geometric segments	animation and patterns
179-180	two-part, ship pointed left	war games
181-182	two-part, ship pointed right NOTE: 179 and 182 can be combined into an aircraft carrier.*	war games
183-210	geometric segments	animation and patterns
211-214	space ship segments	space games
215-228	geometric segments	animation and patterns
229-232	Heart, Club, Spade, Diamond	card games
233-235	geometric segments	animation and patterns
236-239	aircraft	war games, airport controller
240	gun-slinger at ready	"Quick-Draw" games
241	gun-slinger with drawn gun	'' '' ''
242-247	gun turret. in profile	war games
248-255	tank outline, from above	'' ''

* Numbers 180 and 181 have possibilities too.

pg 10 is blank.   page 15 is also blank (not included in this pdf)

This causes the listing of Line 100 and all other lines including Line 200, or the line nearest to 200, if there should not actually be a line 200.

LIST can even be part of a program. This is mentioned during a discussion of the section entitled Cassette Data Files on page 5, but not on page 8, where the BASIC commands are specifically covered. The following little program will demonstrate this capability:

```
10 INPUT N
20 PRINT
30 IF N=5 GOTO 50
40 PRINT N; "TRY AGAIN!" GOTO
   10
50 LIST
60 END
```

Line 50 can be varied to 50 LIST 40 or 50 LIST 20-50, to illustrate the options available. Note that LIST terminates the program, as may be observed by changing line 60 to 60 GOTO 10.

For a program that self-destructs at its termination, or upon branching into a "NO-GO" area, merely include NEW as shown in the following example. If NEW is suitably disguised, it could help in the prevention of unauthorized copying. In the following demonstration game, everyone eventually loses, including the program!

```
10     PRINT "PROGRAM ROULETTE"
       :PRINT
20     S=INT(RND(13)*10)
30     PRINT "ENTER A NUMBER
       BETWEEN 0 AND 9":PRINT
40     INPUT G:PRINT
50     IF G=S GOTO 1313
60     PRINT "GOOD GUESS! NEXT
       SUCKER!":PRINT
70     GOTO 20
1313   PRINT "....SORRY ABOUT
       THAT!"
1400   FOR X=1 TO 2500:NEXT X
1500   FOR Y=1 TO 32:PRINT:
       NEXT Y
13000  NEW
13001  END
```

One can add to the suspense of which message will be printed out by inserting timing loops like 55 FOR T=1 TO 2550:NEXT T and by changing 50 to read IF G=S GOTO 1300 FOR T=1TO3000: NEXT T.

When it comes to printing out data or messages of results, for good or ill, the Superboard BASIC is capable of considerably more versatility than indicated in the manual. Multiple PRINT commands spaced by a semi-colon (;) will result in entirely different formats from when the usual comma is employed. And then there is the PRINT TAB statement, which isn't even mentioned in the manual. To see what these variations on the PRINT Theme can produce, the following program may be run.

```
10 INPUT A,B : PRINT
20 PRINT A,B : PRINT
30 PRINT A;B : PRINT
40 PRINT TAB(A)A;B : PRINT
50 PRINT A; TAB(B)B : PRINT
60 PRINT TAB(A)A; TAB(B)B
70 PRINT : GOTO 10
```

For input, whole numbers from 0 to over 25 may be tried and the results observed. To observe just how creative the PRINT TAB command may be, try the following, using values of 10,.3, and 10 for A,B, and C.

```
10 INPUT A,B,C
20 FOR X=0 TO A STEP B
30 PRINT TAB (C+10*SIN(X));
   "*" :NEXT
40 GOTO 10
```

Once a good waveform is arrived at, it may be made full-time by changing the last line to: 40 GOTO 20. Good Luck!

★

STRUCTURED C1P EXPANSION

by: David Tasker
111 Bass Highway
Tasmania,Australia 7303

For a number of months now I have been reading Peek(65). Admittedly, I borrow from a friend. However, apart from this misdemeanor on my part, I find the Peek very informative and wish to offer a few unsolicited comments and some potentially useful information.

Generally (very) most user groups and newsletters (and magazines) tend to take the single rather that the systems approach to implementing hardware on computer systems. My meaning becomes clear if you consider the expansion arrangements of most C1 or C4 owners. Their systems tend to be a collection of terrific ideas from many sources. These ideas have been developed independently without any overall design philosophy.

The hardware additions that I will describe in the next several issues allow for the expansion of any Superboard (either series) or C1 to be upgraded, progressively thru.

1. Add a motherboard.
2. Add 8K onto the motherboard.

3. 4 - 2716 EPROMs (in 2K sections)
4. 24K CMOS (6116 etc.) Ram/Rom Cards which are plug compatible with 2716 EPROMS.
5. A 6821 and a 6522 PIA/VIA Board.
6. Expansion to the 16 Pin I/O Bus as on the OSI.
7. A very simple EPROM programmer for the 16 pin Bus.
8. A floppy disk board to support 5" Drives or 8" Drives.
9. Data Separator boards for MPIB51 or an interface card with D/Sep for 8" Drives.
10. Parallel Printer Interface.

In addidtion there is a VIDEO board that converts a SBII or C1 to the same video format as a C4.

Currently, I have a C1 with my Videoboard added to make it look like a C4. 2 x 24 Cmos Ram cards to provide 46K. I left my Basic 4 chip in to provide the video driver. This allows me to run the EXTENDED Monitor (EXMON) in EProm at $E800 without the need of the disks. This is great for hardware debugging. I also have my Systems monitor ($F800) on my EPROM board on the motherboard so that I can experiment with different monitors.

I have 2 Double sided 8" Drives (Qume or YE Data - there are many choices). I also support an EPROM programmer on the 16pin I/O bus, and a number of prototyping boards. My printer is a Microline 80 connected to my cassette port (Device 1) as a parallel printer. The printer interface plugs into the ACIA socket on the C1. I left the port at $F000 so that it didn't clash with a new 2K monitor. (A true C4 has its Cass port at $FC000).

