PROTEUS / NEWS

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

FORMERLY SOLUS NEWS

March/April 1981 PUBLISHED BIMONTHLY BY PROTEUS, 1690 WOODSIDE ROAD, SUITE 219, REDWOOD CITY, CA 94061, USA Vol. 4, #2
Single Issue...$4 (US)

NEWS FROM THE COMPUTER FAIRE

by Stan Sokolow

Through the courtesy of the Computer Faire management, Proteus had a meeting room and exhibit booth in the 6th West Coast Computer Faire, April 3-6, in San Francisco, California. This year I had a much more pleasant experience at the Faire than in the past, because I didn't try to do too much. Many members came by to say Hello and ask questions. Numerous newcomers also came by, some who recently bought a Sol and others who just never knew we existed.

The Faire was larger than ever. Over 30,000 registrations were sold--more than they expected. The Faire ran out of programs on Saturday and had to find a printer open on weekends to do a rush job to accommodate the expected Sunday crowd. Pre-Faire radio coverage was excellent. One of the large retailers in the San Francisco area gave many plugs to the Faire in the two weeks before. I guess it paid off for them, because I saw two young men trucking in dozens of Apple computers and overheard that they sold 30 Apples on Saturday alone.

There were a few new computers on exhibit this year, but definitely less excitement was shown than in the early days of personal computing. The biggest interest was around the new Osborne portable system. This is a Z-80, 64K, system with two minifloppies, keyboard, and 5" video monitor, all in a rugged aluminum case that closes up to be carried like luggage. They plan to have a battery pack for it, which will fasten onto the side and allow it to operate 5 hours on one charging. The idea is that business people will be able to carry the computer on-board with them on airline flights, and use the system in the air. (FCC and CAB approval is being sought by Osborne.) The size meets CAB regulations for under-the-seat carry-on luggage. Another interesting point is that the system sells for less than $1800, quantity one, including CP/M and application software tools (a wordprocessor, Microsoft BASIC, a VisiCalc-like program for financial worksheets, etc.).

Another interesting machine was the Micro-Expander, which we've mentioned before in Proteus/News. This is a Sol-like configuration (without the walnut sides though) containing a Z-80, IEEE standard S-100 bus with 4 slots, real-time clock, serial & parallel ports, TRS-80 format cassette interface, 24x80 color video interface, integral keyboard, and an internal expansion bus allowing memory and peripheral expansion without using up S-100 bus slots. Including a 64K S-100 memory board, they were selling this computer for $1875 at the Faire (reg. $2200) including Microsoft BASIC.

It is a high price for a bare-bones computer, but when you put a disk and monitor on it, you have a cost-effective system that has more features than any other on the market near that price. They are talking about adding RGB color output, which allows vivid color display available only on high priced graphics equipment. The signals already are available on the computer; all they need is to add a connector.

The surprising thing about both of these new computers is that they are both products of the same man who designed the Sol! The same theme can be seen running through all 3 computers: the all-in-one package concept of terminal integrated into a quality computer system. Lee Felsenstein has been prolific, and we hope that he finally reaps some financial rewards commensurate with the quality of his work. The Apple boys (Wozniak and Jobs) made a clever little machine, but their Apple II can't compare to the computers done by Lee. Maybe this time Lee will strike the silicon gold-mine. (Wozniak and Jobs began Apple while in their late teens and early twenties. A few years later they were multi-millionaires when Apple became a publicly-owned corporation.)

For more information on these new computers, contact Micro-Expander, Inc., 6835 W. Higgins Ave., Chicago, ILL 60656, (312) 792-1196, and Osborne Computers Corporation, 26500 Corporate Ave., Hayward, CA 94545.

Another booth at the Faire caught my attention, and I can recommend these people since I have subsequently used their service. I have an Epson MX-80 printer and a Diablo daisy-wheel printer. The cost of keeping these printers fed with ribbons has been significant. The Epson ribbon cartridge lists for about $18 and Diablo-compatible ribbons go for $6 to $8. Finally, I found a company that will reload these cartridges with fresh ribbon, and they will handle small quantities. I had an MX-80 ribbon reloaded for only $4.50 and HyType II ribbons refilled for 2.60 to 3.12, depending upon the type of ribbon. (The brown carbon-film "multi-strike" ribbon looks great on my ivory letterhead. It was only $3.12.)

The company that did this is American Ink Products Company, 527 Howard Street, San Francisco, California 94105, (415) 982-0161. The prices I quoted may be slightly higher now, because there was a 10% discount at the Faire. If you ask for it, maybe they'll give you the discount on your first order. They have a complete price list showing all of the ribbons they process. Even if your printer isn't on the list, they may be able to quote a price. My order was filled promptly and the cartridges were all returned in sealed plastic bags to preserve freshness. I recommend them highly.

Another eye-catching booth was the Japanese computer exhibit. Strikingly bold color displays (RGB no doubt), well made consumer-electronics-type cases, and some familiar names, like "Sharp". They aren't ready to push into the American market, but they've already pushed American companies like Commodore out of the Japanese market for personal computers. When they are ready to jump in here, look out. Already they are strong competition in the printer market (Epson, C.Itoh). I hear Centronics is in trouble due to Japanese competition, and I don't doubt it. Epson is air-shipping literally tons of little printers into the US at a price that Centronics can't match.

I N D U S T R Y R U M O R S

Datamation magazine (April 1981) reports that IBM has denied the rumors that it is working on a small computer venture with the Japanese firm of Matsushita, but the Datamation reporter claims IBM will introduce a desk-top computer this summer. It is aimed directly at the Apple, RadioShack, and Commodore computers. Rumored to have a price tag under $5000, it supposedly will be an Intel 8088 processor.

What is Dr. Ken (UCSD Pascal) Bowles up to lately? Datamation says he's started a company called TeleSoftware Inc. to develop the first commercial "Ada" compiler by early 1982. "Ada" is the language adopted by the Department of Defense for all new government projects. The DoD has invested $10 million in developing Ada and will require it to be the computer language used by all Defense contractors on government work. It is bound to be the industry standard in the coming years. Datamation says Bowles moved to Ada due to the difficulties in using Pascal for distributed processing. Pascal was strongly influential in the design of Ada language.

Datamation (same issue) also has an article on DBMS (data-base managment systems) for microcomputers, which gives a brief but favorable description of the Micro Data Base Systems, Inc., package called "MDBS", which we've announced in prior issues of this newsletter. However, Computer Decisions magazine (pages 48-52, April 1981) reports on a study by Andrew Sharp of Tab Products, Inc., benchmarking MDBS on an inventory application he wrote for in-house use at Tab.

Using an 8085-based CP/M system with Microsoft BASIC, it took 9 seconds for the system to locate a record in a 1300 item file, 23 seconds to insert a single record into that file, and 63 minutes to insert a file of 260 records into the 1300 item file. Using MDBS's estimates, an analogous insert into a 10,000 item file would take 100 hours, Sharp estimated. And this is using a fast hard-disk drive, not a floppy!

For comparison, we use MailMaster to maintain our Proteus/News subscriber list. In the worst case, it would take about 2 seconds for our Sol to locate a record in a 1300 item file, and I believe the time would not be much different for a 5000 item file. And this is using a floppy disk system.

With all of its versatility, the MDBS system plus the Microsoft BASIC just must have too much processing overhead for the task. Sharp and the article's author Hazelton believe that the 8-bit word is the culprit, because the IBM 370/148 is no more than 20 times faster than the 8085 but doesn't suffer this slowness in that proportion.

Before you sink $1000-$2000 into MDBS and its options, be sure your application needs that much versatility. It may make the system so slow that it is unusable.

Product Announcement

S O L / N O R T H S T A R U S E R ' S G U I D E

Remember how difficult it was to know where to begin when you were first studying your user's manuals? Sol owners with NorthStar disk drives are faced with the Sol manual, the Solos operating system manual, the NorthStar BASIC manual, the NorthStar DOS manual, and so on. It can be difficult to know how to begin, because you need to know what you are doing before you can figure it out.

The Lindsay Group, back in 1978, tried to guide the novice through the learning process with a 43 page booklet called the "Sol 20 Meets the North Star Disk Drive." This covers the basics of how to turn on the machine, how to talk with the disk operating system, how to use the North Star BASIC, how to use Solos commands, etc. There is a table of commands telling you the command name, which layer of the system uses the command (Solos, DOS, BASIC) and where in the guide you can find a description of what it means. Lots of other information is summarized in the guide, so you don't have to hop through the manuals to find a frequently used item.

I don't know if the guide really clears up the technicalities for the novice, but it may be worth the price for you. Frank de Coster, a Proteus member, has a bunch of these booklets which he will sell just to recover printing costs at $7.50 each including mailing. (They originally retailed for $19.95). If you have a NorthStar/Sol combination, it may even be worth the price to buy one now as a selling point for some future time when you may want to sell the system.

If you want one, send $7.50 to Frank de Coster, 315 Wayne Place, Oakland, CA 94606.

