Editorial

WHERE DO WE GO FROM HERE?

Every time I begin or end a new volume of Proteus News, I ask myself whether it is worth carrying on. As the years have passed, Proteus members have sold their Solis, lost interest, gone beyond the need for a user's group publication, or whatever, resulting in slowly diminishing membership. This is to be expected. The thirst for information was tremendous in the beginning of Sol's life, but by now the industry has matured and people have diversified in their interests and needs.

My original plan after Processor Tech liquidated was to keep the users' group active to disseminate the information we all need or may have need, and I feel this has been done. The Encyclopedia Processor Tech is compiled and available. Other documentation into reference collection. The content of Proteus News was restricted to items of interest to Sol/PTC hardware owners, and there was no other source.

I personally have learned a great deal about microcomputer hardware and software from you who have sent letters and articles for publication. But our members have been contributing fewer articles each year. It appears that the needs of our group have been met pretty well and I would like to free Proteus to expand to a broader scope. This doesn't mean that Proteus will cease to exist or that it will ignore the Sol. In fact, I really don't know what Proteus will evolve into in the coming years. I want to allow it the freedom to go beyond a Processor Tech focus.

There is one important piece of Sol business that I would like to cover in this year. Since Sol was manufactured, the S-100 bus has become standardized by the IEEE society as the IEEE 696 S-100 bus standard. Processor Technology's use of the S-100 bus comes very close to the standard, but does not quite meet it in some respects. My goal for this year will be to examine the Sol and other Processor Technology hardware to document: (1) how it differs from the IEEE 696 standard, (2) how to alter it for full IEEE 696 compatibility, and (3) how to disable the 8080 microprocessor from the Sol's architecture to allow any IEEE 696 standard CPU board to take over the Sol's slave device. This will let the Sol keep up with the rapidly changing technology of microcomputing. (See my article on future shock in this issue.)

I can't do this all myself, but I have looked into the topic already and plan to write enough to stimulate other

(continued on page 2, left column)
members to participate. If you are interested in following along with this discussion and learning more about the SOL and microcomputers in general, then I highly recommend that you obtain some reference material and background knowledge.

If you do not yet understand the basics of electronics (things like voltage, current, resistors, transistors, diodes, etc.), then I recommend the book published by Radio Shack called "Understanding Solid-State Electronics". It is an excellent beginning and will carry you quite far.

For more information about TTL integrated circuits (like the multitude of 74LS... integrated circuits that are the building blocks of the SOL and other boards), I recommend Don Lancaster's "TTL Cookbook" published by the Howard W. Sams company, available through most electronics stores. You needn't master the whole book; the first few chapters will give you what you need.

A very practical book which I highly recommend is the paperback on the IEEE 696 bus written by Sol Libes and Mark Garetz, published by Osborne/McGraw-Hill, entitled "Interfacing to SOL/IEEE 696 Microcomputers". I have the 1981 edition, which was written when the standard was still in development. The final standard (approved at the end of 1982) differs slightly from the original proposal, so I expect that a revised edition of this book will be published. Mark Garetz, as chairman of the IEEE 696 standards committee, wrote an article in the February 1981 issue of Byte magazine, outlining the approved standard and pointing out how it differed from the proposed one. I recommend this article as a supplement to the book.

And of course you should have a copy of the SOL technical manual, giving the schematics and theory of operation. It was distributed with the early SOL computers, but Processor Tech moved into the small business market they stopped giving it away with the computer. If you didn't get a big black notebook with your machine, you can get the same information in Volume 2 of the Encyclopedia Processor Technica, obtainable from Proteus.

I must admit that I have been growing tired after running the show for so long. My first batch of Proteus News, originally called SOLus News, was in August of 1977. That's almost ancient history as microcomputing goes. But I feel revitalized by the decision to let Proteus expand. I will continue to publish all of the SOL articles I can get, but we'll just slowly run out of steam if that's what we want. I want to do more with new programming languages, high resolution graphics, virtual memory, artificial intelligence, and so on. These are going to require hardware beyond the capability of SOL's 880. The Smalltalk-80 programming system from Xerox, for example, needs about 50K of virtual memory. With 256K memory chips becoming cost effective, we'll soon look upon the 64K address space of the 880 as tiny.

Rather than discarding the SOL or dismembering it by removing the SOL-PC, I think the most rational approach is to let the SOL become a generalized SOL-100 machine, keeping all I/O interfaces, keyboard, power supply, walnut shell, and all. Then we can install our chosen microprocessor CPU into a bus slot and update the SOL. Ideally, we should be able to flip a switch, disable the bus CPU, and re-enable the SOL's 880 to run our old software whenever we want.

This project of documenting the conversion of SOL to
(Future Shock continued from page 1)
SMALLTALK-80:

THE LANGUAGE FOR THE NEXT ERA?

by Stan Sokolow

When I first heard the name "Smalltalk", I thought it was a little language to teach kids about computers. You know, something like Tiny Basic. But I was wrong. Smalltalk is not small and it doesn't talk.

Let me quote from the book which is my source for this article:

* Smalltalk is a vision.
* Smalltalk is based on a small number of concepts, but defined by unusual terminology.
* Smalltalk is a graphical, interactive programming environment.
* Smalltalk is a big system.

(Smalltalk-80: The Language and its Implementation, Adele Goldberg and David Robson, Addison-Wesley Publishing Company. 1983, page vii.)

...continued...

Smalltalk is a language and operating environment developed by Xerox at their Palo Alto Research Center (PARC). People tell me it was inspired by some LISP-ish ideas, but it is also kind of FORTH-ish. I'm not enough of a programming language guru to comment on that. But I can say, from what I've read about it, it is a different way of looking at programming than I am used to from the contemporary languages (FORTRAN, BASIC, COBOL, PL/I, Pascal).

What's different about it? It has a different way of looking at computing. Instead of procedures which pass arguments, Smalltalk has "objects" which pass messages to each other. If the object understands the message, it will act upon it and send back a reply. For example, instead of adding 1 to a counter with an arithmetic assignment statement that acts upon variables, in Smalltalk you send the number (which is an object) a message to add 1 to itself. Since numbers know what addition is, the result is computed and returned as the reply.

This may sound absurd, but consider other possibilities. You could send the message "sort" to an array object or to a file object; or you could send the message "rotate" to an object which contains the representation of a molecule on your graphics display.

Now build up a library of definitions which tell the computer what classes of objects exist in your programming universe, and for each class what messages are understood by members of the class, and for each message what methods (here's where the programming comes in) are used to act upon the messages by members of the class.

Include in your definitions a bunch of classes of objects that programmers like to use: Sets, Bags, Dictionaries, OrderedCollections, SortedCollections, LinkedLists, Intervals, Arrays, Bitmaps, ByteArrays, Strings, Symbols, Numbers, FloatingPoint, Fractions, File, Streams, FileDictionaries, and so on.

Add to your objects those items that are often thought of as parts of an operating system: Processors, File, FileStreams, Semaphore, etc. Let programmers send messages to these objects. For example, you could send the message "create" with some arguments to the ProcessorScheduler and create a new process (task) executing concurrently.

Then let the programmer build up methods which send messages to objects, get replies, pass replies to other objects as messages, and so on. Let classes be defined as subclasses of others, so they can inherit the methods of their parent classes without having to redefine them all over again.

What have you got? You are building a modular programming system that is extensible by the programmer and which allows him to think in terms that are appropriate to the problem he is solving. He can create new classes of objects, like "ComplexNumber", and define the operations (messages) it can act upon, in future programs, he can simply use the new class and its messages as though it were a part of the programming language all along.

If a better method for computing something is needed, just that class definition can be changed and all other objects which use that class will use the new method without knowing it.

What have you got? You have the Smalltalk programming language and operating system.

But Smalltalk is not just a language and operating system. It is an interactive programming environment, too. Programmers can edit and debug their programs in their high-level language (Smalltalk). Running programs can be inspected and viewed with a "browser" that will show the messages being processed, protocols which define the methods, etc. All of the programmer's interaction is carried out in a high-resolution bitmapped display screen, with the use of a cursor moving device (mouse), some buttons, and a keyboard.

Most of the interaction is done by selection of items from menus using the mouse to point.

When an error occurs in the execution of a process, the process is suspended and the programmer is notified on the display. The error message (in words the programmer can understand) appears as the heading of a box on the screen, and in the box is a list of objects that are acting on messages and the messages the are processing. This is comparable to seeing a the stack of the procedure calls and arguments that were descended to get to the present place in a Pascal program.

The programmer can then get more information about the suspended state of the process, such as finding out exactly which statement was being executed and the values of all of the variables in the system. He can make changes to the source code of the "methods" (procedures) and resume the program where it left off.

All of this is done on the screen with overlapping boxes that look something like sheets of paper placed down upon each other. When the top one is removed, the next one down becomes visible again, until you finally get back to the original screen. The whole system itself is programmed in Smalltalk, except for the machine-coded interpreter.

Why is this interesting? There are a few reasons.

The interactive video ideas which came from Xerox PARC's work have been incorporated into some of the newest personal computing systems, such as the Apple Lisa and the VisiCorp product called Visi-on.
Intel has developed a new computer (called the IAPX 432) that is said to be an object-oriented machine. In other words, it is a machine which more closely implements the kind of architecture that is needed for the newest high-level languages.

Well if the Xerox PARC people like programming with objects, and Intel is making an object-oriented computer, I guess we ought to start learning a little more about objects. In fact, I suppose Xerox felt the same way, so they have announced the availability of commercial licenses and university licenses to the machine-independent Smalltalk-80 system.

Smalltalk-80 is the 1980 version of Smalltalk. It is implemented much like UCSD Pascal was; that is, in a pseudo-machine code which requires a machine-specific interpreter to execute. (It is NOT the same as the UCSD p-code interpreter.)

I'm sure we'll hear much more about Smalltalk in the future. If any Proteus member has access to a Smalltalk system, I would like to get some comments. Meanwhile, I recommend the book I mentioned, although you should be prepared to be confused by it the first time through. I have the three supplementary books on order, but they are still being written.

THE IEEE 696 STANDARD AND SOL
Part 1: Compliance and Upgrading as a Bus Master
by Stan Sokolow

This is the first in a series of articles on the S-100 bus standard and how it affects Processor Technology products, with emphasis on the SOL computer. In this part, we will look at the standard and compare it with the SOL's use of the S-100 bus. We will show how to upgrade SOL to comply with the standard. Later, we will discuss how to bring other Processor Technology products into various degrees of compliance with the standard, and how to convert the SOL into a bus slave rather than a bus master so a new CPU board can be installed in the bus.

What we won't discuss to any great extent in this part is the timing relationships of signals on the SOL bus. The IEEE standard gives various timing specifications, but I have not investigated these in the SOL yet. If any members have found timing problems other than the few I mention, please let me know about them.

I want these articles to be a dialog among SOL owners so we can pool our knowledge and experience. I don't pretend to be an expert, so I am counting upon you to collectively examine my proposals, find the flaws, make new proposals, comment on the correct ones, etc. After publication of each Proteus News issue, I hope to receive letters from you with corrections and improvements upon my proposals. I will organize the ideas and present the new information in the next issue.

In the series, I must assume a certain level of knowledge. If you are having a hard time following the technical jargon, please see my editorial in this issue for more background and references. The article in Byte, February 1983, by Mark Garett is an excellent place to start.

INTRODUCTION TO THE IEEE 696/S-100 BUS STANDARD

The IEEE is the Institute of Electrical and Electronics Engineers. One function of the IEEE is the establishment of device standards, and the IEEE has been developing the written standard for the S-100 bus. This IEEE 696 standard has been adopted by the IEEE standards board on December 9, 1983. The standard describes the mechanical and electronic specifications for the computer bus commonly known as the S-100 bus.

The SOL computer uses an S-100 bus in the expansion card cage. Obviously, it was designed before the IEEE standard, but the standards committee carefully considered the existing S-100 hardware when proposing the standard. Consequently, most of the existing S-100 equipment (including SOL) comes very close to the adopted standard.

However, there are some differences. Some of the unused lines in the S-100 de facto bus have been defined to have functions in the standard. Some of the lines used by just a few manufacturers for special features have been defined to have new uses. Some of the lines which only had meaning for the 8080 microprocessor have been redefined so that the bus can accommodate a wide variety of microprocessors in an interchangeable manner. We'll examine these differences, signal by signal.

FUNCTIONAL ORGANIZATION OF THE IEEE 696 BUS

The IEEE 696 bus consists of 100 lines that carry the signals which communicate among devices in the computer system. Each line corresponds to a pin position in the 100 pin connector. If you hold an S-100 board in your hand with the component side facing you and the S-100 edge down, the pins will be numbered from left to right, 1 to 50, on the front (component) side. Flipping the board over so that the solder side is facing you and the edge connector is still down, the pins are numbered from right to left, 51 to 100, with pin 51 directly behind pin 1.

Most lines are only supposed to be in one of two states (high or low, also known as +5 or 0 volts) except for brief moments when making the transition between them. In reality, the standard specifies that any voltage less than or equal to +0.8 volts shall be recognized as low, and any voltage greater than or equal to +2.0 volts shall be recognized as high.

NOTATION

Each bus signal has a specific interpretation for the meaning of the high and low states. Those that are "on" (also known as asserted or active) when the line is low are given names that end with an asterisk and are said to be "active low" signals. Signals that are active when the line is in the high state have no asterisk. In electrical engineering notation and in SOL documentation, a bar over the symbol is usually used to designate active low, but the IEEE committee decided it was easier to type their documents on a word-processing machine if they used something instead of the bar, so the asterisk was adopted. For example, SIXTH*. In this series, I will also use the asterisk notation, even when referring to SOL signals that are shown in the SOL manual with a bar. Read the asterisk as "star", but think "bar".
BUS DRIVERS

The devices which generate the high and low voltages on the bus lines are known as bus drivers. There are three types.

Active drivers accept (sink) current in the low output state and provide (source) current in the high output state. If the same active drivers are connected to the same bus line and one goes into the high state while another goes into the low state, a conflict occurs on the bus, which is not to be permitted. For this reason, the other types of drivers are used where several drivers must operate a bus line.

Open collector drivers sink current in the low output state, but they neither accept or provide current in the high output state. Open collectors allow several such drivers to be connected to the same line, which is pulled to a high state by a 1000 ohm resistor (the "pull-up resistor") to prevent the line from floating somewhere in the system. When any of the open collector drivers go into the low state, the line is pulled low and the asserted driver sinks current through the pull-up resistor. Several can be asserted (low) at the same time. Others can be in the high state without creating a bus conflict.

Tri-state drivers are the third type. These devices have a disable input which can put the driver into a high-impedance state. In the high-impedance state, the driver is said to be disabled or floating, and it is virtually isolated from the bus line it drives, neither sourcing nor sinking current from the line. Some bus signals are used to disable the tri-state drivers on other bus lines, such as the address and data drivers from the CPU to the bus.

FUNCTIONAL GROUPING

The lines are functionally divided into 8 groups, although they are intermingled in their physical position within each apparent rhyma or reason. (Actually, the pin positions were determined by the guys who built the first S-100 computer, the Altair. It was simply a matter of layout convenience; they just ran the signals to the nearest bus pin.)

The functional groups are Address, Data, Status, Control Output, Control Input, TMA Control, and Utility.

MASTERS AND SLAVES

The devices that plug into the bus are divided into masters and slaves. Bus masters are devices that take control of the bus, such as microprocessor boards (CPU's). Bus slaves are devices that take orders from the bus master, such as memory boards and input/output ports. In one system, there can be up to 16 temporary bus masters, one permanent bus master, and many bus slaves. Temporary bus masters are devices that can temporarily take control of the bus away from the permanent master in order to communicate directly with the slaves. Right now, temporary bus masters are direct-memory access (DMA) disk controllers or alterate CPU's.

The term "permanent bus master" is somewhat misleading because this master does yield control to the temporary bus masters and therefore is not permanent. The major distinction is that the permanent master generates the system clock, and it is the primary master, having initial control and regaining control between temporary master accesses. (Primary and secondary would have been better terms than permanent and temporary, but I think the standards committee must have liked the sound of "TMA" as a generalization of the DMA concept.)

ADDRESS BUS

There are 24 lines in the address group. The 8-bit microprocessors, such as the 8080 in the Sol, only have 16 address pins since they can only address 2 to the 16th power (65,536) memory cells (bytes). Thus, the older 8-bit S-100 computers (like the Sol) only carried 16 address lines in their bus. The newer microprocessors can directly address more memory than that, so the bus has been extended by another 8 lines, giving the 24th power (16,777,216) directly addressable memory cells in the IEEE 696 bus.

The Intel 8086 microprocessor, for example, has 20 bits of effective address information in its architecture, so it can address about 1 megabyte of memory. The Motorola 68000 can address the full 16 megabyte space. At the moment, this sounds like a ridiculously large amount of memory, but as the cost of memory continues to drop, I'm sure that new programming systems and applications will make use of more memory than we can guess. Sixteen megabytes of address on the bus gives room to grow which is needed last for a few years.

The older 16-bit-address boards can still be handled on the bus if the total memory is restricted to 65,536 bytes (ignoring the 8 extended address bits) or if the boards are modified to recognize the 8 extended address lines. We will discuss extended addressing in more detail later.

The addresses on the bus always refer to bytes, even if the device holds 16 bits in each physical memory cell. That is, in 16-bit memory boards, the low order address bit will refer to the left or right byte of the 16-bit memory cell. The other 15 bits of the address will refer to the actual cell number. The byte accessed with the low order address bit set to 0 is referred to as the even-addressed byte (or the even-data byte), and correspondingly the other byte is referred to as the odd-addressed byte (or odd-data byte) because its address ends in a 1.

DATA BUS

The IEEE 696 bus also has extended the data line group. This was not done by assigning any more data lines, but rather by changing the data bus group into a bidirectional one. The original S-100 bus had separate lines for data flowing to the microprocessor and data flowing from the microprocessor. These were called the Data In and the Data Out lines (DI and DO). Each consisted of 8 separate lines which carried 8 bits of data only one way (to the CPU or away from the CPU). In the IEEE 696 bus, the 16 data lines can carry data to or from the microprocessor, with lines in the control bus acting as the traffic signals. This has effectively doubled the capacity of the data bus.

The designers of the Sol actually used a bidirectional data bus in this way. They have described their reasons for doing this in one of the early articles on Sol that is reprinted in the Volume 1 of the Encyclopedia Processor Technica. So, as you notice in your Sol schematics, the Sol's S-100 data lines are referred to as the DIO lines (data in/out). But still the Sol only has an 8-bit processor, so both halves of the DIO bus carry identical bytes of data. For example, on output the Sol sends a byte of data to memory on the 8 DO lines and on the 8 DI lines. The DI and DO lines always carry identical information in the Sol.

The IEEE standard actually allows for a mixture of 8-bit and 16-bit data devices in the same system. A new pair of
signals (sixteen-request and sixteen-acknowledge) allow bus masters to determine whether the addressed slave device will put 8 bits or 16 bits on the data bus lines. Older devices, like our 8-bit memory boards, will not know about the sixteen-request, so they will not acknowledge, and the master will know to look for only 8 bits of data on the appropriate side of the data bus. This is a clever way to let us make use of our old equipment while extending the system to 16-bit processors.

The first proposal for extending the data bus named the new 16-bit path the DATA lines (rather than D1 and D0) and numbered them from 0 to 15, with 0 being the high-order byte. But the adopted standard has changed this interpretation of the data bus. Since some microprocessors put 8 bits out at a time, the high-order byte first and some with the high-order byte second, there arose a conflict when trying to make the bus independent of microprocessor architecture. Old bus lines DATA 8 through DATA15 contain the high-order byte or the low-order byte?

The standards committee finally solved the dilemma by designating one side of the data bus for the even-addressed byte and the other for the odd-addressed byte. That way, the data is always consistent on the bus and it is up to the CPU to look at the proper part of the bus when interpreting the data. Some CPU boards will have to flip the data around for their microprocessor and others won’t.

The data bus now has two naming conventions. In 16-bit transfers, there are the ED and OD lines, each numbered from 0 to 7. ED lines for 16-bit transfers if desired. 8-bit transfers, the ED lines are designated OD, since they use the same lines as the data-out lines in the pre-standard 8-100 microprocessor. Likewise, the OD lines are the same as the D1 lines. That is, in 16-bit transfers of data, the odd-addressed byte is put out on the data-in lines and the even-addressed byte is put on the data-out lines, regardless of the direction of transfer. In 8-bit transfers, the old unidirectional path (D1 and D0) still is used so old boards will meet the standard.

Confusing? Yes, but we can probably ignore the mess unless we want to put a mixture of 16-bit microprocessors on the same 8-100 bus.

STATUS BUS

The status bus is used by any device in the system to determine which bus cycle the current master is in. A slave device looks at the status lines to determine whether the addressed and data signals. The status lines have names beginning with a lower case "s". Almost all are found in the 8086, except the new sXVRG16 sixteen-request line. The 8 status lines will be covered in detail in the numeric round of the bus pins.

CONTROL OUTPUT BUS

The control output bus gives 5 timing signals known as strobes. The names of these signals all begin with the lower case letter "s" because they originate on the current master (processor) board. They will be covered in the numeric section.

CONTROL INPUT BUS

Six lines make up the control input bus. They send signals from slaves to the master to tell the master to do something special. The RDY and XRDY tell the master to wait longer for the data to be valid during bus transfers from the slave to the master. The HOLD signal requests that the permanent master stop and relinquish control of the bus to another master. The IMT signal is the interrupt signal to the master which can be disabled (masked) under software control. The NMI signal is a non-maskable interrupt request generally reserved for interrupts that must occur regardless of the interrupt to process the power-failure detection condition. And SINT* tells the current master that the slave can send or receive 16-bit data as requested.

VECTORED INTERRUPT BUS

The devices which can interrupt the processor can be ranked into 8 priority levels, each sending an interrupt request on a separate line. Together these make the vectored interrupt. Someplace in the system these lines must be connected as inputs to an interrupt controller that decides when to signal the CPU on the INT line.

The Sol doesn’t contain a vectored interrupt controller to arbitrate among these requests, but the old S-100 bus definition did include the lines because other S-100 manufacturers did make interrupt controller boards. If interrupts are to be used in a Sol, the Sol’s address decoder circuit must be modified by jumpers as described in the numeric section, and an interrupt controller should be added.

TMA CONTROL BUS

The concept of "Direct Memory Access" (DMA) has been extended to cover access to the bus by any of 16 temporary bus masters. The process of transferring control from the permanent master to a temporary master is called "Temporary Memory Access" (TMA). The 16 temporary masters use 4 new lines to signal a lower priority level to the other masters and to decide which one of them is to gain control if several request access at the same time. This is known as bus arbitration. Another four lines are used by the temporary masters to disable the permanent master’s bus drivers, allowing the temporary master to control the bus.

TMA access is too complex for me to explain in this series, but the book by Mark Garetz does it well. The lines used will be mentioned in the numeric section of this article. In a later part, I’ll look at the way the Helios controller gains access to the bus to see how well it complies with the IEEE 696 standard.

UTILITY BUS

The other lines on the bus have been lumped into this miscellaneous category. It includes the power and ground, clocks, reset, clear, and error lines. These are explained in the numeric section.

Three lines have been left explicitly undefined (called the NDEF lines), so that manufacturers can tuck their own unique signals somewhere. The NDEF lines must be clearly explained in the manufacturer’s documentation and it is up to the user to avoid conflicts when integrating various hardware using the NDEF lines.

Four lines have been reserved for future use by the standard and should not be used for any purpose. These are described in the NUF lines. Reserved for Future Use, Reserved for Future Use, Reserved for Future Use, Reserved for Future Use.
COMPLIANCE OF THE SOL BUS, PIN BY PIN, (NUMERIC SECTION)

This is a pin-by-pin comparison of Processor Technology Corporation's use of the S-100 bus with the IEEE 696 standard adopted December 9, 1982.

******************************************************************************
Please note: These updates are still in the proposal stage. Do not make the changes in your SOL, except on a test basis.
******************************************************************************

Pin 1: Standard signal +8 V

This is the main power supply. The standard calls for no less than 7 volts instantaneous minimum, no more than 25 volts instantaneous maximum, and average less than 11 volts.

Compliance: Although the SOL's specifications comply with the standard, some SOL's were made with an improperly specified transformer in the power supply. Processor Tech issued an update calling for these SOL's to be outfitted with an additional transformer (called a bucking transformer) to lower the secondary voltage. Check your SOL's backplane voltage with a DC setting on the voltmeter, pin 1 to pin 50 (ground), being careful not to short circuit any other pin. This will give you the average voltage on the line. To measure instantaneous voltages, an oscilloscope is required.

Upgrading: Methods for lowering excessive backplane voltage have been described in back issues of Proteus (SOLs) News. The bucking transformer update is in the SOL technical manual.

Pin 2: Standard signal +16 V

The standard calls for 14.5 v to 35 v instantaneous, and average voltage less than 21.5 volts.

Compliance: The SOL specifications are within the range. Check your SOL if in doubt. IEEE VERY CAREFUL NOT TO SHORT CIRCUIT YOUR POWER SUPPLY BY ACCIDENTALLY CONNECTING BUS PIN 7 (+16 V) AND BUS PIN 52 (-16 V) SIMULTANEOUSLY WITH YOUR VOLTAGE PROBE! These are exactly opposite each other in the connectors. An in-line fuse would be a wise safety precaution to add between the SOL power supply and the S-100 backplane power connector.

Pin 3: External ready signal, used by an external device to signal the master that the bus is not ready. The bus is ready when both XRDY and FRDY (pin 72) are asserted; when either one is low, the master will wait.

Compliance: SOL complies. SOL only generates wait states by pulling the FRDY line low, not the XRDY, but the SOL does respond to both XRDY and FRDY from the bus when generating the READY signal to the SOL's BIOS microprocessor.

Pin 4 thru 11: Standard signals VI0* thru VI7*

Vectored interrupt request lines.

Compliance: SOL doesn't use the lines, so SOL is in compliance by default. Vectored interrupt signals have no effect on SOL unless an interrupt controller is added and the SINTA line (pin 96) is modified as described below.

Pin 12: Standard signal NMI*

Non-maskable interrupt. SOL designates it the XRDY2 line, but SOL doesn't use it, so SOL complies as a non-implimented feature.

Pin 13: Standard signal PWRFAIL*

The power-failure detection signal. The line goes low at least 16 ms before the power supply lines will drop no lower than the voltage regulators will fall to regulate. It is not specified as an open collector line, but it is required to meet the specifications for an open collector (see description in driver section).

Compliance: SOL doesn't use the line, so SOL complies as an unimplemented feature, but a pull-up resistor should be added since this line acts like an open-collector. See the section below on open-collector lines.

Pin 14: Standard signal TMA3*

One of the TMA arbitration lines. These are used by temporary bus masters to communicate bus-request priorities among each other.

Compliance: Unused by SOL, so SOL itself complies. However, the Helios (which will be examined in a future article) is a non-complying TMA device unless modified. It doesn't use this type of arbitration.

Pin 15: Standard Signal A16

Extended address line 18. Compliance: Unused in SOL. See extended addressing.

Pin 16: Standard Signal A16

Extended address line 16. Compliance: Unused in SOL.

Pin 17: Standard Signal A17

Extended address line 17. Compliance: Unused in SOL.

Pin 18: Standard Signal SDBS*

Status disable: disables the processor's bus drivers on the 8 status lines: smENR (pin 47), sM1 (pin 44), sOUT (pin 45), sINP (pin 46), sW0 (pin 97), sW1 (pin 96), sW2 (pin 48), and sXTRQ (pin 58).

Compliance: The SOL also disables 8 status lines, but they are not quite the same. The SOL has smENR, sM1, sOUT, sINP, sW0, sW1, and sW2; but sXTRQ is not implemented in the SOL because it is a 6-bit processor. See pin 58 below. Instead, the eighth status line in the SOL is pin 98 SSTACK, which has been redefined in the standard. See pin 98 below.

Upgrade: Make the pin 98 and pin 58 upgrades. Since SOL-PC does not generate sXTRQ (pin 58), but other devices placed in the bus may respond to it, sXTRQ must be held high. This is done in the pin 58 upgrade by using the driver made available after pin 98 is upgraded. This driver is disabled by SDBS*, so it will now meet the standard.

Pin 19: Standard Signal CDBS*

Control-output disable: disables the 5 control output lines: pSVA('G) (pin 76), pSTVAL* (pin 25), pDBIN (pin 78), pWR (pin 77), pPHDLA (pin 26).

Compliance: This signal is called C/C DSB (command/control disable) in SOL documentation. SOL disables 6 control lines instead of 5. The extra line in SOL is the PINTE pin 24, which has been dropped from the standard and instead is an RFU line; see pin 28. SOL also disables a control output signal pNINIT (pin 27) which has been re-defined as an RFU line in the standard. And the phase clock which is used for standard signal pSTVAL* (pin 25) is not disabled by C/C.
DSB. So, Sol needs upgrading.
Upgrade: See the upgrades for bus pins 25, 26, 27.

Pin 20: Standard Signal 0 V (ground)
A grounded line.
Compliance: Sol doesn't use this pin. In the Altair and
Imal it was used for the UNPROTECT signal.
Upgrade: Jump pin 20 of the bus to ground pin 100.

Pin 21: Standard Signal NDEF
Not to be defined in the standard. This is reserved for
machine specific use.
Compliance: Not used in the Sol.

Pin 22: Standard Signal ADSB*
Address disable. Disables all 24 address line drivers of
the permanent bus master (CPU).
Compliance: Sol disables the 16 address lines, but not
the 8 extended address lines. However, Sol does not drive the
extended address lines, so it lets them float and thus
complies. See extended addressing section.

Pin 23: Standard Signal DODSB*
Data output disable. Disables the D07 through D00 lines
in 8 bit transfers. Disables all 0D and ED lines in 16 bit
transfers.
Compliance: Sol disables the 8 DIO lines. Since the D1
and D0 lines are tied together in Sol, all 16 lines are
disabled by the DODSB* signal. So, Sol complies.
However, when the lines are driven in 16 bit transfers, there will be
conflicts between the lines tied together.
Upgrade: See the section on the 16-bit data lines.

Pin 24: Standard Signal φ
The master timing signal for the bus.
Compliance: In the Sol, the timing signals are those
needed for the 8080 microprocessor clocks φ1 and φ2
(phase 1 and phase 2). The Sol uses pin 24 for the phase 2 clock, which
meets the requirements of the standard signal φ. So, Sol
complies.

Pin 25: Standard Signal pSTVAL*
Processor status valid strobe.
Compliance: The Sol and other 8080 based S-100 systems use
this for the phase 1 clock, which also meets the requirements of the
signal pSTVAL*. The standard requires that pSTVAL be
 gated by pSYNC to be used. Sol doesn't use this signal for
 anything other than phase 1, but generates it for the bus, so
Sol complies. However, the Sol doesn't disable this signal when
CDSB* pin 19 is asserted (low), so in this respect, Sol doesn't
fully comply. If a temporary bus master tries to take over the
bus and drives the signal pSTVAL*, a bus conflict will occur.
Upgrade: Abandon the driver at pins 4 & 5 of U77 (which
is not disabled by CDSB*), and instead use the driver at pins 2
& 5 of U50 (which is disabled by CDSB*). This U50 driver was
abandoned by the pin 27 upgrade. To do this, cut the traces at
pins 4 & 5 of U77 and install a jumper from pin 6 of U92 to pin
2 of U50 and one from pin 3 of U50 to bus pin 25. You must also
swap the IC's U50 and U77 so that the pin 25 driver remains an
ST79, since this is required by the Sol design. The driver at
pins 4 & 5 of U77 will be used by the SLAVE CLK* pin 54
upgrade.

Pin 26: Standard Signal PHILDA
Hold acknowledge.
Compliance: Sol generates PHILDA, so it complies.

Pin 27: Standard Signal RFU
Reserved for future use by the standard.
Compliance: Sol uses this for the 8080 WAIT status onto this line,
so Sol doesn't comply.
Upgrade: Since Sol doesn't need the WAIT status, cut the trace from pin 3 of U50 to the bus pin 27 and from the
8080 pin 24 to pin 2 of U50. This will remove Sol from this
line and abandon the driver, which is to be re-used for
another purpose. (See the Helios article in subsequent part of
this series for corresponding modification to the Helios
controller.)

Pin 28: Standard Signal RFU
Reserved for future use by the standard.
Compliance: Sol uses this for the 8080 signal PINE, the
interrupt enable line. This is not available from most other
microprocessors, so the standard has dropped the signal.
Upgrade: Simply cut the trace leading to pin 28 of the Sol
S-100 bus.

Pins 29 through 34: Standard signals A5, A4, A3, A15, A12, A9
Address lines.
Compliance: Sol complies.

Pins 35 & 36: Standard signals D01/D01 and D00/D00
Data out / even data lines.
Compliance: See the section on the data lines.

Pin 37: Standard signal A10
Address line
Compliance: Sol complies.

Pins 38 through 43: Standard data lines (various)
Data in /out odd/even lines.
Compliance: See the section on the data lines.

Pin 44: Standard signal #1
The status signal that indicates current cycle is an
op-code fetch.
Compliance: Sol complies.

Pin 45: Standard signal #OUT
The status signal indicating bus transfer to an output
device.
Compliance: Sol complies.

Pin 46: Standard signal #INP
The status signal indicating bus transfer from an input
device.
Compliance: Sol complies.

Pin 47: Standard signal #MEM
The status signal indicating transfer from memory to a bus
master other than interrupt-acknowledge or op-code fetch
cycles.
Compliance: Sol complies.

Pin 48: Standard signal MH7A

The status signal that acknowledges that a halt instruction has been executed.

Compliance: Sol complies.

Pin 49: Standard signal CLOCK

A 2 MHz clock signal (plus or minus 0.5% tolerance with a 40-ns clock cycle) which is not required to be synchronous with any other bus signal, but is available for use by such devices as baud-rate generators or counter-timers.

Compliance: Sol drives CLOCK with the inverse of the signal which can be found at point B of the clock circuit option jumpers (same as pin 10 of U90). The timing diagrams in the Sol manual show that this signal has a period which is either 5, 6, or 7 times the dot clock period of 70 ns, depending upon the clock jumpers. With the usual jumpering, the Sol generates a 2.045 MHz clock, which comes closest to the standard for pin 49 CLOCK but is out of the 0.5% tolerance range. At higher clock frequencies, the Sol is even farther from the standard for pin 49.

Upgrade: For most purposes, 2.045 MHz is close enough, so no upgrade is needed if Sol is kept at the usual 2.045 MHz frequency. However, some systems may use this signal to clock a timer that keeps track of real time, so accuracy may be very important. If you want strict compliance with the standard, or if you have altered your Sol to have a faster clock cycle, you will need to find an alternate way to generate the 2 MHz clock.

There are hybrid circuits available now which generate a TTL clock in one small package the size and format of a 14-pin DIP package. Jameco Electronics (1355 Shorway Road, Belmont, CA 94002, 415-592-8097) sells a 2.000 MHz TTL oscillator for $9.95 in their 1983 catalog (page 21). It only requires connecting power (+5V), ground, and the output line. You can install it in one of the spare spaces in the Sol-PC, either U103 or U104. Since there are 16-pin DIP spaces, you'll have to adjust the oscillator at one end and jumper the oscillator's power or ground pin to the correct hole at the other end.

Upgrade: Cut the trace at S-100 pin 57, jumper that trace to pin 56, and change the Sol documentation. A similar change should be made on the ParaSol debugger board.

Pin 50: Standard signal 0 V (ground)

Signal ground, common with pin 100.

Compliance: Sol complies.

Pin 51: Standard signal +8 V

The 8 volt power supply, common with pin 1.

Compliance: Sol complies.

Pin 52: Standard signal -16 V

Instantaneous maximum less than -14.5 volts, instantaneous minimum greater than -35 volts, average minimum greater than -21.5 volts.

Compliance: Sol specifications comply.

Pin 53: Standard signal 0 V (ground)

Common with pin 100.

Compliance: Sol doesn't use this line, although documentation refers to it as the SSDI sense switch input. (It must have been used in the Altair and Imseal front panels.)