This "system" is really quite easy to put together though it must look frightening to non hardware buffs. The system will run any OSI software either 65D, 65U, Asm/Editor or WP6502.

I appreciate that currently there are a large number of very exciting and excellent products available to expand the rather outdated C1. However, I believe this system to have been one of the first expansion systems to cater to the hobbyist. One of the good things about this approach is that each board is fully designed to interwork with any other, and they are all designed to be as compatible as possible with OSI software and hardware. The expansion system allows for growth from as little as more memory to as great as the system just described, i.e. one or all boards. I have no plans to market any of this in the states, primarily due to the C1 being so old (but not dead) and of course the other expansion boards now on the market. However, in the interest of those that would like to learn from the ground up hardware, I will be presenting a full set of circuits and board overlays of the different boards.

And, finally, I have printed circuit boards available for all of the above mentioned. The EPROM programmer is the one with probably the most interest to your readers. The PCB is $A10.00 with a software source listing (disk based) for $8.00 or on 8" disk for $16.00 includes disk and printed listing. The programmer plugs into any standard 16 pin I/O bus. There are 16 pin I/O bus adaptors available for the system described, the 40 pin output on the C1 and the 48 line bus, plus $2.00 surface postage or $5.00 airmail. Your readers may write direct to me.

The printer board will provide a parallel output to any C1, C2,C3 or C8 that has an ACIA available on it (other than the Disk ACIA).

★

DISK RPM TIMER

by: M. F. Putnan

Submitted by:
Donn Burke Baker
Osmosus News
3128 Silver Lake Road
Minneapolis, MN 55418

An important factor for dependable disk operation is correct rotational speed. The 5-inch MPI minifloppy used in OSI systems is required to spin at a speed of 300 RPM plus/minus 1.5%. The RPM timer given in this article allows a person to sample the speed of his drive to see if an adjustment is needed. The program is written in BASIC and uses a machine code routine which is poked into memory from data statements. The assembly listing of this machine routine has been included for reference purposes. The utility was written for a C4P-MF and runs under the OS65D V3.2 operating system. It will also run on a

C1P-MF thanks to the information and debug work provided by Donn Baker.

The method used by the program to determine disk speed is different than one might expect. The likely method would be to set up a machine code routine which counts how many loops it can execute between successive passes of the diskette index hole. This method, however, runs into problems with instruction execution time. The 2MHz 6502 processor of the C4P loses clock cycles when reading the index hole status from the slower 6820 PIA chip and this results in an inaccurate elapsed time calculation. This situation could have been remedied by introduction of the old calculating aid, the Phuje Factor (pronounced Fudge), however, I decided to come up with some other method to time the disk.

The method used was to employ the Disk's 6850 ACIA chip as a timer. A machine code routine was set up which counts how many bytes can be outputted to the ACIA between passes of the index hole. This method provided accurate timing because the ACIA's clock is never robbed of cycles. The ACIA is clocked to output 1 bit of serial data every 8 microseconds and is formatted to add a start bit, stop bit, and parity bit to each 8-bit byte of data. Thus, each byte of data uses 11 ACIA clock cycles which equals 88 microseconds (on the C1P this value is 89.518 microseconds). Knowing the number of bytes outputted in 1 revolution (index hole to index hole) the following equation is used to determine disk speed.

DISK RPM = 60,000,000 Microseconds/Minute divided by (Nbr bytes/Rev) * (88 Microseconds/Byte)

The RPM program will sample and print the disk speed 10 times and then will print out the average speed and percentage it is off from 300 RPM. Note that even though the drive is not writing, it would be a wise precaution to use a blank or scratch diskette when running the utility.