Product Announcement

I N D U S T R Y S T A N D A R D T A P E D R I V E S

If you want to obtain data from universities, government agencies, public-domain software libraries, and the like, on ANSI-standard IBM-compatible 9-track tapes, you should investigate the interface board manufactured by Pacific Office Systems, 918 Industrial Avenue, Palo Alto, CA 94303, (415) 493-7455.

The POS-100 NRZI Formatter/Controller costs $995 in single quantity, and will enable an S-100 bus 8080 or Z-80 CPU to read and write 9-track magnetic tapes using a "Pertec standard" tape transport. (Apparently, the Pertec transport was so successful, several other manufacturers used an identical interface specification for their drives as well.) Additional transports can be daisy-chained onto one controller. The interface is actually two boards; one for the S-100 bus and one for the tape transport, with a ribbon cable to connect the two.

The price does not include the tape transport, but we have seen refurbished ones being sold for about $1500.

A 2 MHz 8080 can operate a 12.5 ips tape drive, and a 4 MHz can go to 25 ips (inches per second). Other drive speeds up to 37.5 ips can be obtained by software modifications. Timing is in software, so the Sol may require a slightly different delay routine, due to its clock frequency being 2.045 MHz with the 8080A microprocessor. You may be able to adjust for this slight difference when setting the tape speed on the transport. (Pacific Office Supply has a Sol, so specify Sol-compatiblity when you order, and they should be able to check it out in their shop.)

Software routines supplied with the controller allow the computer to rewind, forward space record, back space record, forward space file, etc.. Status of the drive can be read by CPU, including off-line/on-line, invalid command, density,

beginning of tape error, write-protect error, noise record, etc.. Approximately 2K bytes of RAM is needed for all of these routines.

A set of CP/M utilities for diskette to tape, tape to diskette, tape to printer, etc. can be purchased for an additional $100.

POS is a very small company, so Caveat Emptor. But we know the hardware is actually quite simple and full documentation is said to accompany the boards, including theory of operation, testing procedures, service manual, and schematics.

Quantity discounts are available.

Product Announcement

STARKWEATHER'S

D I S K P I L O T L A N G U A G E

FOR MAN/MACHINE DIALOGS

Reviewed by Stan Sokolow

Way back in the early days of interactive computer systems on big computers, researchers began working on computer-aided-instruction (CAI) projects. It soon became evident that the computer programming languages were not well suited to writing the interactive programs needed by teachers for programming a dialog with their on-line students.

One of the pioneers in CAI, John Starkweather, PhD, developed a simple programming language that gave the instructional author just enough computing power and a simple language for making concise programs. The language was not intended as a general-purpose computing language, so it was weak in calculation but strong in character string processing features. Starkweather named the language "PILOT", for Programmed Inquiry, Learning or Teaching, and it has become the most widespread language for CAI.

While Processor Technology was a rising star in the microcomputer field, Dr. Starkweather developed an 8080 implementation of the PILOT language and customized it for the Sol computer. (Dr. Starkweather is one of our members.) Processor Technology Corporation actually distributed the cassette version of PILOT, and the disk version was under development when PTC went out of business.

But Starkweather didn't stop development, and he now has released the language on several different disk formats, taking advantage of the extra features possible with a disk. These versions can be purchased through Proteus, as our item number P20, for $99.95 on PTDOS/Helios disk, or on Lifeboat-CP/M compatible Helios disk, or on CP/M-compatible NorthStar disk. The Sol/CUTS cassette version (SOLOS/CUTER compatible) is still available with original manual, as Proteus item P21 for $49.95, but it lacks some of the new features.

A company in Washington state called Micropi ("micro programmed instruction", I think) offers a version of PILOT called "Common PILOT" for CP/M, NorthStar DOS, TRS-80, Helios II, and TERAK disks at various prices from $275 to $295. Considering that Starkweather's PILOT comes from the inventor, his price seems very reasonable in comparison.

Since few readers may know of the power of PILOT, here is a brief introduction.

The task that PILOT is really designed to do is this. An instructional program needs to let the author present some text or pictures, ask a question, accept a response from the student, analyse the response, and respond accordingly. If the students reply is equivalent to the correct answer expected or to some expected incorrect answers, the program should present instructional text (such as the correct answer, tutorial material, etc.) and branch off to an appropriate part of the instructional program.

What PILOT does for the author/programmer is that it eliminates the programming jargon and allows the author to concentrate on the sequence and content. PILOT also has some powerful string matching capabilities that let the computer decide if the student's response really is equivalent to one of the expected right or wrong answers. The program must be able to accept many variations of grammar, such as singular and plural, and recognize them as the right answer.

For example, if the answer you expect is "running", you may need to accept as equivalent the answers "run", "runs", and even some synonyms such as "jogging", or misspellings such as "joging". You also need to eliminate extraneous words and just look for the key word in the student's response, because you may get various forms of sentence fragments, such as "he is running", "he ran", "running, I think", "he would run", and so on.

To program all of these string manipulations in BASIC would take quite a few instructions and would really obscure the meaning of the program, which is to detect the right answer "running". In PILOT the "match" instruction does it all with this much work:

```
M: run, ran , jog
```

This instruction tells PILOT to examine the student's response (stored in memory) for any word beginning with "run", or the word "ran" not embedded in a longer word, or any word beginning with "jog". The entire response is scanned and the result of the match (yes or no) is set in an internal register for testing by subsequent instructions.

Based on the results of comparisons like these, the program can branch to other instructions, call subroutines, tally scores, record a log of the answers on tape or disk, present other data on the terminal, and so on. A limited amount of integer arithmetic can be done.

PILOT has had various dialects develop, just as BASIC has. A common core of instructions was standardized for PILOT. The core includes "T:" for type a string onto the terminal, "A:" for accept an answer, "M:" for match the pattern parameters against the accepted response, "J:" for jump (go to), "U:" for use (subroutine call), "E:" for end, "C:" for compute, "R:" for remark. All of the instructions can be made conditional by appending the letter "Y" or "N" which tests the result of the last recorded match done. For example, "JY:" means jump if the last match was successful, and simply "Y:" means type this if the last match was successful.

Strings in PILOT don't require quotes around them, since they are used so often. Instead, variables are identified with a leading character (such as $ for string variables and # for numeric ones) that distinguishes them from text in the string. For example,

```
*START
T: Please tell me your name.
A: $NAME
T: Hi, $NAME!
 : How old are you?
A: #a
T:Is it fun to be #a years old?
```

You can see that this is really a dialog programming language, and it can be used for any sort of dialog not just programmed instruction. The current fad of verbal games like "Adventure" could be programmed with some sophistication using PILOT, and the conciseness of the language would allow larger programs to fit into limited memory space. Even applications such as medical history-taking could be done this way.

Disk PILOT version 4 by Starkweather provides many extensions beyond the standard core instructions of PILOT. For example, with PILOT you can open and close files, read and write data files to record answers, control the cursor location on the screen, pause a measured number of seconds, query the system to discover free space available in memory and other system parameters, load and execute other PILOT programs, call machine language routines, and so on.

Starkweather has also extended the meaning of variable names, so that you can address variables indirectly. That is, a string variable can contain the name of the variable you actually want to use; such as "$$NAME", which means use the string whose variable name is in the variable called "$NAME". (This sort of indirection is not found in BASIC, but the MUMPS language has it. In future issues, we will talk more about MUMPS. See the MUMPS article elsewhere in this issue.)

There is even a version of Starkweather's PILOT which allows the Sol to control a particular model of Sony videocassette recorder to rapidly search for a desired point on the video and then play a certain number of frames of video! No special hardware interface is required, except a specially wired cable and plug to connect the remote control socket of the recorder to the Sol's parallel port. (If you want this version, be sure to specify when you order.)

PILOT contains its own Sol-video editor, similar to the one called "EDIT" in PTDOS or the one in ALS-8. Version 4.2 and 4.3 use a serial terminal rather than the display of the Sol or VDM module.

A PILOT library is operated by a PILOT users group. Write to us of you want to find out where you can get in touch with PILOT users.

M U M P S L A N G U A G E F O R C P / M

You've heard of UCSD Pascal, but have you heard about UCD MUMPS? (That's not a typographical error. UCD=University of California, Davis.)

MUMPS is an ANSI-standard language (along with FORTRAN, COBOL, PL/I) that is available on many mini-computers, notably the DEC PDP-11 line. There is a small, fervent, and growing following of MUMPS users who feel that MUMPS is the greatest thing since sliced bread. Some have compared the time required to develop major programming projects, such as a complete hospital information system, with MUMPS versus with COBOL or FORTRAN, and MUMPS has been the winner.

Well, for the past few years, a small crew at University of California, Davis, has been working on an 8080 implementation of MUMPS, and they have distributed about 200 copies to date. The Department of Community Health at UCD is using the language for medical records and education. A number of public domain applications are being adapted to work on the system.