Upgrade: Jumper pin 53 to pin 100 ground.

Pin 54: Standard signal SLAVE CLR*

A reset signal which resets only the bus slaves. Must be active when POC* is active, but may also be asserted by external means (such as a slave reset button).

Compliance: Sol has no connection on pin 54, so it fails to assert SLAVE CLR* when it asserts POC* (Most slaves don't care anyway, however.)

Upgrade: Use the internal gate at pins 4 & 5 of U77 to gate SLAVE CLR* to POC* or SLAVE CLR* also. That is, make the pin 25 pF added upgrade which cuts the traces to and from these pins of U77. Then install a jumper from pin 12 of U77 to pin 4 of U77 and jumper from pin 5 of U77 to pin 16 of U77 so POC* is asserted on the bus, SLAVE CLR* will also be asserted.

Pins 55, 56, & 57: Standard signals TM4*, TM5*, TM6*

Three of the TMA arbitration lines.

Compliance: The Sol does not connect to pin 55 or 56, but it does make non-standard use of pin 57 for a signal called DIG1* (Data Input Gate 1). This is used by the Paraso1 to force PDRIN low in the Sol being tested. Since it is machine specific, it should be relocated to an NDEF line.

Upgrade: Cut the trace at S-100 pin 57, jumper that trace to pin 66, and change the Sol documentation. A similar change should be made on the ParaSol debugger board.

Pin 58: Standard signal AXTNG*

Sixteen request, the signal which the master uses to request a 16-bit data transfer.

Compliance: Conmec Electronics sells this as FRDY*, which disables MWR1TE drivers when a front-panel button is requesting the bus, as in the Altair and IMSeal. The standard provides no way to disable MWR1TE (standard signal MK1*, pin 68). So FRDY* is not needed to meet the standard. However, the ParaSol debugger uses FRDY* to control the Sol being tested. So FRDY* should be relocated to an NDEF line. Also, pin 58 must be driven into the high state to assure that it will not be asserted by floating to a low level. It is not an open-collector line.

Upgrade: The trace from bus pin 58 to pin 1 of U49 should be cut and then pin 1 of U49 should be jumpered to pin 65 of the bus. This trace is from U45 to U14 in the high state to assure that it will not be asserted by floating to a low level. It is not an open-collector line.

Pin 59: Standard Signal A19

One of the extended address lines.

Compliance: No connection in the Sol. See extended addressing section.

Pin 60: Standard Signal SIXTH*

The sixteen-acknowledge signal generated by bus slaves in response to the AXTNG* request signal.

Compliance: Unused in the Sol. Should have a pull-up resistor added to indicate an active 8-bit transfer and to meet the requirement for pull-ups on open collector lines. Jumper a 1000 ohm resistor from pin 60 to +5 volts.

Pins 61 through 64: Standard signals A20 through A23

Extended address lines.
Compliance: No connection in the Sol. See the section on extended addressing.

Pin 65: Standard Signal NDEF

Not to be defined in the standard, so available for machine specific use.

Compliance: Sol does not use this line, but it is needed to carry the signal FRDY* which must be relocated from pin 59 of the Sol bus. See the pin 59 upgrade and re-label this line as FRDY* in the Sol documentation. Make corresponding changes to the Parasol debugger board.

Pin 66: Standard Signal NDEF

Not to be defined in the standard, so available for machine specific use.

Compliance: No connection in the Sol, but needed to carry the Sol-specific signal Digi* which must be relocated from pin 57.

Upgrade: See pin 57 upgrade.

Pin 67: Standard Signal PHANTOM*

The phantom device disable signal. When PHANTOM* is asserted (low), normal slave devices will disable themselves and phantom slave devices will enable. Primarily used to disable RAM which is addressed where a ROM needs to be during initial instruction fetch after RESET* or POC*.

Compliance: The Sol generates PHANTOM* for the first four clock cycles after RESET* or POC*. During this time, the 8080 fetches its first instruction from addresses 0 through 3, which in fact are located in the Sol ROM. PHANTOM* also is used to force the Sol's address decoder to respond to the address bus as though the Sol's ROM were addressed at 0. Therefore, Sol compiles as a master which generate PHANTOM* as a phantom slave. However, if any device on the bus tries to assert PHANTOM*, there will be a conflict with the Sol driver since its a permanently enabled tri-state driver that is normally in the high output state, rather than an open collector driver.

Upgrade: Install a 1000 ohm pull-up resistor from pin 67 to +5 volts, and put a germanium diode in place of the phantom jumper on Sol, cathode (banded end) toward U77. When U77 pin 9 is low, current will flow from the pull-up resistor, through the diode, and into U77. The voltage drop across the germanium diode is small, about 0.25 volts, so pin 9 will be more than 0 volts to be recognized as the low state. When U77 pin 9 is high, the diode prevents current from sourcing back to the bus. Thus U77 acts like an open collector, sinking current when low, but not sourcing current when high.

Pin 68: Standard Signal MWR

Memory write. Generated from two bus signals by the logic equation: MWR = PWR and not SOUT

Compliance: Sol generates MWR just as described. In addition, although the standard does not provide an MWR disable signal, the Sol uses the non-standard FRDY* signal to disable the MWR driver. See pin 58 upgrade.

Pin 69: Standard Signal RFU

Reserved for future use by the standard.

Compliance: No connection in the Sol.

Pin 70: Standard Signal 0 V (ground)

Common with pin 100.

Compliance: Already grounded in the Sol.

Pin 71: Standard Signal RFU

Reserved for future use.

Compliance: No connection in the Sol.

Pin 72: Standard Signal RDY

The processor ready signal which when low causes the CPU to wait for a slow slave.

Compliance: Sol compiles.

Pin 73: Standard Signal INT*

The maskable interrupt request signal.

Compliance: Sol compiles.

Pin 74: Standard Signal HOLD*

The hold-request line used during TMA.

Compliance: Sol compiles.

Pin 75: Standard Signal RESET*

Resets bus masters; must be asserted when POC* is asserted.

Compliance: Sol compiles.

Pin 76: Standard Signal PSYNCH

The control signal identifying the initial clock cycle in a bus cycle.

Compliance: Sol compiles.

Pin 77: Standard Signal PWR*

Identifies the presence of valid data on data bus.

Compliance: Sol compiles. See the data bus section.

Pin 78: Standard Signal PDBIN

Requests data transfer from slave to master using the data bus.

Compliance: Sol compiles. See the data bus section.

Pins 79 through 87: Standard Signal Address lines (various)

Address lines in the primary 16-line address bus.

Compliance: Sol compiles.

Pins 88 through 95: Standard Signal Data bus lines (various)

Bidirectional data lines.

Compliance: Sol compiles. See data bus section.

Pin 96: Standard Signal SINTA

Interrupt acknowledge.

Compliance: Sol generates the SINTA signal, but the address decoder does not properly respond to interrupt acknowledge states unless the Sol is modified.

Upgrade: Revision E Sol's have two jumpers that should be installed; connect A8 to AC and connect A8 to AD. Revision D Sol's do not have these jumpers and the IIC's are used somewhat differently. The following changes should be made to a Rev D Sol-PC: Jumper pin 8 of U57 to pin 6 of U34; jumper bus pin 96 to pin 9 of U57. This uses a spare gate in the Rev D Sol.
Pin 97: Standard Signal SNO*

Identifies a data transfer from master to slave.

Compliance: Sol complies. However, you should be sure to make the change described by Processor Technology in their Sol Rev N to Rev P change notice. This change corrects a potential data bus timing conflict during read cycles. I'm not sure when the conflict arises, but the change apparently alters the generation of the data bus driver disable signal OUT DSBL* found at the U47 output pin 6 (below the 8080 on the Sol CPU schematic). Instead of using the internal signal DSN* which comes from the processor and not the S-100 bus, the modification will use the bus signal SNO* to determine when to disable the drivers.

The change consists of disconnecting the output of U45 pin 10 from the input of U47 pin 5 and jumpering the bus pin 97 (SNO*) to the input pin 5 of U47. This change will be essential if the Sol is to be made a slave of another CPU on the bus, since this disable signal would not be activated from the bus without it.

Pin 98: Standard Signal ERROR*

Signals a bus error condition during the present bus cycle, such as a parity error detected by a memory device.

Compliance: Sol puts the 8080 status SSTACK onto this pin, but it is not needed and should be removed to avoid asserting ERROR*.

Upgrade: Cut the trace leading to pin 98 if the bus installs pull-up resistor as mentioned in section below on open-collector lines.

Pin 99: Standard Signal POC*

Power-on clear signal for all bus devices. When this signal goes low, as during a power failure, it must stay low for at least 10 ms.

Compliance: Sol generates the POC*. I'm not sure about the 10 ms recovery time, but it should be possible to calculate it from the data on the schematic.

Pin 100: Standard Signal 0 V (ground)

Ground.

Compliance: Sol complies.

OPEN COLLECTOR PULL-UP RESISTORS

The Sol generally uses 1500 ohm pull-up resistors, but the standard calls for 1000 ohm resistors on open collector bus lines. This probably will never make any problems. However, for full compliance, the following resistors should be changed to 1000 ohm 1/4 watt resistors (5% tolerance):

R41 (DSBB*)
R20 (DSDB*)
R36 (ADSBB*)
R33 (DDDBS*)
R34 (DDB*)
R31 (INT*)
R56 (HOLD*)
R55 (RESET*).

The open collector bus lines which need to have pull-ups added are:

- pins 4 thru 13 (the interrupt lines),
- pin 13 (PWRFAIL* acts as pseudo-open collector)
- pins 14, 55, 56, and 57 (the four TMA lines),
- pins 54 (SLAVE CLR*),
- pin 60 (EXIT*),
- pin 67 (PHANTOM*),
- pin 98 (ERROR*).

That is, connect a 1000 ohm resistor from these bus pins to a +5 volt source. Since the power to the bus is only the unregulated supply, the pull-up's will have to be connected to the +5 volt source for the Sol-FC itself. See the article on Bob Marsh's backplane board in this issue.

SUMMARY OF PART 1

The S-100 bus in the Sol requires changes to the following bus pins to bring them up to the IEEE standard 696 for a permanent bus master:

- 1 (nominal +8 volt line is overvoltage in some Solas)
- 4-12 (vector/interrupt lines need pull-ups)
- 13 (PWRFAIL* needs pullup)
- 14 (needs pullup)
- 20 (ground)
- 25 (PVTOL* should disable by CDSB*)
- 27 (abandon WAIT status, RFU)
- 28 (abandon PINT*, RFU)
- 49 (2 Mhz clock timing is out of tolerance)
- 53 (ground)
- 54 (assert SLAVE CLR*, add pullup)
- 55-56 (need pullups)
- 57 (relocate D1G*, add pullup)
- 58 (relocate FRDY*, disassert xTMQ*)
- 60 (needs pullup)
- 65 (move FRDY* to this NDEF line)
- 66 (move D1G* to this NDEF line)
- 67 (PHANTOM* fake open collector, add diode and pullup)
- 96 (SINTA jumpers)
- 97 (SNO* gates data bus drivers, REV P update)
- 98 (abandon SSTACK*, add pullup for ERROR*)
- 99 (maybe)

The following are unsupported bus pins (features) of the bus in the Sol:

- 4-11 (vector/interupts)
- 11 (non-maskable interrupt)
- 13 (power-failure)
- 15-17, 59, 61-64 (extended addressing)
- 58 (sixteen-bit data transfer request)
- 98 (bus error).

PREVIEW OF PART 2

Enslave the Sol

In the next part of this series, I will examine the changes necessary to let you disable the Sol's 8080 CPU and allow another CPU board to take over as permanent bus master. The rest of the Sol (video, I/O ports, RAM, ROM) would remain available to the new processor. I think this may be done by flipping a switch so that you can always go back to the 8080 processor to run old software.

If you have some time to look into this, please send me your ideas. Suggestion: look into disabling drivers that control the permanent bus master signals, such as pppppp at U77. One problem is that the dot clock uses a portion of U77, too. We'll have to swap the gates with some other chips, so the dot clock can remain for on-board use, while the bus signals are disabled. The 8080 will have to be put on HOLD so its internal address and data drivers go into the high-impedance state.
Some other problems need to be considered at the same time: 16-bit I/O port addresses, 24-bit memory addresses, and the 16-bit data bus transfers.

**Extended I/O Addressing**

The 8080 microprocessor puts I/O port addresses onto both the A0-A7 lines and the A8-A15 lines. The Sol uses this fact by decoding the A8-A15 lines when looking for an I/O port address. But the standard now defines A8-A15 as the high order side for the first 16-bit port address. A note in the standard warns against using the I/O address way Sol does, because it complicates expansion to extended I/O device addressing. How do we correct this? We'll look at the problem in the next part of the series.

**Extended Memory Addressing**

The Sol only considers the basic 16-bit address bus (A0-A15). The standard now allows 24-bit memory addresses. If Sol is to exist as a slave device in an extended memory system, Sol will have to decode the high order 8 bits of address (A16-A23) when deciding if it should respond to a memory cycle on the bus. In the next part of this series, we'll consider extending the Sol's bus to comply with the standard for address lines A16 thru A23.

**16-bit Bidirectional Data Bus**

In the next part, we'll also examine the problems with Sol's bidirectional tied data lines D100 thru D107. Even though the Sol is permitted to send only 8-bit transfers in a 16-bit system (by not asserting S8694), the fact that the ED and OD data lines are physically wired together will prevent any other device from sending 16-bit data transfers. We'll look into utilizing the two sides of the data bus.

End of Part 1 -- To be continued in next issue.

---

**WHAT'S NEW?**

The Z-80 upgrade piggy-back is in final preparations for production. Prototype PC boards are being checked for accuracy, and an announcement about price and availability in the next issue of Protes New.

Protes New is behind schedule this year, but we'll still deliver 4 issues each 12 months to maintain our status as a periodical. Send in your letters and articles, so I don't have to make those thin issues again, please.

CP/M 3.x (also known as CP/M Plus) supports bank switched memory, bigger files, up to 512 megabytes of disk, hashed directory lookup for faster DDOS operation, automatic disk logging when diskettes are changed (CC not necessary), etc. See article in Byte, July 1983.

Here they come: the LSI hard disk controllers, Western Digital, which made floppies economical with their WD1771 chip, has produced the WD1015 hard disk controller chip for 5-1/4" Winchester. The WD1014 is a companion error correcting code (ECC) chip. Soon we should see single board computers with both hard disk and floppy disk controllers on the CPU.

**NEW S-100 BACKPLANE FOR SOL**

**Extra Reliability and New Feature**

Bob Marsh has arranged to have some backplane boards made for exact replacement of your SOL's present backplane. The problem with the existing backplanes is that the TI S-100 connectors used for the 5 horizontal boards have weak finger contacts. As the S-100 boards plug into the bus warp with heat and age, some contacts fail to make good electrical connections with the edge connectors. This can result in intermittent problems. This is especially true in Sol's that have had the slave boards plugged in and removed many times.

Replacement of the S-100 connectors is virtually impossible since all 100 solder joints must be cleaned de-soldered or heated simultaneously without damaging the board. The only practical solution is to replace the board as well as the connectors.

Bob suggests that we use Sullins H-Rel (high-reliability) connectors, which are advertised for about $4.00 to $4.50 each depending upon quantity, in the California Digital ad in Byte. (Toll free order number: 800-421-5041.) The backplane needs the Inab type (with .250" spacing) for the 5 horizontal board connectors. The top extender connector can be unsoldered more easily than the horizontal ones, so you can probably salvage your old one. On your Sol they are very high quality connectors, making tight contact, so I'm sure I'll want to save my old one. If you need a new top connector, be sure to order the Altair type, with .140" spacing between the rows of pins. The closer spacing between rows allows easier soldering to the bus traces on the top of the board.

As long as we are making new boards, Bob and I discussed adding a new feature to the backplane. In the large empty space to the side of the bus sockets, Bob is adding pads for the pullup resistor packs we need for the open collector lines mentioned in the article on upgrading the Sol in this issue.

Price of the bare backplane board (you add the connectors and other components) will be $35, available from Protes New in 30 to 60 days after receipt of your order. For out of state, add $1.00 for postage. This price includes bare board, instructions, parts list, schematic, and shipping via UPS. If you are out of UPS (United Parcel Service) service area, please add sufficient money to cover up to one pound shipping weight via the carrier of your choice. California residents add sales tax, please. (Visa and MasterCard accepted.) Place your order through Protes New.

If you have intermittent errors which go away when your boards are removed and re-inserted or just moved a bit in the S-100 slot, your problem is probably a faulty S-100 connector. If your system makes errors when it is in one temperature condition (hot/cold) but the problem goes away at the other temperature, you may be experiencing thermal warping of the boards and better backplane sockets may be needed. If you are afraid that you may have this problem and won't be able to find replacement parts, this is your chance. It may be your only chance to buy an exact replacement for the Sol S-100 backplane. After this first batch is made, we may never be able to have an economic quantity of orders again. Don't miss the opportunity.
A couple of weeks ago, disaster struck. My normally faithful Sol showed a Mr. Hyde side, and scribbled garbage over a disk directory. What the...?? But no problem -- I'd just boot one of my backup disks (of COURSE I had a backup). But then it happened again -- another disk crashed! Now I had a real problem. I still hadn't lost any data, but I had no more expendable disks, and I didn't dare boot a disk which I couldn't afford to lose. On most machines, I would just punch out the write-protect hole and not worry; no matter what the stupid CPU did, the disk would be safe. But, unfortunately, the Helios has no write-protect, right? Wrong! It's in there; you just need a way to enable it. So now my Helios has two ugly little switches on the front, and my problem is solved (well, one problem, anyhow; I still haven't figured out why my Sol went berserk, it seems to be fine now).

The Helios write-protect connectors are the two unused connectors along the top of the Persei main PCB, labeled P19 and P20. P19 is for the right drive (drive 1), and P20 is for drive 0. Each connector has 5 pins: 1, 2, and 4 (pin 3 is missing). Pin 2 is ground. If pin 4 is shorted to ground (pin 2), the write and erase circuitry are disabled for that drive. Just run a pair of wires to a front panel DPST switch for each drive; that's all there is to it.

Dave Burton
2317 University Dr.
Durham, NC 27707
Tel: (919) 489-5002

A 7K RAM SOL
by Dave Burton

This modification adds a simple bank-select memory mapping scheme to the Sol which lets a program switch between the normal Sol configuration, with up to 64K of RAM (plus display memory and Solos), and a 64K RAM mode, with locations CO00 hex to FFFF mapped onto a different RAM board.

This allows the Sol to use:
1) The normal 64K RAM below CO00 hex, addressed in either "mapped" or "unmapped" (normal) mode.
2) Solos, the display RAM, the 1K internal RAM, and up to 128K of external RAM, addressed from CO00 to FFFF (or CO00 to EFFFF if Solos has been relocated), all accessed in "unmapped" mode.
3) Up to 16K of additional RAM, addressed from CO00 to FFFF and accessed in "mapped" mode.
For a total of 76K of external RAM plus 1K internal RAM.

The extra memory can be used for I/O buffers or keep parts of a BIOS or UCOM Pascal F-code interpreter "hidden", providing more useful program space.

The modification adds a 1-bit output port, PC, to select between the two modes. The PHANTOM line and IEEE extended address bit A16 are used to select between the two RAM boards which are addressed from CO000 to FFFF. When a "1" is output to port PC, all memory references above F6FF cause A16 to go low (active), and the PHANTOM line to go high (inactive). I call this "mapped" mode. But if output port PC contains a zero (the power-up state), all references to Sol's internal memory (RAM or ROM) cause both PHANTOM* and A16 to go low, and references to any other address from CO000 to FFFF cause only PHANTOM* to go low.

The "unmapped" 12K memory is intended to be a board which supports the new EKE extended (24-bit) addressing (or INEAL-style A16-disable). It should be switchable to reside at address 000000 hex and ignore PHANTOM*. It will still be disabled for "mapped" mode references to the 8080's high 12K of address space, and enabled in unmapped mode. A Processor Tech 16KRA is fine for this if it is altered to take PHANTOM* from A16, though only 8K will be usable to Sol's internal RAM and ROM (take up 4K, usually from CO000 to CFFFF).

Similarly, the "mapped" mode-high 16K memory (from CO000 to FFFF) will be addressed with A16 low and PHANTOM* inactive (high). It should ignore A16, and be deselected by PHANTOM* low (active).

The other 48K (from CO000 to EFFFF) can be addressed either way, since both A16 and PHANTOM* are high during accesses to the low 48K. It must still be strapped to deselect when PHANTOM* (or A16, if you prefer) goes low, since the Sol will still do its famous four-cycles re-map of Solos on power-up. During the power-up "re-map", both PHANTOM* and A16 go low, so the low-48K board can use either one.

Details

Step 1:
You'll need two chips, a 74LS276 dual flip-flop, and a 74951 dual Schottky And-Or-Inverter. They can be mounted on a small perf-board, piggybacked to two other 16-pin Sol chips (with pins 7 and 14 soldered to the chip beneath and the other 12 pins bent out to the sides), or mounted "dead bug" style, somewhere near the rear of the Sol's main PCB. In my Sol, I piggybacked the 74LS276 and the '951 to U54. If you mount these this way, you should first solder a 4" piece of wire (I like wire-wrap wire) to pin 8 of U53, since you will need to make a connection there later.

Step 2:
Disconnect the following IC pins on the Sol main PCB by removing each IC, bending the specified pins out to the side, and replacing the IC in its socket:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>U76</td>
<td>5 (4-cycle power-up remap of Solos)</td>
</tr>
<tr>
<td>U77</td>
<td>7 (PHANTOM* driver's input)</td>
</tr>
<tr>
<td>U82</td>
<td>8 (old simult-de-select buffer)</td>
</tr>
<tr>
<td>U83</td>
<td>9 (OUT-PC line, conveniently already decoded)</td>
</tr>
<tr>
<td>U24</td>
<td>1 (old PHANTOM*/power-up remap Solos)</td>
</tr>
</tbody>
</table>

You will be making connections to the disconnected pins later. All OTHER connections to Sol IC pins must be made WITHOUT disconnecting them.

Step 3:
A) Connect a wire from U76 pin 7 to U24 pin 1. This restores the connection needed to remap Solos down to 0000 for the first four memory fetches after power-up or reset.
B) Connect a wire from U82 pin 8 to U22 pin 6. This connection will be used to disable Sol's internal memory in "mapped" mode.
Step 4:
Connect together S-100 pins 59, 61, 62, 63, 64, 15, 17, and 70. The first 7 pins are RS232 extended address lines A7-A3, and pin 70 is ground. We need this change to use 24-address-bit memory in the Sol.

Step 5 (the big one):
Wire up the circuit shown in the schematic; you must...
A) Connect pins 4 and 14 of the '74574 and pins 3, 13, and 14 of the '7451 to +5V, connect pin 7 of both chips to ground, and connect a ceramic bypass capacitor (0.002 to 0.1 microfarads) between pins 7 and 14 of the '7451.
B) Connect a wire from U53 pin 8 to pins 1 and 2 of the '7451.
C) Connect a wire from U54 pin 6 to pins 9 and 4 of the '7451.
D) Connect a wire from pin 6 of the '74574 to pin 10 of the '7451.
E) Connect a wire from U52 pin 15 to pin 1 of the '74574.
F) Connect a wire from U55 pin 11 to pin 3 of the '74574.
G) Connect a wire from U53 pin 14 to pin 2 of the '74574.
H) Connect a wire from pin 9 of the '7451 to U77 pin 10.
I) Connect a wire from pin 6 of the '7451 to S-100 bus pin 16.
J) Connect a wire from pin 5 of the '74574 to pin 5 of the '7451, and from there to U58 pin 9.

Step 6:
Make a few simple checks to reduce the likelihood that you have made a wiring error. Verify that...
A) pins 8-13 of the new '74LS74 and pins 11 and 12 of the '7451 are unconnected. All other pins on these two ICs should be connected to something.
B) All 6 of the Sol IC pins that you disconnected in Step 2 have wired connected to them.
C) The following Sol IC pins are still firmly seated in their sockets despite having wires soldered to them: U22 pin 6, U34 pin 6, U52 pin 15, U53 pin 8, and U59 pin 14.
D) Pin 7 of the '74LS74 and pin 7 of the '7451 are both connected to S-100 pin number 59 (use an ohmmeter).
E) U77 pin 9 is connected to S-100 pin 67 (use your ohmmeter). This verifies that the Sol's "PHEN" jumper is in place.
F) You can reassemble your Sol and fire it up. It should work normally.

You can now put up to 76K of S-100 RAM in your Sol (or 80K, with 4K usable). Some possible configurations are:
A) 64K extended-address RAM from 0100000 to 01FFFFF, plus an old 16K RAM board from 0000 to FFFF (with PHANTOM disabled)
B) 64K of old-style RAM from 0000 to FFFF (with PHANTOM disabled), plus 16K of extended-address RAM from 0100000 to 01FFFFF.
C) 64K of old-style RAM modified to take its PHANTOM disabled signal from A16, plus a 16K board (with PHANTOM disabled) addressed from 0000 to FFFF.
D) like B, but with an old 5K board (addressed from 0000 to EFFF or from 0000 to FFFF and modified to take PHANTOM from A16) replacing the 16K board. This, of course, would provide "only" 72K of S-100 RAM (64K in "mapped" mode; 56K plus Solos & a 4K gap in "unmapped" mode).

Special Notes
1) Many S-100 RAM boards allow "write-through PHANTOM"; that is, rather than disabling the board when PHANTOM is active, they only disable the output buffers. The R.E. Sales ExpandOram is such a board. To use such a board in the upper 16K, you'll have to modify it to behave the way Procesor Technology boards do: disabling both writes and reads when PHANTOM is active. On the ExpandOram I, this can be done by bridging pin 6 of U7 (a '74LS138) out to the side and connecting a wire from it to the PHANTOM jumper (B2 and B3). (You'll also need to modify the ExpandOram I as described by B. A. Howe on p. 15 of Vol 5, #1, Procesor/New.)
2) If you want to replace the '7451 with a '74LS51, you may, but you'll need to connect pins 11 and 12 to pin 10. I recommend the Schottky part, though, since an LS output is not really adequate for driving a bus line.
3) If you use (or may someday use) interrupts in your Sol, add a pullup to S-100 line A15 by connecting a 1K resistor between pins 10 and 14 of U22. This will prevent spurious responses from Sol's internal memory during 8080 INT4 cycles.
4) If you have relocated Solos to 7000 by connecting pins 2 and 12 to pins 5 and 9 on U22, the memory mapper will still work fine, and Solos will still be at 7000 — ONLYBBS you are also using interrupts in your system! In that case, you should have AB/AD and AB/AC strapped, and you'll have to use something other than pins 9 and 10 of U58 to deselect Sol's internal memory when in "mapped" mode (since that section of U58 is in use). I suggest pin 5 and 6 of U58; this will however, disable bit 1 of the Sol sense switch (which nobody seems to use, anyhow).

Software Notes
1) The program which changes the mapping flip-flop must reside in the lower 64K of course, and should care not to zap its own stack if SP points into the high 16K.
2) Although the flip-flop cannot be directly read, it can be easily tested by a program which tries to write to Solos's ROM. E.G.:
   returns:
   A=FF and Z false if mapped onto RAM
   A=00 and Z set if unmapped (with Solos at C000)
   A=FF and Z set if mapped but no memory at C000
   get old value
   try to change memory
   id id change?
   skip - return Z set (and A=0 if Solos at
   7000, since fat byte of Solos is in a NDR
   restore the old value
   set A=FF (and Z still false)

This would have to be done, for instance, by an interrupt handler which kept a FIFO buffer hidden in high memory, so that it could restore the mapping flip-flop before returning from the interrupt.

See next page for diagram.
CONSOLE I/O ROUTINES FOR MVIVIDEO, INCLUDING SCREEN PRINT
by Charles H. Stembridge

This code contains keyboard input, console output, VIDEO hardcopy and SCREEN scroll speed/stop routines. This code assumes that the Bob Hogg 24 x 80 display has been installed and the F000 page is in use.

Hardcopy of the contents of the CRT may be obtained any time the screen is stable by striking the PRINT key.

This PRINT key must be defined by the user at assembly.

An unused key such as the tilde (~) or other may be used or if the number pad adapter kit has been installed a key can be dedicated to this purpose.

SCROLL speed control may be changed at any time during screen output by striking the numerals 0 through 9 (0 fastest and 9 slowest).

The DISPLAY may be halted at any time by striking the SPACE bar awell as CONTROL-S. Display scrolling may be resumed by striking any key.

The LOAD key causes a JUMP to SOLOS at F004H.

The MODE SELECT key will cause a CP/M system WARM BOOT.

The CLEAR key will cause a SCREEN CLEAR.

*****************************************************************************

* CONSOLE INPUT (This is the CPM CONIN routine)

CINP IN STATP ;Get status
ANT ISDDY ;Loop for
JNZ CINP ;Character
IN DPORT ;Get character
CPI 80H ;MODE SELECT?
JNZ XCT ;Send CTRL-C to CP/M
MV1 A,03 ;Sets CTRL-C to CP/M
NXT CPI 8BH ;CLEAR?
JZ CLR ;Clear
CPI 8CH ;LOAD?
JZ SOLOS ;Exits to SOLOS at F Page
CPI 8BH ;PRINT key? should be changed
POP R ;for your designated key
MV1 A,ODH ;PRINT is screen copy key
JZ RET ;Regets PROMPT

*****************************************************************************

* CONSOLE CONTROL ROUTINE - This routine controls the CONSOLE OUTPUT as well as pro-
viding single key SCREEN CLEAR, WARM BOOT
* and SOLOS JUMP capability. The routines
* were taken from PROTEUS, Vol. 3, No. 5/6
* page 25, 26.
* *****************************************************************************

CCTRL CPI 13H ;CTRL-S?
JZ PAUSE ;The SPACE BAR may also
CP1 20H ;be used to temporarily
JNZ SCROLL ;HALT console output;
PAUSE CALL CONIN ;Any key resumes output.
RET SCROLL CPI 3AH ;This routine regulates
RNC ;dynamically the SPEED
NC CPI 30H ;of SCROLLING.
RAL RC ANI 0FH
NOP RAL
STA RET

*****************************************************************************

* VIDEO hardcopy routine suggested by an article in KILOBAUD
* MICROCOMPUTING for October, 1980, p. 158., but adapted for
* the special case of 80 character line length using memory
* mapped video display.
******************************************************************************

HRDCPY PUSH B ;Save registers
PUSH H ;Get cursor position
CALL VBADD
PUSH H ;Save address of cursor
LXI D,800H
;800H-0-F80H..Result in HL is
LXI DX,OFFBH ;# of bytes past F800 that cur-
SUBB D ;sor is..FB800-0-50H. Do mod
SUBB D ;50H math to determine how many
LXI D,50H ;characters past line start
cursor is.
Enclosed is a check in payment for a list of SOL users in my area, as well as an assembler listing of a NorthStar CP/M User Area, which may help those trying to run Supercalc, dBASE II, and others.

In the listing, provision is made for several functions not found in the usual CP/M BIOS: GOXY (position the cursor on a VDU), CUR (clear screen and home cursor), RCM and ROF (toggle reverse video on/off), and LFP and RGT (cursor left/right 1 position). Both dBASEs and Supercalc can do nicely without a Clear-To-End-Of-Line, so they provide such functions if you can't.

The bulk of the listing is the delivered substance of the Lifeboat/Amoco CP/M user area, but it has been modified. A smaller but substantial part is the subroutine VTERM, including the functions listed above. By changing 'equates', the whole thing is tailored to fit your CP/M and version of SOLAS. It all fits into NorthStar double density sector 19 (where it has to go).

To use this, you initialize/install Supercalc, etc., send an escape sequence to "the terminal" (which is your own SOL). For example, to place the cursor on row 11, column 11, Supercalc must 'send' this:

ESC 002 30H 30H
t( 1BH cll-g 04+20H 04+20H )

The 20H is required, since row and column numbers less than 32-decimal look like control characters, and CP/M traps some for its own use. Adding a 'bias' or 'offset', then removing it in VTERM, allows us to fool CP/M.

Do remember that row and column number from OOHex, i.e. decimal row 10 is hex row 99.

I value my SOL highly. Other machines are newer, faster, produce nice graphics, and so forth. But I have complete control, and access to every function in the machine. Sometimes not so easy with many of the "user-friendly" things on the market. Perhaps one day I'll add another machine to wish list -- but you bet that SOL (with all the nice MCCxx upgrades) will be my workhorse for a long time yet.