DISK SPEED ADJUSTMENT

If you ran the program and found your drive to be 1.5 % or more off from 300 rpm, your drive should probably be adjusted. The motor speed of the MPI Model B51 is adjusted by a pot on the circuit board of the disk drive as shown in

```
10  REM -----------DISK RPM UTILITY----------
20  REM      M.F. PUTNAM - NOVEMBER 1982
29  :
30  REM - TYPE LINE 60 ACCORDING TO YOUR SYSTEM
40  REM IF C4P, USE 'TIME=88'
50  REM IF C1P, USE 'TIME=89.518'
60  TIME=88
69  :
70  REM- STORE MACHINE PROGRAM AT ADDRESS 4400 HEX
80  X=0:M=17408:LOCT=M:HICT=M+1
90  READ D:IFD<>999 THEN POKE M+X,D:X=X+1: GOTO90
99  :
100 PRINT:PRINT:PRINT TAB(10) "DISK DRIVE RPM TIMER"
110 PRINT TAB (10)"---------------------
120 PRINT"THIS UTILITY DETERMINES THE ROTATIONAL
130 PRINT"SPEED OF THE DISK DRIVE.  THE AVERAGE
140 PRINT"SPEED SHOULD BE 300 RPM PLUS/MINUS 1.5%.
150 PRINT:INPUT"WHICH DRIVE (A OR B)";
    X$:IFX$<"A"ORX$>"B"GOTO150
160 PRINT:DISK!"SE "+X$
197 :
198 REM - MAKE 10 SAMPLE MEASUREMENTS OF RPM
199 :
200 PRINT:TL=0:FOR S=1 TO 10:POKE LOCT,0:POKE HICT,0
210 DISK!"80 4402":COUNT=PEEK(LOCT) + 256*PEEK(HICT)
220 R=(60000000)/(COUNT*TIME):TL=TL+R
230 REV=INT(10*R+.5)/10 :REM ROUND OFF TO NEAREST 1/10 OF RPM
240 PRINT"SAMPLE"S TAB(10) "-" REV "RPM":NEXT
257 :
258 REM - CALCULATE AND DISPLAY THE AVERAGE RPM
259 :
260 AV=INT(10*(TL/10)+.5)/10
270 PRINT:PRINT"AVERAGE=" AV "RPM":PRINT TAB(8)"=";
280 IF AV=300.0 THENPRINT"EXACT SPEED":GOTO320
290 X=100*ABS(300.0-AV)/300:Y=INT(100*X+.5)/100
300 IFAV>300THENPRINTY"% FAST"
310 IFAV<300THENPRINTY"% SLOW"
320 PRINT:INPUT'--> AGAIN";A$:IFLEFT$(A$,1)="Y"GOTO200
330 END
497 :
498 REM - DATA BELOW IS MACHINE CODE PROGRAM
499 :
500 DATA 0,0,32,84,39,173,0,192,16,251
510 DATA 173,0,192,48,251,32,46,39,32,29
520 DATA 39,32,35,68,48,251,32,35,68,16
530 DATA 251,32,99,39,96,238,0,68,208,3
540 DATA 238,1,68,32,194,39,173,0,192,96
550 DATA 999
```

```
10              ; This is the assembly source code for the machine
20              ; routine contained in the data statements of the
30              ; Disk RPM Utility.
40              ;
50              ;                    RPM*SR
60              ; THIS ROUTINE COUNTS THE NUMBER OF BYTES THAT CAN
70              ; BE OUTPUT TO THE DISK'S ACIA CHIP DURING 1
75              ; REVOLUTION OF THE DISKETTE.  NO DATA WILL BE
76              ; WRITTEN ON THE DISKETTE ITSELF.
80              ;
90  2754=       LHEAD=$2754 ; OS65D LOAD DISK HEAD
100 2763=       UHEAD=$2763 ; OS65D UNLOAD DISK HEAD
110 271D=       TRKBE8=$271D ; OS65D WAIT FOR TRK BEGIN (END OF
115             INDEX)
120 272E=       I6850 =$272E ; OS65D 6850 ACIA INITIALIZE
130 27C2=       DSKWRT=$27C2 ; OS65D DISK ACIA OUTPUT
140 4400        $=$4400
150             ;
160 4400 00     CNTLD .BYTE 0 ; NBR OF BYTES OUTPUTTED LO BYTE
170 4401 00     CNTHI .BYTE 0 ; NBR OF BYTES OUTPUTTED HI BYTE
180             ;
190 4402 205427    RPM  JSR   LHEAD    ; LOAD DISK HEAD
200                     ;
210                     ; WAIT FOR START OF INDEX HOLE
220                     ;
230 4405 AD00C0  RPM1  LDA  $C000
240 4408 10FB          BPL  RPM1     ; BRANCH IF INDEX HOLE
250 440A AD00C0  RPM2  LDA  $C000
260 440D 30FB          BMI  RPM2     ; BRANCH IF NOT INDEX HOLE
270                    ;
280                    ; INITIALIZE 6850 ACIA CHIP
```

Listing continued

portant, however, is that the length of the connection may be about 15 meters, whereas it is limited to about 2 meters for parallel ports.

I thought it would be an interesting idea to see how parallel ports do work, as this is quite simple and you are not limited to only three of them with complicated circuitry. The circuit here decodes the signals for 16 output and 16 input ports and can easily be extended. It is not yet built and tested so if you find an error in it, don't be surprised and let us know.

The 16 output ports and the 16 input ports have the same addresses. The difference is made with the Read/Write line, which will never be at logic 0 and 1 at the same moment. The ports are located at $F3F0 to $F3FF. A look at the diagram shows you the decoding circuit of those locations, made with U1, U2, N5, N6, N7, N2 and N3. U1 decodes A10 to A15. U2 decodes, together with the inverters N5 and N6, A4 to A9.

U6 and U7 are 4 to 16 demultiplexers, that are enabled by the output of U1 pin 19, and the ORing of the output of U2 and the R/W line on pin 18. For U6 this R/W line is connected directly to the output ports, for U7 it goes first to N7 to get the input ports. Then U6 and U7 decodes then A0 to A3, to give 16 different outputs.

The outputs of U1 and U2 are also ORed in N1, inverted in N8 and ANDed with the R/W and the clock 02 in N4 to get the data direction signal DD through N9 and N10. These buffers are needed to get an open collector output with enough power.

All connections to the left of the diagram are made to the 40 pin extension connector on the 600 board. Do not forget to put two 8T28's in the sockets on U6 and U7 on the 600 board. It would also be wise to put buffers on the address lines, the R/W line and the clock line. I also put buffers U9 and U10 on the data lines. This is necessary only if you already have other circuits connected to the data lines. These extra buffers serve then for the data lines, going to the I/O ports.

The circuit will consume about 110 mA without the two data buffers and about 330 mA with them.