We have just received a letter announcing that the University will provide for an annual rate of $93 the latest copy of 8080 Standard MUMPS for CP/M on 8" diskette, along with the documentation manuals, and three times per year will send you an updated version of the object code with new or revised applications. Source code is available for $25 extra per copy. The letter is reproduced here.

There is an international MUMPS users group, called "MUG", which publishes a quarterly journal, holds annual scientific meetings around the US, publishes educational material on MUMPS, etc. Write to MUMPS User's Group, % The Mitre Corporation, P.O. Box 208, Bedford, Mass. 01730.

H A R D D I S K F O R H E L I O S

At the Computer Faire, I ran into two former employees of Processor Technology Corporation who mentioned that they are in the process of forming a company to service and upgrade Helios disk units. They plan to put an 8" Winchester hard disk drive into the empty side of the Helios II two-slot cabinet and use a Morrow Designs hard disk controller. Modification would be made to PTDOS to accommodate the new disk.

I mentioned that Proteus has the source code to PTDOS, and they were interested in working together with us. I know that many users are interested in adding the extra capacity of a hard disk, and this seems to be the ideal way if you already have a Helios. Those who do not have Helios would be able to use the regular 8" Morrow hard disk, which is available in 10 megabyte and soon in larger capacities.

When the company is ready, you will hear about it in this newsletter. One of PTC's best Helios technicians is involved in the company, so we know the work will be of expert quality.

2 4 x 8 0 V I D E O M O D I F I C A T I O N F O R S O L

We have been told by a southern California manufacturer that a piggy-back board is being designed for modifying the Sol's video display to a standard 24x80 format. This is a modification, not a new display board which would require another slot in the bus. When it is available, Proteus will test it on our system and review the product in the newsletter.

The biggest handicap the Sol suffers right now is the small screen, since most pre-programmed application software assumes an 80 column screen. This modification board will bring Sol up to date. We're looking forward to it.

F O R M E R P T C E N G I N E E R S F O R M S E R V I C E C E N T E R

A group of former Processor Technology technicians and engineers have formed a company for repairing, servicing, and customizing all of PTC's product line and most S-100 bus products. They are experienced in PTC hardware and software, including the Sol, Helios disk, PTC memory boards, printer interfaces, etc. They also have factory experience in Morrow and Dynabyte products, plus other experience in NorthStar, Godbout, Shugart, PerSci, Remex, Micropolis, etc.

They will service hardware, help with interfacing peripherals to computers, customize CP/M for specific hardware configurations, perform preventive maintenance, and so on. Helios alignment has a flat rate of $55 plus shipping; Helios repair flat rate $80 plus parts & shipping (unless the faulty part is not a field-replaceable item). Other service @ $40 per hour. They will give estimate before performing work, if requested.

Contact ACE Computers, Inc., 3388 Moraga Blvd, Lafayette, California 94549. (415) 283-6630.

(Editor's note: I know that when PTC had a tough one to repair, one of these fellows was the man whom they called upon. One of PTC's former corporate officers gave me nothing but the highest personal recommendation for his capability. So I think we are fortunate that this group has gone into the service business. All of these men have worked for some of the biggest names in S-100 microcomputing. -- Stan.)

PROTEUS CASSETTE SOFTWARE LIBRARY SERVICES

At the present time, the Proteus Cassette Software Library is offering services to Proteus members in 4 different areas:

First, we have the original Proteus Library cassettes. We now have available cassettes C1 - C11 (except C9, which has been delayed). These remain priced at $18 without a contribution, and $8 with an acceptable program contribution. These have been described in past issues of PROTEUS NEWS, or you may send me a SASE for a catalog sheet.

Next, we have on tape the full CPM Users Group library. At the present time, CPMUG disks 1 - 47 are available, with several more expected soon. A master catalog of these CPMUG volumes is available, either as hard copy or on tape, for $6.00. Unless you specify tape, you will be sent the hard copy. The CPMUG volumes are priced at $10.00 each (with or without a contribution).

In addition, there are another 7 disks from a new CP/M-oriented users group known as SIG/M. THis includes the famous ADVENTURE game, a 6502 simulator, and a number of CP/M utilities. The SIG/M disk catalog is included on the CPMUG catalog tape, and the SIG/M volumes are also priced at $10.00 each.

I expect soon to have additional volumes from the C Users' Group and one of the Pascal Users' Groups. If you are interested in these, you may write me.

The CPMUG and SIG/M software catalogs are provided on standard CUTS-format cassette tapes, and come with a special tape-to-disk loader program. To load the tapes, you need a SOL (or a compatable machine with a CUTS board and CUTER), some sort of disk system with CP/M v1.4 or v2.2, and your cassette recorder. You should have the motor control cable, too. Hard copy doc with the programs tell you how to get started.

Finally, we offer a CP/M file transfer service. If you have programs on a standard 8" single density CP/M disk, but have a Helios/CPM or a 5" disk system, we can transfer the programs from your disk to a tape format which you can then load into your disk system. The price for this service is $10.00 per 8" disk, regardless of how much or how little is on the disk. The price includes the cassette and postage, and the special tape-to-disk loader program. If you want your 8" disk returned, pack it in a reusable shipping box and enclose extra return postage (say, $1.00 for each 2 disks).

Send orders, inquiries and program contributions to:
PROTEUS Cassette Software Library
C/O Lewis Moseley, Jr., Librarian
2576 Glendale Court, NE.
Conyers, GA 30208

If you want a personal reply, please enclose a self-addressed stamped envelope. Enjoy!!!

PROTEUS LIBRARY CASSETTE C8: MORE ECBASIC PROGRAMS

THIS TAPE CONTAINS A NUMBER OF INTERESTING VIDEO GRAPHICS DEMONSTRATION PROGRAMS, AND ALSO SEVERAL SERIOUS PROGRAMS. EACH IS RECORDED TWICE ON SIDE 1 IN ECBASIC INTERNAL COMPILED FORMAT, AND ONCE ON SIDE 2 IN TEXT (PROTEUS STANDARD BYTE ACCESS) FORMAT.

#	NAME	TYPE	SIZE	DESCRIPTION
1	TCOPY	U	1K	OBJECT CODE FOR THE CUTS TAPE COPY/VERIFY PROGRAM BY LEWIS MOSELEY, JR. BRIEF INSTRUCTIONS FOR USE ARE INCLUDED. SOURCE IS ON CASSETTES C9 AND C10.
2	RNDCH	C	1K	RANDOM CHARACTERS ON THE SCREEN
3	RNDBW	C	1K	RANDOM BLACK AND WHITE ON SCREEN
4	SETSP	C	2K	OBSERVE EFFECTS OF 'SET DS='
5	DICE	C	3K	VIDEO DICE ROLLS ON THE SCREEN
6	BOGGL	C	2K	A NICE VIDEO IMPLEMENTATION OF THE BOARD GAME. ON-SCREEN TIMER. RULES NOT PROVIDED.
7	BINGO	C	2K	CALLS THE NUMBERS FOR A BINGO GAME AND RECORDS THEM ON THE SCREEN FOR CHECKING WINS.
8	DOODL	C	2K	DOODLE ON THE SCREEN. SELECT A CHARACTER AND LEAVE A TRAIL OF THEM ON THE SCREEN.
9	DATA	C	4K	A SOPHISTICATED REAL-TIME DATA GATHERING PROGRAM, CURRENTLY SET UP FOR MEDICAL MONITORING, BUT A GOOD EXAMPLE OF THE TECHNIQUE.
10	TTYPE	C	5K	TOUCH TYPING PRACTICE (WE ALL NEED IT!), WITH DIFFERENT LEVELS OF DIFFICULTY. FULL KEYBOARD SHOWN ON SCREEN.
11	CIRCL	C	1K	THE NEXT GROUP OF PROGRAMS, THROUGH
12	SPIRL	C	1K	LISJ9, ARE VIDEO GRAPHICS, AND SHOW
13	LISJ1	C	1K	THE PLOTTING OF A NUMBER OF TRIG
14	LISJ2	C	1K	FUNCTIONS. OBVIOUSLY, THE LISJ(N)
15	LISJ3	C	1K	PROGRAMS PLOT VARIOUS LISSAJOUS
16	LISJ4	C	1K	FUNCTIONS.
17	LISJ5	C	1K	
18	LISJ6	C	1K	
19	LISJ7	C	1K	
20	LISJ8	C	1K	
21	LISJ9	C	1K	
22	SEAWR	C	4K	GUNNERY PRACTICE AT SEA. CHOOSE THE RIGHT ELEVATION TO SINK THE ENEMY IN FLAMES.
23	FROG	C	3K	GRAPHICS FROG RACE, WITH BETTING, FOR MULTIPLE PLAYERS.
24	END	X	0K	TERMINATOR FOR TCOPY PROGRAM

USE THE TCOPY PROGRAM TO BACK UP YOUR CASSETTE.