Sincerely,  

WILLIAM D. LOUGHMAN

GENEHESS
303GRAYATT DRIVE  BERMELF, CALIFORNIA 94705
**0000** EQU 18H 
**0000** #
**0000** x byt2
**0000** 602 EQU 8H (EESC) (602) (byte3) (byte4) = go to X/Y
**0000** RUN EQU 'N' (EESC) (RUN) = vga rev on
**0000** ROP EQU 'F' (EESC) (ROP) = vga rev off
**0000** CAM EQU 'C' (EESC) (CAM) = vga clear/home
**0000** LFT EQU 'L' (EESC) (LFT) = crsr left 1
**0000** RGT EQU 'R' (EESC) (RGT) = crsr right 1
**0000** x byt3,4 = [col] [XHIDE], [row] [XHIDE]
**0000** XHIDE EQU 20H [MUST] = 20H or more...
**0000** XHIDE EQU XHIDE ...to fool CP/M for VTERM
**0000** #
**0000** #
**0000** #***************
**0000** # SOLOS equi pick 1 swt
**0000** #
**0000** #
**0000** # hi MCS (WDL version)
**0000** # MM EQU 00000H SOLOS org
**0000** # MR EQU MM+0F00H its ram buffers
**0000** # DCCHAR EQU MM+800H byte in B to vga at X/Y
**0000** # VRAM EQU MM+8000H vga org
**0000** # XMAX EQU 90 vga chars wide
**0000** # XMAX EQU 24 vga lines high
**0000** #
**0000** #***************
**0000** # lo MCS (WDL version)
**0000** # MM EQU 00000H
**0000** #
**0000** #**************
**0000** # all PICT
**0000** # MM EQU 00000H lo ; 0F000H hi
**0000** # MR EQU MM+800H
**0000** # DCCHAR EQU MM+800H
**0000** # VRAM EQU MM+8000H
**0000** # XMAX EQU 64
**0000** # XMAX EQU 16
**0000** #
**0000** #**************
**0000** # all SOLOS (versions known to me)
**0000** # AGOT EQU MM+16H psaport out strips per
**0000** # ASIF EQU MM+20H psaport in keeps per
**0000** # START EQU MM+40H gen1 status port
**0000** # USPF EQU MR user psaport adrs
**0000** # UOPF EQU U16P2
**0000** # NCCHAR EQU MM+00H cur col=x
**0000** # LINES EQU MM+04H cur row=y
**0000** # Bote EQU LINE+1 tit dispwnt
**0000** # VBAS EQU MMX+4 for SOL-20 scrolling
**0000** #
**0000** #
**0000** # gen1 wqu
**0000** #
**0000** # PSW EQU "1" ramov if using CP/M ASM or MAC
**0000** #
**0000** # vrities --> NO CHNG
**0000** # FLS EQU 0 integer zero (XRA A)
**0000** # TUE EQU 1 boolean true (XRA A)
**0000** # CLE EQU 40H ctrl ky neg bias
**0000** # SB EQU 100H 8-bit shift
**0000** # RET EQU 0CHR 8000 cod
**0000** #
**0000** # SOLOS speical keys and codes
**0000** # BPS EQU 'H'-CTL non-district bkspc (input)
**0000** # DEL EQU 'F'-CTL non-district bkspc (input)
**0000** # BAK EQU 1FH district bkspc (outpt)
**0000** # CLR EQU 0BH clear vma and reset
**0000** # Hom EQU 00H home crsr
**0000** # LARCM EQU 01H crsr left
**0000** # RADCM EQU 13H crsr right
**0000** # MODE EQU 04H abr, etc
**0000** #
**0000** # SOLOS crsr positioning (outpt)
**0000** # YCGD EQU 1 (EESC) [YCGD] (col)
**0000** # YCGD EQU 2 (EESC) [YCGD] (row)
**0000** #
**0000** # psaports
**0000** # VDM EQU 0 vdm psaport (XRA A)
**0000** # KYB EQU 0 kyb crsr positioning (XRA A)
**0000** #
**0000** # figs
**0000** # B1 EQU NUL by1 fig
**0000** # B2 EQU B1+1 by1 fig
**0000** # B3 EQU B2+1 by1 fig
**0000** # B4 EQU B3+1 by1 fig
**0000** # NOKY EQU MODE n psaport inpt
**0000** #
**0000** # masks
**0000** # LD2 EQU 3 keep 2 low bits
**0000** # FAM EQU 7FH kill parity bit
**0000** # CSS EQU 00H crsr on/off
**0000** #
**0000** # CP/M
**0000** # IOBYT, EQU 3
**0000** #
**0000** #
**0000** # DGG USER
**0000** #
**0000** # CB16S jmp vector; data
**0000** # JMB EQU 4 (keep 11 lines in order)
**0000** # JMP CINIT cold init trnl
**0000** # JMP WINIT warm in it trnl
**0000** # JMP CSTAT CON1 status
**0000** # JMP CONOUT CON1 out
**0000** # JMP CONOUT CON1 out
**0000** # JMP LISTS LST1; out drv
**0000** # JMP PUNCH PUN; out drv
**0000** # JMP READR RDR; in drv
**0000** # JMP PSTAT printer status
**0000** #
**0000** # DW LNSWTH len this code
**0000** #
**0000** # define IOBYT; list punch reader console
**0000** # IOCDD DB 14H 10 01 01 00
**0000** #
**0000** # DSW DB 0 rsvd
**0000** # DSW 0
**0000** #
**0000** # cold init
**0000** # CINIT MVI A, NUL clrr status
**0000** # OUT START
**0000** # LD 1000 set dfil psaports for CP/M
**0000** # STA IOBYT
**0000** # LXI H,CUSTI set SOLDS custe outpt
**0000** # SHLD U16P
**0000** # LXI H,CUSTO
**0000** # SHLD UOPF
**0000** # LXI H,NOKY+B+SHOKY zap byt inpt bufs
**0000** # SHLD PRVHN
**0000** # SHLD PRVHN+2
**0000** #
**0000** # warm init
**0000** # WINIT CALL CLEAR clrr vma
**0000** # CALL VADER locate crsr
**0000** # RET . = CP/M
0000 1
0000 4: ck printer status
0000 PSTAT MVU A,FLS we're always ready
0000 DRA A =,NC
0000 RET . -> CP/M
0000 1
0000 4: ck psuedoport inp status
0000 4: in software, cause SOLDS also reads the by
0000 CSTAT LDA I0BYT CON: status (hybrid)
0000 ANI LO2
0000 1
0000 4: ISTAT CALL STORE sv regs
0000 1
0000 4: MOV C,A ecrire correct byt buf
0000 MVI R,NC
0000 LVI H,PRVIN
0000 DAD B according to psuedoport used
0000 SHLD PNTHI pont to it
0000 1
0000 4: MOV A,R byt inp prvly by AINP?
0000 CPI NOKY
0000 JNZ CSKEY y
0000 1
0000 4: MOV A,C psuedoport fr I0BYT
0000 CALL AINP must read byt to gt status
0000 MOV M,A so sv any byt
0000 JNZ CSKEY for not tym round
0000 1
0000 4: MVI M,NOKY if no inp (=NUL or =MODE)
0000 MVI A,FLS tell CP/M
0000 JMP EXIT1
0000 1
0000 4: CSKEY MVI A,TRI or say byt 'ready'
0000 JMP EXIT1 (actually already read)
0000 1
0000 4: input 1 byt
0000 1
0000 4: CUSTI DB RET SOL csta inppt send/yn
0000 IN 0
0000 1
0000 4: CONIN LDA I0BYT CON: fast inppt
0000 JMP INPUT
0000 1
0000 4: READR LDA I0BYT RD: slow inppt
0000 RAR
0000 RAR
0000 1
0000 4: INPUT ANI LO2
0000 STA ++4
0000 INXX CALL ISTAT await ok
0000 MVI A,XXX
0000 JZ INFXX
0000 1
0000 4: PUSH H
0000 PNTHI EQU ++1 point to PRVIN byt buf
0000 LXI H,XXX gt byt svd by ISTAT
0000 MOV A,K
0000 ANI PAR strip hi bit
0000 MVI M,NOKY tell ISTAT ok
0000 POP H
0000 RET . -> CP/M
0000 1
0000 4: STORE SHLD SAVHL sv regs
0000 MOV H,B
0000 MOV L,C
0000 SHLD SAVHL
0000 MOV B,C SOLDS sends frn B
0000 RET
0000 RET
0000 #
0000 # keep track of byt in vdm
0000 VA02D CALL VBADO crsr adr
0000 SHLH CADDR
0000 RET
0000 #
0000 # remov crsr
0000 NZCRS LSHL HADD crsr adr
0000 LDV A,M
0000 ANI PAR kill crsr
0000 LDV H,A
0000 RET
0000 #
0000 # locate crsr in vdm
0000 VBADO PUSH PSW
0000 PUSH B
0000 PUSH D
0000 #
0000 LDA NCHAR =Y
0000 LDV C,A
0000 LDA LINE =Y
0000 MDV L, A
0000 LDA B OT =Ybias for scroll
0000 ADD L
0000 #
0000 V80 CPI YMAX
0000 JC V91
0000 #
0000 SUI YM.
0000 JMP V90
0000 #
0000 V61 LXI H,YRAM-XMAX
0000 LXI D,XMAX
0000 #
0000 V63 DAD D
0000 DCR A
0000 JP V95
0000 #
0000 MDV E,C
0000 DAD D
0000 #
0000 POP D
0000 POP B
0000 POP PSW
0000 RET . ML= crsr adr in vdm
0000 #
0000 # clear vdm; home crsr
0000 CLEAR PDFW V1 B, CLR
0000 XRA A =vdm
0000 CALL ACUL
0000 MV1 B,HDM
0000 JMP VOUT
0000 #
0000 # position the crsr at col=X row=Y
0000 G02XY PUSC PSW sw XY cpl
0000 CALL XN02S kill crsr
0000 POP PSW
0000 #
0000 # CPI YCOD which X or V7
0000 JZ SDY want Y
0000 #
0000 G02X INR A set for row nut time
0000 STA XYFLS
0000 MV1 A,B4 inform G02XY
0000 STA ESFC
0000 #
0000 MDV A,C set col
0000 SUI XHIDE
0000 STA NCHAR
compile and link really large programs (like the compiler itself), and you must usually use the compiler in "moming" mode (which slows it down considerably).

The "mapped" version runs only on a SOL which has been modified according to my instructions in "A 765 Ram Sol" (also in this issue of Proteus News). It "hides" the BIOS, SOLOS, and the screen memory from the address space of UCSD Pascal programs, leaving much more usable memory; more, in fact, than you are apt to see on any other 8-bit computer running UCSD Pascal.

By the time you read this, I should have received final approval from SofTech. The price will be around $500 including manual, bootable disk, source code(!) for the
F-code interpreter & BIOS, and SofTech's royalty note (note that the package will NOT be supported by SofTech, however). Source code for most of the rest of the package will also be available, for an additional $75 per diskette (probably 4 diskettes to get everything).

David A. Burton
2317 University Drive
Durham, NC 27707
(919) 489-5002

...ON HELP NEEDED TO INTERFACE REAL TIME CLOCK

Dear Sir,

I notice from reading back issues of the SOLUS News recently that you were the editor of this newsletter, and wondered if you could be of any assistance with a few inquiries I have regarding the SOL computer.

First of all, does the SOL User's Group still exist over there, and if so, is it possible for me to join and/or purchase tapes from the software library? I am particularly interested in the tape version of ALSB, as it is impossible to purchase one on the machine due to many attempts.

Secondly, I have been trying to interface National Semiconductor's 58174A Real Time Clock to the SOL and have been having trouble with it. I believe timing is the main problem and wait states need to be generated, but so far all attempts to do this have failed. Has anybody that you know of managed to design an interface for this chip to the SOL, and would they be willing to let me have the circuit involved, as I am most anxious to get the clock working. I have written the software to display the time and date on listings and did have an MM58174 that worked for awhile until the ambient temperature rose, so I was able to get the software working, and would be willing to swap it for the appropriate hardware. I have the clock wired as part of memory at the moment, and would prefer to use it this way rather than have to use input/output ports.

I hope you can be of assistance in these matters as there is no SOL User's Group over here, and most people haven't even heard of the SOL.

Thanking you in anticipation,

Yours faithfully,

Bruce D. Bull

Box 127,
Woodville, 50115
Iowa
26th March 1983
Dear Stan,

Keep up the excellent work. I've been trying to make the effort to write sometime to keep information going to those people that just don't know the difference between their A/B's and C/D's but here goes for me.

Just home from the computer fair. The day of the deal for everyone's overccomputer was the CompuPro RAM-17. The only thing to do before plugging it in is remove Unit B (the 25LS225(2)) use it in your next project -- very handy. Next, set the switches to enable 32Ki of RAM: NORM vs. GLOBAL MODE. Then remove U3 (one of the 74LS138's), chi enable decoder for the uppermost 16K of memory. Send up (or cut) pins 14 and 15 so that when the I.C. is reinserted, they will not go into the socket. By doing this, the two chi enable lines (CE, CE CC) won't ever go ACTIVE LOW (0v) because of the pullup resistor R52, so your Sol's memory can think for itself without any interference. This also saves you two 2K x 8 CMOS static RAMs for spare or test use.

I would like to hear from someone (in Calif. or the Bay Area) that has constructed the conversion to his Sol (listed in Volume I, No. 5 by Ron Parsons), replacing the Tarbell single-density disk-controller board interface for running CP/M. I would like to know if both can be run at once; i.e., reading a CP/M file to memory under CP/M control, then switching to TDDOS control and writing it from memory to a TDDOS disk, using only a HELIOS II! Any information regarding a BIOS that software supports the fast seek mode of the PerSeil drive would save the day.

In closing, thank you for the opportunity to share.

Sincerely,

Jim Tittle

4/15/83

178 Thomas Ave
Pittsburgh, PA 15205
(412) 789-0790

[Ed. reply to Jim Tittle: I think Jack Kinney at UCLA Computer Science Dept, 3413 Boelter Hall, Los Angeles, CA 90024, may be able to give you some advice on the Tarbell multi-plexed controller modification.]

---

Dear Helios Owners,

There is a CP/M version 2.2 available for the HELIOS disk system. There are several companies in the Washington D.C. area who also have it, and have been using it for many months. I am planning a person who grew up in the world of Processor Technology could not take the unbelievably slow speed of Lifeboat's CFM for Helios. Therefore, I began the job of creating one. It works, and has been working for many months. I feel that it is as fast as it can be, and as reliable as I can make it.

How fast is it? Let's use the disk to disk copy program as an example. This program will format the output diskette in under 70 seconds, perform the disk to disk copy in about 35 seconds, and verify that the copy worked by performing a disk to disk compare in 45 seconds. The total time for the complete format, copy and verify takes less than 2 and 1/2 minutes. I am not sure of the numbers but I think Lifeboat's format takes about 6 minutes by itself.

Yes, I do have a version of CP/M 2.2 for Helios but it is not in a state that I am ready to market. There are no user's notes written, and a few enhancements that have been added on the back burner for quite some time that should be added. Items like the ability to read lifeboat format floppy's using this CP/M, and a user oriented interactive configuration program to setup printers, and certain flags like number of retries on disk operation, and whether or not to check for write errors would all be added.

So why did I send this letter? I would be willing to make all of the necessary changes and whip this software into a reliable state if there is enough interest in it. I estimate that the cost per copy if less than 50 people want it will be approximately $25.00. I will have to pay list price for the CP/M, and manuals from Digital Research. This cost will drop to about $17.50 if there are more than 50 people interested.

There is a large amount of software that works very well with under CP/M, especially Microsoft Basic, and Fortran compilers work with a bit of setup WordStar works very nicely on a Sol. I prefer to use the Sol for editing because of the speed with which WordStar updates the screen.

If you are interested please send me a letter of interest as soon as possible (DO NOT SEND MONEY) and I will figure up an exact price based on the number of responses. If there is very little interest, I might just offer it at a loss, inform all interested parties, either through the newsletter or by mail, of my intentions and the exact price as soon as I can.

Sincerely,

Lewis Bean

Computer Problems Resolutions
1924 Crystal Rose Drive, #23
Germantown, Maryland 20874

---

Dear Helios Owners,

There is a CP/M version 2.2 available for the HELIOS disk system. There are several companies in the Washington D.C. area who also have it, and have been using it for many months. I am planning a person who grew up in the world of Processor Technology could not take the unbelievably slow speed of Lifeboat's CFM for Helios. Therefore, I began the job of creating one. It works, and has been working for many months. I feel that it is as fast as it can be, and as reliable as I can make it.

How fast is it? Let's use the disk to disk copy program as an example. This program will format the output diskette in under 70 seconds, perform the disk to disk copy in about 35 seconds, and verify that the copy worked by performing a disk to disk compare in 45 seconds. The total time for the complete format, copy and verify takes less than 2 and 1/2 minutes. I am not sure of the numbers but I think Lifeboat's format takes about 6 minutes by itself.

Yes, I do have a version of CP/M 2.2 for Helios but it is not in a state that I am ready to market. There are no user's notes written, and a few enhancements that have been added on the back burner for quite some time that should be added. Items like the ability to read lifeboat format floppy's using this CP/M, and a user oriented interactive configuration program to setup printers, and certain flags like number of retries on disk operation, and whether or not to check for write errors would all be added.

So why did I send this letter? I would be willing to make all of the necessary changes and whip this software into a reliable state if there is enough interest in it. I estimate that the cost per copy if less than 50 people want it will be approximately $25.00. I will have to pay list price for the CP/M, and manuals from Digital Research. This cost will drop to about $17.50 if there are more than 50 people interested.

There is a large amount of software that works very well with under CP/M, especially Microsoft Basic, and Fortran compilers work with a bit of setup WordStar works very nicely on a Sol. I prefer to use the Sol for editing because of the speed with which WordStar updates the screen.

If you are interested please send me a letter of interest as soon as possible (DO NOT SEND MONEY) and I will figure up an exact price based on the number of responses. If there is very little interest, I might just offer it at a loss, inform all interested parties, either through the newsletter or by mail, of my intentions and the exact price as soon as I can.

Sincerely,

Lewis Bean

Computer Problems Resolutions
1924 Crystal Rose Drive, #23
Germantown, Maryland 20874
...ON SCREEN PRINT ROUTINE, BIOS FOR MCVIDEO AND MCSVOLOS, CCS DISK CONTROLLER

Dear Stan,

Enclosed is my check for renewal of my subscription to PROTEUS. Add my thanks to those of other subscribers for the good work you are doing.

Also enclosed is the source code for a screen copy routine to be added to a CPM BIOS (or to PIDOS I suppose) which will allow hardcopy to be made on a printer of the contents of the display screen any time it is not scrolling. It will need to be entered into your own BIOS and reassembled using Digital Research's ASM.COM or any other assembler. I have used it with CPM 1.4 and 2.2 and with the VISTA disk controller for 5.25 inch disks and the California Computer Systems disk controller for 8 inch disks. I hope it will be of use to someone.

I have adapted CPM 2.2 to run on the SOL with McVideo and McSolos using the CCS 2422 disk controller driving 8 inch disks in double density. While it was not difficult it was time consuming. If anyone would like to save themselves this effort for their SOL I would be glad to help. Just write to me at the above address.

One additional note of interest. The CCS controller will not read double density tracks at the standard SOL speed of 2Mhz. Remember, the SOL has jumpers which allow changing the clock speed to as high as 2.8Mhz. I purchased an 8080A-1 and tried the higher speed. The SOL ran fine but there were some random screen content changes while editing files (not in the file content - just in the display). However, I found that the original (2Mhz) 8080A which came with my SOL ran at 2.08Hz just fine. As a result my SOL is now operating at the higher speed and reads the double density disks with no problems.

Sincerely,

Charles H. Stembridge

2427 Frances St.
La Crescenta, CA
91214

...A COMPLAINT ABOUT THE DUAL PERSONALITY MODULE

I recently purchased and installed the video upgrade and corresponding Dual Personality Module for my Sol-20. After some initial problems getting the pins to make proper contact, the video display has worked reliably and well. I am currently rewriting many of my programs to take advantage of the added display space.

I am not nearly as pleased with the new Dual Personality Module - it does not execute custom user output routines. This has forced me to consume much time rewriting interface routines. What I really want to know is can get my present program replaced with one that has the old pseudo-output port 3 custom user routine?

Sincerely,

Merle L. Bowen
5825 Keith Avenue
Oakland, CA 94618

[Ed. reply to Merle Bowen: Bob added some new features in the latest version of the DPM, such as emulation of a Televideo 950 terminal, so some more Solos features had to be deleted. Bob assumed that most users would be selecting I/O re-direction through the disk operating system rather than through the Solos ports. He will reprogram your module to the old version if you wish. Please contact Bob Hogg at Micro Complex, 25651 Minos St., Mission Viejo, CA 92691, (714) 770-2168.]
UNCLASSIFIED ADS

**WANTED**

Cassette Electric Pencil II program to run matrix printer from SOL-20 48K.

Contact with someone who has converted SOL/Electric Pencil data tape to IBM PC.

Walter Jessel, 1500 Bluebell, Boulder, CO 80302
(303) 442-5757

**FOR SALE**

North Star Controller DD, 2 SS DD Drives, 16k RAM, Sol Micro
All for $595.00
Argonaut Computer Center
1104 Buchanan Road, Antioch, CA 94509
(415) 778-2595

**FOR SALE**

SOL-20, Rev.E wt 32K Static Ram, North Star DOS wt Dual Bingle Density Discs, including all documentation. Excellent used and in excellent condition. It is now surplus to me and I will entertain any reasonable offer. Also have an unused Bunny International Co. BV-18A, 16V-2.5A+K-+-+ assembled power supply (never used) for #49.50. Contact J.D. Knight at 3405 Doral Drive, Waterloo, Iowa 50701 or Phone 319-233-6123.

**FOR SALE**

2 ea. SOL-20 (Rev. E) with SOLos module, 8K Dynabyte Memory, Dual North Star Single Density Floppies, X5 Dos 2.7$5, NS Basic 3.6, SolStar word processor, Games, CP/M Distribution Diskettes, Spare floppy drive, 14" Javelin monitor, 50 diskettes, complete manuals for all of the above. System runs like a top when warm, occasional startup problem when cold. $600.

Richard B Kern
314 W. Howe St.
Seattle, WA 98119
(206) 284-1558

**FOR SALE**

VDM-1 1-10G board - $50
Fivar-SOL hardware debugger - $30
complete back issues SOLUS News/PROTEUS - $20
original manuals and cassette for
BASICS
Extended BASIC (w/ CP/M patch)
GamePac 1
GamePac 2
EDIT
EED
FOCAL
ALD-8
Poly-Morph Basic, BGI patched for SOLOS
$5 each, everything for $25

Contact: Bob Stick
19 Mayfield Road
Holland, Saskatchewan
SOL 0B5 Canada
(306) 352-7184

SOL2D WITH 48K HITTACHI MONITOR, DISCUS II, BIN DISK.
CP/M, DISKATR, EC BASIC, BASIC V, VIRTUAL BASIC,omas,
BASIC-E, WSTHR, CHESS, TRK-80, TEC'S. SOME CP/M U.G. SOFTWARE.
ALL MANUALS, EXCEPT WSTHR, PROTEUS ISSUE 1 THRU CURRENT.
$100 F.O.R. BOX OXNARD.
DICK LINZER 1051 KERN ST OXNARD
CALIF 93033. (805) 943-6993.
TABLE OF CONTENTS

EDITORIAL: WHERE DO WE GO FROM HERE? by Stan Sokolow...........1
FUTURE SHOCK or ARE 16 ADDRESS BITS ENOUGH by Stan Sokolow...1
SMALLTALK-80: THE LANGUAGE FOR THE NEXT ERA? by Stan Sokolow.3
THE IEEE 696 STANDARD AND SOL — PART 1: COMPLIANCE AND
UPGRADING AS A BUS MASTER by Stan Sokolow....................4
WHAT'S NEW — EDITOR..................................................12
NEW S-100 BACKPLANE FOR SOL — Editor........................12
WRITE PROTECT FOR HELIOS by Dave Burton.....................13
A 77K-RAM SOL by Dave Burton..................................13
CONSOLE 1/0 ROUTINES FOR MVVIDEO, INCLUDING SCREEN PRINT by
Charles B. Stembridge.................................................15
CP/M 2.2 BIOS USER AREA ROUTINES FOR SOL/NORTHSTAR by
Bill Loughman..........................................................16
UCSD PASCAL FOR SOL/HELIOS by Dave Burton..................20
LETTERS TO THE EDITOR:
...ON HELP NEEDED TO INTERFACE REAL TIME CLOCK by B. D. Bul1.20
...ON COMPUGRAPH RAM 17 MODIFIED FOR SOL & REQUEST FOR HELP
WITH TARBEll/HELIOS DISK by Jim Tittle.........................21
...ON FINDING THE 6574 CHARACTER GENERATOR by Ron Burford...21
...ON A FAST IMPLEMENTATION OF CP/M 2.2 FOR THE HELIOS by
Lewis Bean............................................................21
...ON SCREEN PRINT ROUTINES, BIOS FOR MVVIDEO AND MVCOLO5
...CCS DISK CONTROLLER by Charles Stembridge....................22
...ON HELP NEEDED TO MOVE FTC BOARDS TO HIGHER ADDRESS by
John E. Breden.........................................................22
...ON A COMPLAINT ABOUT THE DUAL PERSONALITY MODULE by
Merle Bowen........................................................22
...ON MVVIDEO JITTER, IPSON MX80 DOWNTIME by J. Jackson....22
BARGAIN PRICE ON 1979 ISSUES OF PROTEUS NEWS..............23
UNCLASSIFIED ADS.....................................................23

PROTEUS/NEWS

A news journal for owners and users of Processor Technology
Corporation computer equipment. Published by Proteus, 1690
Woodside Road, Suite 219, Redwood City, California
94061-3483, USA, telephone (415) 368-2300.

Submit items for publication to Proteus News, Attn: Stan
Sokolow, 1690 Woodside Road, Suite 219, Redwood City,
California 94061-3483, USA. Please make submissions as
camera-ready as possible by using a fresh, black ribbon and
typing single-spaced.

Copyright (C) 1982 by Proteus. All rights reserved.
Permission is hereby granted to reproduce any computer programs
contained herein, provided that Proteus and the program's
author are given credit.

FROM:
PROTEUS
1690 WOODSIDE ROAD, SUITE 219
REDWOOD CITY, CALIFORNIA 94061-3483
USA

FIRST CLASS MAIL

James D. McElroy
2826 Crest Ave. North
Allentown, PA
18104
PROTEUS NEWS

AN INDEPENDENT NEWSLETTER FOR OWNERS AND USERS OF PROCESSOR TECHNOLOGY CORPORATION COMPUTERS
FORMERLY SOLUS NEWS

VOL. 6 #2
PUBLISHED BIMONTHLY BY PROTEUS, 1690 WOODSIDE ROAD, SUITE 213, REDWOOD CITY, CA 94061, USA
SINGLE ISSUE...$2.50 (US)
SINGLE ISSUE...$3.00 (FOREIGN)

1973

SOL AND THE IEEE 696 STANDARD

Part 2: Upgrading the Sol
Compliance of Memory and Peripherals

In this part, we examined the extent to which the Sol complies with the IEEE 696/S-100 bus standard. We saw that the standard was defined in such a way that most of the Sol's design complies and only a few signals had to be moved because of conflicts. Some of the conflicts only happen if you try to use fancy things, such as multiple DMA devices (temporary bus masters) all trying to take the bus concurrently.

In this part, we'll look at the question of extending the Sol to handle the features of the bus that it doesn't yet implement: extended memory addressing, DMA arbitration, and so on. We'll also look at the Helios disk system and Processor Technology memory boards. This first part really was too short for this part, so I may leave out some details. I'm counting on you to let me know and fill in the gaps.

EXTENDED MEMORY ADDRESSING

The IEEE standard now provides for 24 bits of address lines, whereas the Sol was designed with 16 bits because that is all that the 8080 microprocessor can address. With 16 bits, you can address 64K of memory. With 24 bits, you can address 256 times that much, amounting to about 16 megabytes. Can the Sol be modified to address more than 64K?

Sure. It is possible to add circuitry to let the programmer output an 8-bit byte to a certain I/O port, where it would be latched (stored) onto the extended 8 address lines. The Sol's address decoder could then be modified to compare the extended lines when deciding if the bus is addressing memory space in the Sol's 64K block (normally at 0000, optionally at 8000 with the DFM module we sell). This would let the Sol's memory respond only to addresses in one of the 256 possible 64K segments. The particular segment would best be the first one or the last one if only one were hardwired into the circuit, but it could also be designed to be switch selectable with 8 DIP switches or even latched by program control.

But then the question comes, what would you do with more memory? Dave Burton in a previous Proteus News issue has given me his school assignment of designing an extension to the Sol for 16K for RISC p-code interpreter in his implementation of USCL p-system version 1.4.

(continued on page 7, right column)

MEET ADA, THE NEW PROGRAMMING LANGUAGE

If you've been reading the computing journals over the past few years, you may have come across the ongoing story of the U.S. government's project to develop a standard language for programming large defense systems. Beginning in 1974, the Department of Defense (known as DoD to the in-group) has been working on requirements, specifications, and implementation of a computer language for large, real-time computer systems that control missiles, radar networks, and smart weapons of all types. This effort has culminated in the adoption of the Ada language standard on February 17, 1983, by the American National Standards Institute (ANSI).

There has been controversy surrounding Ada throughout the process. Some people claim that the language is too big, having too many features that are difficult to implement and validate. They worry that an obscure compiler bug will result in a program error that causes a weapon system to go wild and start World War III or cause some other loss of life and property. These critics prefer the clean simplicity of Pascal or Modula.

Other experts have been disappointed by the project's rejection of their favorite language feature, in essence saying that the language isn't big enough.

Nevertheless, in spite of the criticism, Ada is here to stay. The DoD has decreed from the highest levels that Ada shall be used on all mission-critical applications after January 1, 1984. There are now five defined mission-critical areas: intelligence systems, cryptologic systems related to national security, command and control of military forces, computer systems that are an integral part of weapon systems, and other applications critical to the direct fulfillment of military or intelligence missions.

This doesn't mean that all of the billions of dollars worth of COBOL programs will be scrapped. COBOL will probably still be used for the business of running this arm of government, such as doing the payroll, although that may even be interpreted as mission-critical one of these days. FORTRAN may still be the language of choice on scientific research projects under government contract since the program libraries in FORTRAN are immense.

But for the new high-tech weapons, communication networks, and such, the control programs will probably be written in Ada for the foreseeable future. (There will be exceptions made by the DoD for contractors who must use computers for which there is not yet an Ada implementation, but an extra bank of 16K for hiding the p-code interpreter in his implementation of USCL p-system version 1.4.)

(continued on page 4, left column)
From time to time we've received questions about the various configurations of the MCSDM Dual Personality Module. The MCSDM allows you to modify your Sol to permit up to 60K of contiguous RAM memory. The standard Sol has Solos ROM and the Sol's monitor, screen and scratchpad RAM addressed in the ROM beginning with C000. This only allows 48K of contiguous memory below it. The MCSDM allows you to switch the base address from C000 to F000, thus allowing the 60K space below F000 to be available to your system programs.

In addition, the MCSDM comes with a new version of Solos (I've dubbed "McSolas") that eliminates the tape commands and some other unused features and instead provides support for the 24x80 screen module. It also includes a boot command for booting your floppy disk.

The usual configuration to buy is Configuration A, which comes with two EPROMs programmed with McSolas, one for the C000 node to be compatible with screen software assuming a standard Sol and one for the F000 origin node to allow 60K. The MCSDM switches between these two EPROMs at the same time the address is changed from C000 to F000.

Here is a reprint of the catalog description. The full catalog was printed in Proteus News, Volume 5, Number 1.

Item M2: MCSDM -- Dual Personality Module. $95.00

Replaces Solos personality module and has two EPROMs (2716). Switch on back of module (accessible without opening Sol covers) selects which EPROM is active. When used with the MCVideo upgrade (Proteus item M1), the switch also selects Sol ROM, RAM, and video to be at C000 or F000 address, and sets video display font (16x64 versus 24x80). Programmed with various versions of Solos (see below).

The new McSolas is an alternate version of Solos which replaces the tape I/O commands with other command, Test Memory, Dump Assemble, Move Memory, etc. It also provides a select autoboostload on power-on/reset, or bootload on LOAD key. See description of McSolas in Proteus News, Volume 5, Number 1. (Costum versions available, request price quotation.)

Standard configurations:

Configuration A = 2716 EPROMs in both C000 and F000 socket, programmed with same version of McSolas except the ROM origin McSolas provides 16x64 video routines, while F000 McSolas provides 24x80 routines for use with the MCVideo board.

Select this configuration if you want to have identical Solos monitor features in the C and F nodes. To use tape operations, you should load them from disk or replace your old personality module temporarily. (Note: A Processor Tech personality module will still work after installation of the Dual Personality Module, but a minor change must be made to it first; explained in installation guide.)

Configuration B = Empty socket C000 and Board jumped to accept 2716 masked ROM in C000 socket; you move your original Solos ROM into this. Socket F000 has a programmed 2716 EPROM containing McSolas with video output routines for 24x80 screen.

Select this configuration if you want your Sol to function completely normally with the Dual Personality Module switched to C000 setting, including tape routines, but to talk to the 24x80 screen properly when Sol is relocated to F000.

Configuration C = unprogrammed 2716 EPROMs in both sockets. Select this configuration if you don't plan to use the 24x80 video upgrade, can program your own 2716's, and want to relocate Sol's address space to F000 for more contiguous RAM space or if you want to do something special in Solos.

When ordering, specify which configuration and which disk controller bootup routine to include in the McSolas:
1. Helios controller.
3. Terbell single density controller.
4. Versatile Disk Controller (Proteus item M5).
5. Mc floppy (NorthStar compatible).
Other controllers boot routines available on special order.

__________________________

INTERESTING ADVERTISING WE'VE RECEIVED

Here are things you may wish to inquire about. If not, at least you didn't have to sort through the junk mail.

Glen Buie, a Proteus member, has a repair shop for Sol and Helios equipment. Doing business as "ProTech of Texas", Glen can do all repairs, including PerSol drive repairs. Maximum charge for a drive is $200 and for a Sol console $225. He has a flat-rate table for the more common problems, so you won't be hit with a surprise and for unusual debugging problems his hourly rate is $40/hour. He says his turnaround time is about 6 working days, but recommends that you call him before shipping your machine so he can tell you when to ship for quickest turnaround. He is offering a Special 25% off Labor charges for a limited time. Contact Glen at ProTech of Texas, 3625 Sunset Lane, Arlington, TX 76011, (817) 275-1015.

Cleo is here! The flyer proclaims, CLEO is "Computer Listings of Employment Opportunities". It is free to the user, except for your phone call. Use your terminal and modem to access the CLEO listings of job openings. Completely confidential. Your job inquiries and applications are known only to the prospective employers you contact, the ad states. CLEO requires a 300 baud, full-duplex ASCII terminal. Access phone numbers are available in the following area codes: 415, 408, 213, 714, 619. For access assistance: (213) 618-1923.

FileDriver is an integrated collection of file management utilities for CP/M and similar operating systems. In menu mode or command mode, you can achieve file operations with a minimum of keystrokes. Utilities include: Archive, Attr., Compress, Copy, Csub, Compress, Default, Erase, Listf (lists file directory info in various formats), Print, Rename, Verify, Write. Features include: command line interface allowing multiple commands per line, can be run from submit file, input/output redirection, tape commands from text files, help facility to teach you command line construction, create your own commands, clear error messages without need for a manual, requires no modification to CP/M. Also has some modifications to CP/M, familiar syntax, reasonable price ($69). For more information, contact Dunbar-Ridge Corp., 102 Sterling Court, Syosset, NY 11771, phone (516) 496-4441.

Micro-Grip is an inexpensive friction feed for Epson MX-70, MX-80, and RX-80 printers which only come with forms tractors. This device easily installs easily, indefinitely, and still allows you to use the forms tractor. Now you can feed your output sheets of letterhead, use roll paper, etc. without upgrading to the more expensive versions of Epson printers that provide friction and tractor feeding. Micro-Grip II comes in at $39.95 plus $2.00 shipping, and is available directly from Micro-Grip, Ltd., 3164 Dumbarton Avenue, San Bernardino, CA 92404, phone (714) 964-6643.
COMPUTING!, the publisher of "POWER!", a CP/M utility package, has announced version 3.3 with many updates and new features, including password protection. The password facility actually makes the protected files disappear from the directory. Anyone wanting to change or even print them must now know the password to put them back into use. They can't spy on the files without the password. The password is not visible to prying eyes either, not even with SOT. The new POWER! costs $159, but an update for older versions is available for $25. Contact COMPUTING!, 2519 Greenwich, San Francisco, CA 94122, phone (415) 567-1934.

COMPUTING! also has announced a PC-DOS version of POWER!, known as MENU/POWER!. It is available for PC-DOS, MS-DOS, as well as CP/M-86. Address above.

LINK-LABEL is a unique, two-part, serially-numbered label which lets you match your diskette to its jacket. The label can contain descriptions of the contents of the diskette's data. The label system makes it easy to match the two, quickly and accurately, so you are sure each diskette is stored in the appropriate jacket. Contact HEXCO, Inc., P.O. Box 199-PL, Huntley, IL 60142, phone (312) 780-2048. A descriptive price-list/order form is available for the asking. The illustrated catalog of the full product line of specialized data processing products is $1 postpaid First Class Mail.

PARROT is a programmable speech generator for the Timex-Sinclair ZX80, ZX81, or 7S-1000, computers. Price $89.95 plus $4 handling. Contact D.A.G.T., Inc., P.O. Box 499, Pt. Washington Station, Brooklyn, NY 11209.

SIG/M and the CP/M Users Group have been collecting CP/M software in public domain for several years. Eiam Associates now have the complete set of disks, which includes over 2000 programs and files on over 140 disks, and are offering to copy them for you. 8" single density disks and 5-1/4" disks with over 1000 disks are $10.00 per disk. Prices for listings of these are available for $10 each, or order diskettes CP/M-CAT and SIG/MUG-CAT at $10 per diskette. Contact Eiam Associates, 4600 Bommarito Street, Woodland Hills, CA 91367, phone (213) 348-4276.

Disk Niche is a solid hardwood flip-top storage bin for 5-1/4" floppy diskettes. Available in solid Walnut, Oak, or Cherry, with five movable tabbed dividers, it holds up to 50 diskettes. The Disk Niche offers a functional alternative in a "plastic world." Price $49.95 each plus $3 each for postage and handling. Visa & MasterCard accepted. Contact Systems Integration, 1519 North Nevada Avenue, Colorado Springs, CO 80907, phone (303) 635-4477, S.I.S. computer/CLUB. PRICE: $44.95 if you mention you heard about it through Proteus, the Processor Technology Users Group.

Macroma makes a TV Projection system that converts your ordinary TV into a projection TV that throws the screen image onto your wall or movie screen. "It's so easy a 12 year old can do it." The complete kit, including lens, is only $19.95. You supply wood or cardboard for cabinet and a common mirror obtainable locally for about $2. Kit contains precision lens system, detailed plans, complete instructions. Works with TV sets up to 26", even color sets. [Editor's note: This looks like a simple mirror-and-fresnel-lens camera obscura projector. The image won't be super bright, but in a dark room it should work okay. Clarity will depend upon lens quality. A curved screen is required if distortion is to be avoided. The company also offers plans for a do-it-yourself curved screen. Contact Macroma TV System, 15 N. Main Street, Washington Crossing, PA 18977, phone (215) 736-3997.