The 16 input lines I0 to I15

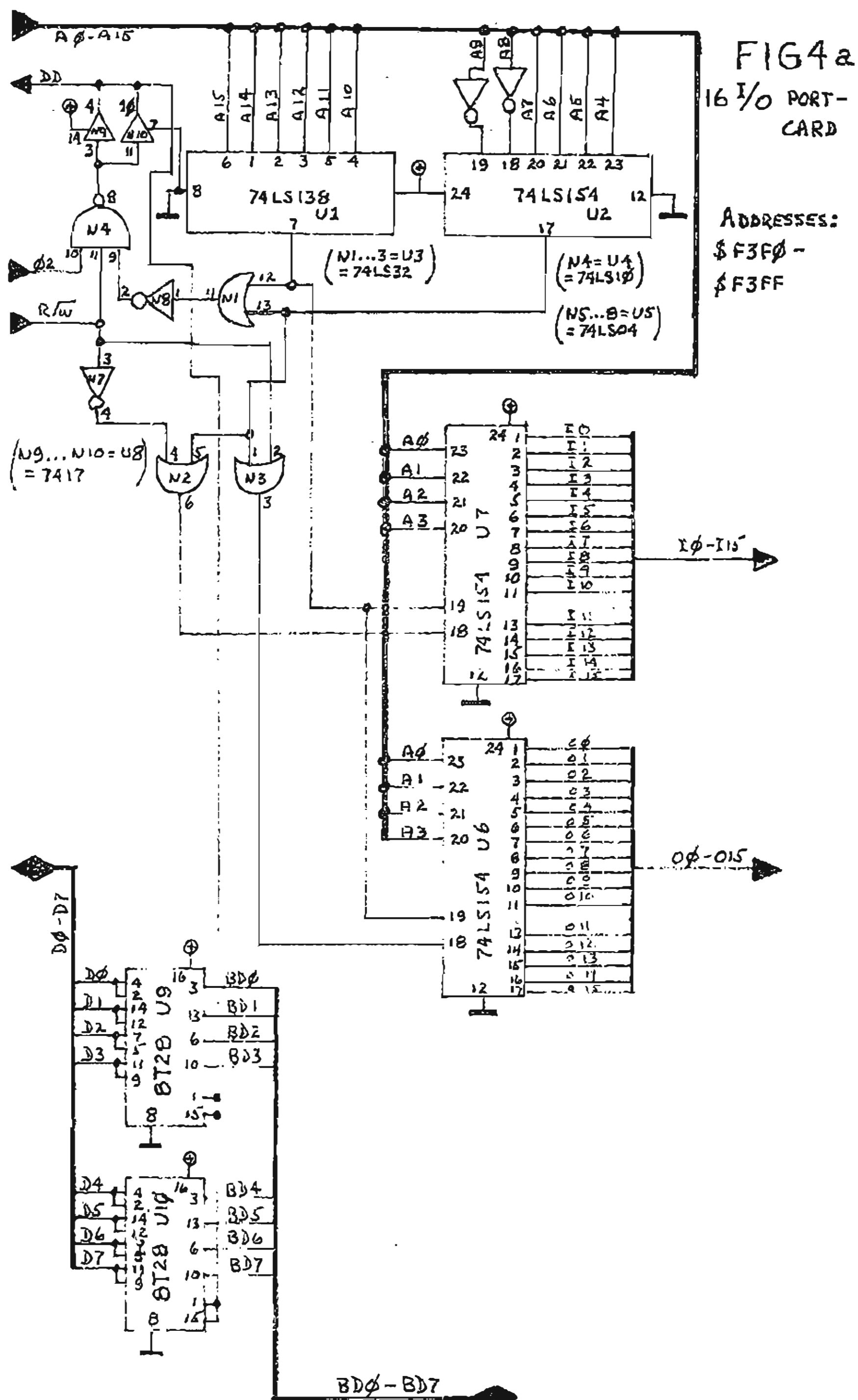


Fig 4b and 4c on page 17

and the 16 output lines O0 to O15 have to be connected to the correct circuits to form the ports themselves. For an input port this is usually an 8 bit tri-state buffer. For an output port an 8 bit latch. The lines I0 to I15 form the output enable (active low) for the tri-state buffers, the lines O0 to O15 the enable (active low) for the latches. An example of a possible input port is given in FIG 4b and of an output port in FIG 4c. Of course, there are other chips you can use. Some easy ones are also the 8212 and the 8T31 which can be connected as input or output port.

The technique used here is known as memory mapping. It consists of decoding one or more unused memory locations and putting a port there. In software you just write to or read from this location as if it was a normal memory place. For the output port, latches are almost always needed, because most external devices

are much slower than the CPU. The data is written only for a short moment at the memory location so you need to keep it there long enough to be able to use it. For input ports this is unnecessary, because you will usually have your data there longer than the CPU needs to effect a read.

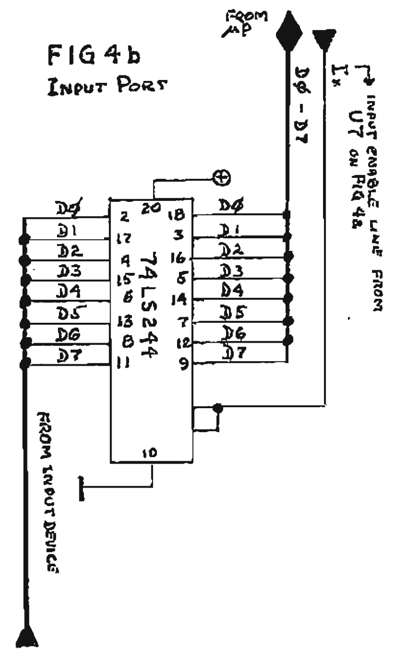
With these ports it is now possible to connect such devices as D/A converters, A/D converters, a real time clock, music synthesizers, etc. Only your imagination is the limit. If you understand the system used, it should also be easy to make more ports if you need them. Good luck.

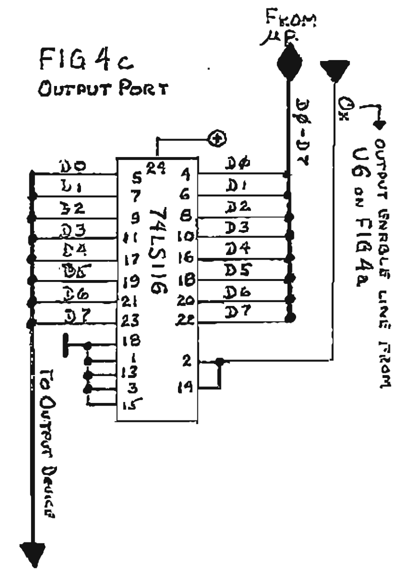
Ed's Note

2708 and 2716 EPROMs are or will be becoming scarce, due to the fact that manufacturers of chips are busy manufacturing 2732 (4K x 8) and 2764 (8K x 8) chips and are gearing up for 64 x 8 EPROMS.

6K EPROM FOR SBII OR C1P

It can be very easy to have useful assembly language programs or subroutines saved permanently in memory. Unfortunately, when you shut off the power, you lose everything





in RAM. Therefore, I designed a circuit with EPROMs, where I can store those subroutines more permanently. Of course, you see the next problem, - how to program EPROMs. This, however, is the subject of another story.