PROTEUS LIBRARY CASSETTE C10: CP/M PROGRAMS FOR SOL

THIS IS OUR FIRST TAPE OF CP/M PROGRAMS, IN RECOGNITION OF THE FACT THAT MANY OF OUR MEMBERS HAVE GONE TO DISK. THERE ARE 5 MAJOR PROGRAM SYSTEMS, ANY ONE OF WHICH IS WORTH THE PRICE OF THE CASSETTE, AND SEVERAL UTILITIES. THE FIRST FILE ON THE TAPE IS THE 'TAPEDISK.COM' FILE, WHICH IS MANUALLY LOADED, AND WHICH THEN LOADS THE REST OF THE FILES TO THE DISK. ALL OF THE FILES HAVE THE SAME TAPE NAME - CPM - BUT DON'T WORRY; TAPEDISK KNOWS THEIR TRUE NAME. REFER TO THE HARD COPY DOCUMENTATION FOR LOADING INSTRUCTIONS.

#	NAME	TYPE	SIZE	DESCRIPTION
1	BYTE	.ASM	5K	THIS ROUTINE, BY LEWIS MOSELEY,
2	BYTE	.DOC	4K	ALLOWS CP/M'S PIP.COM TO READ A PROTEUS STANDARD BYTE ACCESS TAPE FILE ONTO THE CP/M DISK, OR VICE VERSA. NOW YOU CAN USE ALL OF THOSE PROGRAMS YOU WROTE FOR ECBASIC. ALSO, LOAD SOURCE FILES PROCESSED BY 'UNPACK', AND TRANSFER TO/FROM OTHER SYSTEMS.
3	COMLINK	.COM	6K	THIS PROGRAM INTERFACES CP/M WITH
4	COMLINK	.DOC	30K	A D.C. HAYES 80-103A OR MICROMODEM 100 MODEM BOARD. IT ALLOWS SOFTWARE SELECTION OF LINE CHARACTERISTICS, SUPPORTS AUTO DIAL AND AUTO ANSWER, AND ALLOWS A TEXT FILE TO BE SENT TO OR CAPTURED FROM THE MODEM. FULL SOURCE CODE IS ON TAPE C11.
5	DISKTAPE	.ASM	5K	THIS GROUP OF PROGRAMS, BY GREENLAW,
6	DISKTAPE	.COM	1K	ALLOWS THE TRANSFER OF ANY KIND OF
7	DISKTAPE	.DOC	14K	CP/M DISK FILE BETWEEN DIFFERENT
8	TAPEDISK	.ASM	7K	DISK SYSTEMS. THE CP/M FILE IS
9	TAPEDISK	.COM	2K	BROKEN INTO BLOCKS AND RECORDED TO THE CASSETTE. VERY LARGE FILES CAN BE TRANSFERRED, EVEN WITH A MINIMUM 16K SYSTEM. FULL INTERNAL ERROR CHECKING. ALSO, A CHEAP, IF SLOW, BACKUP FOR DISKS.
10	MFT	.ASM	17K	A VERY NICE SINGLE-DRIVE COPY
11	MFT	.COM	2K	PROGRAM BY LARRY HUGHES. ALLOWS
12	MFT	.DOC	5K	COPYING ALL OR SPECIFIED PARTS OF A DISK TO ANOTHER. USES ALL AVAILABLE MEMORY AS A DISK BUFFER TO MINIMIZE DISK SWAPPING.
13	TCOPY	.ASM	10K	CUTS TAPE COPY/VERIFY UTILITY BY
14	TCOPY	.COM	1K	LEWIS MOSELEY, JR., MODIFIED TO LOAD FROM CP/M. ALLOWS UNATTENDED COPYING AND VERIFYING OF CASSETTE TAPES. USE TO BACK UP CASSETTES. OBJECT WILL BE INCLUDED ON MOST FUTURE LIBRARY CASSETTES. DOCUMENTED IN THE .ASM FILE.
15	PRINT	.COM	2K	A PRETTY-PRINTER WHICH GIVES NICE
16	PRINT	.DOC	5K	FORMATTED AND PAGINATED HARD COPY FROM A DISK TEXT FILE. THE .DOC FILE TELLS HOW TO CUSTOMIZE THE FORMAT TO YOUR NEEDS.
17	WD	.COM	1K	TWO NICE EXTENDED DIRECTORY ROUTINES
18	XD	.COM	2K	TO SUPPLEMENT THE BUILT IN DIR FUNCTION. WDIR GIVES A 4 ACROSS ALPHA-SORTED DIRECTORY. XDIR GIVES A COLUMNAR FORMAT, WITH FILE SIZES, WHICH IS BEST FOR HARD COPY. FROM CPMUG.
19	PACK	.COM	2K	DONATED BY PROCESSOR TECHNOLOGY
20	UNPACK	.COM	2K	CORP., AND MODIFIED TO LOAD FROM
21	PACK	.DOC	3K	CP/M DISK. THESE PROGRAMS CONVERT BLOCK ACCESS FILES TO BYTE ACCESS FILES, AND VICE VERSA. THEY WERE INCLUDED WITH PT'S EDIT AND CASSM PROGRAMS. USE TO CONVERT EXISTING FILES FOR TRANSFER.
22	C10	.PRN	5K	THIS CATALOG LISTING

IN ALL, ABOUT 125K OF MATERIAL. ALL PROGRAMS ARE KNOWN TO WORK ON AN 8" SINGLE DENSITY SOFT SECTOR CP/M SYSTEM (TARBELL). ALL, WITH THE EXCEPTION OF XD.COM, ARE BELIEVED TO WORK ON ALL CP/M SYSTEMS. XD WILL PROBABLY GIVE SPURIOUS SIZE INFORMATION ON SYSTEMS OTHER THAN 8" SINGLE DENSITY SOFT SECTOR.

PROTEUS LIBRARY CASSETTE C11: MORE CP/M PROGRAMS FOR SOL

THIS TAPE IS PRIMARILY FOR PERSONS WHO WANT THE FULL SOURCE LISTING FOR COMLINK. THE SOURCE, ITSELF ALMOST 60K, WOULD NOT FIT ON C10. ALSO INCLUDED ARE THE TAPEDISK/DISKTAPE PROGRAMS TO LOAD THE TAPE TO CP/M DISK, AND A MEMORY TEST.

#	NAME	TYPE	SIZE	DESCRIPTION
1	COMLINK	.ASM	57K	THIS PROGRAM INTERFACES CP/M WITH
2	COMLINK	.COM	6K	A D.C. HAYES 80-103A OR MICROMODEM
3	COMLINK	.DOC	30K	100 MODEM BOARD. IT ALLOWS SOFTWARE SELECTION OF LINE CHARACTERISTICS, SUPPORTS AUTO DIAL AND AUTO ANSWER, AND ALLOWS A TEXT FILE TO BE SENT TO OR CAPTURED FROM THE MODEM.
4	DISKTAPE	.COM	1K	THESE PROGRAMS, BY RICHARD GREENLAW,
5	TAPEDISK	.COM	2K	ALLOW THE TRANSFER OF ANY KIND OF
6	DISKTAPE	.DOC	14K	CP/M FILE BETWEEN DIFFERENT DISK SYSTEMS. THE CP/M FILE IS BROKEN INTO BLOCKS AND RECORDED TO CASSETTE BY DISKTAPE. TAPEDISK REVERSES THE PROCESS. FULL SOURCE CODE ON CASSETTE C10.
7	QUIKTEST	.COM	1K	A MEMORY TEST PROGRAM WHICH TESTS ALL MEMORY FROM THE START OF THE TPA UP TO THE BASE OF THE CBIOS. RESET AND REBOOT TO STOP THE TEST.
8	TCOPY	.COM	1K	CUTS TAPE COPY/VERIFY UTILITY BY LEWIS MOSELEY, JR. USE TO VERIFY TAPE FILES AND TO MAKE BACKUP COPIES. FULL SOURCE CODE ON C10.
9	C11	.PRN	3K	THIS CATALOG LISTING

ALL PROGRAMS ARE KNOWN TO WORK ON AN 8" SINGLE DENSITY SOFT SECTOR CP/M SYSTEM, NORTHSTAR CP/M AND MICROPOLIS CP/M, AND ARE BELIEVED TO WORK ON ALL CP/M SYSTEMS.

C O N T E N T S O F H E L I O S L I B R A R Y H 5

This diskette contains the source and/or object code for a few device drivers. Those with type IW are WordWizard-compatible printer drivers. The source code for these begins with the lower case "w", as in "wSol3". To use the object code, GET the driver onto your system disk, RETYPE it type "D" for driver, and then use it.

The SolPrinter drivers (mSol3, wSol3, mSol2, etc.) were written by the Basic Computer Group, Ltd., in Vancouver, B.C., for use in WordWizard, MailMaster, etc. They may be used with any software, but be careful where they load to be sure they don't overlap other programs. The source code may be re-ORG'd to other locations. The programs beginning with "m" meet the specifications of MailMaster and the AccPac programs. The drivers beginning with "w" are for WordWizard and support bi-directional logic-seeking printing in a foreground/background mode. This is described in PTC updates specifying requirements for WordWizard drivers. They may be used by PTDOS in general, but will only give the special features for WordWizard if set in word-processing mode by a control/status call.