Optronics Technology makes a Disk Control Unit that turns off the motor of your 8" floppy disk drive when the disk has not been used for 9 seconds. The circuit board is the size of a business card and fits neatly into the drive, according to the manufacturer. It uses a zero-crossover and a built-in activity monitor. It is available for Shugart 800/801/850/851, Qume DB, Siemons and other drives. You MUST state the type of drive you order. As a kit, the price is $29.95; assembled and tested, it is $49.95 (you install in your drive, of course). Contact Optronics Technology, P.O. Box 81, Pittsford, NY 14534, (716) 377-0369.

SOL'S CO-DESIGNER ON COVER OF INFOWORLD

Lee Felsenstein, the co-designer of Processor Technology's SOL computer, appeared as the cover story of the November 7, 1982, issue of Infoworld, the newspaper for computer users. Dubbed the "Popular Engineer", the biographical story reveals Lee's background and interests as well as his involvement in microcomputer projects, such as the "Pennywhistle" modems, the SOL, the Osborne computer, and Community Memory. Old timers will remember the Pennywhistle modem, we all know the SOL, who can forget the Osborne, but what is Community Memory? Community Memory embodies one of Lee's long-term dreams: decentralization of computing power; in other words, computing power for the masses. A group of counter-culture people in Berkeley and San Francisco have been working for years, apparently at their own expense, to develop an open-ended, computerized building-board system with terminals located throughout a community and available to anyone without charge.
For example, in a scientific program you may be dealing with variables that have angular measurements in degrees and others with angular measurements in radians. Yet both variables would be declared with the same type "real". If your main program accidentally passed a degrees measurement to a subroutine where a radian parameter value was assumed, a program error would exist, but the compiler would have no way to catch it. In a strongly typed language, each of these types could be declared to be distinctly different, although they would all be single numbers as their base type. The compiler would enforce type compatibility.

A weakly typed language, on the other hand, has a few general classes of objects (integer, real, character, etc.) and it forces the programmer to keep their uses straight without giving him any help. A very weak programmer could get two variables confused and try assigning a time measurement in seconds (declared as a real number) to a linear variable with units of feet (also a real number) and the compiler would let it get through.

Ada avoids any such type confusion by requiring that all type conversions be explicit (unlike PL/I which has been criticized for making default type conversions with reckless abandon). That way, type errors can be caught by the compiler instead of sneaking through to become program bugs. Also, a name has only one type and that type defines not only that it represents a particular kind of data, such as an integer, but also it defines the valid range and precision of the permitted values. New instances of data types can be declared distinct from other similar types. For example:

**CIRCUMFERENCE** is new **DISTANCE IN FEET**;

defines two data types that are both based upon a previously defined type called **DISTANCE IN FEET**, but you can't pass:

**CIRCUMFERENCE** variable where a procedure expects a **HEIGHT** variable for its compiler could catch that error.

For example, here's a type definition from the Ada manual:

```
for example:
This means that **COEFFICIENT** is a programmer-defined data type which can hold a real number of at least 18 significant digits and ranging from 1e-1 to 1e+10. Later, you could apply this type to declare other types or variables. For example:
```

**SUBTYPE** SHORT **COEFF** IS **COEFFICIENT** digits 5;

defines a constrained version of the **COEFFICIENT**, this subtype having only 5 significant digits, (for example):

```
```

```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
Statements can have labels that are real identifiers, not just numbers. Labels are indicated by brackets made of double less-than signs: <<START>>. You can:

goto START;

Literal numeric constants can be expressed in any base from 2 (binary) through 16, much like the way it is done in the C language. In Ada:

1E0008
represents the hexadecimal number C000.

Procedure and function calls are done as in Pascal, just by naming the procedure and giving actual parameters in parentheses. But unlike Pascal, Ada allows the actual parameter to be either in the position corresponding to the formal parameter in the procedure definition, or to be identified by the name of the formal parameter it matches. For example:

PRINT HEADER(HEADER => TITLE, CENTER => TRUE);

is a call of the procedure named PRINT HEADER with the actual parameters being TITLE and TRUE, which respectively are to be substituted for the formal parameters HEADER and CENTER in the procedure definition. Formal parameters are the ones you give as dummy variables when you declare your procedure and give its body of statements. (Read the => symbol as "becomes".) Here it doesn't matter in what order you've placed HEADER and CENTER in the procedure parameter list. They will still be matched on the basis of the keyword.

This keyword matching of parameters is really helpful when using library routines that have lots of parameters, many of which can take default values. Ada lets you define default values for these parameters. In any called procedure, you can identify whether the parameter is an input value to the called procedure, an output value back from the routine, or an input/output value which is modified by the routine. You don't have to resort to error-prone calls like:

SORT (TABLE,A,"Table of Results",1,10,TRUE,"A",5)

where a couple of parameters could be reversed. You can still use the position to make the correspondence between actual and formal parameters, but it is optional.

Case statements can have an "others" alternative to catch any values that didn't meet the other case alternatives. Each alternative can have several values, or expressions, or a range of values, as the selecting choices. For example,

case COUNTER is
when 0 => INITIALIZE(BIN_NUMBER);
when 1..100 => UPDATE(BIN_NUMBER,COUNT,NEW_COUNTER);
when <110 => EMPTY BINS;
when 12..147 => raise NO MORE BINS(COUNTER);
when others => raise COUNTER_ERROR;
end case;

Arithmetic operators in Ada are similar to Pascal's, but with a few changes. The assignment operator is the same = symbol. The equal relation is the equals sign =, but the not-equal relation uses a slash equal /= pair. There is an exponentiation operator as in FORTRAN and PL/I, the double asterisk (**) symbol. There are no increment or decrement operators like the ones in C (++, --).

You can declare and initialize arrays of constants. You can declare arrays that have variables as subscript bounds (that is, you can have dynamically sized arrays). Array bounds can be passed as parameters.

Ada lets you get at the declared attributes of an identifier. For example, suppose you have a subroutine that

can use a quick algorithm if it is given an actual parameter of 5 significant digits or less, but has to use a more lengthy
algorithm to satisfy the precision of a larger number of digits. You would like to check the number of "significant" that were declared for the actual parameter (see the examples COEFFICIENT and SHORT COEFFICIENT given previously) and then branch accordingly in the procedure. Ada lets you do this. For any declared T, the expression

**DIGITS**

yields the number of decimal digits declared. Similarly other attributes can be obtained:

**LARGE** yields the largest number that can fit in T;
**EMAX** yields the largest exponent value in the type;

Array types have special attributes that can be expressed:

**FIRST**(N) yields the lower bound of the first index range;
**LAST**(N) yields the lower bound of the Mth index range;
**LENGTH**(N) yields the number of values in the Mth index range.

For example, if you have an array MESSAGE_TABLE that is declared:

**MESSAGE** : TABLE (1..MAX_MESSAGES,1..M_LENGTH),

then **MESSAGE_TABLE** (LAST(M) has the value of **MAX_MESSAGES**.

Ada lets you define your own types and pass parameters to the type definition. These parameters are known as discriminants. For example,

**type BUFFER( size : BUFFER_SIZE = 100) is record**

**POSITION : BUFFER_SIZE := 0;**
**VALUE : STRING (1..BUFFER_SIZE);**
**end record;**

defines BUFFER to be a record type having a parameter **size** which is of type BUFFER_SIZE with default value of size being 100. BUFFER contains 2 variables called **POSITION** and **BUFFER_SIZE** with default value zero. BUFFER also contains a **VALUE** which is a string of character size BUFFER_SIZE. Later in the program, I can use BUFFER to define a variable that is an actual buffer:

**MESSAGE_BUF : BUFFER (80) := (1,80 => ' ');**

This declares **MESSAGE_BUF** to be a **BUFFER** of size 80, that is 80 characters should be allocated for its **VALUE** string. The expression after the := assignment symbol is the initial value of **MESSAGE_BUF**. It tells the compiler that all 80 characters (subscripts 1 to 80) to blanks.

I hope this gives you a glimpse of the extent that Ada is like Pascal but goes beyond Pascal in giving the programmer greater power to express exactly what meanings are imposed on identifiers. Without going any further on data types, let me just say that Ada has scalar types, real (fixed and floating), enumeration, array types, string types, record types (with variant parts), access types (like pointers), private types (known only inside a limited scope, as in a library routine), and task types (which permit concurrent execution of separate parts of the program).

Another feature of Ada is the pragma. A pragma is a statement which conveys information to the compiler. It is similar to Pascal's pragramatic comments, things like:

"**40g,C,-g,-g**

which could mean to generate code, suppress the listing, and insert debugging code. These are non-standard features of many Pascal compilers. In Ada, these sorts of instructions to the compiler appear in a pragma statement. For example,

**pragma SUPPRESS (INDEX_CHECK, ON) => TABLE;**

will suppress the compiler generated code to check that the bounds of the array TABLE are not exceeded.

Another pragma **INLINE** instructs the compiler to make a procedure into an "inline" routine, which is compiled into an assembly language macro in that it generates code at each call, rather than one reusable procedure.

There are 14 standard pragmas and implementers can add their own.
Another unique feature of Ada is the "overloading" of operators. In Ada-esque, this means that an ordinary arithmetic operator, such as the multiplication sign "+", is given an additional, context-sensitive meaning. For example, if you define your own data type which you call MATRIX, you can also define a multiplication meaning of multiplying two objects of your type as done in a function definition using the operator in double quotes as the name of the function. For example:

```ada
function "**" (LEFT, RIGHT : MATRIX) return MATRIX is ...
end;  
```

This function would be introduced into a function definition for matrix multiplication of two matrices which are given as parameters (dummy) names LEFT and RIGHT, in actual use, you would "call" this function by using the operator in its customary way:

```
A := B ** C;  
```

meaning that matrix A should be set to the value of matrix C multiplied by matrix B, wherein A, C, and B are all previously declared of type MATRIX.

You can also overload relational operators, this lets you define your own ordering operation associated with a new data type.

Likewise, subprograms (procedures, functions) can be overloaded. For example, the IO packages supplied with the system define many different procedures all with the same name PUT. But there is a PUT that lets you output a floating point number to a file, a PUT that sends a floating point number to a string variable instead of a file, a PUT that outputs an integer to a file, and so on. When the compiler encounters a procedure call with the name PUT it looks at the types of arguments that are being passed and compiles the code to activate the appropriate PUT procedure that matches those argument types.

The matching of parameter type profiles to pair a call with one of the overloaded subprogram definitions mimics the SmallTalk scheme for identifying which messages an object can understand. (See my article SmallTalk in a Prior Issue.)

This overloading of operators and subprograms will let a programmer develop a very natural means of expressing operations on his own data types.

The concepts of information hiding and library units are useful in developing modular programs employing general purpose routines. In Ada, the "package" is a program unit that contains type definitions, object definitions (constants, variables, entries, etc.), and/or subprograms (procedures, functions). These subprograms are separated out so that they are available to the use of the code within the package.

For example, you may want to program a general purpose graphics package which has some externally known function names such as FILL and PLINE, and/or subprograms (procedures, functions). These are not only used for internal use by the code within the package, there may be some internal variables that are not needed outside the package. (A classic case of hidden variables is the random number generator which keeps a secret internal pattern that it uses to generate the next random number on the next call, modifying the pattern on each call.)

In Ada, a package declares which named objects are to be public and which are private. The "use" clause lets you bring into your program the definitions of names which are public (visible) in the packages you want to use.

Ada also provides certain library packages. The package STANDARD contains all predefined identifiers in the language and follows the rules of the language. The routines, SYSTEM, includes certain implementation-dependent characteristics such as SYSTEM.MEMORY_SIZE, and MACHINE_CODE allows insertion of native code in the program. CALS (Computer Aided Software Engineering) routines, also predefined in the package for addition or subtraction of TIME objects.

Selective waits allow the task to wait for certain conditions. TIME entry calls attempt to rendezvous but cancel if it is impossible. Tasks may have priorities, which may be by abort the statement, and may have variables for communication.
Exceptions handlers are routines that respond to unusual conditions, such as interrupts or execution errors (zero-divide, etc.). The exceptions can be predefined in any program (as in Pascal) or defined by the programmer in any program (as in C). The effective range (scope) of an exception handler is indicated by where it is defined, and the function to begin and end the handler are called EXCEPTION and END respectively.

For example,

```
COMPUTE X
COMPUTE Y
exception
  when OVERFLOW => PUT ("OVERFLOW ERROR");
  when others  => PUT ("FATAL ERROR");
  raise ERROREXIT;
end;
```

The raise statement lets the programmer voluntarily signal that an exception has occurred and that handler should be activated. Exceptions are one way that library packages can communicate with the calling procedures. For example, the exception STATUS ERROR is defined in the IO package and is raised if the IO routines if the file to be used is not open; the exception MODE ERROR is raised if an attempt is made to read an input file or write to a file that is not an output file.

And this brings up the IO packages. IO (input/output) is defined with library procedures rather than with special input/output statements. Procedures are supplied to create, open, close, delete, reset, read, write, set index (seek to a specified point in the file), and so on. Functions are provided to inquire on the mode, mode, current index, size, end-of-file condition, and so on.

The IO package defines routines for reading files composed as a sequence of pages, each having a sequence of pages. Lines of text of 80 characters are not specified, but the language provides a machine independent way of detecting and page terminator. The routine implementation of line and page termination is not specified, but the language provides a machine independent way of detecting and line and page length, create and test for terminators, read current line page and any location of the cursor, etc.

Limited formatted output, similar to Pascal's, is available, but surprisingly cumbersome. This may be a big flaw in the portability area. Even if all implementations of its own library routines for easy-to-use formatted text like that in PL/I/PICTURE clauses, users will have to transport the library too.

I hope this gives you a feeling for the flavor and power of the Ada language. If you are interested in learning more about Ada, you may want to purchase the Ada Reference Manual that has been mentioned, as well as the library text. I'm sorry I don't yet have a reference for an introductory text, but I've read that one is available published as "Ada - An Introduction" by Nara Gehani, 920.23, 325 pages, which is good once you understand the basic parts of Ada.

You may also wish to subscribe to the Journal of Pascal and Ada, published every other month for $14.00 per year ($21.00 outside U.S.). Send subscription orders to the Journal of Pascal and Ada, P.O. Box 384, Oron, Utah 84057.

Digital Research has provided a new version of CP/M-3.0, which has support for a bank selected memory. This is CP/M-3.0, which has support for a bank selected memory. If you know of some PASCAL ERROR handler (as in, SINGULAR_MATRIX), the effective range (scope) of an exception handler is indicated by where it is defined, and the function to begin and end the handler are called EXCEPTION and END respectively.

For example,

```
COMPUTE X
COMPUTE Y
exception
  when OVERFLOW => PUT ("OVERFLOW ERROR");
  when others  => PUT ("FATAL ERROR");
  raise ERROREXIT;
end;
```

The raise statement lets the programmer voluntarily signal that an exception has occurred and that handler should be activated. Exceptions are one way that library packages can communicate with the calling procedures. For example, the exception STATUS ERROR is defined in the IO package and is raised if the IO routines if the file to be used is not open; the exception MODE ERROR is raised if an attempt is made to read an input file or write to a file that is not an output file.

And this brings up the IO packages. IO (input/output) is defined with library procedures rather than with special input/output statements. Procedures are supplied to create, open, close, delete, reset, read, write, set index (seek to a specified point in the file), and so on. Functions are provided to inquire on the mode, mode, current index, size, end-of-file condition, and so on.

The IO package defines routines for reading files composed as a sequence of pages, each having a sequence of pages. Lines of text of 80 characters are not specified, but the language provides a machine independent way of detecting and line and page length, create and test for terminators, read current line page and any location of the cursor, etc.

Limited formatted output, similar to Pascal's, is available, but surprisingly cumbersome. This may be a big flaw in the portability area. Even if all implementations of its own library routines for easy-to-use formatted text like that in PL/I/PICTURE clauses, users will have to transport the library too.

I hope this gives you a feeling for the flavor and power of the Ada language. If you are interested in learning more about Ada, you may want to purchase the Ada Reference Manual that has been mentioned, as well as the library text. I'm sorry I don't yet have a reference for an introductory text, but I've read that one is available published as "Ada - An Introduction" by Nara Gehani, 920.23, 325 pages, which is good once you understand the basic parts of Ada.

You may also wish to subscribe to the Journal of Pascal and Ada, published every other month for $14.00 per year ($21.00 outside U.S.). Send subscription orders to the Journal of Pascal and Ada, P.O. Box 384, Oron, Utah 84057.

Digital Research has provided a new version of CP/M-3.0, which has support for a bank selected memory. This is CP/M-3.0, which has support for a bank selected memory. If you know of some PASCAL ERROR handler (as in, SINGULAR_MATRIX), the effective range (scope) of an exception handler is indicated by where it is defined, and the function to begin and end the handler are called EXCEPTION and END respectively.

For example,