A look at the diagram of the EPROM circuit tells you that there are two 2708 type EPROMs and two 2716 type EPROMs. The 2708 is a 1K memory and the 2716 a 2K. The reason why I did not use three 2716 EPROMs is the memory map. OSI was a little wasteful with the dedication of memory to keyboard, cassette and disk controllers. I guess the reason was that the decodification

was simpler this way and thus cheaper, but it leaves us with some trouble to find some spaces 2K long.

The solution was to use 1K EPROMs instead. The first one (U1) is located on $C400 to $C7FF, the second (U2) on $F400 to $F7FF, the third and the fourth (U3, U4) on $E000 to EFFF.

However, since it is not necessary to put them all in the circuit, you can put in only those you need, and complete the others in the future when you want to put more programs or subroutines permanently in memory.

A little inconvenience is that the 2708 needs 3 different power supplies: +5V, +12V and -5V, but you can replace it with 2758, which needs only +5V.

A closer view at the diagram shows that the four EPROMs are connected in parallel with their address lines (A0 to A9 or A10, depending on which type), their data lines (D0 to D7) and the power lines. The lines 18 on U1 and U2 and lines 20 on U3 and U4 are output enables, which are permanently connected to ground. (They are active low). Each of the memories has also a chip select line (CS1 to CS4), which is generated in different ways.

CS1 is generated in a decoder (U5) of A10 to A15 and the R/W line. This together with N7 and N8. N7 ORs A15 and the R/W line, N8 inverts this signal.

The output of N4 and A13 to A15 and the R/W line goes then to N5 to be inverted. This signal, together with A10 to A12 (All inverted in N6) goes to N3, whose output is CS2.

The output of N4 is also used for the decoder U9, together with A11 and A12 to get the chip select signals for the two 2716 EPROMs, CS3 and CS4. This is possible because the three highest bits are the same in $E000-$EFFF and $F400-$F7FF.

The 4 chip selects are ORed with the R/W line and the face 2 (02) of the clock in N1 and N2 to make the data direction signal (DD) through N9 and N10 (to get an open collector output with enough power).

All connections (address lines, data lines, R/W line, 02 line and DD line) are made to the 40 pin extension connector on the 600 board. Just remember to put two 8T28's on U6 and U7 of the 600 board. They are the data buffers. It could be recommendable to put buffers on the address lines the R/W line and the 02 line.

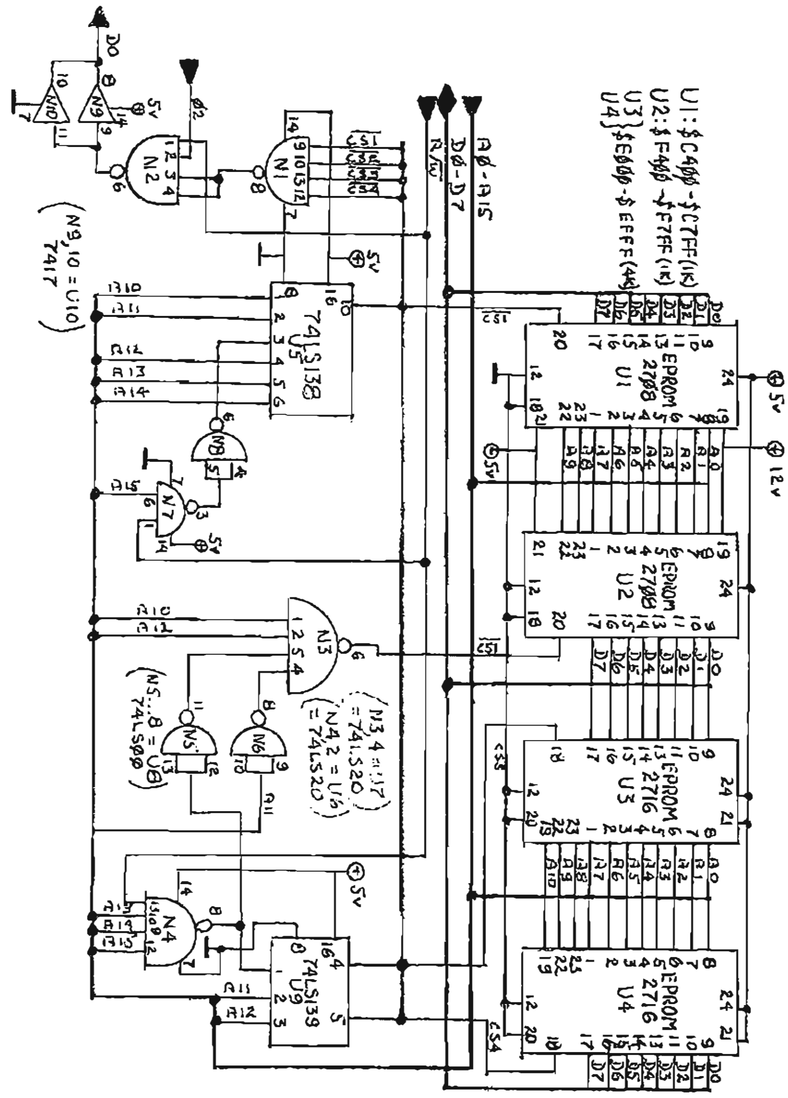
The whole circuit should consume about 200 mA at +5V, 100 mA at +12V and 60 mA at -5V if you use the LS version of the TTL chips.

One last thing, this circuit is not yet tried out, so it is possible that there are some errors in it. Any remarks about it will be welcomed.

Next time an EPROM programmer for 2708 and 2716.