The Sol2 is a Diablo Hytype II printer metal printwheel printer, interfaced to revision E Sol parallel port using the PTC interface for the printer. If you have this printer but a revision D Sol, you will need to make an adapter cable that reverses one set of data lines as described in the Sol manual. This driver assumes you have the revised Hytype interface which was named the SolPrinter interface. The original Hytype interface did not support some of the status conditions, such as paper-out, so the status test in the driver will need to be modified.

The Sol2E is the same as Sol2, but using the plastic printwheel Diablo.

Hytype driver is a Sol2E driver modified to support the original Hytype interface designed by PTC, not the later SolPrinter one.

Sol3 is a driver for a serial printer attached to the Sol serial port. It was designed for the SolPrinter3 which was a Diablo 2300 matrix printer, but it has also been used successfully with other printers, including the Epson MX-80.

DEC is a DecWriter driver for the Digital Equipment Corporation Decwriter. It can easily be modified for most common printers on the serial or parallel port.

XEROX or 1610 or WPXER are drivers for the Diablo/Xerox 1610 or 1620 daisy-wheel terminals. They support ETX/ACK protocol and bidirectional printing.

CDC is driver for Control Data 9317 matrix printer.

SPIN is for NEC Spinwriter.

TI810 is for Texas Instruments 810 printer.

02/17/81 FILES ON: H5

NAME	TYPE	SIZE	BLKZ	ID	SEC	TRK	ATTRI	INDEX
!	IS	11	0100	0041	5	76		
1610	IW	4	04C0	0044	0	9	KWN	
CDC	IW	4	04C0	0046	8	14	KWN	
CONTENTS	.	12	04C0	0049	0	15		
CTAPE.A	T	64	04C0	001F	9	9	KWN	
CTAPE1	D	4	0100	001E	5	9	KN	
CTAPE2	D	4	0100	0020	13	9	KN	
DEC	IW	4	04C0	0047	12	14	KWN	
DEC.A	S	68	04C0	002E	0	37		
FEEDBACK	.	8	04C0	004A	12	15		
NOTICES	T	8	04C0	004C	8	16		
OKIDATA	T	16	04C0	003C	3	0	KWN	
SETPRT.A	T	20	04C0	001C	8	7		
SPIN	D	5	0100	003F	7	0		
SPIN:D	T	36	04C0	003D	0	18	K	
SPINWR:S	T	84	04C0	003E	4	20	K	
TERM:S	T	4	04C0	0028	12	22		
TERMINAL	I.	1	0100	002B	14	26		
TI810.A	T	36	04C0	0016	10	0		
WARRANTY	T	4	04C0	004B	0	16		
WPXER.A	S	44	04C0	002D	4	34		
WPXER.A2	S	60	04C0	002C	8	30		
XEROX.A	S	36	04C0	0029	0	23		
XEROX.A2	S	56	04C0	00_F	4	41		
m1610:s	S	36	04C0	0030	12	44		
m1610e:s	S	56	04C0	0031	0	47		
mHytype2	I.	3	0100	001A	14	0		
mS154C	I.	3	0100	0024	13	13		
mS154C:S	.	64	04C0	002A	6	26		
mSol2:S	.	4	04C0	0037	8	65		
mSol2E:S	.	4	04C0	0019	12	6		
mSol3:S	.	4	04C0	0038	12	65		
mSolp:S	.	24	04C0	0017	4	3		
msp2d:s	.	32	04C0	0018	12	4		
msp3d:s	.	12	04C0	0039	0	66		
w1610:s	S	44	04C0	0033	4	54		
w1610e:s	S	60	04C0	0032	8	50		
wSOL2	IW	8	04C0	0045	0	14		
wSOL2E	IW	8	04C0	0043	4	7		
wSOL3	IW	4	04C0	0048	0	7		
wSol2:S	.	40	04C0	0035	8	59		
wSol2E:S	.	40	04C0	003A	12	66		
wSol3:S	.	40	04C0	0034	0	57		
wsp2d:s	.	56	04C0	0036	0	62		
wsp2ed:s	.	44	04C0	003B	4	69		

If you modify these programs and reassemble them, please remember that PTDOS requires all drivers to be loadable as one logical block. The assembler doesn't usually create image files in this format. To convert the object file into a single-block image file, use the command:

```
EXTRACT file,S
```

which will "scrunch" the file into one block if possible. The scrunched file can be RETYPE'd into type "D" for driver. The physical blocksize doesn't matter, but you should chose a block-size for efficiency. See the PTDOS manual for more info on this matter.

C O N T E N T S O F H E L I O S L I B R A R Y H 7

This file lists the contents of this diskette, H-7 from the Proteus Library.

This diskette contains programs which were donated by a number of people. I feel that due credit should be given to the authors and have therefore grouped the programs by author.

For further information please read the files WARRANTY and FEEDBACK.

Charles L. Athey, III
Proteus Librarian

The following were donated by Frank J. Sanders. These programs handle personal finances.

ACCOUNTS - SAMPLE LIST OF ACCOUNTS

BANKERS - PROGRAM TO ENTER DEPOSITS AND WITHDRAWALS, AND OBTAIN READOUT OF TOTAL AND PERCENTAGES

CREATFIL - CREATE RANDOM FILE OF ACCOUNTS

EXPENSES - SAMPLE LIST OF EXPENDITURES

PERSONAL - PROGRAM TO ENTER DAILY OR WEEKLY EXPENDITURES,AND OBTAIN READOUT OF TOTALS AND PERCENTAGES

SERIAL - PROGRAM TO CREATE SERIAL FILE OF NAMES OR TITLES WHICH MAY THEN BE CONVERTED TO A RANDOM FILE IF DESIRED USING SER.RNDM PROGRAM

SER.RNDM - PROGRAM TO CONVERT SERIAL FILES TO RANDOM ACCESS FILES

Here is a group of programs from Preston Briggs of Interactive Computing.

PATCH1.5 is a program to allow the reatributing of attribute protected files on PTDOS1.5 (not 1.4). I don't have the source code unfortunately but it is pretty simple. Type PATCH1.5 and it will execute and return to PTDOS. Then REATR the file(s). For safety, you should re-boot afterwards as PTDOS will continue to ignore the attribute protects.

MESSAGE, !MESSAGE, BUILD, MES.S, and MES.TEMP are a group of useless programs I did for fun. Studying the source is a good way (maybe) to understand how to use the overlay handler and how to interface to PTDOS in general.

MESSAGE will type a random (almost) message on the screen whenever run. I use it in my START.UP file to avoid the same old boot-ups over and over.

!MESSAGE is a utility file that contains the 32 messages that MESSAGE may choose from. !MESSAGE should be on the default diskette.

MES.S is the source code for MESSAGE.

BUILD is a DO file that I use to create or replace messages in the utility file !MESSAGE. It expects to run on the default disk and requires !MESSAGE and MES.TEMP.

MES.TEMP is the source file for a message.

FORMAT is the text formatter originally described in Software Tools, by Kernighan and Plauger. This version was written by Mike Gabrielson and printed in the May 79 issue of Dr. Dobb's. I added the necessary interfacing to work with PTDOS. Mostly what I'm donating here, is the typing effort. I don't think it violates anything and Gabrielson includes no copyright message so I assume it is for general use. I did not include the comments when I entered the code so one should reference Dr. Dobbs and Software Tools for help with the program. Type FORMAT sourcefile, outfile. Enter #1 in outfile to run to screen. FORMAT.S is the source file. TEXT is a sample file to be formatted.

Extended Disk BASIC programs:
(actually, these should all run on extended cassette BASIC too)

PRIMES is a fast program for generating prime numbers. The algorithm is from a fairly recent CACM article by Gries. I'm sorry I don't have the date. The program could be extended by using PEEKs and POKEs instead of an array to represent the sieve as each element in the sieve can have only two values.

FACTOR is a program to factor an integer into it's prime components. It utilizes the same algorithm as the PRIMES program and could be extended in the same way.

KWIKSORT is a quiksort or partition-exchange sort. Is neat in that it utilizes user-definable multi-line functions recursively, with aoutomatic stacking of local variables.

QUIKSORT same as above but with modifications suggested by Knuth.

HEAPSORT from Knuth

SHELSORT from Knuth

SORT is another sort suggested by Knuth which seems to approach the speed of the quiksort but may not have the disadvantages(when the file is in order, for example) Knuth rates the mathematical evaluation of this sort at 50 points, his maximum.

MAZE is another example using recursive functions. It was inspired by a contest a friend entered in which the object was to find the longest possible word in a given matrix of random letters. The words are allowed to twist and turn as much as necessary but must not use the same letter twice. Try words like: location, tatterdemalion, pharmaceutical to see it work. (Should be entered in lower-case)

FIND+ was originally inspired by the IEEE mico-mouse contest. This was the best program I came up with, and it naturally uses a recursive function. Note that it will require lots of memory to run! SET your BUffer = 9000h.