```
COMPUTE X
COMPUTE Y
exception
  when OVERFLOW => PUT ("OVERFLOW ERROR");
  when others  => PUT ("FATAL ERROR");
  raise ERROREXIT;
end;
```

The raise statement lets the programmer voluntarily signal that an exception has occurred and that handler should be activated. Exceptions are one way that library packages can communicate with the calling procedures. For example, the exception STATUS ERROR is defined in the IO package and is raised if the IO routines if the file to be used is not open; the exception MODE ERROR is raised if an attempt is made to read an input file or write to a file that is not an output file.

And this brings up the IO packages. IO (input/output) is defined with library procedures rather than with special input/output statements. Procedures are supplied to create, open, close, delete, reset, read, write, set index (seek to a specified point in the file), and so on. Functions are provided to inquire on the mode, mode, current index, size, end-of-file condition, and so on.

The IO package defines routines for reading files composed as a sequence of pages, each having a sequence of pages. Lines of text of 80 characters are not specified, but the language provides a machine independent way of detecting and line and page length, create and test for terminators, read current line page and any location of the cursor, etc.

Limited formatted output, similar to Pascal's, is available, but surprisingly cumbersome. This may be a big flaw in the portability area. Even if all implementations of its own library routines for easy-to-use formatted text like that in PL/I/PICTURE clauses, users will have to transport the library too.

I hope this gives you a feeling for the flavor and power of the Ada language. If you are interested in learning more about Ada, you may want to purchase the Ada Reference Manual that has been mentioned, as well as the library text. I'm sorry I don't yet have a reference for an introductory text, but I've read that one is available published as "Ada - An Introduction" by Nara Gehani, 920.23, 325 pages, which is good once you understand the basic parts of Ada.

You may also wish to subscribe to the Journal of Pascal and Ada, published every other month for $14.00 per year ($21.00 outside U.S.). Send subscription orders to the Journal of Pascal and Ada, P.O. Box 384, Oron, Utah 84057.
EXTENDED I/O ADDRESSING

The 8080 processor puts I/O port addresses out onto the 16 bit address bus in what is called a "mirrored" fashion. The familiar port address, which is only 8 bits wide for 256 ports, is repeated on the lower half and upper half of the 16 bit bus. In the Sol, the port decoder actually looks at the high order 64K of the Sol's port addresses. As a 16 bit address, the Sol ports are FFFF, FFFF, ..., FFPA. Just stay away from these addresses, and there will be no conflict. If you go to a 16 bit processor and disable the 8080, as suggested in Part I of this series and as described by Ed Bolton in this issue, you should just refer to the Sol ports by the 16 bit version of their addresses.

New I/O boards that use consecutive 16 bit addresses may have to be modified to ignore the high-order address lines when decoding the port address if they are used in an unmodified Sol, since the Sol can't generate consecutive 16 bit addresses due to the mirroring. This can probably be done by pulling a few IC pins and tying them to ground or +5V.

THM CONTROL

The standard allows up to 16 temporary bus masters to take over the bus from the permanent master (CPU). This allows devices such as disk controllers to temporarily use the bus to input or output data between the controller and the memory. The Helios disk controller is designed to use direct memory access (DMA) when it is used to be called, but it does not use a standard method for coordinating the activity of competitive controllers trying to take over the bus simultaneously. The coordination is called bus arbitration. As we mentioned in Part I, it uses some of the bus lines that the Sol has been using for other signals, but the signals can be relocated or abandoned. The Helios can co-exist with other DMA devices (temporal masters) as long as they are not both activated at the same time. You will have no trouble if you just be sure to let the Helios finish its entire sector transfer before you start any other DMA device going. Then there will never be a bus contention, so arbitration is irrelevant.

HELIOS DISK CONTROLLER

The Helios controller will work in a system with other TMA devices as I mentioned above. It will not work in a system that can't supply it with a 2MHz clock, though. But, the controller, I am told, will work just fine if you have it take its clock from the IEEE 696 clock signal (pin 49) instead of the master system clock 0 on pin 72.

Apparantly the Helios just needs the clock for its internal timing, but the bus timing is obtained from other timing signals on the bus. I haven't tried it myself, but you could easily make the change. It just involves cutting the trace to pin 49 on the Helios controller board and jumpering that trace to pin 24 instead.

Bob Mogg tells me this will let the Helios work just fine in a Sol with the new 2-80 upgrade installed, even when the Sol's clock runs at 4MHz. I'm installing a 2-80 upgrade into a Sol, so I'll let you know how it goes with the Helios. If it works, you can conclude that it will work on other processors as well.

MEMORY BOARDS

Processor Tech made provisions for extended addressing on the NTKA line of memory boards (not the old 15KRA or 32KRA, but the 8KRA-1, 32KRA-1, and 4KRA). However, the circuitry is in a skeletal fashion and you must add to it. It is in light of my earlier comments about extended memory boards for the Sol, you can see why I haven't bothered to figure out the exact changes to make in an NTKA into an IEEE 696 standard 24-bit address memory board.

Another handicap is that the NTKA is designed to run at system clock rates of more than twice that of 4MHz. Thus, with a 2MHz system clock, you couldn't replace the memory with faster boards.

My advice is use the Processor Tech boards in the good ol' Sol and sell them or give them away when you must upgrade. Otherwise, you'll be stuck with a 2MHz system clock, and with 6K, 8, and 16 MHz processors available now, who wants to cripple a speedy processor just to let it slow down for a little memory board.

WHY UPGRADE?

In summary, my feelings are this. It only makes sense to upgrade your system if you need something you don't now have. If you want to run a particular kind of software that isn't available for 8080 systems, then consider the M25UL (2-80 upgrade) for a 2-80 microprocessor, or the trick of disconnecting the bus from the Sol motherboard and using the Sol as a terminal to your 16-bit microprocessor 5-100 board system in the bus.

If you need to run software that uses more than 64K RAM, you should upgrade the processor to a 16-bit CPU. Or consider buying another computer.

Meanwhile, it still does everything it did the day you bought it. What do you do with the Sol when you no longer are satisfied with it? My feelings are sentimental and pragmatic. Imagine what it would be like now if you had your grandfather's first car in mint condition or how about one of those old radios with the huge glowing tubes and horn-shaped speaker. It would be a classic, or at least an antique.

Go into a computer store these days. They never heard of the Sol. Not even the Intel 8080 microprocessor. It is fast becoming an antique.

Keep the Sol. Keep it in working condition. Keep all the documentation you have about it. Collect other artifacts about it - magazine ads, articles, stories. Have them for historical purposes. Someday it will all be worth more than you paid for it.

(shall continued from page 7)

An article in that journal has reviewed some of the Ada compilers under development. The Janus/Ada compiler by (P.O.Box 1512 Madison, WI 53701, 608-244-6636) runs on 2-60's under QM as well as the popular 16 bit operating systems for the 8088/8086. It only compiles a subset, of course, but it was commended by an expert panel at the Ada compiler, and the Ada compiler, and its separate feature is used in many different compilers, including Marcom Corporation, 3340 Thorton Drive, Suite 102, San Jose, CA 95123, (408) 226-0170.
Dear Stan:

When I noted that it was renewal time for the subscription to PROTEUS and saw your request for material to put into it, I thought that some fellow readers might be interested in this in that work. At the time I started on it, the BSR X-10 control system was just out and Steve Garcia's article in Radio-Electronics for an ultrasonic interface was an attractive method of connection. Since my home controller wasn't an ultra-sonic controller I just added the missing connector to the P.I.C. board and used a high frequency transformer to couple the signal from the interface. I found that his data field was slightly in error but when corrected, this system worked fine. I thought that this arrangement used rather a lot of hardware for a simple function so I decided to experiment with the latest new Mitel MT88048 cross point switch. This chip ties in very well with the PICO 542C controller chip used in the BSR X-10 control module. Power for the system can be drawn from the S-100 bus. The output to the line is interfaced via the output transformer from the module and two 50V0F disk capacitors.

Since I was interfacing this system to my SQL, I thought I would also need a real time clock to use the computer as a multi-point time controller. The NATIONAL MKS167 clock chip seemed to have all the functions one might need and it is easy to interface to the bus as well. I found that this clock chip must be put into the standby mode before the system power falls or the clock timekeeping will be lost. I added a power fail detector circuit, INTERFACIAL ICQ-9211, to monitor the S-100 8V rail and shut down as soon as the rail begins to drop. Unfortunately, the S-100 power system is quite independent of the S-100 bus power, and since my S-100 bus is quite lightly loaded, the main board logic fails well before the clock is affected. It is necessary, therefore, to add a 10k ohm resistor from the positive on C9 in the SQL power supply to line 21 of the S-100 bus to feed the power monitor.

I also had just acquired a modem, and since the serial port on the SQL was used as a custom driver port for my printer, I needed another serial port. I decided that the widely used 8251A uart was the way to go on this port as it is more flexible than the fixed function uart's such as the 8255 series.

I have attached a copy of the schematic for my board, which I built up on an S-100 prototyping board using a variation on the VDC-700 wiring pencil system. As is common practice on digital circuits, the layout is not too critical as long as the leads subject to interference are kept away from the data and address buses. Be sure to use bypassing, 0.1uF per chip, on the power supply lines. As can be seen, the various modules all use the common circuits ( addresstester, data buffering ). One can easily add more modules not required or, indeed, add other functions, as there is lots of room left on the board.

Note that the D I/O lines from the 74LS241 buffers are connected together. The alternate pin numbers show the connections for full 8-100 implementation of the unidirectional data buses, but only those shown are needed on a SQL.

All component values are shown on the schematic except the line transformers. The H.P. transformer one is from the BSR X-10 control module as is the PICO 542C chip and their associated components. Be sure to trace the pinout of the H.P. transformer before removing it from the BSR module. The two H.P. output capacitors must be 500V ceramic for safety.

The power transformer is a reworkied, cheap, "AC adaptor." Cut the case open along the joint with a thin saw (not blade sharp) and carefully remove the transformer and filter, noting the lead connections. Very carefully separate the core laminations and slip them out of the bobbin—heat may help if they're really stuck. If there is enough room over the wrapper insulation, add a few hundred turns of #38 AWG or smaller magnet wire to the bobbin, making sure the winding does not extend beyond the lips of the bobbin. Cover the winding with one layer of hard tape e.g., Magic tape. Restack the core just as you found it, being careful not to nick the new winding. Carefully test the winding insulation to be sure there is no leakage between the secondaries and the primary. You don't want to kill your SQL or yourself. Alternatively you can use two transformers such as the RADIC SHACK 273-1385 connected back to back as shown, and mounted in a mini-box. Mount the H.P. transformer and its associated components on some vector board and mount it next to the transformer in the case. Connect the HF coupling capacitors to the line with fine wire ( #38 ) to act as fuses. Run a length of ribbon cable from this assembly to the this 'n that board. Remember, 120V AC around the logic cards is a no-no for everybody's safety!

A part that was at first difficult to find was the 32k crystal for the clock. Go to an electronic watch repair shop for one. They are about the size of a rice grain, and though they are quite sturdy they should be added to the board last.

Attached are the lists of the access codes for each of the modules, all in decimal notation. The programs I use for the BSR and clock modules are written in Pro Tech PC Basic running under CP/M ( Tad Enterprises ). I have also included the source listing for the modem driver I use for accessing bulletin boards. It contains a ratrace buffer for the printer so that a 300 baud input can get to a slower printer, since most BB's have numerous pauses in their output. To allow the printer to catch up. The printer is turned on with \ and off with @ any time during the program. If the buffer is overfilled, the data is overwritten.

I hope this brief description of the this 'n that card and its functions is of some use to our fellow readers, and will encourage them to submit material to PROTEUS.

Regards,

Walt Batschke

See documentation on next page...
CONTROL CODES FOR 8051A UART

ADDRESS: DATA = 96
UNDECODED ADDRESS FOR 32 PORTS

MODE/CONTROL = 97

MODE WORD: 1st CONTROL WORD AFTER A RESET (WRITE)
M.B. HARDWARE RESET IS NOT IMPLEMENTED - USE SOFT RST.

D7 D6 D5 D4 D3 D2 D1 D0
BAUD RATE FACTOR

0 1 0 1
0 0 1 1
SYNC 1X 16X 64X

TYPICAL WORD
FOR STD ASYNC.
= 78

CHARACTER LENGTH

0 0 0 1
0 0 1 3
5 6 7 8

PARITY ENABLE
1=ENAB 0=DISAB

EVEN PARITY (GEN/CHECK)
1=EVEN 0=ODD

# OF STOP BITS

0 0 1 0
1 0 1 0

INVAL: 1 1.5 2

CONTROL WORD: (WRITE)

D0 - TX ENABLE 1=EN 0=DIS
D1 - DTR 1=DTR 0=DT-2
D2 - RX ENABLE 1=EN 0=DIS
D3 - 2ND BREAK 1=BRK 0=NORM
D4 - ER 1=ERROR 0=NORM (ERROR RESET PE, OE, PE)
D5 - RTS 1=RTS 0=RTS-
D6 - IR (RESET) 1=RESET 0=NORM
D7 - ER 1=SEARCH FOR SYNC CHARACTER (ENTER HUNT MODE)

STATUS: (READ)

D0 - Tx RDY 1=DATA BUFFER EMPTY
D1 - Rx RDY 1=CHRI READY IN RECEIVER
D2 - RX EMPTY 1=TRANSMITTER EMPTY
D3 - PARITY ERR 1=ERROR
D4 - OVER RUN 1=ERROR (CHR NOT READ BEFORE NEXT RCVD.)
D5 - FRAMING 1=ERROR (NO VALID STOP BIT)
D6 - SYNCHR 1=SYNC CHR (SYNC MODE ONLY)
D7 - BSR 1=BASIC INTERNAL 0=BSR INPUT--

NOTES:
LATCHES SET WITH 1 IN TWO
MSB'S OF THE BYTE CAUSE A 'DON'T
CARE' STATE IN THE LATCH.
STATUS: BIT: A 1 IN D0 INDICATES A
ROLL OVER DURING THE LAST READ OP
OF A COUNTER. D1-D7 = 0
GO: A GO COMMAND RESETS 1/1000,
1/100,1/10 AND SEC THEN STARTS
THE CLOCK.
STOP INTERRUPT: A 1 IN D0 SETS THE
INTERRUPT; A 0 DISABLES IT. THIS
OVERRIDES DURING POWER DOWN.
10 REM TEST AND DIRECT CONTROL PROGRAM FOR BSR SECTION IF
20 REM THIS N' TH CARD THAT CARD
30 INPUT "STATION ", B
40 IF B=0 THEN GOTO 220
50 ON D GOTO 60,70,80,90,100,110,120,130,140,150,160,170,180,190,200,210
60 LET D=121 GOTO 310
70 LET D=21 GOTO 310
80 LET D=4 GOTO 310
90 LET D=20 GOTO 310
100 LET D=2 GOTO 310
110 LET D=18 GOTO 310
120 LET D=10 GOTO 310
130 LET D=26 GOTO 310
140 LET D=1 GOTO 310
150 LET D=30 GOTO 310
160 LET D=61 GOTO 310
170 LET D=221 GOTO 310
180 LET D=01 GOTO 310
190 LET D=16 GOTO 310
200 LET D=81 GOTO 310
210 LET D=241 GOTO 310
220 INPUT "FUNCTION ", D
230 IF D="" THEN LET D=51 GOTO 310
240 IF D="ALL ON" THEN LET D=51 GOTO 310
250 IF D="ALL OFF" THEN LET D=01 GOTO 310
260 IF D="OFF" THEN LET D=01 GOTO 310
270 IF D="ON" THEN LET D=51 GOTO 310
280 IF D="TRUE" THEN LET D=51 GOTO 310
290 IF D="FALSE" THEN LET D=01 GOTO 310
300 PRINT "NOT Valid FUNCTION"; GOTO 220
310 OUT 255, 0
320 PAUSE 10
330 OUT 253, 128
340 IF S=0 THEN GOTO 220
350 IF S=20 THEN GOTO 370
360 GOTO 30
370 INPUT "MORE? ", D
380 IF D="Y" THEN GOTO 310
390 IF D="N" THEN GOTO 310
400 LET S=GOTO 220
410 END

LI.
10 REM: BSR 100 EVENT TIME OF DAY CONTROL IN FTC EXTENDED BASIC
20 REM: INPUTS MUST BE IN CHRONOLOGICAL ORDER
30 DIM D(20),S(20),A(100),B(100),C(100),E(100)
40 REM: TRANSLATION TABLE FROM POINT NUMBER TO CONTROL CODE
50 DATA 12,28,4,20,2,18,10,26,14,30,6,22,0,16,8,24
60 FOR X=1 TO 16
70 NEXT X
80 FOR X=1 TO 10
90 NEXT X
100 LET B=5 REM. OFF CODE
110 INPUT (1,0) UNIT , A
120 LET A(L)=A(L)+CONTROL CODE FOR POINT
130 INPUT (1,0) FUNCTION , B
140 IF B="ON" THEN LET B=51 REM. ON CODE
150 LET A(L)=B
160 INPUT (1,0) HOURS , D
170 LET E(L)=E(L)+HOURS IN BCD
180 INPUT (1,0) MINUTE , C
190 LET D(L)=D+MINUTES IN BCD
200 INPUT (1,0) HOUR , D
210 IF D="N" THEN LET D=EXIT 300
220 NEXT L
230 REM: DECIMAL TO BCD CONVERSION
240 LET X=INT(C/10) REM: FIND 10'S
250 IF X=0 THEN LET C=RETURN
260 LET Y=16*X REM: MULTIPLY 10'S BY 16
270 LET Z=C-2*X REM: STRIP 10'S FROM C
280 LET A(0)=Z REM: FORM TWO NIBBLE BCD VALUE
290 RETURN

LI.
300 FOR D=1 TO P
310 REM: CLOCKS READ AND CONTROL OUT LOOP
320 IF INF(1)=E(L) THEN GOTO 320 REM: TEST FOR HOUR MATCH
330 IF INF(2)=E(L) THEN GOTO 330 REM: TEST FOR MINUTE MATCH
340 IF INF(3)=E(L) THEN GOTO 340 REM: TEST FOR HOUR MATCH
350 IF INF(4)=E(L) THEN GOTO 350 REM: TEST FOR MINUTE MATCH
360 IF INF(5)=E(L) THEN GOTO 360 REM: TEST FOR HOUR MATCH
370 IF INF(6)=E(L) THEN GOTO 370 REM: TEST FOR MINUTE MATCH
380 REM: DO OUTPUT ROUTINE TO BSR CHIP WITH APPROPRIATE CLEAR
390 REM: AND WAIT FOR THE MODULES TO REACT.
400 OUT 253, A(R); PAUSE 10; OUT 253, 128; PAUSE 2
410 OUT 253, B(R); PAUSE 10; OUT 253, 128; PAUSE 2
420 NEXT R
430 END

LI.
10 REM: CLOCKS READ ROUTINE
20 CURSOR 1,12
30 FOR X=7 TO 2 STEP -1
40 LET X=INF(X)
50 LET X=INT(N/16)
60 IF X=0 THEN LET M=01 GOTO 80
70 LET M=X
80 LET L=N-X16
90 ON R GOSUB 600,600,500,400,350,200
100 NEXT R
110 PAUSE 10 GOTO 20
200 ON (M10+L) GOSUB 220,230,240,250,260,270,280,290,300,310,320,330,340
210 RETURN
220 PRINT "JANUARY "; RETURN
230 PRINT "FEBRUARY "; RETURN
240 PRINT "MARCH "; RETURN
250 PRINT "APRIL "; RETURN
260 PRINT "MAY "; RETURN
270 PRINT "JUNE "; RETURN
280 PRINT "JULY "; RETURN
290 PRINT "AUGUST "; RETURN
300 PRINT "SEPTEMBER "; RETURN
310 PRINT "OCTOBER "; RETURN
320 PRINT "NOVEMBER "; RETURN
330 PRINT "DECEMBER "; RETURN
340 PRINT "DECEMBER "; RETURN
350 IF M=0 THEN PRINT " "; RETURN
360 PRINT (M10+L)" "; RETURN
370 PRINT (M10+L)" "; RETURN
400 ON (M10+L) GOSUB 420,430,440,450,460,470,480
410 RETURN
420 PRINT "SUNDAY "; RETURN
430 PRINT "MONDAY "; RETURN
440 PRINT "TUESDAY "; RETURN
450 PRINT "WEDNESDAY "; RETURN
460 PRINT "THURSDAY "; RETURN
470 PRINT "FRIDAY "; RETURN
480 PRINT "SATURDAY "; RETURN
500 IF M=0 THEN PRINT " "; RETURN
510 PRINT (M10+L)" "; RETURN
520 PRINT (M10+L)" "; RETURN
600 IF M=0 THEN PRINT " "; RETURN
610 PRINT (M10+L)" "; RETURN

LI.
10 REM: SET CLOCK ROUTINE
20 FOR F=7 TO 3 STEP -1
30 ON D GOSUB 200,200,240,220,210,200
50 LET A=N=INT(N/10)
60 IF X=0 THEN LET M=01 GOTO 100
70 LET L=161X
80 LET Z=N-10X
90 LET G=Y
100 LET L=11G
110 NEXT D
120 INPUT (1,0) "INPUT 0 TO START CLOCK", 0
130 OUT 210, 0
140 END
200 INPUT (2,0)"MONTH ",N:LN RETURN
210 INPUT (2,0)"DATE ",N:LN RETURN
220 INPUT (1,0)"DAY OF WEEK (SUNDAY = 1 ) ",N:LN RETURN
230 INPUT (2,0)"HOUR ",N:LN RETURN
240 INPUT (2,0)"MINUTE ",N:LN RETURN
250 INPUT ".N:LN RETURN

1: MODEM AND PRINTER CONTROL PROGRAM FOR 8251A
2: RVY N,STAND.FIFE
3: ID:DATE 810820
4: THIS PROGRAM IS LOADED AT 1004 TO RUN A
5: MODEM THROUGH THE THIS N PUT CARD
6: HARD PRINTER THROUGH PORT #5
7: 1
8: 1
9: 1
10: 1
11: 1
12: 1
13: 1
14: 1
15: 1
16: 1
17: 1
18: 1
19: 1
20: 1
21: 1
22: 1
23: 1
24: 1
25: 1
26: 1
27: 1
28: 1
29: 1
30: 1
31: 1
32: 1
33: 1
34: 1
35: 1
36: 1
37: 1
38: 1
39: 1
40: 1
41: 1
42: 1
43: 1
44: 1
45: 1
46: 1
47: 1
48: 1
49: 1
50: 1
51: 1
52: 1
53: 1
54: 1
55: 1
56: 1
57: 1
58: 1
59: 1
60: 1
61: 1
62: 1
63: 1
64: 1
65: 1
66: 1
67: 1
68: 1
69: 1
70: 1
71: 1
72: 1
73: 1
74: 1
75: 1
76: 1
77: 1
78: 1
79: 1
80: 1
81: 1
82: 1
83: 1
84: 1
85: 1
86: 1
87: 1
88: 1
89: 1
90: 1
91: 1
92: 1
93: 1
94: 1
95: 1
96: 1
97: 1
98: 1
99: 1
100: 1
101: 1
102: 1
103: 1
104: 1
105: 1
106: 1
107: 1
108: 1
109: 1
110: 1
111: 1
112: 1
113: 1
114: 1
115: 1
116: 1
117: 1
118: 1
119: 1
120: 1
121: 1
122: 1
123: 1
124: 1
125: 1
126: 1
127: 1
128: 1
129: 1
130: 1
131: 1

13
Record stores, supermarkets, libraries, etc. would have terminals all tied to a central computer in the city. People would be able to enter any sort of message they want, just as they now can place a note upon a common bulletin board.

"Community Memory's ultimate role," the story states, "may be to pull together all of the bulletin boards and other information sources in the country into one network that is accessible to everyone."

It was through the Community Memory group that Felsenein and Bob Marsh, the other co-designer of the Sol, met and began to collaborate on computer add-ons for the MIT's Altair computer. Marsh and another firm friend and former Digital Technology Corporation, contracted with Felsenein for the design work, and the OSRA memory boards, the CUTF tape interface, and VDM video display were born.

Late in 1975, Les Solomon, the technical editor of Popular Electronics (now called Computers & Electronics) magazine, challenged Bob Marsh to come up with a low-cost design for a computer terminal within one month and promised to make it a cover story on his magazine. This fit into one of Lee Felsenein's earlier goals, namely to design a low-cost "Tom Swift Terminal" for Community Memory. So, working days and nights, Marsh and Felsenein designed and built the first Sol. The rest of the Sol story we know all too well.

What's Lee Felsenein doing these days? He still moderates the Homebrew Computer Club, which was the breeding ground for many microcomputer companies and where I got my first introduction to the Sol.

I recall the night when Steve Jobs came to the Homebrew meeting with the first production version of the Apple II case and held it up in the air for all to see. Everyone was quite impressed that these two kids, Wozniak and Jobs, had gotten together the big bucks to have a real structural-foam case designed and built. And in those days, microcomputers all had low-tech, sheet-metal and screws, cases which were cheap to make in small quantities. The big savings for the foam cases cost a bundle, but would pay off when production was huge. No one dreamed how huge it would become. Apple had sold more than a million Apple II's, and the end is not in sight.

And Lee is also working on another of his long-term ideas through his company called "Golems." The name draws from the ancient Hebrew tale about a "golem," a creature made of clay and brought to life by religious rituals, but which got into a water-fetching infinite loop and flooded the entire village. Of course, today we call these things robots.

Unlike a robot, Golems' idea of a golem would be a machine which has some autonomous features of a robot but also depends upon a close interaction with a human operator. Golems is what the golem lacked.

I hope these highlights of the Infoworld story will encourage you to find a copy of this issue and read more about Lee's life and aspirations. He's an interesting guy, and I'm glad that Infoworld is bringing out some of the human side of this high-technology era.

His story reminds me of the frontier men in the early days of settling into this continent. These loners would go out for long times into the beautiful but wild, uncharted country, sometimes returning to work for one of the civilized entrepreneurs who needed their knowledge of the territory, yet always going back out beyond the frontier following their dreams.

I'm sure we haven't heard the last about Lee. He seems to have a great resiliency and bounces back into the visible universe every once in a while from the fringes of his frontier.
...OH A DATA BASE MANAGER AND ELECTRONIC SPREAD SHEET

Dear Stan,

I would like to have the use of a data base manager and an electronic spread sheet. I have a SOL/Helios system. Could you tell me what I need to do to get this on my SOL? I am trying to convert the spread sheet program from the last Proteus but am having some difficulty. It seems that I need to have a character in the string variable in order to specify particular parts of the string to point to (e.g., A(2,4) must have a length of at least 4 characters or else I get an OUT OF BOUNDS error). This becomes a problem with a Data Base program I am trying to work out also.

Sincerely,

Roy Haybrock
2118 Lakeshore Drive
Greensboro, NC 27407

--Oh Amateur Radio Software for SOL

October 17, 1983

Francis R. Bialetto
135 West 35th Street
New York City, NY 10001
(212) 736-8500

Dear 'Proteus':

Can anyone in your organization steer me toward any SOL programs suitable for Amateur Radio work. My SOL has 48K of memory and 2 drives (HELIOS) also tape.

Enclosed is a copy of a letter from Joe A. Elliott of the Fleisher Corporation. His outfit makes the hardware necessary to connect the SOL to the ham radio set. If you can help me with the software I'll have all the ingredients.

Any assistance you can lend will be very much appreciated.

Sincerely,

Francis R. Bialetto

---

PTC BASIC RISES AGAIN AS "NEVADA BASIC"

Processor Technology's Extended Disk BASIC with Business Basic features, has been enhanced and adapted to CP/M by Ellis Computing, Inc. If you are looking for a transition from your Extended Cassette or Extended Disk BASIC programs to a CP/M compatible BASIC, this is the way to go. See the press releases that follow in this issue. (Chuck Ellis is a Proteus member and has been licensed for commercial use of the PTC BASIC and PTDOR EDIT.) PTC FORTRAN is also available for CP/M from Ellis, under the name "NEVADA FORTRAN."

(Editor's reply: For a real, commercially produced database manager and spread-sheet program, I recommended getting CP/M for your Helios and buying those programs from Lifeboat Associates. For a do-it-yourself, simple spread-sheet, we now have H8R's ESS program converted to Helios/EDBASIC. It was donated to the Helios library by Leonard Cole, and will be described in the next issue. --Stan.)

(News Release)

ELLIS COMPUTING, INC.
SOFTWARE TECHNOLOGY
911 Noriega Street
San Francisco, CA 94112 U.S.A. (415) 753-0186

COMMODORE-64 GETS COBOL AND FORTRAN

August 16, 1983

Ellis Computing, Inc., has signed two agreements with Commodore Business Machines, Inc., to provide Nevada COBOL and Nevada FORTRAN under the CP/M operating system for the Commodore-64.

Both Compilers will be made available worldwide, exclusively through Commodore's standard distribution channels. Production is well underway, with delivery set for early this fall.

Ellis Computing, Inc., is a San Francisco based Software Development Firm located at 911 Noriega Street, San Francisco, Ca., 94122.
Dear P. T. Users:

I have six P. T. boards that I need documentation for. They are bare boards: one is a CPU board #PC11201 Rev B, a 5 slot backplane, a GM, a CUPS and a 16K to 64K DRAM, and a 2708 personality module. I would like to know if any M.R.D.E. members would have any circuit drawings and parts lists for these. I am willing to trade the CPU's personality module for documentation on the $100 boards, or if no one wants to trade, I am willing to pay the cost of copying and postage.

Dear Mr. Nimsberger:

I have no documentation for these boards. If you have any, I would be grateful.

State University of New York
Downstate Medical Center

MAILING ADDRESS: 655 Clarkson Avenue, Brooklyn, New York 11203/Phone: (212) 278-1009

Dear Sir:

I have been trying to interface National Semiconductors 58174A Real Time Clock to the SOL and have been having trouble with it. The interface is the main problem and wait periods need to be generated, but so far all attempts to do this have met with failure. Has anybody that you know managed to design an interface for this chip to the SOL and would they be willing to let me have the circuit involved, as I am most anxious to get the clock working. I have written the software to display the time and date on listings and did them all on SOLs that worked for a while until the ambient temperature rose, so I was able to get the software working, and would be willing to swap it for the appropriate hardware. I have the clock wired as part of memory at the moment, and would prefer to use it this way rather than have to use input/output ports.

I hope you can be of assistance in these matters as there is no SOL User's Group here, and most people haven't even heard of the SOL.

Thank you in anticipation,

Yours faithfully,

Bruce S. Ball

(continued from bottom left)

4) Which format is most useful/cost efficient and has the least software support by users or manufacturers?

5) Which format has Basic available from a reliable source?

6) Does such basic allow usage of cassette Basic programs?

7) Is CP/M available for both formats? Yes. Usually from the vendor who sells the controller board.

8) To what extent must the BIOS be tailored to the various parts of the system? What features are there?

9) Which memory boards are recommended/most compatible for all use in SOL? Static memory more compatibility because it doesn't depend upon bus cycle timing as much.

10) I purchased an IMSAI mainframe to expand SOL's base. Is there any special configuration for the SOL extension that I'm going to construct? See my article on the Sol 100 standard in last issue. Also see back issues. A few people have done this.

11) Which video board would give best results/dollar with little or no modifications to SOL? That depends upon what features you want in a second video. For consoles only, a single board would do.

12) What video hardware/software addition to SOL is most recommended? CP/M/Minibyd

(continued from bottom right)

[Note about disk formats: Before IBM produced the IBM PC, there was no one format for 5 1/4" drives that emerged as an overwhelming standard. However, the IBM PC and its clones have such a huge market, that one could say that the IBM PC format is the standard. Softdisk Microsystems (the USP/PC System vendor) calls it "The Universal Medium."]

(continued from bottom right)
Dear Stan,

Enclosed is a photostat of a printout of a manual of a graphic upgrade for the SOL/UV2. I am sorry that the quality is not better. I do not know if there is a copywriter, or who would own it. I found it in the back of the manual when I bought another SOL. It might be of use to someone. Also enclosed is $24. I hope it is enough for next year. You never seem to bill me in time. I will send more if you raise the price.

For those who have 6/2 BASIC, this may be of some use. If you plug two SOLs together as described in the Chess-88 manual, and load 6/2 BASIC into one, and Extended Cassette BASIC into the other, you may translate files to a format that can be read by ECB. Set the baud rate as slow as possible, 75 baud, and the rest of the serial port switches the same. The SOL with ECB type SET D11, in the other SOL type RENUM 100,18, SET D11, and then LIST R10. The SOL with ECB will treat the SOL with 6/2 BASIC as a text type. You will also lose the ends of some long lines, and the line numbers of the following lines, as the transfer rate is still too fast for BASIC. Try to keep track of those lines and list them separately later. Erase these lines below 100. They will be spurious copies of higher numbers.

It is a very slow and tedious job, as well as being not very rewarding, so make sure the program is worth the effort.

From a Christmas wish list. How about the three upgrades from Mr. Hogg on a single new PC board? Their total price is about $99 and it seems that the three together could be made for no and 1/2. I have McVideo and very much like it, but I am not sure I trust the piggyback aspect. I rather think that a 2-88 rather than a 2-48 should do the job. Does PROTEUS own the copyright for the SOL pdf? Is it in the public domain? Such a mod would probably void the copyright in any case. It would be very extensive.

Keep up the good work. Thanks for the last four years.

PS. VISTA seems to have sold the rights to the Mac controller to a Mr. Byron Wagner, Genius Co., 1-213-462-1426.

PPS. What about the cassette library? I have been unable to contact Lou Moseley. I now have an 8 inch disk drive, and I was counting on Leuv for format conversion to VISTA format. If that no longer exists, I would now be able to download to cassette using TFABDISP, or perhaps in VISTA V-200 manual, if no one else is willing or able.

[Editorial remarks:

The Graphic Add was made by K.E.A. Mexico Designs in Canada. Not much market for it now. It converts some of the inverse video symbols into 2 pixels per character graphics, leaving the manual into Encyclopedia Processor Technique Volume 2.

About the piggybacks versus a new SOL/PC, Bob has mentioned long range plans to make a new PC board, but it will probably be a 16/8 CPU. See my thoughts about SOL upgrades in this issue. The piggybacks let you add some reasonable new features to SOL, selecting just those you want. Possibly does not own any SOL hardware rights, this anyone heard from Lou Moseley?]

---

ON USING COPRO 805/8088 CPU BOARD TO UPGRADE SOL

Dear Stan,

10/18/83

Your articles on the IEEE 486 standard in PROTEUS prompted me to try my CoproPro 805/88 CPU board in the SOL. It works fine. Of course there is a modification required to the SOL, but no permanent changes are required. The modification involves inserting seven ICs, lifting several pins on two ICs, and adding two jumpers. The ICs are easily be made without removing the SOL motherboard. In fact — excluding the time to make the jumpers — the SOL can be moded or demod'd in about fifteen minutes.

Based on your article, I realized that the SOL is close enough to being IEEE 486 compatible that any IEEE 486 CPU board would work in the SOL's backplane if the SOL's 8038 CPU is disabled. And that is all my modification accomplishes. I remove the 8080 and its associated bus drivers, jumper PBDR1 to U48 to disable the data line drivers except for data inputs from the SOL's internal bus, jumper the keyboard reset to the S-100 reset, and add one more jumper for housekeeping. That is it all takes!

The 8085/88 board has a power-on jump option which can make the 8085 jump to COMO on power-on or reset so the SOL's four phase motion isn't needed. Another option allows selection of a 2 MHz clock for the 8085 so the old PT dynamic memory boards can be used — at least for check out when you are just trying to make the 8085 work. Since I am currently running with a PT 64K board, I have only checked out the operation of the 8085. The 8085 uses a 5 MHz clock. There are only two possibilities; change the 8085 clock to 2 MHz by changing a crystal on the 8085/88 board or get faster memory and access the display plexus memory only with the 8085. I am going to try the crystal change first and go for the faster memory later.

I am using the IDE 105 disk drive. I have also shown that DMA without arbitration works ok.

I think the CoproPro 805/88 CPU board is the next logical upgrade to the SOL. As Godbaut says, "The CPU 805/88 Dual Processor board was specifically designed to make it easy for the S-100 Bus user to get into the world of 16 bit micro, while at the same time preserving compatibility with existing hardware and software..."

"The user may switch back and forth between the two processors with a simple software command. For example, this allows the use to let the SOL run on a currently available and familiar disk operating system while letting the 8088 run the more advanced applications software..."

To unify the SOL; remove ICs U30, U63, U67, U68, U94, U105 and U107, lift pins 3, 5, 7, and 9 on U63; lift pin 10 on U67; add a jumper from pin 5 of 850 socket to pin 10 of U68; add a jumper to pin 5 of U68 must be lifted from the socket and the jumper must also be lifted to the jumper attached to the lifted pin. The easiest way to do this is to solder the jumper to an Augat pin and plug pin 10 of U68 into the Augat place. I soldered pin 10 from the Augat pin which is KST jack K3 to pin 14 of U63 socket (KST jack K3 is less to the right and down from 653); add a jumper from pin 17 to pin 23 of U63. I soldered pin 17 from a dead IC to the jumpers for all jumper ends which need to be plugged into sockets.

To setup the 805/88 CPU board: DIP switch 1, 1-5 off, 6-8 on; DIP switch 2, 1-4 off, 7-8 on; DIP switch 3, 1-5 off, 6-8 on. DIP switch 3 sets the port address for memory mapping for any other port address in the system.
I think this modification is a good start toward making the SOL completely IEEE 696 compatible while bringing the board into the 16 bit world. Additionally, this change opens several options for the SOL motherboard to be used for extended I/O and memory addressing. One problem with this change is that the SOL's pullup resistors are left on the S-100 bus. These resistors make the SOL addressable, but they can be disabled by putting 5.5v in the trace to either side of the cut to permit reconfiguration or simply adding jumpers between the pins. It would be nice to see an enhancement to the SOL implemented in a similar way so that reconfiguration will always remain visible. As time permits, I'll continue to work on the other changes required to bring the SOLS into full IEEE 696 compatibility.

Over the years I've gotten a lot more from PROTEUS than I have contributed. Each time I write a letter to PROTEUS I again understand the great effort that you have put into PROTEUS. Your program service to PROTEUS is greatly appreciated.

Sincerely,
Ed Botto
4255 Moore St.
Los Angeles, CA 90066
213-366-7788

Jan 12 1983

--- A SOL KEYPAD MODIFICATION

Dear Sirs:

I have expanded my memory space to 60K using the Motorola 1200 chip and it works well. I'm sure that much of the hardware and software that I've done is available in some other places, but I'd like to share some of the details of what I've done.

Speaking of memory, I've only gotten two 8K boards to fully work on the SOL. I'm thinking of buying a third one, but I'm not sure if it will work on the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.

I've also been working on the keyboard, trying to make it compatible with the SOL. I've added a couple of switches to make it work with the SOL. I've also added a few extra switches to make it work with the SOL.
DEAR PROTEUS,

I WISH TO ADD A DISK DRIVE AND A PROGRAMMABLE CHARACTER GENERATOR TO MY 8KB SOL SYSTEM FROM YOUR EXTENSIVE RESOURCE POOL, WHICH ARE THE MOST POPULAR/BEST/EASIEST INTERFACED DEVICES AVAILABLE. MICROPOLIS SEEMS TO BE ABLE TO TAILOR THE DESIGN TO SOl'S SPECIFIC NEEDS. OBJECTIVE DESIGNS Pcg IS THE ONLY ONE I'VE SEEN THAT MAKES DISKS WITH DISKS IN YOUR SYSTEM. PLEASE COMMENT.

C. Jason Slade
32 Berkeley Cres.
Sincere, Ontario N3Y 2K4
Canada

[Editor's reply: For 5-1/4" disk, I recommend the MicroComplex disk controller. This is a flexible, reliable, and easy-to-use controller. The controller is designed to interface directly with most disk drives and systems. It provides a standard interface that allows for easy integration into existing systems. The controller is compatible with a variety of disk drives, including high-capacity drives, and is designed for use with East Coast Solid-state systems. Please contact MicroComplex for more information.]

DEAR IMP, I FELT FOR THE FIRST TIME IN YEARS, I WISH TO SEE YOU AGAIN. I HOPE TO BE ABLE TO JOIN YOU SOON.

J. E. Minor
32 Berkeley Cres.
Sincere, Ontario N3Y 2K4
Canada

[Editor's reply: I am pleased to hear that you wish to see me again. I look forward to our meeting soon.]


taste the networking

James E. Minor
32 Berkeley Cres.
Sincere, Ontario N3Y 2K4

[Editor's reply: I am happy to hear that networking is becoming more popular. Please let me know if you need any assistance with networking or hardware design.]

DEAR EMILE,

I THINK THAT I WOULD BE HELPFUL TO ME, COULDN'T, AND MY DIMENSIONS WERE NOT THAT EXACT. STILL IT WORKS.

...ON DECK KEYS

EMILE ROTH
1011 EVELYN TERRACE EAST, #104
SUNNYVALE, CA 94086

...ON NETWORKING

James E. Minor
32 Berkeley Cres.
Sincere, Ontario N3Y 2K4

DEAR EMILE,

I NOTE IN A BACK ISSUE OF PROTEUS THAT YOU WERE TROUBLED WITH A GOOGLE KEYS ON YOUR SOL. PROBABLY YOU HAD RELOCATED A TROUBLE. IF NOT, MY EXPERIENCE SHOWS IT IS PROBABLE TO REPLACE THE FORMER.

...ON IMP KEYS OCTOBER 8, 1983

EMILE ROTH
1011 EVELYN TERRACE EAST, #104
SUNNYVALE, CA 94086

THANK YOU FOR ALL THE BACK ISSUES, WHICH I AM STUDYING AND WRITING LETTERS, WHEREVER I THINK I CAN HELP. ENCLOSURES ARE COPIES OF SEVERAL YOU MAY WISH TO PUBLISH.

IN YOUR MAY/JUNE 1983 ISSUE IS A QUERY BY DANIEL HENDRICKS OF IRVINE, CA AND YOU ABOUT IMP KEYS.

VOLUME 34 OF THE CP/M USER IS DEVOTED TO IMP KEYS, AND HAS VERSION 3.16 WHICH I STUDIED FOR ABOUT A WEEK. IT ALSO HAS PROGRAMS AND INSTRUCTIVE MATERIALS. CREATIVE COMPUTING, MAY/JUNE OF 1983 ALSO HAS AN ARTICLE STARTING ON PAGE 30. DR. DOBB'S ALSO HAS MATERIAL ON IMP KEYS IN VOLUME 3, ISSUE 3, PAGE 446; VOLUME 3, ISSUE 4, PAGE 191; AND THE MAIN DOCUMENTATION IN VOLUME 3, ISSUE 2, PAGE 10. ALSO AVAILABLE FOR #24 IN A WELL WRITTEN HANDBOOK FROM IMP76 INC. BOX 257 RRI PENNINGTON, N.J. 08071.

I TOLD THE LANGUAGE PRETTY WELL, AND ABOUT 90% OF THE FUNCTIONS I TRIED WORKED. HOWEVER, I WAS NOT ABLE TO TELL THE PRINTING, ALTHOUGH THE BOOK SAYS THIS IS POSSIBLE BY CHANGING A CONTROL TO LIST. IT IS POSSIBLE THAT THE VERSION I HAVE FROM CP/M NO 3.04 IS NOT THE LATEST.

...ON THE SOL PENCIL PROGRAM OCTOBER 9, 1983

BOB JOHNSON
55 MONARCH DR. A-5
BOULDER, COLORADO 80303

DEAR BOB:

I SEE YOUR MAY 6, 1993 LETTER TO PROTEUS ASKING ABOUT HELP IN GETTING YOUR VERSION OF PENCIL TO PASS CONTROL CHARACTERS TO YOUR CP/M IBM.

I HAVE A SOL PENCIL CASSETTE PROGRAM, WHICH I CHANGED TO USE WITH MY CP/M DISKS. I WANT TO DISASSMINGLE MY VERSION AND UNDERSTAND IT PRETTY WELL. ALL 26 CONTROL - ALPHABET CHARACTERS ARE USED BY IT FOR PENCIL FUNCTIONS. THIS WAY YOU CAN TELL WHICH CHARACTERS ARE INTERPRETED AS PENCIL FUNCTION SPECIFICATIONS. FURTHER PENCIL PLACES YOUR TEXT IN A BUFFER, AND MANIPULATES THIS AS NEEDED TO PROPERLY APPEAR ON YOUR VIEWER. I BELIEVE THE LOGIC DIVIDING UP THE SENTENCES COULD NOT MATION CONTROLS, AS EVERYTHING IS HARDCODE. THE PROOF IS THAT WHEN IN PENCIL YOU ARE NOT ABLE TO SEND A CONTROL CHAR TO THE SCREEN. IF WE CAN DO THIS, IT WOULD BE IN THE BUFFER. I THINK THE ALPHABET, AND SINCE I USE WORDSTAR, WHICH IS EASY TO INTERFACE WITH THE PRINTER CONTROLS, I DON'T MISS IT. THE ONLY THING I CAN THINK OF IS TO SACRIFICE ONE PENCIL FUNCTION AND USE IT TO SET THE PRINTER. THE OTHER THING WOULD BE TO TAKE SOME OF THE CONTROL CHARACTERS LIKE "C", "D", "E" AND "F" AS PRINTER CONTROLS, THEN PLACE A FILTER ON YOUR OUTPUT ROUTE, AND INTERCEPT WITH THE PRINTER FUNCTION.

HOPE THESE MUSINGS HELP. I AM RUNNING TWO THINKERTOY DISKS WITH 4KB RAM, AND LOTS OF PROGRAMS. IF YOU HAVE CP/M, WOULD BE HAPPY TO SHARE WITH YOU.

BERNARD PLOTHK
5245 N. BAY RD.
MIAMI BEACH, FL 33140
TABLE OF CONTENTS

SOL AND THE IEEE 596 STANDARD—PART 2: UPGRAADING THE SOL: COMPLIANCE OF MEMORY AND PERIPHERALS By Stan Sokolow .............................................. 1
MEET ADA, THE NEW PROGRAMMING LANGUAGE by Stan Sokolow ............................... 1
CLARIFICATION OF THE DUAL PERSONALITY MODULE by Stan Sokolow .................. 2
INTERESTING ADVERTISING WE'VE RECEIVED by Stan Sokolow .................................. 2
SOL'S CO-DESIGNER ON COVER OF INFO WORLD ............................................ 3
THIS 'N THAT BOARD by Neil Switcliffe .............................................................. 9
OFFICE SYSTEMS RESEARCH ASSOCIATION CONFERENCE ................................. 14
SAY, WHAT'S ROB MARCH UP TO THESE DAYS? .............................................. 15
PIT BASIC RISES AGAIN AS "NEVADA BASIC" ............................................. 15
ELLIS COMPUTING, INC., Press Releases ............................................................ 15

LETTERS TO THE EDITOR:
...ON A DATA BASE MANAGER AND ELECTRONIC SPREAD SHEET by Roy Hey brokerage ........................................... 15
...ON AMATEUR RADIO SOFTWARE FOR SOL by Francis Blaslotto .................... 15
...ON PC BOARD DOCUMENTATION by Ed Rainsberger .................................. 16
...ON DISK DRIVERS OTHER THAN HELION by Harold Lyons ......................... 16
...ON QUESTIONS ABOUT DISK CONTROLLERS by Jason Slade ...................... 16
...ON A REAL TIME CLOCK INTERFACE by Bruce Hull .................................. 16
...ON CBR FILES WITH Q/F BASIC by John Barber ...................................... 17
...ON USING COMPUTED BASES/8088 CPU BOARD TO UPGRADE SOL by Ed Belton .................................................. 17
...ON A SOL KEYPAD MODIFICATION by Greg Heli ...................................... 18
...ON SANTAG by Bernard Plotkin ................................................................. 19
...ON THE SOL PENCIL PROGRAM by Bernard Plotkin .................................. 19
...ON DEAD KEYS by Bernard Plotkin ............................................................. 19
...ON NETWORKING by James McElroy ........................................................... 19
...ON DISK CONTROLLERS AVAILABLE by Jason Slade ................................ 19

NOTICE TO MEMBERS

VISA and Master Charge for Proteus Orders will no longer be available after 1983. Newsletter format may also change to 8 1/2 x 11 xerox copy. These changes are due to financial and staffing difficulties. However, be assured that Proteus will remain a source of materials for and a vital communication link between all interested SOL owners.

20

PROTEUS/NEWS

A news journal for owners and users of Processor Technology Corporation computer equipment. Published by Proteus, 1690 Woodside Road, Suite 219, Redwood City, California 94061-3483, USA, telephone (415) 368-2300.

Submit items for publication to Proteus News, Attn: Stan Sokolow, 1690 Woodside Road, Suite 219, Redwood City, California 94061-3483, USA. Please make submissions as camera-ready as possible by using a fresh, black ribbon and typing single-spaced.

Copyright (C) 1982 by Proteus. All rights reserved. Permission is hereby granted to reproduce any computer programs contained herein, provided that Proteus and the program's author are given credit.

FROM:
PROTEUS
1690 WOODSIDE ROAD, SUITE 219
REDWOOD CITY, CALIFORNIA 94061-3483
USA

Z-80 CONVERSION "MCZOL" IS IN PRODUCTION

Proteus has received the first Z-80 module for upgrading the SOL without using any slots. Bob Rocq once again has made a superb engineering design for retrofitting the SOL-PC by plug-backing. It works in conjunction with the Z-80B upgrade (MCVideo) and dual personality module (MC/DM).

I'll be testing it in my SOL and will write an article on it for the next newsletter. Due to the lateness of this issue, the next issue of Proteus News will be shorter to expedite getting the news out on the Z-80 module.

James D. McElroy
2826 Crest Ave. North
Allentown, PA
18104

PRIORITY MAIL
37
EDITOR'S MESSAGE

This is the new look for Proteus News. To help reduce the production cost as our readership slowly declines, I've changed to a direct printing method that doesn't require the more costly photo-reduction. This will simplify production of the newsletter.

CRYSTAL BALL GAZING

UNIX IN THE NEWS

It's always fun to try to guess what's happening to microcomputing in the future. Here are some of the recent visions that came in on my crystal ball.

Digital Research Incorporated (DRI), the system software company that brought you the CP/M family of operating systems, has announced that it is transporting (otherwise known as "porting") the UNIX System V operating system onto the Intel 286 microprocessor under an agreement between Intel and DRI.

UNIX is the popular operating system developed for internal use by AT&T employees at the Bell Labs facility, but it became the joy of many college students because AT&T was allowed to distribute it to universities. Since AT&T is deregulated, the new AT&T Technologies company that grew out of Western Electric is gearing up for competition with IBM in the computer market. Naturally, it is to AT&T's advantage to promote UNIX as a standard operating system for the 16 and 32-bit microprocessors.

A DRI newsletter for Independent Software Vendors (ISV Forum Vol 4, No 1) headlines the idea that UNIX System V is to be promoted as a standard. DRI and AT&T have agreed to develop a UNIX System V Applications Library to be marketed through DRI's retail outlets.

DRI has already released versions of its CP/M compilers that run on UNIX. This allows ISV's to recompile their applications on a UNIX machine and create an instant new market. DRI is counting on the thousands of applications it fostered in the CP/M world to help it build a UNIX library quickly.

What about IBM in all of this. It is well known that IBM owns a substantial chunk of Intel and has a director seat on the Intel board. IBM chose Intel as a rapid way to pick up microprocessor technology for its PC products. It is rumored that IBM is developing a small business multi-user computer based upon the Intel 286 microprocessor. Will it use UNIX?

Based upon prior history with the IBM-PC and the new wrinkle of AT&T competition, my crystal ball has been receiving visions that say the IBM multi-user microcomputer will offer UNIX as an extra-cost option, but a unique operating system developed by or for IBM will be the standard OS with the machine. It may even be an enhanced version of the Microsoft IBM PC-DOS.

I can't see IBM giving support to AT&T by dependence upon the UNIX operating system. IBM will want to avoid paying large royalties and being at the mercy of a competitor.

Meanwhile, DRI will try to cover all bases by continuing to extend its family of operating systems to other processors and to provide new features, such as windowing. It is also porting its compilers to competitors OS's, specifically MS-DOS and PC-DOS.

But, my guess is that DRI really is hoping to be the prime vendor for the AT&T world, since it lost the battle for IBM to Microsoft.

ALIGN YOUR OWN DISK DRIVES

by Stan Sokolow

I own six Sol/Heinos systems, all but one of which I picked up at reasonable prices after PTC went out of business. These have served me faithfully for word-processing and accounts receivable mostly, with perhaps one breakdown a year among the six.

But the fellow who has been doing my disk repairs has gotten progressively busier himself, and the turnaround ground to a halt. Over a several month period, five of the systems went out of service, leaving me on the brink of disaster. (Well, not disaster, because I have carefully planned a fail-safe manual system to fall back upon in the case that none of the machines worked. But, it sure would be inconvenient.)
I tried a few other disk repair places, but most of them were not able to repair the PerSci drives used in my Helios systems. I found a place that had the equipment and know-how (they said), but when I sent my drive, they let it sit on the shelf for several weeks and finally returned it un repaired -- they said their PerSci exerciser was broken.

I discovered that the remains of PerSci, Inc. is a skeletal crew of the former manufacturer. They will service drives, but their minimum rate is $400 per drive! With five drives out, that was a minimum of $2000.

Well, this was enough motivation for me to get off of my duff and see if I could do the drive repairs myself. I have the PIC technical manual (Encyclopedia Processor Technica volume 9) which describes the alignment procedure and gives the schematics and some repair guidelines. I rented an appropriate oscilloscope and sat down to try it.

I'm pleased to report that even mere mortals can learn how to align a PerSci drive! I have worked on them in spare time on two weekends and have 3 of the 5 back in service. The 4th has a problem which I've zeroed in on and expect to fix this weekend. The 5th is the hardest because it needs new bearings,

I want to give you the results of my experience.

First, let me shatter a myth. Not one of the five drives was "out of alignment". I mean that I always thought the term "alignment" meant whether the head was properly lined up in the standard location on the diskette. In the jargon of the disk drive world, this is called radial alignment and azimuth. None of the drives had this problem.

What was wrong then? A variety of little things broke down. One drive wouldn't seek -- its positioner lamp had burned out. One got persistent ABORT errors when testing with DISKT test program -- it had a bad TTL chip in the data separator board within the drive. Another also got persistent ABORT errors -- it has a bad TTL chip on the data & interface board in the drive. Another has a bad power supply in the Helios box (the +5 volt regulated supply is giving only +3.2 volts). And the fifth one has the bad spindle bearings and the positioner mechanism whistles (oscillates).

Now, the positioner oscillation is often due to a weak positioner lamp, and this can be adjusted by twisting a screw on a potentiometer in the drive (unless the lamp has gone too far). But other than that, all of the other problems were just worn out electronic components, not "alignments". They simply needed repair.

In doing the repairs, though, it is necessary to check and readjust all of the alignment steps. In fact, that's how I discovered the parts that were bad. I couldn't adjust the signal to meet the specification, or there was no signal at all.

In the greater sense, alignment also means adjustment of the electronics to meet operating specifications, so in this sense many drive problems may be solved by "alignment". As components age, their characteristics change a little, and adjustment of the circuit may bring the drive back into working condition.

However, eventually something is going to break down. So, think of repair rather than just alignment or adjustment. If you have a disk drive made by a company still in operation, you're in luck. If your drive is an orphan, you may be faced with the choice of paying through the nose, junking the drive, or treating it as a learning experience as I did.

I also discovered another bonus in doing it myself. I suspect that most repair facilities simply get the drive back into operation and check only the most common alignment specifications. On my drives, I found several with the index photo-detector out of specification. This did not affect their passing of all the other tests.

I have had a nagging problem with the drives over the past several years which has only happened once a year, but which was devastating when it hit. I now suspect that the index timing was the culprit in this bug (which resulted in the drive wiping out the first header on a track when it wrote a long sector at the end of the track). So, if you want to be sure it is done thoroughly, I guess you have to do it yourself.

What do you need to be able to do alignments of the PerSci drives? (I suppose alignment of the other brands of drives would be similar but easier because they use simple stepper motor positioning rather than the faster, more complicated, voice-coil positioner of the PerSci.) You need a dual trace oscilloscope with at least 25 MHz bandwidth, external trigger, and the ability to display the difference between the two channels (add with inverting channel B). You need a Dyan 240A alignment diskette. You need a few small hand tools like screwdrivers, plastic bladed screwdriver, allen wrenches, etc. You need the SIMU-ciser program written by Processor Tech, or a similar program you write that will step the drive to whatever track you want or automatically step at full speed between two designated tracks. And you need the manual I mentioned.
You also need to understand how to read the schematics if you are going to have any hope of doing repairs, rather than just adjustments. I have no engineering background, but I’ve picked up enough electronics understanding over the years since I bought my computer by reading things like the Don Lancaster TTL Cookbook, the Radio Shack "Understanding Solid-State Electronics" book written by Texas Instruments, Byte magazine, the Sol manual, Proteus News, etc.

I rented my scope from Continental Resources, which has home office at 175 Middlesex Turnpike, Bedford, MA 01730 (telephone 617-275-0850). They have a branch in Santa Clara, which is where I got mine. You probably won't find a scope with exactly those specifications, so treat them as minimum requirements. Look for a rental company under "Electronic Instruments -- Rental" in the yellow pages directory.

The cheapest scope that was on hand when I called to rent was a Tektronix 2215, which is 60 MHz dual trace and has other fancy features, but it does the job. It rents for $100 per month, but I’ve rented a similar but simpler scope for $80 before. You should ask to rent a third probe, which you’ll need for the trigger signal.

Now, some of you may have heard about the Dysan Digital Diagnostic Diskette (DDD). This is a device that lets you check alignment of the drive without an oscilloscope. There was a good article on it in Dr. Dobb's Journal, December 1983. Why didn't I use it?

As I mentioned, my drive problems turned out not to be "alignment" problems, but rather electronic problems for the most part. The DDD is of no help in trouble-shooting a faulty circuit. Also, the PerSci positioner mechanism is different from that in other drives. To adjust it and check its performance, it is necessary to look at certain voltages at test points on the board. But also, the DDD is designed to be read by a standard soft-sector disk controller, which the Helios is definitely not. (But the main reason I didn't use the DDD was that I wanted to do it by the book since I was a novice. This turned out to be a wise decision.)

FOR SALE:

SOL 20, (Rev.E), with manuals and original box-$325+shipping; Central Data Corp. 32K board (expandable to 64K with 4116's, 2117's, etc.) with manual-$60+shipping; GPM board with ALS-9, SIM-1 & TXT-2, all manuals and updates-$35+shipping. All the above for $375+shipping. All equipment above in very good condition, hardly used, and just kept as a backup system for my main SOL systems: Curt Kobylarz-Schmidt 1710 N. Wisconsin Ave. Peoria, IL 61603 (309)692-5258

WANTED:

To buy old SOLs for parts. No memory boards, add-on components or accessories wanted. Must be in working condition, especially keyboards and backplanes. Reply with price, including packaging and delivery.

Please call or write to: KUZIAK CONSULTING, 100 Connaught Crescent, Regina, Saskatchewan, S4T 6M9, Canada, 306-527-0154. 9 a.m. to 12 a.m., 1 p.m. to 5
9 February 84

PROTEUS
1690 Woodside Road, Suite 219
Redwood City, CA 94061-3483

Dear Stan,

It's been a while since I've written about my ongoing experiences with my SOL, so I thought this would be a good time to catch up. I have enclosed a check for $30 to cover my 1984 subscription. I thought I renewed for 1983, but only received one issue (#1). Could you check your records, and mail any back issues—thanks.

I have a couple of new products that I would like to report on. The first is a low-priced daisy wheel printer. The one I'm referring to is the DWP-210 from Radio Shack. Although, I've never been a fan of Radio Shack computer equipment, the past few years have seen dramatic improvements in their products. One such product is the DWP-210.

The DWP-210 is a full daisy-wheel printer with all of the niceties you would expect in a printer costing twice as much. Although it is not as fast as the high priced units, the quality of the print is superior. That was one feature that I was looking for when I went shopping for my daisy-wheel. The impression of the characters is far more important than the speed. The unit is solid—weighing in at 38 pounds. This should tell you something about the quality of construction of the 210.

The printer types at 20 cps average, although that figure can change depending on the type of document you are printing. I have found that it is considerably slower when printing a file that is all CAPS. However, for a normal business letter the speed is very satisfactory. The pitch is both hardware and software controllable. The pitches available are 10, 12 an PS(Proportional Spacing). A switch on the top panel of the printer selects between the three. There is also an On-line/Off line switch.

The 210 uses standard Diablo ribbons, both the fabric, and multistrike. Ribbons can also be purchased at any Radio Shack Computer Center, and selected office supply stores. There are two interfaces that come standard with the unit. An RS-232, and the standard Centronics type parallel port. I'm currently using the parallel port—so I had to customize my own cable—which was not that big of a deal.

There are three warning sensors on the printer. The first is a "paper out", the second "replace ribbon", and the third indicates that the top cover is open. When turned on the printer initiates a self-test routine that checks out the printer mechanism, carriage, etc. This all takes about 3 seconds. There are also some DIP
switch selectable options such as BAUD rate (600/1200) for the serial interface, self test, logic seek, serial/parallel interface, etc. The switch is located under a small sliding door on the top of the printer in the back. This allows easy access, though generally it is not used once the initial settings have been selected.

Another real nice feature of the printer is that it is bidirectional. This makes the total throughput somewhat more efficient. Certainly a bonus for a printer in this price range. The only option available at this writing is a bidirectional tractor feed. The printer is priced at $799 and the tractor feed is $150. One irritating feature that is lacking is the means to turn off the CR/LF sequence mechanically. This can done under software control though. I have successfully patched both Wordstar, and Electric Pencil to not issue an extra line feed. I also patched Wordstar to recognize all of the features of the printer - boldface, overstrike, underscore, etc.

I am very satisfied with this new addition to my computer peripherals. It has worked beautifully for over 9 months without any problems. Radio Shack products do deserve a second look.

Another product that SOL owners will be interested in is the RAM disk from Digital Research Computers of Garland, Texas. The Light Speed Disk Simulator or LS-100 is a 256k byte RAM board (S-100) that emulates a disk drive. The software uses CP/M as the operating system, although other disk system may well be patched to operate it as well.

The LS-100 board uses the new 4164 64k x 8 bit DRAMs(Dynamic) chips. It also uses a new Dynamic Ram controller chip that is manufactured by Intel, the 8203-1. This eliminates a lot of the timing problems previously associated with dynamic RAM boards. It has been working fine in my highly modified SOL for the past 6 months.

The LS-100 is available one of two ways: (1) the complete kit of all parts, including the S-100 board and software on an 8" IBM format disk, or (2) a bare board with the same software. The complete kit is $399, while the bare board is $69.95. I opted for the bare board, and managed to save about $100 over the kit price. However, there is always a tradeoff in order to save money. If you decide to build the board from scratch the Intel controller will be hard to find. It is also the single most expensive component - expect to pay around $45. I finally found one through my local Wyle Labs distributor.

Once the board is operational, the software must be properly patched to your CP/M system. This was not as difficult as first thought. More specifically, a few EQUATES need to be changed in the supplied software to that of your CP/M size, etc. After the software has been patched and reassembled, following the routines described in the manual the board is ready to go.

There are three basic programs used with the board. The first is a diagnostic(DIAG) program that checks the functionality of the board
and also performs a memory test of the chips. Once executed, the program will display a memory map indicating a good(G), or bad(B) memory chip. This need only be run if you suspect a failing memory chip or get a checksum error during install.

The other two programs must be run before the board can be used. The first program is a format routine(FMAT) that formats the board, just as you would a regular disk. It takes about 3 seconds to format the board. The second program is the install routine(INSTALL) that actually allows CP/M to access the LS-100. When this is run a signon message will appear noting that the LS-100 drive is active. At that time files can be PIPed over and that board accessed as a regular disk drive.

I have used several word processors, spelling checkers, and assemblers with it and the performance is really - unbelievable. To give some timing comparisons I used a 600 unique word Electric Pencil file, and checked the spelling using The Word Plus. With a regular 5 1/4" floppy, the time was 2.5 minutes. With the LS-100 the time was reduced to 19 seconds. That is a dramatic improvement!

I am very pleased with the overall performance the LS-100 has added to my system. It even has battery backup capabilities that would allow you to always be on-line. In fact as many as eight of these boards can be installed in the same system giving you a total of over 2M bytes of on-line storage. I wonder if the new 256K x 8 bit chips can be substituted?

Well, I'd better stop and give someone else room in the newsletter. Thanks for your continued efforts on the newsletter. I'm still traveling quit a bit so I don't spend as much time as I would like at the keyboard. However, I plan to write more about other applications I'm working with that may be of interest to other SOL users.

With all the new computers on the market today, and the proliferation of PC compatible machines, sometimes I feel left out. Everything I know about computers, is because I gambled on PT way back in 1975. I guess we're a dying breed - now that the little MAC is out.

Regards,

Rick

Rick Downs
Denver Colorado
(303) 750-1838 evenings
Dear Stan,

I have been using a new Basic called Basic/z written by Bob Zale of Systemation. This has to be close to the ultimate Basic in capability, speed of execution, and ease of use. There are 232 key or reserved words in the current version. Which is 1.11

Among one of the more annoying problems that I have had in MBasic is the my cursor keys are not usable in editing, cursor positioning is not as direct or as easy as it was with some dedicated Basics, reverse video and other terminal characteristics require the use of CHR$ characters which are terminal dependent. Basic/z has commands to clear screen, formfeed printed pages. It is able to determine where the cursor or print head is at any time so that you can write code that if the cursor is at a particular location to take a specific action or if a procedure is interrupted to return, after an action, to the original location. You may through the use of UINASCE statement force all entry to be in upper case. Basic/z supports both CHAIN and COMMON. It does not require, as does MBasic, the use of LPRINT with each line to direct output to the printer. It allows a simple statement, LPRINTER, to direct the output to the printer, while CONSOLE sends output to the screen. This is very similar to the SET OF = statement in PTEBasic.

Basic/z has a number of very interesting features, among them is that it is an interactive compiler, that is supports two different modes of execution. First, in the testing mode, you compile and immediately execute the program. When you have the final version you may execute a one step BND command which will link the the object program with the needed support routine into an executable command (.COM) file. For maximum efficiency and security, all generated code is in the form of native machine instructions.

Another feature is that each line tested for correct syntax as the line is finished. If there is an error the editor points to the position of the syntax error. There are seventeen local edit commands that allow movement anywhere within the edit line. The edit commands, all of which are invoked by one key, include 
advance, back-space, tab, abort, append, change, delete, insert, jump, kill, zap, replace, list, quit, search and move (go directly to the error in the new line).

Basic/z has program definable precision in decimal (BEC) math from 6 to 18 digits.

Some of the more significant keywords are DEEBUG, EDITS, FMT, LABEL, LINK, SORT, SPOOL, DEBUG - To select various debugging options.