★

FIG 2
5K EPROM CARD



LETTERS

ED:

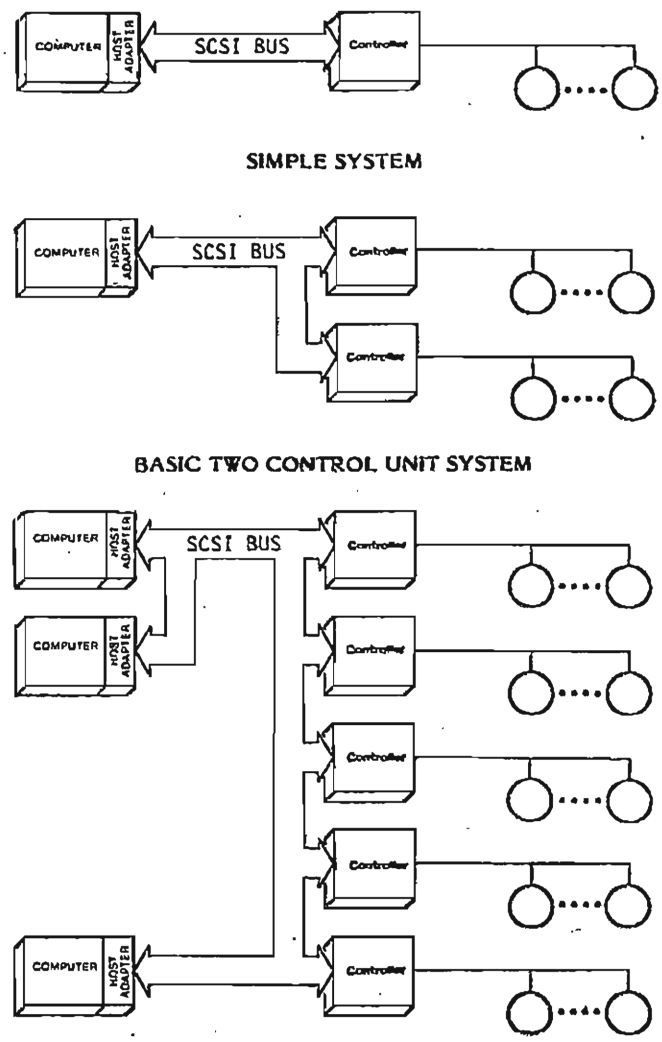
With reference to PEEK's May '83 article on page 15, HOW TO BUILD A DATA SEPARATOR AND USE IT TO INTERFACE A C1P WITH SASI-COMPATIBLE DRIVES, I would like to correct the description of the SASI bus.

The Shugart Associates System Interface is a system bus which connects between computers and peripheral controllers. It is now being standardized by ANSI under the name SCSI (Small Computer System Interface). This interface offers much more powerful commands (such as FORMAT DRIVE

OR READ BLOCKS OF DATA) than a SA-400 type drive level interface.

James Korpi
Sunnyvale, CA 94086

Peripheral Units such as Rigid Disks, Flexible Disks, Magnetic Tape, Removable Disks, Printers, Optical Disks or Communication Units



Up to 8 SCSI DEVICES can be supported by the SCSI bus. They can be any combination of host CPUs and intelligent controllers.

Figure 1
Sample SCSI Configurations

★ ★

ED:

PROBLEMS !!

There have been a number of problems and OSI manuals are of questionable value in solving them. When I read the front page of the April PEEK (65) I must admit to laughing out loud at what reads, certainly to me, as absolute nonsense. Talk of a "different company", "exciting developments", "highest management support", "I made a lot of promises to the industry and we're keeping all of them" just don't compare with my dealings with OSI. I wrote to them on October 11, 1982 detailing what I believed were failings on their part. Nothing happened until shortly before Christmas when I received a cryptic telex message from them. After a couple of such fruitless exchanges, which concluded in OSI admitting that they had lost my

letter, I wrote on January 17, 1983, personally to Ken Wortz, whose name had just been published in PEEK(65). He replied on February 23, 1983, telling me that my letter was dated January 28 (perhaps they had lost that one too - my copy is clearly dated 17 January !) and asking me to accept his apologies for delays in the past. I was advised that someone had been asked to research my problems and respond accordingly. Mr Wortz also wrote: "I'm very sensitive in relation to how our customers regard our product and support". At the date of writing, 11 April 1983. I have not yet received an answer to my letters of October and January. I know how I regard the product, but I still have not seen any support !

Maybe PEEK(65) readers can be of assistance in my problems, I believe that I have given OSI ample opportunity and they have failed miserably.

The first problem was with WP-3. We were using WP-3 on C3 floppy disc and it was a clear stipulation that we should be able to use both WP-3 and 65U on our new hard disc system. When the 230E was installed in August 1982 we discovered (as I had suggested to the NZ dealers in May 1982) that WP-3, being based on 65D, could not run in a 65U multi-user system. For one month we retained the C3 just for word processing work until a disc arrived from USA. This turned out to be WP-3-3 adapted for 65U time sharing systems. It worked, in a manner of speaking, and our WP work was transferred to the 230E. A week after receiving WP-3-3 we received the unexpected blow - a substantial account for WP-3-3. Despite my protests we were obliged to pay and that prompted my first complaint that OSI had created new equipment which was incompatible with existing systems. Surely we should not have been obliged to purchase expensive "new" software simply to continue doing work we had already set up on OSI equipment!

In addition, the new WP-3-3, delivered in September, was subsequently upgraded to version 1.1, (not for us though) and in December we were advised by the OSI telex message that WP-3-3 was obsolete and no longer supported by OSI! You've really got to move fast to keep up with this crowd.

WP-3-3 is less versatile than WP-3-2. It cannot access files on floppy discs, device 8 turns up as device 3, printing to device 8 turns up on user 1 screen, and it is totally incompatible with 65U v1.42 utilities such as PRTMAP. We have many WP files which are accessed only occasionally and which would properly be stored on floppy - but you must run a special file copy program to install the files on the hard disc, then recopy back to the floppy when complete - very cumbersome. Also, WP-3 has always had a dreadful problem in underlining headings longer than 60 characters: half of the heading turns up typed on top of itself at the start of the line. This was not fixed in the expensive new WP-3-3.
I had many problems with WP-3 when I used PRTMAP to allow both user 0 and user 1 to access their individual printers (#5 and #8) via the device 5 driver to allow paging. Naturally the semaphores had to be turned off as the system would otherwise allow them to access device 5 alternately (it thinks there's only one printer). This worked well until WP-3-3 came along and blundered into the semaphores making printers lock up left, right and centre ! I can't even delve into WP-3-3 as there is no EXIT or EXXX command to get out to DOS or BASIC. The only way out is to return to BASIC thus 65U is reloaded and WP-3-3 is gone. WP-3-3 was obviously thrown together in a hurry. I wonder how OSI could take a 65D word processor, place it into 65U with 24K operating system, and yet manage to lose some of the facilities !