FIND another mouse program but with a unique idea that causes interesting behaviour sometimes. Consider an array where the walls are valued at 999 and the corriders are set at 0 and the "cookie" is set at -1. Have the mouse increment any location he's at by 1 and then go to the lowest adjacent number. It works but looks funny sometimes. Also takes much less memory than FIND+. Watch its behaviour in the top right corner.

PERMTEST generates the all the permutations of a given
array in order. Makes a good problem. This
algorithm from Dijkstra.

The following program was donated by Larry McDavid of
LMC Engineering, Anaheim, Ca.

LOADM - PTDOS image-file load to memory. This program reads
PTDOS image-type files into system memory starting at a
user-selected address. The image-file block headers are
used to control the loading of each file block so that
the final loaded format is identical to that resulting
from entering the filename as a command. The source is
LOADM.C, and the documentation file is LOADM.D.

The following program was donated by Ben C. Stapleton Jr. of
Office Supply Inc., Portsmouth, Ohio

PHONUM:S EDBASIC program converts Phone Numbers to Words.
Each Phone Number generates 2187 different words.

The following programs were donated by Jay Parsons of Somerset
Data Systems, Inc. Bernardsville, NJ.

ROBOTS is an old game involving hiding from killer robots.

TENSORTS compares ten sorting algorithms in EDBASIC, including
three versions of the Shell-Metzner, quicksort, heapsort, plain
and Woodrum merges, delayed-replacement, selection and bubble.

The following program to help convert between CP/M and PTDOS
format files was donated by Gib Zeratsky, GreenLake, WI.

CPM-TXT Documentation in file CPM-TX.D

The following programs were donated by Earl J. Dunham of La
Habra, Ca.

Weekly Reporting programs: ADD.WK, ANYMO, NUMSTR, WK.DOC

CONVERT : a comprehensive english<=>metric convertsion program.

STR-SORT : an unusual way of sorting strings, using the Shel-
Metzer sort algorithm.

03/25/81 FILES ON: H-7

NAME	TYPE	SIZE	BLKZ	ID	SEC	TRK	ATTRI	INDEX
Imessage	M	17	0240	0080	13	11	KWN	0B0F
ACCOUNTS	08	5	04C0	004D	0	2	KWN	000F
ADD.WK	05	12	04C0	009A	0	27	KWN	
ANYMO	08	33	0100	009B	13	20	KWN	1D07
BANKERS	06	12	04C0	0050	12	3	KWN	
CONTENTS	P	28	04C0	0024	12	24	KWANEU	
CONVERT	05	36	04C0	009D	0	30	KWN	
CPM-TX.D	T	8	04C0	0097	9	20	KWN	
CPM-TXT	T	4	04C0	0096	5	20	KWN	
CREATFIL	06	4	04C0	004F	8	3	KWN	
EXPENSES	08	21	04C0	004E	4	2	KWN	010C
FACTOR	05	4	04C0	0075	12	5	KWN	
FEEDBACK	T	8	04C0	0047	3	0	KWANEU	
FIND	05	8	04C0	0076	8	6	KWN	
FIND+	05	8	04C0	0077	0	7	KWN	
FORMAT	I.	8	09C0	0074	0	6	KWN	
FORMAT.S	T	40	04C0	007C	8	8	KWN	
HEAPSORT	05	4	04C0	0085	10	13	KWN	
KWIKSORT	05	4	04C0	0086	0	14	KWN	
LOADM	IC	3	0380	0089	8	7	KWN	
LOADM.C	T	2	0100	008B	11	7	KWN	
LOADM.D	T	16	04C0	008A	4	14	KWN	
MAZE	05	4	04C0	007A	0	8	KWN	
NOTICES	T	8	04C0	0042	0	73	KWANEU	
NUMSTR	05	40	04C0	0098	4	21	KWN	
PATCH1.5	IS	4	04C0	0079	12	7	KWN	
PERMTEST	05	4	04C0	007B	4	8	KWN	
PERSONAL	06	12	04C0	0051	8	4	KWN	
PHONUM:D	T	4	04C0	008F	0	17	KWN	
PHONUM:S	05	8	04C0	008E	4	15	KWN	
PRIMES	05	4	04C0	007D	0	11	KWN	
QUIKSORT	05	4	04C0	0083	2	13	KWN	
ROBOTS	05	16	04C0	0092	4	17	KWN	
ROBOTS:C	T	1	0100	0094	15	13	KWN	
SER.RNDM	05	4	04C0	0052	4	5	KWN	
SERIAL	05	4	04C0	0053	8	5	KWN	
SHELSORT	05	4	04C0	0084	6	13	KWN	
SORT	05	4	04C0	0073	8	1	KWN	
STR-SORT	05	8	04C0	009C	8	29	KWN	
TENSOR:C	T	1	0100	0095	4	20	KWN	
TENSORTS	05	28	04C0	0093	8	18	KWN	
WARRANTY	T	4	04C0	0040	4	72	KWANEU	
WK.DOC	T	28	04C0	0099	12	23	KWN	
build	$	2	0100	0081	14	12	KWN	
mes.s	T	8	04C0	007E	4	11	KWN	
mes.temp	T	2	0100	0082	0	13	KWN	
message	I	1	0100	007F	12	11	KWN	
text	T	4	04C0	0088	8	15	KWN	

C O N T E N T S O F H E L I O S L I B R A R Y H 8

This disk contains the small C compiler as implemented by Ron Cain and inhanced by Ed Hirgelt.

Cc - The running compiler/
Cc?.c - C source for the compiler.
Cc?.a - 8080 ASSM source for the compiler, the results of compiling the compiler.
C80LIB.A - The runtime support package for the Sol-Helios system.
CcDef - The common definitions needed by the compiler.
Cc.Txt - A description of each routine in the compiler.
Cc.Use - A short description on how to use the compiler.
Setup.a - An assembly routine which the compiler generates a call to to setup the C environment.
Test.c - A sample C program.

Please address any questions to Chuck Athey (415) 449-8337, 5571 Shorehaven Circle, Livermore, Ca 94550.

04/16/81 FILES ON: H-8

NAME	TYPE	SIZE	BLKZ	ID	SEC	TRK	ATTRI	INDEX
C80LIB.A	T	32	09C0	0017	8	7	KWN	
CC	Ic	80	09C0	0046	0	48	KWN	
CC.DEF	T	8	04C0	0027	9	1	KWN	
CONTENTS	T	4	04C0	004A	4	57	KWANEU	
Cc.A	T	1	0100	0026	15	0	KWN	
Cc1.A	T	56	09C0	0015	0	1	KWN	
Cc1.C	T	56	09C0	0016	0	4	KWN	
Cc2.A	C	48	04C0	0043	0	7	KWN	
Cc2.C	T	48	09C0	001C	0	20	KWN	
Cc3.A	T	48	09C0	001B	8	17	KWN	
Cc3.C	T	32	09C0	001D	8	22	KWN	
Cc4.A	T	56	09C0	0019	8	11	KWN	
Cc4.C	T	32	09C0	0021	8	32	KWN	
Cc5.A	T	48	09C0	001E	0	24	KWN	
Cc5.C	T	24	04C0	0030	12	42	KWN	
Cc6.A	T	40	09C0	001F	8	27	KWN	
Cc6.C	T	24	04C0	0034	12	44	KWN	
Cc7.A	T	48	09C0	0020	0	30	KWN	
Cc7.C	T	24	04C0	0035	4	46	KWN	
Cc8.A	T	32	09C0	001A	8	14	KWN	
Cc8.C	T	32	09C0	0025	8	38	KWN	
FEEDBACK	T	8	04C0	004B	4	7	KWANEU	
WARRANTY	T	4	04C0	004C	12	57	KWANEU	
XASSM	I	28	04C0	0048	12	47	KWN	
cc.txt	T	52	04C0	003A	0	37	KWN	
cc.use	T	4	04C0	0047	8	40	KWN	
setup.a	C	12	04C0	0044	12	35	KWN	
test.c	T	1	0100	003E	8	1	KWN	

— BOOK REVIEW —
by Jeff Tom

SOFTWARE DEBUGGING FOR MICROCOMPUTERS
Robert C. Bruce $17.95
ISBN 0-8359-7021-3 or 0-8359-7020-5 pbk

I was wandering around near the University of Arizona in Tucson when I stopped to look in the window of this bookstore and there it was. I went in to see if it pertained to 8080 microprocessors, and found that it was for debugging programs written in BASIC. But wait...what is this?? Processor Technology Extended Cassette Basic? Yes! For that reason alone, I bought the book. Was it worth it? Read on.

First let me say that the book is not intended to be a tutorial on BASIC, but any newly presented, or unique statements are briefly described before being used. On the other hand, one could probably learn the function of many of the statements by seeing them used in the programs if nothing else.

I found the approach of the book to be very much in line with human nature. The author "writes" a program, and then when it doesn't work, he starts to debug it, often resorting to the user's manual when all else fails. How many of us haven't done that at one time or another?