```
DEBUG "L", *, *
```

Used in conjunction with the &D option at compile time would specify entire program and would list the line where the first error occurred. Other options are Pause, Trace, Single-step.

EDITS - Allows user editing of a string value, may also be used for input and will place a default value in string.

```
text$ = EDITS(text$,20)
```

Will print old or default value of texts, will accept correction or new entry of up to twenty characters and

7
will then change value to new entry. During editing you may use cursor keys for position, insert, delete, and for special actions such as help.

**FMT** - return a formatted string from a numeric value

```
telephone$ = FMT(telephone, tel.mask$)
```

when

```
tel.mask$ = "(999) 999-9999"
telephone = 3124818085
```

Will output telephone$ = (312) 481-8085

**LABEL** - To create a symbolic destination

**LABEL @print.routine**

Used in conjunction with GOTO, GOSUB, RESTORE and the like. By use of descriptive labels and variable names it almost possible to eliminate use of REM statements. Basic/z will recognize any length variable name and all characters are significant. The keyword LABEL is not necessary and @with.any.name may be used instead.

**LINK** - To load and transfer control to an object program.

```
LINK "A:BZ.COM"
```

Will revoke the Basic/z compiler

**SORT** - To sort an array

```
SORT Text.element$(0), Text.element$(1024), "DU", 12
```

Will sort the first 1025 elements of the array Text.element in descending order forcing the characters into upper case for comparison only. Only the first 12 characters will be compared.

**SPOOL** - To direct printed output to a file