A further problem is that we were originally (with the C3) supplied by the OSI dealers with a Spinwriter 5530 (Centronics parallel) as a word processing printer. I now realise that we haven't a hope of using it as a word processing printer since the proportional spacing facility of WP-3 will never match the proportional spacing facility of the Spinwriter. The 230E has a Diablo word processor parallel output, but that's no use to the Spinwriter ! How annoying to be told that there is all that electronics in the machine, XTRA ON available in WP-3, but no way to use them !

A final complaint - the 230E manual has errors, no illustrations, ("insert illustration here") and is generally scrappy - surely there will be a better manual issued and surely existing users are

entitled to receive manuals with the illustrations.

Frankly, I'm disgusted with OSI. There must be something wrong with a company that can't manage to reply properly by April to a letter sent to them in October! Reading between the lines I get the impression that OSI are putting all their effort into selling equipment and no time or effort into supporting what they have already sent out. That policy will rebound on OSI very quickly, they will find that those who have been caught once won't be back for more, and they can't keep on finding totally new clients.

Colin Law
Aukland, New Zealand

* * * * *

ED:

I would like to respond to the article on RIGHT JUSTIFIED TEXT FOR WP-6502 on C1P/C4P. I have spent a lot of blood, sweat, and tears, learning about the tape and disk versions of WP-6502. I have found it necessary to do this because I have an OSI C1P which I have modified with the Aardvark video modification and at different times a C1S and currently a C1E Monitor ROM. I currently have versions of WP-6502 which will run with the C1S, C1E Monitor ROMs and a version which runs with HEXDOS and the C1E. As you may guess by now, I know WP-6502 quite well. I have also modified my version so it has a Print command rather than getting to print through the View command. It numbers pages at the center of the bottom of the page and allows spaces at the beginning of a new line.

The entry points mentioned by Leo Jankowski in his article do not correspond to the entry points in my tape copy of WP-6502.

View	$0795
Type	$0A80
Move	$0B3A
Zap	$0F67
L/Edit	$0B9C
G/Edit	$0E03
Blk View	$0D70
R/Tape	$0EB2
W/Tape	$0EF3

I hope these comments have been of some help to your readers.

John T. Roecker
Minneapolis, MN 55410

* * * * *

AC 612 925 3490

ED:

We love your publication over here - if you think OSI dealers feel cut-off or neglected in the U.S., imagine us in Europe where OSI cannot afford to market their products yet (all emphasis is now on the U.S. market, and properly so)!

We have a number of 4 and 5 user installations here in London, all based upon the 74 megabyte Winchester. Most of them use at least 60 megabytes for data files, and this leads to back-up time of nearly 2 hours on the Alloy Tape Cartridge system (4 tapes x 28 mins/tape). I realize that Alloy have a reel-to-reel system which would cut the time in half (and I understand each reel can hold 30-40 megabytes?), but I wonder if anyone out there in the serious business-user world of OSI:

1. Has hands-on experience with the Alloy reel-to-reel.

2. Has another tried and tested back-up system been found? Has anyone interfaced a streamer?

Next time any of your readers are in London, I hope they'll give me a call!

Nicholas Clough
Salient Systems Ltd.
London, England

Nicholas:

Yes! lots of people have had excellent experiences with the Alloy reel-to-reel. It even uses the same software you already have. It is the only one we know of that's really mated to OSI although there are others, and other systems (even video tape players) that do the job, but not as quickly or as safely. One nice feature of the Alloy reel-to-reel is that the data is in standard IBM format. Hence, data can easily be transferred.

Ed.

* * * * *

ED:

I have an OSI C4PMF and was considering getting a second 527 (2MHz) memory card for it when they suddenly became unavailable. Does anyone still supply them? I'm still interested.

I'd also like to hear from anyone who's replaced the 2316 boot ROM on the 505 board with a 2716 EPROM. I want to replace the ridiculous keyboard scan routine (only used at boot time anyway OS65D v3.21) with a much shorter one of my own, and I want to stick a couple "handies" up there, too.

According to the 505 schematics, the 2716 is pin-compatible with the 2316 except for the polarity of the chip select signals. On my 505 I can trace these to some wire-wrap pins. Is rewiring these all there is to the reconfiguration? I'm curious.

Charles Kline
Urbana, IL 61801

Charles:

The 527 (2Mhz) memory board is available from Cleveland Consumer Computer and Components, 1333 S. Chillicothe Road, Aurora, OHio 44202. Telephone 1-800-321-5805. The populated board sells for $180.00 and the bare board sells for $28.00.

The answer to your second question is also, yes! Rewiring is all that is necessary.

Brian.

* * * * *

ED:

First, Robert Crane who wrote last month looking for an Assembler/Disassembler should check out Steve Hendrix's HEXASM. It is a very powerful micro assembler which comes on a 5 1/2" disk also containing a simple editor, a fast machine code program renumber utility, a disk test program and disk copiers for single and dual drive systems. A disassembler is also available on the HEXDOS disk. I'm not sure why Steve has them on two separate disks.

Second, I need some help from your readers. Has anyone successfully interfaced slow (>1micro sec) memory or other peripherals to a C1? If so, how? I've been trying to interface a slow real time clock without success. When I try to use the RDY line, the computer hangs. I tried a clock stretcher from a 6502 app note, but BASIC gives me a ?SN Error and prints two separated zeros everytime I do a print PEEK (clock). The clock works great if I run the entire machine at 500 khz instead of 1 mhz. Next, does anyone know where in ROM the following three routines lie and what their inputs and outputs are?