The book doesn't present any earth-shaking plan for writing error-free code, but stresses the use of flowcharts, modular programming, PRINT statements, and playing computer with paper and pencil. He does recommend using simple numbers like 0 and 1 where possible to simplify catching math errors, which makes good sense to me. He encourages the use of REM statements and indented FOR-NEXT loops, but there are places in the programs where REM statements are few and far between, and the only place his loops are indented is when he shows how nice it makes tracing program flow. This was unusual because somehow, E.C. BASIC's "print pretty" feature had been defeated.

About half the book is devoted to developing a data base management system, and these chapters deal rather well with the subjects of string manipulation, and use of cassette files.

Few of the programs presented are anything that you'd run to your computer to type in, but most could be adapted for personal use with a little thought. As a matter of fact, changing programs from one use to another is one of the subjects covered, along with some of the pitfalls that can be encountered.

I won't guarantee that you'll learn anything new from this book, but it does make you think about your programming practices and it may point out a trick or two for you to use. One thing is fairly certain, this is the first and last book to be devoted to E.C. BASIC.

COMPUTER COURSES FOR THE DEAF

Rochester Institute of Technology (RIT) will offer two computer courses for deaf adults this summer through the National Technical Institute for the Deaf (NTID).

Introduction to Data Processing - August 3 - 7
Advanced Data Processing - August 10 - 14

For more information, contact Donald Beil, NTID Data Processing Dept., Rochester Institute of Technology, One Lomb Memorial Dr., Rochester, NY 14623 or (716)475-6373.

Repairing your Sol (Part 2)

by Joe Maguire

In the previous installment of this series, we concentrated on isolating the problem area. We left off just as we were going to tackle the Sol PC board.

The operation of the Sol can be divided into a number of functional blocks. From the point of view of the operator they are:

1. The video display
2. The CPU
3. The cassette tape I/O
4. The serial port
5. The parallel port
6. The SOLOS ROM
7. The C800-CBFF memory
8. The keyboard

Beg, borrow or steal a copy of the Sol PC block diagram from the Sol manual. (X-24) This diagram has each of the ICs associated with the above blocks listed in tabular form. It's possible to proceed without it but it's going to be tougher.

Now is the time to ask the PC, "where does it hurt?" In other words, what isn't working properly. Probably the most serious fault is no video. Without that to assist in trouble shooting you are groping in the dark. The first thing to check here is the monitor. The best way to check it is to try it on another computer. This is where friends and computer clubs really become valuable! When taking your monitor to your friend's, be sure to take your video connector cable along too. I've never found a bad monitor but I've found many shorted cables. Assuming the monitor is OK, we now start the real search for the problem area.

A completely dead Sol (one with no CPU activity) will not generate any video. So how can we tell whether to start with the video circuits or the CPU logic? Try some commands from SOLOS which will produce a response outside the computer. For example: Can you boot up your disks? Can you SAVE something on a cassette tape? If it seems like these activities are working but the screen remains blank, it's most likely the VDM. If nothing happens, you've got a dead one.

Assuming that the proper voltages are arriving from the power supply, the most likely cause of death would be no clock signal. The Sol is a digital computer, which means, all circuits within its innards must march in step with the "drummer". If the drummer (crystal oscillator) gets tired, everybody else takes a break too. About the only way to check if the crystal and clock circuits are working is with a logic probe or an oscilloscope. A rough idea can be had by holding a small transistor radio near the PC board. If there is any clock activity, the radio will produce all sorts of whines and buzzes.

Third Step, Repair

I'm going to let you in on a secret. I'm going to tell you how the "experts" find the bad ICs. Now I know you are visualizing racks of test equipment: oscilloscopes, digital analyzers, logic emulators and the like. But that's not the secret. Oh, those things are nice for quickly getting to the bad functional area but, when it gets down to picking out the bad IC, do you know how the "experts" do it? I'll tell you. They get another Sol that's working and start swapping ICs until the trouble disappears! The great thing about this method is that you don't need any test equipment. The absolutely worst case would be that you had to swap every IC on the PC board. If you can narrow down the problem area even a little you can have your Sol up and running in no time. Now you know the measure of a real friend. He is one who will lend you his Sol for testing!

As I said, the worst problem is no video. There are 29 ICs associated with the VDM circuit and here is where that page X-24 from the Sol manual can really save you some time. It lists all of them in a neat group. Start swapping until the video returns to normal. I recommend swapping one at a time, of course turning the power off between each change. Two other conditions can cause strange video displays so if all 29 ICs have been swapped and the problem isn't cured, it could be one of them.

The VDM in the Sol is known as a "memory mapped" display circuit. This means that the characters shown on the screen are actually stored in a section of the computer's memory. In the Sol, this video memory resides between addresses CC00 and CFFF hex. Eight memory ICs make up this video block. (U14-U21) If one of them goes bad, the characters cannot be formed properly and you get a crazy looking display. A memory test of this 1K block should turn up the bad chip. Have you ever tested the video memory? It gives a wild display! The other thing which can cause funny characters to appear on the VDM is a bad keyboard. There are two ICs on the keyboard which are very sensitive to static charges. (U19,U22) I have found a number of these bad and when they go, all the wrong characters are sent out. Try exchanging keyboards from the test Sol to see if this is the problem.

The CPU can be checked by giving any of the SOLOS commands. If even one works OK then the CPU is probably not at fault. Look suspiciously at the SOLOS ROM instead. Of course, if you are trying a tape command or the TERM command then the problem might be with the tape I/O circuit or the serial UART. A program that runs amuck frequently may indicate a bad memory IC in the SOLOS RAM area. This 1K memory block (C800-CBFF) is often used for the program stack and only one bad bit can send the CPU off into never never land. If you can't get any SOLOS command to work or even get the prompt to appear, this is a good place to suspect trouble as all SOLOS commands use this memory for stack purposes. CPU support ICs can cause problems such as failure to jump to SOLOS on RESET, bad address or data signals and some really weird symptoms. This is another area to check if nothing will work.

The cassette tape circuit as well as the serial and parallel ports can best be checked by connecting a device, known to be working properly, to their respective connectors and running some test programs. Things to look for if you suspect too much voltage got on the printer cable would be the line driver ICs for the serial port, (U38,U56) or the six ICs associated with parallel I/O. The cassette tape motor relays are known to get stuck occasionally and when this happens they usually need to be replaced.

Barring a catastrophic accident, (like being hit by lightning) changing ICs should find most of the problems. Shorted circuit traces, bad IC sockets and the like of those ills generally will not be found in a computer which has been operating sucessfully for some time. Those are the bane of kit builders.

In the next issue: The memory boards.

...DEAD KEYS ON MY SOL

I would appreciate it if someone would explain what to do about dead keys on the Sol keyboard.