```
SPOOL 19
```

All output from the LPRINTER statement will be sent to file 19.

**Some other keywords are:**

**ECHO** - Print on both screen and printer.

**INCR** - To increment a numeric variable.

**INCLUDE** - Allows additional source program to be included in the compilation.

**LOWCASE$** - return a string in which all upper case is changed to lower case.

**SELECT** - To select a default drive.

**PAGESIZE** - To allow program control of printer page size.

Also are supported are **WHILE - WEND, DO - UNTIL, IF - THEN - ELSE**

**INCHAR$** - permits recognition of a single character.

**INKEY$** - will allow a specific number of keystrokes. Which will not be echo on the screen.

The only slight deficiency is the lack of detail examples and suggestions on using some of the more complex keywords.

Basic/z is available from: System/z, Inc.
P.O. Box 11
Richton Park, IL 60471

Sincerely;

David I. Dalva II
Dear Stan,

I have received Bob Hogg's new Z-80 update for the Sol and have been using it for the last month. It is completely transparent to the user except for two important features. First, you can run Z-80 software, which seems to be replacing 8080 for machine dependant code. Second, the processor can be run at either 5 mhz or 6 mhz instead of 2 mhz. At this speed the Sol is a absolute joy, as the screen updates almost instantaneously, additionally you can change the processor speed under software control. Also there are available, as an option, non-maskable interrupts to enable single stepping of software under development. There are some necessary mods to be made to the Sol mother board in order to achieve this speed. I sent my mother board to Bob so that he could do both the modifications and testing as some of the chips are critical at the higher speeds.

It is also possible to modify the controller boards for the Helios to work at the higher speeds. At 5 mhz the Helios is very fast. So far, I have not experienced any difficulty in using my old Helios software. Also I have also obtained from Bob a new keyboard enclosure, with a extension cable, which allows me to place the Sol itself out of the way and just have the monitor and keyboard on my desk.

Bob has also supplied me with a 15 meg hard disk which I use in conjunction with his Northstar compatible 5.25 inch drives. After seven months operation I am convinced that this is the only way to go. There is a noticeable speed up on disk access and disk-intensive compilation are noticeable quicker. I have just about completely switched over to CP/M and use it for almost all my work. The big problem is that many of my original programs were written in PT Disk Business Basic which allowed both CHAIN and COMMON.

Now, there is a PTBasic for CP/M is available from Ellis Computing. At this time they have only implemented the standard PT Disk Basic but have promises to add the CHAIN and COMMON features of the Business Basic. To convert programs you have to save them in ASCII, move them to CP/M and then run under Nevada Basic which is Ellis name for PT Basic. Ellis has also included a full screen editor which is invoked when you generate an error. Nevada Basic also uses the same abbreviation that PT Basic used. I have converted a few programs and have found no significant problems, but I am waiting for the new version, as most of my programs were written in Business Basic to take advantage of its CHAIN feature. Nevada Basic is available from Ellis Computing, 3917 Noriega Street, San Francisco, CA 94122, (415) 753-0186. The price is $39.95. They have also converted PT Fortran, PT Pilot and PT Edit to run under CP/M. The Fortran is the same implementation that was used by PT and is written by the same author. I have used both Nevada Fortran and Microsoft Fortran and found that there was no sufficient advantage to the Microsoft over Nevada.
I have modified WordStar to use the various function keys on the Sol keyboard.

The modifications are implemented as follows:

In version 3.3 of Wordstar, the Installation Menu offers you the following choices 'A,B,C,D,E,F or X'. If you enter a '+' instead you will enter the Custom Modification routine. You may make changes by either entering the Wordstar symbol proceeding by ':' for example ':HITE'

In earlier version you can use DDT to effect the changes. The changes are as follows. They presume an 80 by 24 screen and Bob Hogg's operating system and the Solus at F000.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Version</th>
<th>location</th>
<th>Code in hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>HITE</td>
<td>0248</td>
<td>0232</td>
<td>18</td>
</tr>
<tr>
<td>WID</td>
<td>0249</td>
<td>0233</td>
<td>50</td>
</tr>
<tr>
<td>CLEAC1</td>
<td>024A</td>
<td>0234</td>
<td>02</td>
</tr>
<tr>
<td></td>
<td>024B</td>
<td>0235</td>
<td>1B</td>
</tr>
<tr>
<td></td>
<td>024C</td>
<td>0236</td>
<td>3D</td>
</tr>
<tr>
<td>LINOFF</td>
<td>025E</td>
<td>0246</td>
<td>20</td>
</tr>
<tr>
<td>COLOR</td>
<td>025F</td>
<td>0249</td>
<td>20</td>
</tr>
<tr>
<td>ASCUR</td>
<td>0260</td>
<td>024A</td>
<td>00</td>
</tr>
<tr>
<td>ERASEC</td>
<td>026C</td>
<td>0250</td>
<td>00</td>
</tr>
<tr>
<td>IVON</td>
<td>0248</td>
<td>0267</td>
<td>00</td>
</tr>
<tr>
<td>IVOFF</td>
<td>025B</td>
<td>026E</td>
<td>00</td>
</tr>
<tr>
<td>USELST</td>
<td>02AA</td>
<td>0280</td>
<td>00</td>
</tr>
<tr>
<td>MEMAPV</td>
<td>02B0</td>
<td>0280</td>
<td>00</td>
</tr>
<tr>
<td>MEMADR</td>
<td>02B1</td>
<td>0291</td>
<td>00</td>
</tr>
<tr>
<td>HIBIV</td>
<td>02B2</td>
<td>0292</td>
<td>00</td>
</tr>
<tr>
<td>HIBCUR</td>
<td>02B3</td>
<td>0293</td>
<td>00</td>
</tr>
<tr>
<td>CRELIV</td>
<td>02B4</td>
<td>0294</td>
<td>00</td>
</tr>
<tr>
<td>DEL3</td>
<td>02B5</td>
<td>0295</td>
<td>00</td>
</tr>
<tr>
<td>DEL4</td>
<td>02B6</td>
<td>0296</td>
<td>00</td>
</tr>
<tr>
<td>DEL5</td>
<td>02B7</td>
<td>0297</td>
<td>00</td>
</tr>
<tr>
<td>ITHELP</td>
<td>02B8</td>
<td>0298</td>
<td>00</td>
</tr>
<tr>
<td>ITITOG</td>
<td>02B9</td>
<td>0299</td>
<td>00</td>
</tr>
<tr>
<td>NONDOC</td>
<td>02BA</td>
<td>029A</td>
<td>00</td>
</tr>
</tbody>
</table>

NE: To suppress SIGNON and Warning in Wordstar 3.3, enter 00 at 02E2h, 3F1Dh and 411Fh.
EDITOR'S MESSAGE by Stan Sokolow ......................... 1
CRYSTAL BALL GAZING: UNIX IN THE NEWS by Stan Sokolow ..... 1
ALIGN YOUR OWN DISK DRIVES by Stan Sokolow ............... 1
UNCLASSIFIED ADS ............................................. 3
ON A DAISY WHEEL PRINTER AND A RAM DISK by Rick Downs .... 4
ON BASIC/Z by David Dalva .................................... 7
ON A HARD DISC AND CP/M by David Dalva ..................... 9
QUESTIONS FROM MEMBERS TO MEMBERS ........................ 12

BACKPLANE REFUNDS

Bob Marsh has been too busy to have the backplanes manufactured. We don't want to hold your funds any longer. We will send refunds soon. Sorry.

PROTEUS/NEWS

A news journal for owners and users of Processor Technology.
Corporation computer equipment. Published by Proteus, 1690
Woodside Road, Suite 219, Redwood City, California
94061-3483, USA, telephone (415) 368-2300.

Submit items for publication to Proteus News, Attn: Stan
Sokolow, 1690 Woodside Road, Suite 219, Redwood City,
California 94061-3483, USA. Please make submissions as
camera-ready as possible by using a fresh, black ribbon and
typing single-spaced.

Copyright (C) 1984 by Proteus. All rights reserved.
Permission is hereby granted to reproduce any computer programs
contained herein, provided that Proteus and the program's
author are given credit.

FROM:
PROTEUS
1690 WOODSIDE ROAD, SUITE 219
REDWOOD CITY, CALIFORNIA 94061-3483
USA

FIRST CLASS MAIL
PROTEUS QUESTIONNAIRE #1

To determine the direction our members would like Proteus to take in the coming year, please complete the following questions and return to Proteus, Attn: Stan Sokolow, 1690 Woodside Road, Suite 219, Redwood City, CA 94061. You may remain anonymous or sign the questionnaire. The tabulations will be published in Proteus News. Thanks for your active participation.

1. Describe your system:
   ( ) Sol
   ( ) Northstar floppy
   ( ) Micropolis MetaFloppy
   ( ) Standard 8" floppy such as Tarbell, Discus, etc.
   ( ) Other storage medium:
   ( ) Daisy-wheel printer
   ( ) Selectric
   ( ) Dot-matrix printer
   ( ) Other printer

   Number of kilobytes of RAM:

2. Software you own:
   ( ) Extended Cassette BASIC
   ( ) ALS-8
   ( ) CP/M for Northstar
   ( ) CP/M for Micropolis
   ( ) CP/M for Helios
   ( ) Micropolis MDOS and BASIC
   ( ) Microsoft FORTRAN
   ( ) Proc. Tech. PILOT
   ( ) UCSD Pascal
   ( ) COBOL
   ( ) WordWizard
   ( ) Others you often use:

   ( ) Microsoft BASIC
   ( ) CP/M for standard 8" disk
   ( ) CP/M for Micropolis
   ( ) Northstar DOS and BASIC
   ( ) PTDOS for Helios
   ( ) Proc. Tech. FORTRAN
   ( ) Other PILOT
   ( ) Other Pascal
   ( ) Electric Pencil
   ( ) TEX CP/M text processor

3. Please estimate the relative amount of time your computer is used for business versus recreational purposes:

   % business  % recreation

4. CP/M users: Do you have access to the CP/M users group library on media your computer can use?
   ( ) Yes, I can get these programs...
   ( ) direct from the CP/M Users Group
   ( ) from a computer club in my area
   ( ) from a friend
   ( ) from:

   ( ) No, the users group doesn't have my kind of diskettes.

   If not, would you like access to these programs?
   ( ) No
   ( ) Not sure. I'd like to see what the library has.
   ( ) Yes, but only on my kind of diskettes.
   ( ) Yes, I'd even take them on cassettes in a format suitable for loading onto my disk (assuming the price is reasonable).

(Continued on other side)
5. Would you buy cassettes containing public-domain library programs for cassette systems?  ( ) Yes.  ( ) No.  ( ) Perhaps.

If so, what do you consider a reasonable price for a C-60 cassette from the library, including postage?  From $__ to $__.

6. What kinds of programs would you like to see in a library?  Indicate by showing the percentage of Proteus library space you personally would like devoted to each topic:

   8 Games
   8 Business applications (accounting, analysis, etc.)
   8 Programming systems and aids (compilers, debuggers, etc.)
   8 Statistical, engineering, scientific
   8 System utilities (file manipulations, device drivers, etc.)
   8
   8
   8
   8

7. What kinds of articles would you like to see in Proteus News?

8. What other services would you like Proteus to perform?

9. What didn't you like about our performance as "Solus" in the past year?

10. What did you like most about Solus?

11. What do you think about the current issue of Proteus/News?

12. Would you participate in a software-swap meeting of Proteus at the 4th West Coast Computer Faire, May 11-13, in San Francisco?  It would be arranged so that you could either mail your software ahead of time, or drop it off when you arrive at the show, and then pick up copies of all submitted software at the end of the show.

   ( ) No, I don't have any original software to contribute.
   ( ) Yes, but by mail only.
   ( ) Yes, I'll attend and can drop off my software there.

What else would you like to have happen at such a meeting?

   ( ) A tutorial course on the following topic:

   ( ) A panel discussion by Proteus members on:

   ( ) A presentation by Processor Tech on:

   ( ) Other:
PROTEUS ORDER FORM

PROTEUS LIBRARY CASSETTES (PUBLIC-DOMAIN PROGRAMS)
An annotated catalog of each cassette is published in
Vol. 2, Number 1, and subsequent issues.
Price: Each cassette is $8 with a donated program, or $18
without an acceptable donated program. See the Copyright
Statement on the other side if you are donating a program.
To order: Circle the item codes.
  Items ordered: C1 C2 C3 C4 C5
  Number of cassettes @ $18 each (without donation):
  Number of cassettes @ $8 each (with donation):
  (One donated program must be submitted for each
tape purchased at the $8 price.)
TOTAL PRICE OF CASSETTES FROM LIBRARY:

HELIOS LIBRARY DISKS (PUBLIC-DOMAIN)
An annotated catalog of diskette H1 is in Solus News (the precursor
of Proteus/News) Vol. 1, Number 6. Diskette H2 will be described
in the next issue of Proteus/News. It will contain the source
code for the SLAC PASCAL system and miscellaneous donated programs.
Price: $25 each diskette without acceptable donated program,
or $10 with an acceptable donated program. See Copyright
Statement for donated programs, on reverse side.
To order: Circle the diskette item number.
Helios diskettes ordered: H1 H2
Number of diskettes @ $25 each:
Number of diskettes @ $10 each:
(An acceptable program or data file must be submitted for each
diskette ordered at $10 price.)
TOTAL PRICE OF DISKETTES ORDERED:

COMMERCIAL PROGRAMS (PROPRIETARY, NOT PUBLIC-DOMAIN)
G/2 Extended BASIC cassette. (List $49.95) @ $42.46 ___
G/2 Beat the House cassette. (List $14.95) @ $12.71 ___
G/2 Clinic cassette. (List $14.95) @ $12.71 ___
G/2 Outwit cassette. (List $14.95) @ $12.71 ___
TOTAL PRICE OF ALL PROGRAMS ORDERED: $ SUB-TOTAL $

Calif. residents sales tax (6% of previous line) TAX $

PERIODICALS
Current subscription to Proteus/News (1979)
  In the U.S.A., $12 per calendar year.
  Foreign addresses, $15 per calendar year.
Subscriptions expire at end of calendar year (Dec 31).
Mid-year subscribers will receive back-issues of current
year. Subscription includes membership in Proteus,
required to purchase library programs or obtain discounts
on commercial products.
Back-issues of 1977 (Vol. 0, Solus News) @ $2/set
Back-issues of 1978 (Vol. 1, Solus News) @ $10/set

(PERIODICALS ARE NOT SUBJECT TO CALIF. SALES TAX) $

TOTAL AMOUNT OF PURCHASE
  (MAKE PAYABLE TO PROTEUS, US FUNDS ONLY PLEASE.) $ TOTALS$

OVER
LIMITED WARRANTY ON PROTEUS LIBRARY PROGRAMS

The programs included in the Proteus Library have been
donated by members as original works or from the public domain.
Proteus has screened the programs to check that they work, but
Proteus does not warrant them to be correct. Some of the programs
may bear restrictions on their use and so indicate in the text
of the program. Media that cannot be read by the purchaser's
appropriate equipment may be returned within 30 days from date
shipped, for replacement or repair.

Proteus assumes no liability to anyone with respect to any
loss, damage, or liability resulting directly or indirectly
from the use of these products, including but not limited to
interruption of service, loss of income, or consequential damages
arising from the use of the product.

COPYRIGHT STATEMENT (REQUIRED WHEN FILES ARE DONATED TO LIBRARY)

If you are donating a program or data file to Proteus for
a reduced price on purchase of a Proteus product, please complete
the following statement:

As donor of the computer file(s) or program(s) named below,
I certify that no copyrighted work is contained therein, other
than my own. Furthermore, if my submission is accepted by Proteus
(in whole or in part), I hereby transfer, assign, or otherwise convey
all copyright ownership to Proteus.

Name of files or programs donated:

Donor's signature: ____________________________

Date: __________

MAIL COMPLETED ORDER FORM TO:

PROTEUS
1690 WOODSIDE ROAD, SUITE 219
REDWOOD CITY, CALIFORNIA 94061
U.S.A.

ALL PRICES INCLUDE POSTAGE BY UPS OR SURFACE MAIL. IF AIR SHIPMENT
IS DESIRED, LET US KNOW AND WE WILL NOTIFY YOU OF ADDITIONAL CHARGES.

IF PAYMENT ACCOMPANIES ORDER, WE PAY POSTAGE.
IF C.O.D. IS DESIRED, WE WILL ADD SHIPPING AND C.O.D. CHARGES.
C.O.D. NOT AVAILABLE TO FOREIGN ADDRESSES.

U.S. FUNDS ONLY, PLEASE.
LATE NEWS! >>>> PTC SOURCE CODE FOR SALE <<< LATE NEWS!

For the past 6 months, we've been trying to get in touch with someone at Processor Tech who has authority to negotiate sale of their source code, and finally we succeeded! The source code for PTDOS (and its subordinate commands), ECBASIC, EDBASIC, ALS-8, and in fact for any PTC program that was produced in-house, is for sale by PTC for an amount in 5 figures, depending upon exactly what we buy. What we will try to do is put together enough buyers to collectively purchase what we want. If we don't act fast, the Corporation may be forced into bankruptcy court by its creditors and we will lose the opportunity to negotiate this purchase.

The purchase agreement will probably state that buyers can use the source code in any way (commercially or otherwise), but they cannot resell or otherwise disclose the source to anyone else. The sale is non-exclusive, that is, PTC is selling this to anyone who wants it, but no buyer can turn around and sell or give away the source, just object code. I'll try to negotiate a time limit on the non-disclosure clause, so that after some time the package could go into public domain.

The agreement will include the right to reproduce the user's manual, and hopefully a machine-readable copy of it.

To keep the price within reach of the greatest number of members, we are splitting the software into a few packages. We haven't seen the programs yet, so we have no idea of the amount of code that we will have to reproduce. Hence, we have separated the cost for reproduction and mailing from the purchase of the source code itself. After we see what we have, we will let you know the reproduction cost, but the rates will be our usual—that is, $0.15 per page, $8 per cassette, $10 per diskette.

So, here are the packages and prices, with their Proteus item numbers.

Proteus item P10:
Source code and object code for the cassette versions of BASIC/5 language, FOCAL language, Software Package #1 (precursor of ALS-8), and ASSM (cassette-to-cassette assembler). Price $45 plus reproduction costs.

Proteus item P11:
Source code and object code for GAMEPAC 1, GAMEPAC 2, and TREK-80. Price $15 plus reproduction costs.

Proteus item P12:
Source code and object code for Extended Cassette BASIC, Extended Disk BASIC, Optional Precision Extended Disk BASIC, and possibly the new Level I Business BASIC (Disk) which allows program chaining with passing of data in COMMON area and other enhancements for modular business systems. Price $90 plus reproduction costs.

Proteus item P13:
Source code and object code for Extended Cassette BASIC only. Price $45 plus reproduction costs.

Proteus item P14:
Source code and object code for ALS-8, the Assembly Language System consisting of an assembler (memory-to-memory), video editor, and 8080 simulator. Price $45 plus reproduction costs.
Proteus item P15:
Source code for PTDOS, the Processor Technology Disk Operating System. We presume this will be version 1.5, although it might also include the version 2.0 which was under development when PTC went out of business. It will contain all of the source modules for the command and device files on the distributed disk except the source of the Extended Disk BASIC, which is a separate item listed above. Price $90 plus reproduction costs.

Proteus item P16:
Source code and object code for Sol diagnostic and test programs that were designed to aid the technician in trouble-shooting a malfunctioning Sol system. Includes 16KRA, 16KDT, 32KRA, 48K (all preceding are memory tests), DMARD and DMAWR Helios DMA tests, SOlT Sol Diagnostic Test program, SOL-B Sol burn-in test program, DOST Sol System III automatic tests, DISKT Helios Disk Diagnostic Program, DISKCHK Helios file checker, SIMU (Simu-Ciser) a program to exercise the PerSci floppy disk drive during alignment of the drive, PSRCI thru PSRC3 ParaSol debugger programs for the ParaSol board. Price $25 plus reproduction costs.

User's manuals for all of the acquired programs will be made available as Proteus document items (prefix "D" in item number), both individually and in groups corresponding to the above item groupings. In addition, we plan to produce a "Book of Sol" in several volumes, containing portions of PTC technical manuals that have not been generally available. As part of the deal with PTC, we plan to acquire the right to reproduce these manuals, and part of the purchase prices you pay will go toward this.

To purchase your copy of the items above, send us your order immediately. Payment of the above mentioned prices should accompany your order. Once the programs are acquired, we'll let you know how much extra it will cost to reproduce them, but the rates will be as mentioned above. If we can't put together enough money to make the deal, we'll refund your money. In your order, tell us what media you want the programs on: paper listing only, cassette, Helios disk. We'll also make the programs available on standard 8" soft-sectored CP/M diskette and NorthStar diskette, but we can't promise swift delivery of this form because of the time consuming process of transferring the data to the other media.

 Acquisition of these programs will help prolong the useful lives of our systems. I hope we have adequate participation by the members. This is the most significant event in the history of Proteus.

Sincerely yours,

Stan

Stanley M. Sokolow
A FAREWELL NOTE

This issue marks the end of an era in the history of the Sol computer. We started publication of this newsletter in August, 1977, after the first formal meeting of the Sol user's group. This issue, 7 years later, is the last one we are going to print. In its place, we will continue to send announcements of general interest to Sol owners, but only at irregular intervals and only when we have sufficient material to warrant it.

The original Sol user's group, called Solus back then, began with about 30 owners. Over the years, our numbers swelled to a high point of about 1200 and gradually declined to the present level of about 200 members. Of course, there were many more Sols made, but only some went to personal users who were interested in a user's group.

In the early days, we desperately needed each other for information and help. After Processor Technology went out of business, I felt we still needed each other more than ever. But today, most people have other options for handling their computing needs. A whole new generation of computers has come and will soon pass into obsolescence. Mass market computers are available with more computing power than our Sols.

The microcomputer world has moved on beyond the Sol, but the excellent design of the Sol has stood the test of time for those who continue to use it. By avoiding unusual components and by incorporating the S-100 bus, the designers created a rugged computer which will continue to be a servicable machine for many years to come.

It has been a great educational experience for me to be the focal point of information about the Sol. I started with a background in computer programming from my college years, but very little electronic knowledge and virtually none about digital logic and computer hardware. By reading Adam Osborne's original microcomputer books, articles in Byte magazine, Don Lancaster's TTL cookbook, etc., and by absorbing the comments of the skilled Sol owners who contributed material to the newsletter, I eventually could understand the Sol technical manual.

Out of necessity, I learned how to maintain my hardware that was orphaned. Just recently, I purchased my own oscilloscope for maintenance of my PerSci disk drives. I can read the schematics and trace the signals down to the faulty gate or transistor. I've come a long way. I look at the oscilloscope as another educational tool that will carry me into the future of computing. We all have a long way yet to go.
If you haven't yet taken the plunge into understanding the insides of your computer, but you consider computers your hobby, think about making a move in that direction. It's a good feeling to know that, as long as you can find the parts, you can be self-sufficient with your computer, even if it is an orphan.

Some day, our Sol's will be antiques. In fact, when asked what kind of computer I have, I usually tell people "Oh, you never heard of it; it's an antique made before the guys at Apple sold their first Apple II." Let's never lose sight of this fact. We've got the equivalent of the antique classic car.

I should give special thanks to a few people whose help made this journey last as long as it has. First, Bill Burns deserves special mention as the founder of Solus. It was his impetus that pulled together the Sol owners at the Homebrew Computer Club to form the first Solus meeting and subsequently to extend it to the world-wide Sol owners' group.

Second, the designers and entrepeneurs who started Processor Technology Corporation: Gary Ingram, Bob Marsh, Lee Felsenstein, and any others. I'm sure I speak for all Sol owners when I say that we were sorry to see Processor Tech go under, and we still have dreams of what it may have become, if only ....

Third, Les Solomon, technical editor of Popular Electronics (now known as Computers and Electronics) magazine. Those who don't know the story should read the historical material in the Encyclopedia Processor Technica, volume 1. The Sol is named after Les, and it owes a large part of its success to his early support in the magazine.

Fourth, I want to thank the major contributors of articles and time: Joe Maguire, Father Thomas McGahee, and others. You helped to make Solus News and Proteus News one of the best users' group newsletters, according to the many positive comments I received in letters from readers.

And certainly I want to thank all of you members who are reading this issue. Without your diehard support, many people would have been left stranded without any help after investing a large amount of their money in a Sol system. The continued membership support made this all possible.

One person deserves special mention because he is part of your future with the Sol: Bob Hogg. Bob is a retired engineer who has an uncommon level of expertise in digital electronics and in the Sol in particular. He is the proprietor of Micro Complex, the firm that is still supporting Sol owners with service and new products: the 64/80 column video upgrade, the dual personality module, his NorthStar work-alike improved disk controller (Phase Lock II), and so on. If you still love your Sol and want to keep it alive and growing, look to Bob for the help you need.
One last word: I'm quite busy now and not able to spend time with Proteus, but I'll always be reachable by phone. Here's a tip to tuck away with your Sol documentation. I'm a California licensed dentist, an orthodontist to be specific, so you'll always be able to track me down, even in the unlikely event that I move away. The California State Board of Dental Examiners (in Sacramento, California) will always know my business address. If you are absolutely stuck and can't seem to find anyone to help with a Sol problem, I'll be glad to give you any advice I can muster up out of my memory or to point you toward another contact.

So, best regards to you, especially the old-timers who've been with us since the start. It's been good to know ya'.

Sincerely,

Stan Sokolow

================================

McZOL

THE Z-80 UPGRADE FOR SOL FROM MICROCOMPLEX

Bob Hogg has been working on his Z-80 upgrade for a long time. It wasn't easy to get the Z-80 to work reliably with the Sol and its most common peripheral equipment. Now you can obtain a Z-80 upgrade from MicroComplex, but you should read this article before you buy one.

The Z-80 upgrade is a piggy back board that plugs into the main Sol PC board, just like the 64/80 column video board does. It takes up no slots. Since Bob designed the two upgrades to work together, you can install both in the same Sol and have a computer that will run your existing software as well as the new software designed for the Z-80 and 24 x 80 video screen.

The Z-80 replaces your present 8080 microprocessor, but since it has an expanded version of the 8080 instruction set, the Z-80 can run both 8080 and Z-80 programs. Bob has designed the board to run at 2 or 4 MHz clock rate, or optionally at higher clock frequencies. The faster the clock, the faster the computer.

The Z-80 upgrade can be used with a Sol that does not have the video upgrade, and instructions are given in the installation guide. I installed my Z-80 piggy-back in a Sol that has the video upgrade, without removing the Sol-PC board. This requires some care to be sure you don't bend the pins that insert in the Sol's sockets, but it doesn't require uncommon skill. It's smaller and easier to insert than the video board.
You need to remove the Sol covers, keyboard, and several IC's. A few IC's have to be removed, certain pins bent outward to avoid the socket, and re-inserted. A few soldered wires must be added. The keyboard needs to have a couple of passive components (R & C) replaced to be compatible with both the original and the faster clock speed.

You may have a problem with your S-100 memory boards. Processor Tech dynamic memory boards were designed only to work with a 2 MHz clock and they absolutely won't go at 4 MHz. This is also true of many other dynamic memory. I have static memory in the modified Sol, with memory chips rated at 300 ns. These work fine. If you need new memory, I've seen boards with the newer CMOS static chips advertised for about $300 for a 64K board. Check the back of Byte magazine for ads from Jade, Advanced Computer Products, etc.

You may also have trouble with your disk controller. Bob has been beating his head against the oscilloscope trying to fix the design to work with the NorthStar double density disk controller. NorthStar took shortcuts that make their board too sensitive to transients on the bus, and the Z-80 board just overwhelms the disk controller. Bob's Phase Lock II controller (which I dubbed the McFloppy) is a more robust design than NorthStar's, and it is compatible with NorthStar's format, so that's an escape. But it adds to your cost if you must go that route. The single density NorthStar controller works fine with the Z-80 upgrade.

I haven't fully checked out the nature of the problem yet, but when I tried to bootload my Helios disk system using PT DOS in the Z-80 modified Sol (which I'll call my McZol from now on), the bootload failed to come up. A modification to the Helios controller is needed for compatibility (two jumpers), but I made that and still no luck bootloading. I haven't had time to try other controllers or to look at possible problems with my installation.

So, if you have a Helios, proceed with caution. I know that PT DOS has some timing loops in software, and this may be the problem. Bob Hogg tells me that at least one other McZol is running a Helios with Lifeboat CP/M, so it sounds to me like a software problem, not hardware. The fix may only take modification of a single byte to change a timing loop counter, but I haven't looked into the details. Bob's hardware allows a program to switch the clock speed from 2 to 4 MHz, so I can try booting up a slow speed, but I haven't done that yet.

When you order your McZol upgrade, be sure to tell Bob what kind of IC sockets you have. The Texas Instruments low-profile (edge-wiped contacts for the IC pins rather than surface wiped contacts) require a different setting of the pin height when the upgrade is assembled by Bob.

Bob has tried his Phase Lock II controller in a McZol at speeds up to 10 MHz with no problem. (The video ram and proms have to be replaced with newer ones that are faster than the ones in a standard Sol, of course.) Z-80 chips that run at that speed over the full commercial temperature range are not available yet, but this fact gives you the assurance that the hardware can handle more than you are likely to give it.
I ran some simple BASIC programs with Extended Cassette BASIC, and yep, they ran twice as fast at 4 MHz as they do at 2 MHz. But, when I asked Solos for a memory dump, I noticed that it didn't run as fast as I expected. Then I remembered! The Sol PC introduces wait states when running programs in the Solos ROM.

The wait states were designed into the Sol to let it use slow EPROMs at the normal 2MHz clock speed and are not needed if you use faster ROMs. But at 4 MHz, you need the wait states again; the installation instructions tell you how to re-introduce the wait states if they have been jumpered out of your Sol. The wait states for I/O can be removed (the instructions tell you how). The wait state circuitry on the Sol only slows down programs when they access the on-board Solos ROM, not when they access S-100 memory. Your programs will of course reside in S-100 memory, so you will take full advantage of the new speed.

You can order the Z-80 upgrade from MicroComplex, 25651 Minos Street, Mission Viejo, CA 92691, telephone (714) 770-2168. Price: $150 for the basic board fully assembled and tested, but you install it; optional extra features available; installation available at modest additional cost. (Bob is a night person. The best time to call is in the afternoon or evening, Pacific time.)

Unclassified Ad

FOR SALE - Operational computer system consisting of:
SOL-20 Rev E. 56K User RAM from various manufacturers. Northstar single-density controller with dual drives. Sanyo 9" b-w monitor. Assorted software including: EC BASIC, EDIT modified with text processor, ALS-8, DEBUG, ASSM, Gamepac 1 and 2, Trek-80, assorted machine language and BASIC games and utilities. Programs that had cassette access will also be supplied with disk drivers. All pertinent manuals plus ALS-8 Users group letters and notes. JADE Bus Probe.

Will sell entire system for $500 or best offer received one month from the publication date. Will not sell parts individually.
Dear Stan and SOL Brothers,

Enclosed is the manual and complete source listing for the SUPER-USER AREA routines that Rick Downs mentioned in his letter to PROTEUS. You have my permission to reprint all of this in its entirety. I am sure that many SOL users will find things of interest in the program. As usual, it is heavily commented so that others may make changes more readily.

Although this program states that it is for LIFEBOAT CP/M version 2.X, the actual interface is consistent with ALL versions of CP/M that I have tested it with. I have, for instance, used it with the HELIOS CP/M 1.4, MICROPOLIS CP/M version 2.2, NORTHSTAR CP/M version 2.2, as well as with version 2.2 CP/M for TARBEILL, MICROMATION, VERSAFLOPPY and others. There is one section marked with $<;$$$$> that will have to be changed for each specific system, but that is a minor undertaking. Just look up what your CURRENT system equates are, and put those in. The program has lots of goodies in it, and so it eats up almost 512 bytes. In systems where the BIOS does not have sufficient room for all this, it can be kept in a separate file and relocated into the top of memory and then patched into the BIOS using a short program. It can be used as-is by any LIFEBOAT version of CP/M that supports a separate USER AREA.

I originally wrote this program to specifications supplied by Rick Downs. Because it was a custom program, I charged him $65 to help cover the time I spent writing it. I am now releasing it through PROTEUS in the hope that other SOL users may benefit from it. You guys out there who find it useful can thank Rick Downs for getting me to write it in the first place. I have added some additions and improvements to the original program specifications, and as far as I can determine the program is bug-free.

I would like to remind our readers that we here at DON BOSCO TECHNICAL HIGH offer a three-year course in computer programming as well as a three-year electronics course in which we teach computer fundamentals and interfacing. Any donations of computer equipment (especially SOLS) is greatly appreciated. You would not believe how much use we get out of our equipment! Our machines are running from 8 AM to 10 PM almost every day of the week. We need some additional
equipment to handle replacement of machines while they are being repaired. In addition, we have several projects underway that require the use of a computer as a controller. For example, current projects are using the computers to control ROBOTS, our own homemade video games, a LASER beam that writes on the wall, computerized music synthesis, a vector-beam graphics system, and a model computer language. We keep pretty busy around here.

When sending any donated equipment, include an itemized list of your original cost. I will see to it that you receive an itemized acknowledgement for your tax records.

Some people like Dr. Richard Black and Jim Blackwood have already donated SOL systems to us, and these systems are now in daily use here at DBT.

We can even use certain so-called BASKET CASES as spare-parts sources. (Besides taking care of our own SOLs here at DBT, I also help other SOL users get their systems modified and repaired). Almost anything related to SOLs is of use to us.

I will try to get some more articles out in the near future, as time permits.

Sincerely yours,

Fr. Thomas McGahee

Fr. Thomas McGahee

P.S. We have an IBM PC here that I have been doing a lot of programming on, but it is still not as useful as our SOLs at this time because I have much to learn about its innards yet. I have also been heavily into DBASE II programming the past two years, and have developed a lot of neat programs and techniques that I eventually want to share.
CUSTOMIZING USER.ASM TO MEET YOUR NEEDS

USER.ASM is provided in SOURCE CODE so you can easily make changes to it to customize it to your needs. Some sophisticated users may wish to actually change some of the program code, but most persons will find that they can effect the necessary customization by simply setting a few EQUATE statements. Be aware of the fact that, depending on the options you select, up to 511 of the 512 bytes available in the USER area may be used. If you make changes to the actual PROGRAM (as opposed to just choosing options with the EQUATE statements), then make sure that you examine the value of <LFTOVR> at the VERY END of the USER.PRN listing the assembler generates. Recall that the assembler uses two-s complement notation. If the value of <LFTOVR> (The amount of space LEFT-OVER) starts off with an "F", then you have a NEGATIVE amount... meaning you have exceeded the 512 byte limit.

Using whatever editing facilities you have available, load in the USER.ASM file. In systems with very little memory, you may wish to just append the first 300 lines, as the entire program is very large. This is due to the fact that I have heavily commented the program so people can figure out what I am doing. All lines that might possibly require customization are marked with <;***> or <;$$$>. Thus you may use the search facility of your editor to search for <;***> or <;$$$> to locate all such lines.

I will now outline each such line and give a brief description of its use and possible options. To make things easier, I will give them in the order they actually appear within the listing.

MSIZE Set for the amount of contiguous memory in your system.

SCRN16 Set <TRUE> if using 64x16 screen, or <FALSE> if using the MICROCOMPLEX 80x24 board by BOB HOGG. Setting this will cause automatic generation of the SYSTEM RAM addresses as required.
PAGE. Set for the number of actually printed lines desired per page of printed text. A normal sheet of paper is 66 lines long. I suggest setting this to 55, as this allows a decent margin at the top and bottom of the sheet.

NEWPAGE Set this <TRUE> if you want an automatic FORMFEED generated whenever you use LOAD/F, LOAD/S, or LOAD/^ to setup the printer.

SOLOS Set this for the starting address of the SOLOS ROM in your system. <OC000H> or <OF000H> are usually used.

CONSIN Set to the device you want used for console input. May be KBD, SERIN, PARLIN, or CSTMIN.

** NOTE ** KBD=SOL KEYBOARD, SERIN=SOL SERIAL INPUT PORT, PARLIN=SOL PARALLEL INPUT PORT, CSTMIN=USER-DEFINED CUSTOM INPUT PORT WHOSE ADDRESS IS AT SOLOS <UIPRT>.

CONSOUT Set to the device you want used for console output. May be VDM, SEROUT, PARLOUT, or CSTMOUT.

** NOTE ** VDM=SOL VIDEO SCREEN, SEROUT=SOL SERIAL OUTPUT PORT, PARLOUT=SOL PARALLEL OUTPUT PORT, CSTMOUT=USER-DEFINED CUSTOM OUTPUT PORT WHOSE ADDRESS IS AT SOLOS <UOPRT>.

LISTOUT Set to the device you want used as your LIST device (PRINTER). May be VDM, SEROUT, PARLOUT, or CSTMOUT.

LISTIN Set to the device used to read from LIST device. This is used for handshaking purposes, and can generally be set as follows: <LISTIN EQU LISTOUT>. This is the default setting. You may also explicitly set it to SERIN, PARLIN, or CSTMIN.

RDRIN Set to the device you want used as the READER. It may be SERIN, PARLIN, or CSTMIN. I suggest that if you are using PARLIN for the LISTIN device, then use SERIN as the RDRIN device. If using SERIN as the LISTIN device, then use PARLIN as the RDRIN device. There is nothing wrong, of course in having LISTIN and RDRIN assigned to the same device. In many systems it is useful to use the READER for reading in from MODEMS.
PUNCHOUT

Set to the device you want used as the PUNCH. It may be SEROUT, PARLOUT, or CSTMOUT. I suggest that if you are using PARLOUT for the LISTOUT device, then use SEROUT as the PUNCHOUT device. If using SEROUT as the LISTOUT device, then use PARLOUT as the PUNCHOUT device. There is nothing wrong, of course in having LISTOUT and PUNCHOUT assigned to the same device. In many systems it is useful to use the PUNCH for sending to MODEMS.

NONEH

Set <TRUE> only if your LIST device (PRINTER) requires no handshaking. Printers using the PARALLEL port do not generally require handshaking, as this is done in hardware by the parallel port driver in SOLOS. Also, certain non-intelligent printers such as the venerable TTY do not require any handshaking.

ETXH

Set <TRUE> only if your LIST device (PRINTER) is set up to handle ETX/ACK handshaking. For example, DIABLO and most DAISY WHEEL and THIMBLE printers can generate and use this protocol. Some such printers need to have the protocol explicitly set by switch somewhere, whereas others use this protocol without need for any switch settings. See your printer manual for details.

XONH

Set <TRUE> only if your LIST device (PRINTER) is set up to handle XON/XOFF or DC1/DC3 or CTL-Q/CTL-S handshaking. Most DOT MATRIX printers use this protocol, and many DAISY-WHEEL and THIMBLE printers can select this protocol as an option. See your printer manual for details.

BUFFR

Set this to the length of your printer's buffer. This is only needed for ETX/ACK handshaking. The default value of 80 is for DIABLO printers and most DAISY WHEEL and THIMBLE printers. If not sure about your printer's buffer size, leave it at 80... it will still print at its maximum speed! Buffer size is used to determine how often ETX/ACK handshaking must be done, and that's all. Maximum value for <BUFFR> is 255.

BANK

Set <TRUE> only if you are using a bank-select scheme such as my SUPER-PHANTOM to allow RAM to overlay SOLOS ROM/RAM. This allows CP/M systems greater than 48K to be generated and run. Please note that the entire USER area must reside outside the SOLOS ROM/RAM address space for the overlay scheme to work properly. In addition, unless the CPM system has been patched to change the way stacks are assigned, then the entire CPM operating system must reside outside the SOLOS ROM/RAM area. If this is not done, then stack control may be lost when banks are switched.
BKPRT  Set to value of port used for bank-selecting. In my
SUPER-PHANTOM technique, OFFH is used.

SOLON  Set to value to be sent to bank-select port to cause
SOLOS ROM/RAM to be ACTIVATED.

SOLOFF  Set to value to be sent to bank-select port to cause
SOLOS ROM/RAM to be DE-ACTIVATED and OVERLAYED by RAM.

That takes care of the actual system EQUATES. There is also a set of
INITIALIZED values that determine the DEFAULT conditions when the
USER area is first loaded from disk. The following detail these
locations and their default values:

ATOGGLE  <00>=OFF <01>=ON. When ON, the video screen will scroll
up a maximum of one full page and then stop. Hitting any key will