1. A routine to find the current value of a given variable.

2. A routine to create a new variable and give it a value.

3. A routine to do large data block moves.

Thanks!

Jim McConkey
Rockville, MD 20855

* * * * *

AD$

SALE: 32K Superboard II w/2 Shugart drives; 12" monitor; supply (5V@10A,12V@4A); joysticks; uninstalled 540 board w/out documentation; OS65D V3.1; 100+ games and utilities; manuals; Aardvark and Peek(65) journals. $1000 or best offer. James Williams (201) 766-3292

* * * * *

FOR SALE: 48K Challenger with terminal. System in excellent condition and is available for $1,200.00. For more information, contact Mike Guidry (318) 988-1300 during office hours.

FREE PEEK(65) SOFTWARE LISTING FORM

PROGRAM NAME ______________________________

BASIC - VERSION /_/_/_/_/

/D/ OS65-D /U/ OS65-U /C/ CP/M

CATEGORY

/G/ GAME /B/ BUSINESS /U/ UTILITY /O/ OTHER

MINIMUM COMPUTER

/1/ SB,SBII, C1P,C2/4P /4/ C4P /8/ C8P /O/ C2/3OEM

/D/ C2/3D /2/ C200,C3A/B /3/ C300

MINIMUM STORAGE REQUIREMENTS

/C/_/ CASSETTE /5/_/ 5 1/4" /4/_/ 8" FD /7/_/ CD-7

/2/_/ CD23/28/36/74 (USE 2ND BOX FOR NO. UNITS REQUIRED)

TERMINAL SYS. /S/ SERIAL /V/ VIDEO /B/ BOTH

SYSTEMS SUPPORTED - MAX 2 CODES - RECORD LOCK ASSUMES MULTI-USER

/S/ ONE USER /M/ MULTI-USER /H/ HARD DISK /R/ RECORD LOCK

SOFTWARE SUPPORTED BY:

/D/ DEALER /P/ PHONE /M/ MODEM /N/ NONE

/O/ OTHER

SOLD BY

/A/ AUTHOR /D/ DEALER /M/ MAIL ORDER /O/ OTHER

COPIES IN CIRCULATION (# / 10); 1 = <11, 11 = 100-110 /_/_/

PRICE, RETAIL, PLUS TAX & HANDLING OR SASE $/_/_/_/_/NO CENTS

NAME AND ADDRESSES

AUTHOR	SELLER (or SAME)
NAME ____________	NAME ____________
STREET ____________	STREET ____________
CITY ____________	CITY ____________
STATE /_/_/ ZIP ______	STATE /_/_/ ZIP ______
PHONE ( ) -	PHONE ( ) -

DESCRIPTION - MAXIMUM 10 LINES OF 30 CHARACTERS

1 ______________________________
2 ______________________________
3 ______________________________
4 ______________________________
5 ______________________________
6 ______________________________
7 ______________________________
8 ______________________________
9 ______________________________
10 ______________________________

PEEK (65)
The Unofficial OSI Users Journal

P.O. Box 347
Owings Mills, Md. 21117

BULK RATE
U.S. POSTAGE
PAID
Owings Mills, MD
PERMIT NO. 18

DELIVER TO:

19608WLYJ300DG 3001*8404:8
DG WILEY, JR.
3001 LINDA LN.
SINKING SPRING, PA. 19608

S

GOODIES for OSI Users!

PEEK (65)
The Unofficial OSI Users Journal
P.O. Box 347 • Owings Mills, Md. 21117 • (301) 363-3268

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

( ) **C4P Sams Photo-Facts Manual.** Includes pinouts, photos, schematics for the 502, 505, 527, 540 and 542 boards. A bargain at — $15.00 $ ________

( ) **C2/C3 Sams Photo-Facts Manual.** The facts you need to repair the larger OSI computers. Fat with useful information, but just — $30.00 $ ________

( ) **OSI's Small Systems Journals.** The complete set, July 1977 through April 1978, bound and reproduced by PEEK (65). Full set only — $16.00 $ ________

( ) **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 OS-65U. Complete disk sort and merge, documentation shows you how to call from any BASIC program on any disk and return it or any other BASIC program on any disk, floppy or hard. Most versatile disk sort yet. Will run under LEVEL I, II, or III. It should cost more but Sanders says, "...sell it for just..." — $89.00 $ ________

( ) **KYUTIL** - The ultimate OS-DMS keyfile utility package. This implementation of Sander's SORT/MERGE creates, loads and sorts multiple-field, conditionally loaded keyfiles. KYUTIL will load and sort a keyfile of over 15000 ZIP codes in under three hours. Never sort another Master File. — $100.00 $ ________

BOOKS AND MANUALS (while quantities last)

( ) **65V Primer.** Introduces machine language programming. — $4.95 $ ________

( ) **C4P Introductory Manual** — $5.95 $ ________

( ) **Basic Reference Manual** — (ROM, 65D and 65U) — $5.95 $ ________

( ) **C1P, C4P, C8P Users Manuals** — ($7.95 each, please specify) — $7.95 $ ________

( ) **How to program Microcomputers.** The C-3 Series — $7.95 $ ________

( ) **Professional Computers Set Up & Operations Manual** — C2-OEM/C2-D/C3-OEM/C3-D/C3-A/C3-B/C3-C/C3-C' — $8.95 $ ________

( ) Cash enclosed ( ) Master Charge ( ) VISA

Account No. ________ Expiration Date ________

Signature ________

Name ________

Street ________

City ________ State ________ Zip ________

TOTAL	$ ________
MD Residents add 5% Tax	$ ________
C.O.D. orders add $1.65	$ ________
Postage & Handling	$ 3.50
TOTAL DUE	$ ________

POSTAGE MAY VARY FOR OVERSEAS