cause it to start scrolling the next full page. It is recommended
that this be set to <00>. Please note that when using the
SHEET-PAPER option and doing a listing, this should be toggled OFF,
as the SHEET-PAPER option will also suspend listing until a key is
hit, and you may get confused as to what hitting the key will do!

LTOGGLE  Set this to <01> ONLY if you are setting <PAPER> to
"F", and you want perforation skipping (Automatic Formfeed every
<PAGEL> number of lines). Otherwise set it to <00>.

PTOGGLE  <00>=OFF <01>=ON. This determines whether or not the
PRINTER is active. When active, everything that appears on the
CONSOLE OUTPUT device will be echoed on the PRINTER. NOTE: This
feature is INDEPENDENT of the CPM CTL/P function. This is toggled
using the LOAD/P sequence. Don't have CTL/P and LOAD/P BOTH ACTIVE,
or the printer will double-print text.

PAGE    <PAGEL>. Leave this set to <PAGEL>. This starts the
program logic off so that it thinks it is at the top of a page.
NOTE: You still have to set your printer Top-Of-Form control if you
have one.

PAPER   <'S'>=SHEET <'F'>=FAN-FOLD. Set this to the kind of
paper you generally use. LOAD/S and LOAD/F and LOAD/" can be used to
change the value under program control later.

*** The remaining INITIALIZED data items should be left as-is.
USING THE NEW FEATURES

MODE-SELECT This is converted to a CONTROL/C. Since CONTROL/C is used so often, it is nice to have a single-key that can be used in its place. The MODE-SELECT key was chosen because it is so large and conveniently located.

LEFT-ARROW The SOL LEFT-ARROW CURSOR KEY is converted to a BACKSPACE, since the SOL has no backspace key. Note that in CPM a backspace will cause the cursor to back-up and ERASE the last character. (DELETE and LEFT-ARROW are equivalent functions now, as both act like backspace).

DELETE The DELETE KEY is converted to a BACKSPACE. Note that in CPM a backspace will cause the cursor to back-up and ERASE the last character. (DELETE and LEFT-ARROW are equivalent functions now, as both act like backspace).

NOTE The output routines have been designed so that a BACKSPACE will be properly handled. Also, multiple carriage-returns no longer cause stuff on the screen to be erased.

THE LOAD KEY HAS BEEN IMPLEMENTED AS A KIND OF "ESCAPE" FUNCTION. THE FOLLOWING LOAD SEQUENCES HAVE MEANING:

LOAD/A Toggle the AUTO-DISPLAY feature. When this is ON, it causes the screen to stop scrolling after one full page. To resume scrolling of next page, hit any key except LOAD.

NOTE Any LOAD sequence that produces a TOGGLE-ACTION will result in a reverse-video <1> or <0> being displayed to indicate whether it is now ON or OFF. The actual LOAD sequence itself is always done without any echoing of characters on the screen. Please note that the program will convert LOWER CASE to UPPER CASE automatically within the LOAD SEQUENCE. This is a convenience feature so you don't have to worry about what case you're in.

LOAD/0 – LOAD/9 Sets the speed at which scrolling takes place. <0>=FASTEST <9>=SLOWEST.

LOAD/P Toggles the PRINTER ON and OFF. Note that this is independent of the CPM CONTROL/P function. If your implementation of CPM supports the CONTROL/P function, then do not activate it and this at the same time, or else double-printing will occur. Unlike CONTROL/P, LOAD/P may be invoked at any time, even in the middle of a listing!
LOAD/S  Unconditionally sets the printer up to handle SHEET-PAPER. If <NEWPAGE> was set <TRUE> in the assembly listing, then an automatic FORMFEED will also be generated at this time. In any case, the <PAGE> count is set to <PAGEL> to indicate that we are at the top of a new page. When SHEET PAPER is being used, the PRINTER will automatically receive a FORMFEED signal at the end of each page. It will then suspend further printing until any key is hit on the CONSOLE INPUT device (usually the keyboard).

LOAD/F  Unconditionally sets the printer up to handle FAN-FOLD PAPER with continuous printing. If <NEWPAGE> was set <TRUE> in the assembly listing, then an automatic FORMFEED will also be generated at this time. This mode is very handy when printing MAILING LABELS and the like, because it will NOT generate an automatic FORMFEED at the end of a page.

LOAD/^  Unconditionally sets the printer up to handle FAN-FOLD PAPER with AUTOMATIC FORMFEED generation at the end of each PAGE. (That is, it will skip over the perforations in the paper, making for a much neater listing). If <NEWPAGE> was set <TRUE> in the assembly listing, then an automatic FORMFEED will also be generated at this time. In any case, the <PAGE> count is set to <PAGEL> to indicate that we are at the top of a new page.

LOAD/MODE-SELECT  Causes a JUMP to SOLOS. (This is an entry to the SOLOS <RETRN> entry point at SOLOS+4). Think of it as being like a SYSTEMS RESET, except that the screen is not erased, and the SOLOS SYSTEMS RAM is not cleared. Should you prefer the screen erasure and clearing action, change the assembly listing after the label <IN4:> from <JMP SOLOS+4> to <JMP SOLOS>.

LOAD/LOAD  This allows you to temporarily suspend screen operations. Whenever you hit the first LOAD, the display will freeze. When you hit the second LOAD, it will continue. This is much more convenient that using the CPM CONTROL/S feature. Since printing is tied in with the display, this is also a way of temporarily suspending printing.

ANY OTHER LOAD SEQUENCE NOT OUTLINED ABOVE WILL ACT LIKE THE LOAD/LOAD SEQUENCE AND MERELY PAUSE THE DISPLAY.

One last NOTE: I have incorporated the key conversion and LOAD sequence interpreter into the CONSOLE STATUS routine instead of the CONSOLE INPUT ROUTINE. This allows all features to be activated AT ANY TIME. In addition, whenever a LOAD sequence is entered, the return to CPM is done in such a way that CPM does not know that anything happened. Since a TOGGLE can be either ON or OFF, I have included a routine that displays a REVERSE-VIDEO <O> for OFF and <I> for ON. CPM is totally unaware that these REVERSE-VIDEO codes have been displayed. They are simply there for your convenience.
CUSTOM USER AREA FOR LIFEBOAT CP/M 2.X

***** MAY SERVE AS THE I/O SECTION IN NON-LIFEBOAT CP/M
***** BY JUST CHANGING THE EQUATES MARKED WITH $$ in THE
***** COMMENTS.

***** CUSTOMIZED FOR USE WITH SOL-20 AND MOST PRINTERS

*****
***** WRITTEN BY FR. THOMAS MCGAHEE
***** DON BOSCO TECHNICAL HIGH
***** 202 UNION AVE.
***** PATERN, NJ 07502
***** (201) 595-8800


*****
***** LAST REVISED JUNE 4, 1984


***** MODIFIED TO ALLOW NULLS AFTER CR
***** GENERATES FORMFEED VIA CALCULATED LINEFEEDS


***** IF YOU FIND THIS SOFTWARE USEFUL AND WISH TO MAKE A
***** $5 DONATION TO HELP ME CONTINUE PRODUCING SUCH STUFF
***** I WOULD APPRECIATE SUCH A KIND GESTURE OF SUPPORT.


***** IF YOU HAVE AN OLD SOL YOU NO LONGER USE, WE HERE AT
***** DON BOSCO TECH WOULD LOVE TO GIVE IT A NEW HOME
***** WHERE IT WILL GET LOTS OF USE AND TENDER LOVING CARE
***** <DON BOSCO TECH HAS A THREE-YEAR COMPUTER COURSE IN
***** WHICH WE GIVE STUDENTS OVER 700 HOURS OF INSTRUCTION
***** AND ON-HANDS EXPERIENCE IN COMPUTER PROGRAMMING. WE
***** CAN MAKE VERY GOOD USE OF ANY SOLS AND OTHER LIKE
***** EQUIPMENT. DONATIONS ARE TAX DEDUCTIBLE>


;DEFINE TRUE AND FALSE.
TRUE EQU OFFFFH ;DEFINE TRUE.
FALSE EQU NOT TRUE ;DEFINE FALSE.

;DEFINE SOL PSEUDO PORTS.
KBD EQU 0 ;SOL KEYBOARD.
VDM EQU 0 ;SOL VIDEO DISPLAY.
SERIN EQU 1 ;SOL SERIAL INPUT.
SEROUT EQU 1 ;SOL SERIAL OUTPUT.
PARLIN EQU 2 ;SOL PARALLEL INPUT.
PARLOUT EQU 2 ; SOL PARALLEL OUTPUT.
CSTMIN EQU 3 ; SOL CUSTOM INPUT DRIVER.
CSTMOUT EQU 3 ; SOL CUSTOM OUTPUT DRIVER.

******************************************************************************
***** THE USER MAY WISH TO ADJUST THE FOLLOWING EQUATES *****
******************************************************************************

; Change MSIZE to the desired CP/M memory size in K.
MSIZE EQU 56 ; *** Distribution size

; SET <SHOW> TRUE IF YOU WANT TOGGLS FLAGGED IN REVERSE VIDEO.
SHOW EQU TRUE ; SET FALSE ONLY TO SAVE SPACE.

; SET <SCRN16> TRUE IF 16X64, OR FALSE IF 24X80.
SCRN16 EQU FALSE ; *** SET TRUE IF 16X64

; SET <NULLS> TO # OF NULLS AFTER EACH LINEFEED.
NULLS EQU 6 ; *** NUMBER OF NULLS AFTER <CR>.

; SET <DONULLS> TRUE IF YOU WANT NULLS AFTER CR.
DONULLS EQU TRUE ; *** SET TRUE IF NULLS ARE DESIRED.

; SET <PAGEL> TO # OF LINES TO BE PRINTED PER PAGE OF SHEET PAPER.
PAGEL EQU 55 ; *** SET PAGE LENGTH FOR SHEET.

; SET <MAX> TO MAXIMUM # OF LINES PER PAGE (INCLUDING MARGINS).
MAX EQU 66 ; *** SET MAX FORM LENGTH.

; ALLOW-automatic page-eject on receipt of load/f, load/s, or load/^.
NEWPAGE EQU TRUE ; *** SET TRUE IF NEW PAGE DESIRED.

; SET <SOLOS> TO THE BASE ADDRESS OF YOUR SOLOS ROM.
SOLOS EQU 0F000H ; *** BASE OF SOLOS ROM.

; SELECT WHICH PORTS/DEVICES YOU WANT FOR STANDARD CP/M I/O.
CONSLN EQU KBD ; *** KEYBOARD IS USUAL CONSOLE INPUT DEVICE.
CONSOOUT EQU VDM ; *** VIDEO IS USUAL CONSOLE OUTPUT DEVICE.
LISTOUT EQU SEROUT ; *** SELECT YOUR PRINTER PORT.
LISTIN EQU LISTOUT ; *** NEEDED FOR XON/XOFF OR ETX/ACK
; *** TYPE HANDSHAKING.
RDRIN EQU SERIN ;*** <USUALLY SAME AS LISTOUT>.
PNCHOUT EQU SEROUT ;*** SELECT YOUR PUNCH PORT.

;SET ONLY ONE OF THE FOLLOWING HANDSHAKE CODES TRUE.
NONEH EQU TRUE ;*** SET TRUE IF PARALLEL OR NO HANDSHAKING.
ETXH EQU FALSE ;*** SET TRUE IF USING ETX/ACK HANDSHAKING.
XONH EQU FALSE ;*** SET TRUE IF USING XON/XOFF HANDSHAKING.

;SET <BUFFR> TO SIZE OF YOUR PRINTER BUFFER.
BUFFR EQU 80 ;*** SIZE OF PRINTER BUFFER
               ;***** MAXIMUM SHOULD BE 255 *****
               ;*** <USED BY ETX/ACK PROTOCOL>.

;THE FOLLOWING ARE FOR THOSE WHO ARE USING SUPER-PHANTOM
;OR SOME OTHER FORM OF BANK-SELECT SCHEME FOR ALLOWING
;RAM TO OVERLAY THE SOLOS ROM/RAM AREA. SET <BANK> TO <FALSE> IF
;NOT USING THIS FEATURE. IF <TRUE>, THEN FILL IN THE OTHER
;EQUATES SO BANK-SWITCHING CAN BE DONE PROPERLY.

BANK EQU FALSE ;*** SET TRUE IF USING BANK-SELECT.
BNKPRT EQU OFFH ;*** BANK-SELECT PORT #.
SOLON EQU 00 ;*** CODE TO ACTIVATE SOLOS.
SOLOFF EQU 01 ;*** CODE TO DE-ACTIVATE SOLOS
               ;*** AND ACTIVATE OVERLAY RAM.

;THE FOLLOWING ARE SOLOS ENTRY POINTS.
AINP EQU SOLOS+22H
AOUT EQU SOLOS+1CH
SINP EQU SOLOS+1FH
SOUT EQU SOLOS+19H

IF EQU SCRN16
SYSRAM EQU SOLOS+800H ;LOCATION OF SOLOS SYSTEMS RAM.
SCRNSZ EQU 16 ;SCREEN SIZE.

16
ENDF
IF NOT SCR宁16 ;IF 24X80...
SYSRAM EQU SOLOS+0F80H ;BOB HOGG'S MCVIDEO IS ASSUMED.
SCRNSZ EQU 24 ;ASSUME 24X80.
ENDIF
SOLSDP EQU SYSRAM+0BH ;LOCATION OF <SPEED> BYTE.
UIPRT EQU SYSRAM ;LOCATION OF CUSTOM INPUT ROUTINE.
UOPIE EQU SYSRAM+2 ;LOCATION OF CUSTOM OUTPUT ROUTINE.

;ASSIGN KEY/CODE VALUES.
MODE EQU 80H ;MODE-SELECT KEY.
CTRLC EQU 03 ;CONTROL/C.
CTRLG EQU 07H ;CONTROL/G
DELETE EQU 7FH ;DELETE KEY.
LOAD EQU 8CH ;LOAD KEY.
LARROW EQU 81H ;LEFT-ARROW KEY.
FFFD EQU 0CH ;FORM-FEED CODE.
BS EQU 08H ;NORMAL BACKSPACE.
ETX EQU 03H ;ETX CODE.
ACK EQU 06H ;ACK CODE.
XON EQU 11H ;XON CODE.
XOFF EQU 13H ;XOFF CODE.
ESC EQU 1BH ;ESCAPE CODE.
LF EQU 0AH ;LINE-FEED.
CR EQU 0DH ;CARRIAGE-RETURN.

******************************************************************************

;The following equates are automatically changed by MSIZE.

BIOS EQU 5300H+(MSIZE-24)*1024 ;$$
CCP EQU BIOS-1600H ;$$
BDOS EQU CCP+800H ;$$
USER EQU BIOS+700H ;$$For double/quad density
;USER EQU BIOS+500H ;$$For single density
OFFSET EQU 2000H-BIOS ;$$To overlay SYSGEN IMAGE
IOBYTE EQU 3 ;$$Storage location
<<IOBYTE NOT USED >>.

******************************************************************************

**** BEGINNING OF ACTUAL PROGRAM ****
******************************************************************************

ORG USER ;Start of USER AREA

******************************************************************************
; JUMP TABLE - Jumps MUST remain here in same order.

CINIT JMP CBOOT ; COLD BOOT INIT.
WINIT JMP WBOOT ; WARM BOOT INIT.
JCONST JMP CONST ; CONSOLE STATUS.
JCONIN JMP CONIN ; CONSOLE INPUT.
JCONOUT JMP CONOUT ; CONSOLE OUTPUT.
JLIST JMP LIST ; LIST DEVICE.
JPUNCH JMP PUNCH ; PUNCH DEVICE.
JREADER JMP READER ; READER DEVICE.
JLISTST JMP LISTST ; LIST STATUS.

; THIS DATA AREA MUST BE HERE IN THIS ORDER.
LENUE: DW USRLEN ; Length of USER AREA
USRIOB: DB 80H ; "I" Initial IOBYTE
HSTYPE: DB OFFH ; "H" Handshaking type
NULLS: DB NULLS ; "J" Printer nulls
LINES: DB PAGEL ; (#lines)/PAGE
PMAX: DB MAX ; (MAX # LINES)/PAGE.

*******************************************************************************
*******************************************************************************

; THE FOLLOWING INITIALIZED STORAGE AREA CONTAINS ITEMS
; THAT ARE CHANGED DYNAMICALLY BY THE PROGRAM.
; *** THE USER MAY WISH TO CHANGE THE DEFAULT VALUES GIVEN.

ATOGGLE DB 00 ; *** AUTO-DISPLAY TOGGLE INITIALLY OFF.
LToggle DB 01 ; *** LINE-TOGGLE INITIALLY ON.
PToggle DB 00 ; *** PRINTER INITIALLY OFF.
PAGE DB PAGEL ; *** PAGE SET TO FULL LENGTH.
PAPER DB 'F'; *** 'F'=FANFOLD 'S'=SHEET.
LNCNT DB SCRNSZ-1; *** LINE-COUNT FOR SCREEN SCROLL.
CHRCNT DB BUFFR-3 ; *** CHARACTER COUNT <FOR ETX/ACK>.

*******************************************************************************
*******************************************************************************

; <BON> AND <BOFF> DO BANK-SELECTION OF SOLOS.

IF BANK ; <ONLY NEEDED IF BANK=TRUE>.

BON:
    PUSH PSW
    MVI A,SOLOH ; PRESERVE "A" AND FLAGS.

BSEL:
    OUT BNKPRT
    POP PSW
    RET

BOFF:
    PUSH PSW
    MVI A,SOLOOFF ; PRESERVE "A" AND FLAGS.

; GET CODE FOR TURNING SOLOS ROM/RAM "OFF".
JMP BSEL ;DO IT.

ENDIF

;CONSOLE STATUS ROUTINE... USES CONSIM DEVICE.

*** <CONST> RETURNS ZFLAG ON AND "A"=00 WHEN NOT READY.
*** IT RETURNS "A"=OFFH AND <CHARAC>=DATA WHEN READY.
***
*** SPECIAL CHARACTERS ARE CONVERTED/INTERPRETED AS NEEDED,
*** AND THEN A RETURN IS MADE AS IF NOTHING HAPPENED.
*** THIS FEATURE ALLOWS THE SPECIAL KEYS TO BE USED ANYTIME
*** WITHOUT CAUSING ANY DIFFICULTY WITHIN CP/M.
***
*** SPECIAL FEATURES INCLUDE:
*** MODE-SELECT = CONTROL/C
*** LEFT-ARROW = BACKSPACE
*** DELETE = DELETING BACKSPACE UNDER CP/M
***
*** THE LOAD KEY ACTS AS A SPECIAL "ESCAPE" KEY:
*** LOAD/P = TOGGLE PRINTER ON/OFF
*** LOAD/S = SHEET PAPER
*** LOAD/F = CONTINUOUS FAN-FOLD PAPER
*** LOAD/^ = FAN-FOLD WITH PAGE-LENGTH
*** LOAD/A = TOGGLE SCREEN AUTO-PAGING <(HIT ANY KEY TO RESUME)>
*** LOAD/O THRU LOAD/9 = SET SCROLLING SPEED. 0=FASTEST
*** LOAD/MODE = RETURN TO SOLOS
*** LOAD/LOAD = PAUSE SCREEN
*** LOAD/ANYTHING ELSE = PAUSE SCREEN

CONST:
CALL XXAINP ;DO INPUT.
STA CHARAC ;STORE RETURNED CHARACTER...
 ; ZFLAG IS ON AND BOTH "A" AND <CHARAC>
 ; ARE = 00 IF NO CHARACTER READY.
RZ

CONST0:
CPI MODE ;CONVERT MODE-SELECT
JNZ CONST1 ;TO A CONTROL/C.
MVI A,CETLC
JMP GOTIT ;WE GOT IT... RETURN TO CP/M.

CONST1:
CPI DELETE ;CONVERT DELETE TO BACKSPACE.
JNZ CONST2

BSOUT:
MVI A,BS ;(A NORMAL-TYPE BACKSPACE).
JMP GOTIT ;WE GOT IT... RETURN TO CP/M.

CONST2:
CPI LARROW ;ALLOW LEFT-ARROW AS BACKSPACE TOO.
JZ BSOUT
CONST3:
  CPI  LOAD ;LOAD KEY IS LIKE AN "ESCAPE"...
  JZ   IN1
GOTIT:
  ANI  7FH ;STRIP OFF MSB.
  STA  CHARAC ;SAVE IT.
LISTST:
  MVI  A,OFFH ;TELL CP/M WE GOT A CHARACTER.
CBOOT:
WBOOT:
  RET ;RETURN TO CP/M.

IN1:
  CALL XXAINP ;<USE SOLOS>.
  JZ   IN1 ;WAIT IF NONE READY.
  ANI  5FH ;STRIP IT.
        ;<<CONVERT LOWER TO UPPER CASE>>.
  CPI  '0' AND 5FH ;LOAD/0 = FASTEST SPEED.
                    ;LOAD/9 = SLOWEST SPEED.
  JC   IN2 ;IF LESS THAN '0', NOT A SPEED.
  CPI  ('9'+1) AND 5FH ;IF >9 THEN NOT A SPEED.
  JNC  IN2 ;ALL WE NEED IS LAST PART.
  ANI  OFH ;ONLY NEEDED IF BANK=TRUE.
  IF   BANK
  CALL BON ;TURN SOLOS ROM/RAM ON.
ENDIF
  STA  SOLSPD ;STUFF IT INTO SPEED REGISTER.
  IF   BANK ;TURN SOLOS ROM/RAM OFF.
  CALL BOFF
ENDIF
  JMP  NOTHING ;ACT LIKE NOTHING HAPPENED.

IN2:
  CPI  'S' AND 5FH ;LOAD/S = SET FOR SINGLE-SHEET.
  JNZ  IN3
SETUP:
  PUSH PSW ;SAVE CODE.
  MVI  A,1 ;<LOAD/S AND LOAD/^ SET LTOGGLE ON>.
SETUP1:
  STA  LTOGGLE ;RECOVER CODE.
  POP  PSW
  STA  PAPER ;SET PAPER TYPE.
        ;<CONDITIONAL CLEAN-UP>
  LDA  LINES ;SET UP FRESH PAGE.
  STA  PAGE
ENDIF
  IF   NEWPAGE ;<CONDITIONAL PAPER-EJECT>.
  PUSH  B
  MVI  C,FFEED
  CALL  LIST ;FORCE A PAGE EJECT ON LIST DEVICE.
ENDIF
  JMP  NOTHING ;ACT LIKE NOTHING HAPPENED.

IN3:
CPI   '^' AND 5FH ;LOAD/^ = SET FOR FAN-FOLD/P.
JZ    SETUP ;FINISH ELSEWHERE.

IN4:
CPI   MODE AND 5FH ;LOAD/MODE-SELECT = GO TO SOLOS.
JNZ   IN5
IF    BANK ;TURN ON SOLOS FIRST.
CALL  BON
ENDIF
JMP   SOLOS+4 ;USE PROPER RETURN POINT.

IN5:
CPI   'A' AND 5FH ;LOAD/A = TOGGLE AUTO-PAUSE.
JNZ   IN6
MVI   A,SCRNSZ-1 ;RESET LINE-COUNT.
STA   LNCNT
LDA   ATOGGLE ;GET CURRENT STATUS.
INR   A ;UPDATE IT.
STA   ATOGGLE ;STORE IT.
JMP   PNOTHING ;ACT LIKE NOTHING HAPPENED.

IN6:
CPI   'F' AND 5FH ;LOAD/F = CONTINUOUS FAN-FOLD PAPER.
JNZ   IN7
PUSH  PSW ;SAVE CODE.
XRA   A ;RESET LTOGGLE.
JMP   SETUP1 ;FINISH UP ELSEWHERE.

IN7:
CPI   'P' AND 5FH ;LOAD/P = TOGGLE PRINTER.
JNZ   NOTHING ;LOAD/ANYTHING ELSE = PAUSE.
LDA   PTOGGLE ;GET CURRENT P-TOGGLE STATUS
INR   A ;TOGGLE IT.
STA   PTOGGLE ;MAKE THAT PERMANENT.

PNOTHING:
IF    SHOW

PUSH  B ;SAVE TOGGLE VALUE.
PUSH  PSW
MVI   C,ESC ;SEND ESCAPE CODE TO SOLOS.
XRA   A
CALL  XAOUT
MVI   C,CTLG ;SEND CONTROL/G TO SOLOS
XRA   A
CALL  XAOUT ;<ALLOWS REVERSE-VIDEO>.
POP   PSW ;RECOVER TOGGLE VALUE.
ANI   01 ;LOOK AT LSB ONLY.
ADI   'O' OR 80H ;SEND REVERSE-VIDEO "0" OR "1".
MOV   C,A ;PRINT IT FROM C.
XRA   A
CALL  XAOUT
POP   B

ENDIF

NOTHING:
XRA   A ;SET ZFLAG FOR SURE.

SCHARAC:
ORA A ;SET ZFLAG IF A=00
PUSH PSW ;(SEVERAL ROUTINES USE <SCHARAC>).
XRA A
STA CHARAC ; RESET <CHARAC>.
POP PSW
RET ;ZFLAG IS ON AND "A"=00.
;(IF ENTERED THROUGH <NOTHING>).

*******************************************************************************
CONSOLE INPUT ROUTINE.

*** THE CONSOLE INPUT ROUTINE RELIES HEAVILY UPON THE
*** <CONST> ROUTINE. ALL CODE CONVERSIONS AND CHECKS FOR
*** SPECIAL CHARACTERS ARE DONE INSIDE <CONST>.

CONIN:
LDA CHARAC ;CHECK FOR A CHARACTER FROM CONST.
CALL SCHARAC ;RESET <CHARAC>.
RNZ ;(AND SET Z FLAG IF NO CHARACTER).
CALL CONST ;IF WE GOT SOMETHING, LOOK NO FURTHER.
JMP CONIN ;ULTIMATELY IT COMES FROM CONST.

*******************************************************************************
CONSOLE OUTPUT ROUTINE.

*** BACKSPACES ARE PROPERLY HANDLED.
*** MULTIPLE CARRIAGE-RETURNS ARE IGNORED.
*** SEE CONST FOR DETAILS ON PAUSING DISPLAY AND SETTING SPEED.
***

CONOUT:
LDA PTOGGLE ;CHECK PRINTER TOGGLE.
ANI 1 ;ONLY LSB IS USED.
CNZ LIST ;IF ON, LIST IT.

OUT1:
PUSH B ;SAVE ORIGINAL.
MOV A,C ;WE NEED IT IN "A" FOR COMPARSES.
CPI BS ;SOL NEEDS HELP WITH BACKSPACE.
JNZ OUT2 ;CONVERT TO SOL LEFT-ARROW.
MVI C,LARROW

OUT2:
CPI CR ;DON'T ALLOW CR/CR.
JNZ SENDIT ;RECOVER LAST CHARACTER DISPLAYED.
LDA LCHAR
CPI CR
JNZ SENDIT ;TRAP CR/CR
JMP PO PB ;JUST SKIP DISPLAYING THEM.

SENDIT:
STA LCHAR ;SAVE LAST CHARACTER DISPLAYED.
MVI A,CONOUT ;OUTPUT TO SELECTED DEVICE.
CALL XAOUT ;LINE FEEDS ARE SPECIAL.
CPI LF
JNZ POPB ;GET AUTO-DISPLAY TOGGLE.
LDA ATOGGLE
ANI 1 ;JUST LSB.
JZ POPB ;IGNORE IF OFF.
LDA LNCNT
DCR A ;UPDATE LINECOUNT.
STA LNCNT
JNZ POPB ;NO ACTION UNTIL = 0.
MVI A,SCRNSZ-1 ;RESET LINECOUNT TO SCREENSIZE.
STA LNCNT
CALL CONIN ;...PAUSE UNTIL A KEY IS HIT.
;ALSO ALLOWS SPECIAL KEYS.

POPB:
POP B ;RECOVER ORIGINAL.
RSTORE:
MOV A,C ;CP/M LIKES IT IN "A".
RET

*****************************************************************************
*****************************************************************************

;LIST ROUTINE.

LIST:
PUSH B ;SAVE BC.
FFCHK:
MOV A,C ;CHECK FOR FORMFEED.
CPI FFEED
JNZ LISTIT ;IF NOT FORMFEED, CONTINUE.

FFCHK1:
LDA LINES ;GET NORMAL PRINT LENGTH...
CMA ;MAKE IT NEGATIVE...
INR A ;...TWO'S COMPLEMENT TYPE...
MOV B,A
LDA PMAX ;GET <MAX> FORM LENGTH...
ADD B ;CALCULATE MAX-LINES...
MOV B,A
LDA PAGE ;GET LINES LEFT TO PRINT...
ADD B ;CALCULATE LINES FOR FORMFEED.
INR A ;ALLOW A CR TOO.
PUSH PSW ;SAVE IT.
MVI A,OFFH
STA PAGE ;PREVENT "HICCUPS".
POP PSW ;RECOVER FORMFEED LENGTH.
MVI C,CR ;ALWAYS DO ONE CARRIAGE-RETURN.

FFLOOP:
PUSH PSW ;SAVE COUNT.
CALL LIST ;PRINT A LINEFEED (OR CR).
MVI C,LF ;NEXT TIME IT'S A LINE- FEED.
POP PSW ;RECOVER COUNT.
DCR A ;UPDATE IT.
JNZ FFLOOP ;DO LINE FEEDS AS NEEDED.

LWAIT:
LDA LINES ;RESET LINE COUNTER.
STA PAGE
LDA PAPER ;CHECK WHAT KIND OF PAPER...
CPI 'S' AND 5FH ;...IS IT SHEET?
JNZ POPB ;IF NOT, ALL DONE.
CALL CONIN ;IF SHEET, WAIT FOR ANY KEY TO BE HIT...
;...<ALLOW SPECIAL FUNCTIONS TOO>.
JMP POPB ;WHEN DONE, BE NEAT.

LISTIT:
IF XONH ;XON/XOFF IS DONE EARLY.

XOFFL:
MVI A,LISTIN ;IS XOFF BEING RECEIVED?
CALL XAINP
JZ CNTNU ;IF NOT, GO AHEAD.
ANI 7FH ;STRIP OFF MSB.
CPI XOFF ;MAKE SURE IT'S XOFF.
JNZ CNTNU ;IGNORE IF NOT.

XONL:
MVI A,LISTIN ;IS XON BEING RECEIVED?
CALL XAINP
JZ XOXL ;IF NOT, WAIT.
ANI 7FH ;STRIP OFF MSB.
CPI XON
JNZ XOXL ;WAIT UNTIL XON.

CNTNU:
;OK, GO ON.
ENDIF ;END OF XONH ROUTINE.
CALL XLAOUT ;USE SELECTED LIST DEVICE.
IF ETXH ;USE ETX/ACK PROTOCOL.
;THIS VERSION INCLUDES CHECK FOR ESC
;SO ESCAPE CODES GET THROUGH OK.

MOV A,C ;GET CHARACTER INTO "A".
CPI ESC ;ESCAPE NEEDS HELP.
JNZ ETX1
LDA CHRCNT ;GET CURRENT COUNT.
ADI 3 ;ADD 3 TO IT...
STA CHRCNT ;... SO ESC SEQUENCE MAKES IT.

ETX1:
LDA CHRCNT ;GET CHARACTER COUNT.
DCR A ;UPDATE IT.
STA CHRCNT ;SAVE IT.
JNZ LISTON ;IF NOT EMPTY, GO HOME MAYBE.
MVI A,BUFFR-3 ;RESET CHRCNT.
STA CHRCNT

ETXL:
MVI C,ETX ;SEND ETX CODE.
CALL XLAOUT ;USE SELECTED LIST DEVICE.

ACKL:
MVI A,LISTIN ;GET INPUT FROM LIST DEVICE.
CALL XAINP
JZ ACKL ;WAIT IF NECESSARY.
ANI 7FH ;STRIP OFF MSB.
CPI ACK ;IF NOT ACK...
JNZ ACKL ;... WAIT 'TIL IT IS.
ENDEF ;END OF ETXH ROUTINE.

LISTON:
POP B ;RECOVER ORIGINAL CHARACTER.
PUSH B ;PUT IT BACK FOR NOW.
IF DONULLS ;<CONDITIONAL> NULL GENERATOR.
MVI A,CR ;DO NULLS AFTER <CR>.
CMP C
JNZ NONULLS

;***** SPECIAL NULL ROUTINE *****
LDA NULLOC ;GET NULL COUNT.
MOV B,A ;<B> IS HARMLESS.
PUSH B ;DON'T LOSE IT.

NULLLOOP:
MVI C,0 ;SEND A NULL.
CALL XLAOUT ;USE SELECTED LIST DEVICE.
PDP B ;RECOVER THE COUNTER.
DCR B ;UPDATE COUNT.
PUSH B ;SAVE IT.
JNZ NULLLOOP ;DO SEVERAL.
PDP B ;KEEP THE STACK CLEAN.

;***** END OF NULL ROUTINE *****
ENDEF

NONULLS:
LDA LTOGGLE
ORA A
JZ POPB

LFEED:
MOV A,C ;USE IT FOR COMPARES.
CPI LF ;LINE-FEEDS ARE ALSO SPECIAL.
JNZ POPB
LDA PAGE ;GET PAGE LENGTH LEFT.
DCR A ;REDUCE BY ONE.
STA PAGE ;STORE IT.
JNZ POPB ;IF DONE, BE NEAT.
JMP FFCHK1 ;DO A FORMFEED.

*****************************************************************************
*****************************************************************************

;PUNCH IS USEFUL FOR DRIVING MODEMS.

PUNCH:
MVI A,PNCCHOUT ;PUNCH IS WHATEVER YOU SELECTED.
JMP XAOUT

******
; READER IS USEFUL FOR LISTENING TO MODEMS.

READER:
MVI A,RDRIN ; READER IS WHATEVER YOU SELECTED.
CALL XAINP
JZ READER ; WAIT UNTIL YOU GET SOMETHING.
RET

; XAOUT ALLOWS ANY SOL PSEUDO OUTPUT PORT TO BE USED FOR OUTPUT.

*** ENTER WITH PSEUDO OUTPUT PORT # IN "A".
*** 0 = VDM
*** 1 = SERIAL PORT
*** 2 = PARALLEL PORT
*** 3 = CUSTOM DRIVER <<YOU MUST DEFINE CUSTOUT=CUSTOM ADDRESS>>

XAOUT:
MVI A, LISTOUT

XOUT:
MVI A, LISTOUT
IF BANK ; ONLY NEEDED IF BANK=TRUE.
CALL BON ; TURN SOLOS ROM/RAM ON.
ENDIF
PUSH B
MOV B, C ; SOLOS NEEDS IT IN "B".
CALL AOUT ; USE SPECIFIED PSEUDO PORT.
POP B
IF BANK ; ONLY NEEDED IF BANK=TRUE.
CALL BOFF ; TURN SOLOS ROM/RAM OFF.
ENDIF
MOV A, C ; CP/M LIKES IT IN "A".
RET

; XAINP ALLOWS ANY SOL PSEUDO INPUT PORT TO BE USED FOR INPUT.

*** ENTER WITH PSEUDO OUTPUT PORT # IN "A".
*** 0 = KEYBOARD
*** 1 = SERIAL PORT
*** 2 = PARALLEL PORT
*** 3 = CUSTOM DRIVER <<YOU MUST DEFINE CUSTIN=CUSTOM ADDRESS>>.

XXAINP:
MVI A, CONSIN ; PREPARE FOR CONSOLE INPUT.
XAINP:
    IF  BANK
    CALL  BON     ;<ONLY NEEDED IF BANK=TRUE>.
    ENDIF
    ;TURN SOLOS ROM/RAM ON.

XAINP1:
    CALL  AINP    ;USE SOLOS AINP PSEUDO PORT.
    IF  BANK    ;<ONLY NEEDED IF BANK=TRUE>.
    CALL  BOFF   ;TURN SOLOS ROM/RAM OFF.
    ENDIF
    RET

************************************************************************************
************************************************************************************
************************************************************************************
*****    END OF PROGRAM  *****
************************************************************************************
************************************************************************************
************************************************************************************

;THE FOLLOWING ARE UNINITIALIZED DATA.

CHARAC  DB    00     ;*** CHARACTER FROM CONST.
LCHAR   DB    00     ;*** LAST CHARACTER DISPLAYED.

*** THE FOLLOWING DETERMINE THE AMOUNT OF USER AREA LEFT.

ENDUSER  EQU   $
MAXLEN   EQU   512
USRLEN   EQU   ENDUSER-USRLEN
LFTOVR   EQU   MAXLEN+2-USRLEN
;IF <LFTOVR>=FXXX THEN OPTIONS SELECTED HAVE CAUSED
;LENGTH TO EXCEED 512 BYTES. TRY CHANGING OPTIONS.

************************************************************************************
************************************************************************************
************************************************************************************

END

C>
April 5, 1984

Stanley M. Sokolow, D.D.S.  
PROTEUS  
1690 Woodside Road, Suite 219  
Redwood City, CA 94061

Dear Stan,

The accompanying diskette contains Franz J. Hirner's General Ledger system programs in Processor Technology's EDBASIC source. I obtained printed listings from Mr. Hirner in April, 1983, and have been entering and testing the programs since then. You may copy the diskette for Proteus members as you see fit. If you need a formal release from Mr. Hirner, please contact him.

File 'message2' on the diskette contains information about the system. Enter 'Bo' to see it during START.UP or print it directly on screen or on hardcopy. The original reference to the system appeared in Proteus September/October/November/December 1981, Vol. 4, #5/6, pp. 3-5. It is a very nice addition to my collection of application programs. I hope other Proteus members like the system as much as I do.

Sincerely,

Leonard

Leonard E. Cole  
644 San Fernando Ave.  
Berkeley, CA 94707  
Phone: 415-527-2110

cc: Mr. Franz J. Hirner  
631 Matsonia Drive  
Foster City, CA 94404

Editor's note...

Please contact Franz Horner or Leonard Cole to obtain this disc.
SOL Is A Valuable Antique

WILLIAM D. LOUGHMAN

20 July 1984

Dr. Stan Sokolow
PROTEUS, Inc.
1690 Woodside Road, Suite 219
Redwood City, CA 94061

Dear Stan:

As you have said, Proteus may 'wind down'. Given your own steady investment in time and energy, to say nothing of the head-over-heels advancement of the entire micro-computer industry, that can't be too surprising. But there remains a place for Proteus, and I cast my vote for continuation. Perhaps as a quarterly, for those of us who value it enough to pay (and contribute). Perhaps expanded, as a newsletter for other 'antique' machines. There are only two publications that have been of outstanding value to me: Proteus News, and Dr. Dobbs (I subscribe to over a dozen more).

So in my view, Proteus should continue. For one, I'll pay increased fees if necessary. The SOL remains a practical machine, and like you, Stan, I intend to keep my investment useful. It continues to be a capable and very adaptable machine.

As to contributing ... Well, yes, this letter is years overdue. We've talked on the phone many times over the years, but I've written only one letter (a CP/M BIOS modification). That generated several fast responses, so I felt pleased to have helped others, and guilty that I hadn't done more. In some measure, this letter is meant to allay my guilt. In some measure, I hope its tone will encourage others to write.

To add some substance to my own desire for more 'collegiality', let me detail a few of the areas in which I might help (local) SOL owners.

I own the Buss Probe (an auxiliary board from Jade which displays status of and signals on the S-100 bus), the ParaSol board and software (a PTCO machine debugger), a logic probe, and a fast triggered 'scope. My acquaintance with computer hardware is distant, but these, the Sol manual, and some thoughtful jiggling have helped me more than once to avoid expensive down-time.

My special delight, computer-wise, runs to assembler code. I've written BIOSs, full-screen editors, assemblers, disassemblers, telecommunication modules, and even a complete operating system. Time permitting, I could help those who have some problems with interfacing
the SOL to commercial software.

As to common commercial (or other) software, there are Things I Like, and Things I Don't Like.

Things I Like (and use a lot):
Spellbinder (word-processor - see below); Supercalc (spreadsheet); Modem7 (telecommunication); and the SOLOS monitor (MicroComplex mod).
In re Spellbinder: Their customer service is the pits. There is a bright spot on the Lexisoft stonewall however: The current Female Voice On The Phone. Don't try to bypass this gem, ask HER to solve your problems. She's good, and deserves a raise.

Things I Like (and use a little):
PILOT (Starkweather's 'Nevada' beta-test version, also Stok PILOT); NorthStar Basic (Soho CP/M conversion); PTCo BASIC (for the matrix functions); and fig-FORTH.

Things I Like (I think... but only a little experience with):
BDS 'C' (not quite Kernighan and Ritchie, but a huge base of programs; and it is fast).

Utilities that are 'the greatest':
DUXx (CP/M disk editor - Ward's second greatest gift); and CATxx (disk catalog - Ward's third greatest gift). Maybe best of all: like Modemxx, these useful and reliable programs are 'public domain'.

Which brings me to People I Like:
First, Stan Sokolow (alias SOLUS, alias PROTEUS). This newsletter and its past and ongoing contents justify that 'First'. Ward Christiansen, for reasons cited above. Bob Hogg (MicroComplex), for continuing to support the SOL. I have some of his upgrades; they work reliably and as advertised, and make a hell of a difference.

Finally, Things I Don't Like: (and avoid whenever possible):
The absolute FIRST dislike has to be dBaseII v.2.3B and v.2.4. Ashton-Tate is a marketing corporation, not a software author. The difference shows in their regard for both their product and their customers. I haven't the space here to say much more, but the product is UNRELIABLE. It trashes disk files in unpredictable ways, at unpredictable times. I've circulated a letter to PICONET and the San Francisco dBase User's Group detailing my experiences, and some experiences of others, and my conversations with Ashton-Tate. There is a popular book on dBase which supports my views, and so does an article in the current 'Computer Currents' (a Bay Area tabloid). Further, A-T's Chief Programmer called me on 22 June 84, confirming my views. Should any readers wish more information, I'll try to comply. SEND ENOUGH POSTAGE, with a SASE, to cover a 9-page document plus miscellaneous supporting material.

WordStar is another, though lesser, marketing triumph. Too big, somewhat clumsy, and 'busy-busy'. Wordmaster, Nevada EDIT, or even
Electric Pencil are better (faster) editors; Spellbinder is a better and faster word-processor. Spellbinder topped WordStar in a poll of 'computer-professionals' published last year in InfoWorld.

My final dislike is aimed at all those programs published by people who simply want their names in print. In my experience, at least half simply don't work. The rest often require extensive debugging. There is a rare well-written gem, and I am thankful for same. But they ARE rare. (My own contribution to PROTEUS has an error in the documentation - my fault - but the program works as written.)

All of the programs cited above, and dozens more, I've put onto my machine. The point here is NOT to toot my own horn, but rather to offer my experience to others. Several people have written PROTEUS, concerned that they couldn't get SuperCalc working. I have a CP/M BIOS (NorthStar) which allows one to run SuperCalc or dBase on a SOL.

Ask. PROTEUS/NEWS is a newsletter, after all.

So I promised to say more about the SOL. Indirectly, I just did. Perhaps the bottom line has something to do with the following:

I use two 32k Godbout boards with the MicroComplex VDM mods to get a 64k SOL, and an aging NorthStar disk system for over a half-megabyte external storage. Thus I have two slots and the connector atop the backplane for expansion (bank select, color, etc.; whatever I want). The serial and parallel ports don't have to be added, and I have cassette I/O too (for archive at 20% of disk costs). There are LOTS of newer machines that can't offer that, even if you pay extra. Sure, the SOL is aging...but like a fine wine. We all should age so well. If you don't believe me, ask Rob Hoco (MCsOLOS, McVideo, etc.) who has a wealth of machines to play with. His preference for real work remains...yes, Virginia...the SOL. My own friends often ask me to find/fix some problem for their 'modern' machines. I just don't have the heart to tell them the solution came out of a seven-year old 'antique'.

Sincerely,

Bill

William N. Loughman, Ph.D.
Burglary Notice

would you let this man, Glen T. Buie, work on your Sol? Just thought you would like to know where he got all of the spare parts from. I hope you let the Public know about this burglary.

(Anonymous)

Thank You.

Stolen computer gear found; man jailed

ARLINGTON — An Arlington man was arrested on suspicion of burglary Wednesday and $74,000 in computer equipment was recovered from his apartment, police said.

Deputy Police Chief Marion Rettig said investigators will consult with the district attorney’s office Thursday on possible charges to be filed.

“It’s most unusual for us to have computer equipment stolen, but I suspect we’ll see it more and more now that people are getting into the home computer business,” Rettig said.

Burglars went through the roof of Computer Port, 2142 N. Collins, during the weekend, and the burglary was discovered Monday, police said.

Taken in the theft were visual monitors, Apple II computers and Horizon computer units, police report.

Computers recovered; man arrested

ARLINGTON — Police here have recovered $74,000 in computer equipment in an apartment of an Arlington man arrested on suspicion of burglary.

Deputy Police Chief Marion Rettig said investigators were expected to consult today with the district attorney's office as to what charges might be filed.

Someone went through the roof of Computer Port, 2142 N. Collins, during the weekend, and the burglary was discovered Monday, police said.

Taken in the theft were visual monitors, Apple II computers and Horizon computer units, police report. The items were recovered in the attic of the suspect’s apartment, Rettig said.
Burglar’s defense

By JIM MORRIS
Star-Telegram Writer

An Arlington computer technician found guilty Wednesday of stealing $75,000 in computer equipment from his former employer claimed he was simply trying to convince his former boss to purchase a burglar alarm system.

The testimony of defendant Glen T. Buie failed to convince Judge Gordon Gray Wednesday in state district court, however.

After hearing testimony in Buie’s burglary trial, Gray pronounced the defendant “insane.”

“I’d send him to the pen in a New
York minute if he didn’t have the capability of making a decent citizen,” Gray ruled.

Gray assessed a six-year probated sentence and a $500 fine and ordered Buie to make $3,000 restitution. In addition, the judge said, Buie must spend 30 days in jail as a term of his probation.

Buie, 25, of 315-B N. Davis, was charged with burglarizing Computer Port in Arlington on Aug. 8. Buie admitted Wednesday that he had been fired immediately prior to the burglary. However, he said he decided to take the equipment because he had been unable to convince the store’s owner to install a burglar alarm system that Buie had been designing.

Buie denied he was seeking revenge on his employer and said he did not intend to keep the various computers, terminals, disc drives and monitors he swiped. The equipment was returned to the store after police discovered it in Buie’s apartment.

Buie said another man, Charles Rawls, entered the Computer Port building through an air conditioning duct Aug. 8 and then let him in through a side door. The two spent the next three hours loading the delicate equipment into their cars, Buie said, before taking it to his apartment.

When police searched the apartment Aug. 11, they found the equipment mostly undamaged in the attic.

“I told store owner Phil Dorcas he was inviting trouble without some sort of security system,” Buie testified Wednesday. “I did it to prove a point to the owner . . . that he needed a burglar alarm system.”

Defense attorney Joe Shannon asked Buie why he took so many valuable items.

“I wanted to show them that a burglary of that magnitude could wipe them out,” Buie said.

After Shannon had rested his case, Gray said he thought Buie was “definitely insane to do that. I think he was fixing to go into business for himself. It was a novel, ingenious defense.”

Dorcas earlier had testified that the equipment not been returned, Computer Port would have gone bankrupt. Dorcas said he had agreed with Buie that the store needed a security system. Dorcas said he had permitted Buie to continue working on the system even after Buie had been fired.

The store owner said only the most current, valuable items were taken. The older, “obsolete” equipment was untouched, Dorcas said.

Shannon noted that Buie voluntarily allowed police to search his apartment and made sure all the stolen items were returned to the store.

Assistant District Attorney Paul Conner asked why Buie initially had placed the blame on Rawls, who pleaded guilty to burglary last month and received a four-year probated sentence and was ordered to make $1,000 restitution. Buie had told police they should question Rawls and friends of his who were members of a “witchcraft cult.”

Buie said he wanted to “buy us some time” until he could talk to Dorcas about returning the equipment.

In his closing argument, Conner contended Buie stole the merchandise to start his own computer repair business. Computer Port manager Jay Weiner had testified that there is a “good market” for second-hand computers.

Shannon told Gray that Buie was guilty of using “darned poor judgment,” not burglary. “He’s obviously not a criminal because the planning was terrible,” Shannon said.

Conner, however, said Buie concocted his story to “save his hide.”

Buie declined to comment after being sentenced. On the witness stand, he said, “I went about this in a totally dumb, stupid manner. If I had it to do over again, I definitely would think of some other way.”
1. **TYPE OF OFFENSE**: Burglary Business  
   **DATE**: 8/10/81

2. **LOCATION OF OCCURRENCE**: 213 1/2 E. Collind

3. **WEATHER CONDITION AT TIME OF OFFENSE**: Hot and Dry

4. **ARRESTEE**: Glen T. Date and Charles E. Rawls

5. **VICTIM**: Computer Fort

### CODES:
- **F**: PROPERTY ROOM
- **L**: LABORATORY
- **R**: RELEASED TO OWNER OR REPRESENTATIVE

<table>
<thead>
<tr>
<th>ITEM</th>
<th>DESCRIPTION</th>
<th>CODE</th>
<th>PROP. USE ONLY</th>
</tr>
</thead>
<tbody>
<tr>
<td>08</td>
<td>Disk Drive Model 277 S/N 1465</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>09</td>
<td>Disk Drive Model 270 S/N 2382</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Disk Drive Model 270 S/N 14526</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Disk Drive Model 270 S/N 1451</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Disk Drive Model 270 S/N 1431</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Disk Drive Model 270 S/N 14680</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Disk Drive Model 270 513.2 S/N 2593</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Helios II Disk Drive Memory System S/N 501512</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Helios II Disk Drive Memory System S/N 501484</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Helios II Disk Drive Memory System S/N 501763</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Sol Terminal Computer Model 20 S/N 401189</td>
<td>P</td>
<td></td>
</tr>
</tbody>
</table>

6. **REMARKS**:

7. **CHAIN OF CUSTODY**
   - **ITEMS (8-18)**: RELEASED TO: Property Room  
     **DATE**: 8/13/81  
     **TIME**: 1400 AM/PM  
     **PURPOSE**: Evidence Storage

   - **ITEMS (8-18)**: RELEASED TO:  
     **DATE**: 8/15/81  
     **TIME**: 1000 AM/PM  
     **PURPOSE**: Return to Owner

   - **ITEMS (8-18)**: RELEASED TO:  
     **DATE**: / /  
     **TIME**: / AM/PM  
     **PURPOSE**:

   - **ITEMS (8-18)**: RELEASED TO:  
     **DATE**: / /  
     **TIME**: / AM/PM  
     **PURPOSE**:

   - **ITEMS (8-18)**: RELEASED TO:  
     **DATE**: / /  
     **TIME**: / AM/PM  
     **PURPOSE**:

8. **ITEMS ENTERED INTO EVIDENCE BY**:  
   **D.M. Sustaire**  
   **DATE**: 8/13/81  
   **TIME**: 1400 AM/PM
Announcements

1) COMPUTER LANGUAGE is a new magazine especially for people who write serious code. It covers major development in the software design field. For more information, contact Computer Language, 2443 Fillmore St, 346 San Francisco, CA 94115.

2) World Disk Drives has purchased the manufacturing rights to all Siemens 8" disk drives, including the FDD100-8 and the FDD200-8. Special prices are available to clubs.

SIEMENS FDD200-8 (single or double density)
1-50 -------- $169.00
51-100-------- $159.00
100+ -------- $SPECIAL DISCOUNT DEPENDING ON SIZE

SIEMENS FDD100-8 (double density)
1-50 -------- $154.00
51-100-------- $144.00
100+ -------- $SPECIAL DISCOUNT DEPENDING ON SIZE

ALSO! Included in this special offer: (1 of each with each drive)
*1---COUPON FOR 30% OFF ON PARTS
*2---60 DAY GUARANTEE
*3---COUPON FOR 1 DRIVE REPAIR FOR $80.00
*4---1 FREE OPERATORS MANUAL (ORIGINAL)
*5---ORIGINAL MANUFACTURERS MAINTENANCE MANUAL
(INCLUDES SCHEMATIC) FOR ONLY $15.00

For more information contact World Disk Drives, P.O. Box 2000-154, Mission Viejo, CA 92690; (714) 855-1761.

3) Electronic Specialists, Inc., now offer a KLEEN LINE MODEM protection. Models are available for standard 4 pin telephone modular connectors (RJ-11) and wider professional 8 pin connectors (RJ-45). These are intended to suppress software altering telephone and power line spikes caused by lightning, spherics, or phone office switch gear. For more information, contact Frank Stifter, Electronic Specialists, Inc., 171 South Main Street, Natick, Mass. 01760, (617)655-1532.

4) This will be the last issue for Volume 6 of PROTEUS NEWS. PROTEUS members who have paid dues for 1984 (amount same as 1983) will continue to receive flyers and announcements at various times throughout the year, however a formal newsletter will not be assembled unless more contributions from members are received.
TABLE OF CONTENTS

A Farewell Note by Stan Sokolow (Editor) ..................... 1
McZol: The Z-80 Upgrade For SOL From MICROCOMPLEX .......... 3
Unclassified Ad .................................................. 5
Super-User Area for Lifeboat CP/M by Gr. McGahee ............... 6
Hirner's General Ledger System by Leonard Cole ................ 28
SOL Is A Valuable Antique by Wm. Loughman ..................... 29
Burglary Notice .................................................. 32
Announcements .................................................... 35

PROTEUS/NEWS

A news journal for owners and users of Processor Technology Corporation computer equipment. Published by Proteus, 1690 Woodside Road, Suite 219, Redwood City, California 94061-3483, USA, telephone (415) 368-2300.

Submit items for publication to Proteus News, Attn: Stan Sokolow, 1690 Woodside Road, Suite 219, Redwood City, California 94061-3483, USA. Please make submissions as camera-ready as possible by using a fresh, black ribbon and typing single-spaced.

Copyright (C) 1984 by Proteus. All rights reserved. Permission is hereby granted to reproduce any computer programs contained herein, provided that Proteus and the program's author are given credit.

FROM:
PROTEUS
1690 WOODSIDE ROAD, SUITE 219
REDWOOD CITY, CALIFORNIA 94061-3483
USA

FIRST CLASS MAIL

James D. McElroy
2826 Crest Ave. North
Allentown, PA

18104