Emile Roth, 1001 Evelyn Terrace East, #104, Sunnyvale, CA 94086

(Editor's note: The dead keys are usually due to a bad foam pad inside the key plunger. The keytops pull straight off. Beneath, will see the mechanism which is screwed down to the keyboard. Remove the mechanism and inside you'll find a cylindrical piece of foam plastic with a self-adhering layer of metal foil on it. The foil is pressed down against the printed-circuit pads to make the k register in the circuit. Old pads fail to make good proximity. Keytronics, the manufacturer of the keyboard, will only sell the pads in large quantity (unless you happen to get to a salesperson with a soft heart that day).
IF ANYONE HAS EXTRA FOAM KEYPADS, PLEASE CONTACT EMILE.
--Stan.)

```
0000 *
0001 *  North Star PASCAL Input/Output Routine
0002 *
0003 *  This I/O routine is for use with North
0004 *  Star Pascal Ver. 1.0 and a Sol computer.
0005 *
0006 *  Written by: Stephen Maguire     July, 1980
0007 *              P.O. Box 3742 DT
0008 *              Anchorage, AK 99510
0009 *
0010 *  It provides the following support:
0011 *
0012 *  a)  It correctly interprets and performs
0013 *      the GOTOXY procedure that comes with
0014 *      the system so that BINDER need never
0015 *      be used.  This allows instant cursor
0016 *      positioning without the need to write
0017 *      the "necessary" GOTOXY procedure de-
0018 *      scribed in the manual.
0019 *
0020 *  b)  It supports PRINTER: so that output
0021 *      can be printed out.  The routine is
0022 *      for an NEC Spinwriter 5510, but will
0023 *      work for any serial printer if wired
0024 *      according to protocol shown below.
0025 *
0026 *      Spinwrtr pin     Sol Serial pin
0027 *
0028 *      TX DATA   2      3   RX DATA
0029 *      RX DATA   3      2   TX DATA
0030 *          GND   7      7   GND
0031 *          CTS   5|
0032 *          DSR   6|     20  DTR   | denotes common
0033 *          CD    8|                 connection
0034 *                     |6  DSR
0035 *          DTR  20    |8  CD
0036 *      REV CHA  19      5   CTS   See Note below:
0037 *
0038 *      Note: In this driver, the Reverse Channel
0039 *      pin of the Spinwriter is used in the "LOW"
0040 *      mode by setting #5 of SW1 to "ON" (up) on
0041 *      the control panel circuit board. (G9BNF)
0042 *      This results in a "high" to the Sol when
0043 *      characters can be accepted. If the printer
0044 *      is unplugged or turned off, the Sol still
0045 *      sees a high because of its own internal
0046 *      circuitry and will continue sending char-
0047 *      acters. This prevents a program hang (and
0048 *      possibly a crash) if the printer is not
0049 *      available.
0050 *
0051 *  c)  If input is asked of the printer, input
0052 *      from the keyboard is checked for instead.
0053 *
0054 *  d)  If a control-p is sent to CONSOLE:, output
0055 *      to CONSOLE: is sent to PRINTER: instead.
0056 *      This continues until another control-p is
0057 *      encountered.  (The control-p may be typed
0058 *      at the keyboard or output in a program.)
0059 *
0060 *  e)  On initialization, the memory is sized and
0061 *      then waits for either a carriage return
0062 *      or a hex value to set the memory limit.
0063 *      This allows the user to "protect" high
0064 *      memory if necessary.
0065 *
0066 *  f)  The underline character can be printed
0067 *      in order to facilitate compatibility
0068 *      with other systems.
0069 *
0070 *  g)  Control-L erases to end of line.
0071 *
0072 *  h)  The bell character is sent to PRINTER:
0073 *      instead of to CONSOLE:.
0074 *
0075 *  org 2400H - SYSTEM.NSTAR2  (memory at 2000H)
0076 *  org 400H - SYSTEM.NSTAR0  (memory at 0000H)
0077 *
0400            0078        ORG     400H
                0079 *
00FF            0080 TRUE   EQU     0FFH
0000            0081 FALSE  EQU     0
                0082 *
0007            0083 CHBEL  EQU     07H     The bell
000A            0084 CHLFE  EQU     0AH     Linefeed
000B            0085 CHCLR  EQU     0BH     CLEAR screen character
000C            0086 CHFFD  EQU     0CH     Formfeed
000D            0087 CHCR   EQU     0DH     Carriage return
000E            0088 CHOME  EQU     0EH     HOME CURSOR character
0010            0089 CHDLE  EQU     10H     Control-p
001B            0090 CHESC  EQU     1BH     ESCAPE character
005F            0091 CHUND  EQU     5FH     underline character
007F            0092 CHDEL  EQU     7FH     The DEL character
007F            0093 ASCII  EQU     7FH     largest ASCII
                0094 *
                0095 *  Equates determined by STANDARD SOLOS
                0096 *
C019            0097 SOLOUT EQU     0C019H  Solos output routine
C098            0098 OCHAR  EQU     0C098H  print an underline
C0F4            0099 CLINE  EQU     0C0F4H  erase to end of the line
C11C            0100 VDADD  EQU     0C11CH  calculate screen address
C340            0101 SHEX   EQU     0C340H  convert ASCII to binary
C40B            0102 HEOUT  EQU     0C40BH  print register A in ASCII
C808            0103 NCHAR  EQU     0C808H  X coordinate of cursor
C809            0104 LINE   EQU     0C809H  Y coordinate of cursor
C80A            0105 BOT    EQU     0C80AH  Text offset
                0106 *
                0107 *  CONSOLE: device (keyboard)
                0108 *
00FA            0109 CSTAT  EQU     0FAH    Keyboard status port
00FC            0110 CDATA  EQU     0FCH    Keyboard data port
0001            0111 CRDYINP EQU    1
                0112 * CRDYOUT EQU 2  (Not used)
                0113 *
                0114 *  PRINTER: (no input -- CONSOLE: input
                0115 *            is used instead)
                0116 *
00F8            0117 PSTAT  EQU     0F8H    Printer status port
00F9            0118 PDATA  EQU     0F9H
                0119 * PRDYINP EQU 2  (Not used)
00A0            0120 PRDYOUT EQU    0A0H    "high" on CTS and TBE pins
                0121 *
                0122 *  REMOTE:  device (not supported)
                0123 *
                0124 * RSTAT EQU 6  (For optional extra device.
                0125 * RDATA EQU 0   Values are from the sample I/O
                0126 * RRDYINP EQU 2   routine given in the manual.
                0127 * RRDYOUT EQU 1   OK for tape I/O if the
                0128 * RSTROBE EQU 80H   required code is written.)
                0129 * RPOFLG EQU 20H
                0130 *
                0131 *
0009            0132 NOTRDY EQU     9       Not-ready value
00E8            0133 DCTRLB EQU     0E8H    High byte of standard PROM
0080            0134 DDENS  EQU     80H     Double density
0000            0135 SDENS  EQU     0       Single density
0000            0136 ONESIDE EQU    0       For single sided drives
0040            0137 TWOSIDE EQU    40H     Quad capacity drives
0005            0138 SBLKTRK EQU    5       blocks/track in single-dens
000A            0139 DBLKTRK EQU    10      blocks/track in double-dens
```

```
                        0140 *
                        0141 * * * * * * * * * * * * * * * * * * * * *
                        0142 * Set line below according to your drives *
     00CA               0143 CHARACS EQU   TWOSIDE+DDENS+DBLKTRK  (Quad) *
                        0144 * * * * * * * * * * * * * * * * * * * * *
     0400               0145 NSJTST EQU    $
     0600               0146 STRTSR EQU    NSJTST+512
                        0147 *
                        0148 *  CONSOLE:  routine addresses
                        0149 *
0400 C3 2E 05           0150 CONOCL JMP    ONLINE  Keyboard is always "ready"
0403 C3 2D 04           0151        JMP    CONINP  Keyboard in
0406 C3 3D 04           0152 CONESC JMP    CONOUT  Video out
0409 C3 F1 04           0153        JMP    CONST
                        0154 *
                        0155 *  PRINTER:  routine addresses
                        0156 *
040C C3 2E 05           0157 PTRONL JMP    ONLINE  Keyboard is always "ready"
040F C3 2D 04           0158        JMP    CONINP  If input, go to CONSOLE: in
0412 C3 20 05           0159        JMP    PRNT2
                        0160 *
                        0161 *  REMOTE:  routine addresses
                        0162 *
0415 C3 30 05           0163 REMONL JMP    OFFLIN  Offline (not supported)
0418 C3 FB 04           0164        JMP    REMINP
041B C3 FB 04           0165        JMP    REMOUT
                        0166 *
041E C3 33 05           0167        JMP    NSMSIZ  How much memory is available?
0421 C3 30 05           0168        JMP    OFFLIN  No system clock, it is offline
0424 C3 38 05           0169        JMP    MACINT  The initialization routine
                        0170 *
0427 CA                 0171 DV4CHR DB     CHARACS  Device characteristics
0428 CA                 0172 DV5CHR DB     CHARACS  (set for your drives)
0429 CA                 0173 DV9CHR DB     CHARACS
042A CA                 0174 DV10CHR DB    CHARACS
                        0175 *
042B 00 00              0176 EXPANSN DW    0       For future use
                        0177 *
                        0178 *  CONSOLE: input  (PRINTER: input)
                        0179 *
042D DB FA              0180 CONINP IN     CSTAT  Has a key been typed?
042F 2F                 0181        CMA    .      Inverse the value
0430 E6 01              0182        ANI    CRDYINP  Strip the value
0432 CA 2D 04           0183        JZ     CONINP  No, keep waiting
0435 DB FC              0184        IN     CDATA  Yes, get the character
0437 FE 10              0185        CPI    CHDLE  Has a control-p been entered
0439 C0                 0186        RNZ    .
043A F6 80              0187        ORI    80H    Ctrl-p's have high bit set
043C C9                 0188        RET    .
                        0189 *
                        0190 *  CONSOLE: output
                        0191 *
043D 79                 0192 CONOUT MOV    A,C    Get the character
043E E6 7F              0193        ANI    ASCII
0440 4F                 0194        MOV    C,A    Save the stripped value
                        0195 *
0441 FE 07              0196        CPI    CHBEL  If bell, send to printer
0443 CA 20 05           0197        JZ     PRNT2
                        0198 *
0446 FE 1B              0199        CPI    CHESC  Escape says GOTOXY
0448 C2 57 04           0200        JNZ    CON0
044B CD 1C C1           0201        CALL   VDADD
044E 7E                 0202        MOV    A,M    Remove the cursor
044F C6 80              0203        ADI    80H
0451 77                 0204        MOV    M,A
0452 3E 01              0205        MVI    A,1    Set flag to indicate so
0454 C3 AB 04           0206        JMP    GOTO3
                        0207 *
0457 FE 10              0208 CON0   CPI    CHDLE  ctrl-p, toggle
0459 C2 66 04           0209        JNZ    CON1
```

```
045C 3A FC 05           0210        LDA    TOGGLE  Get the switch
045F C6 80              0211        ADI    80H     Now flip it
0461 32 FC 05           0212        STA    TOGGLE
0464 AF                 0213        XRA    A
0465 C9                 0214        RET
                        0215 *
0466 3A FB 05           0216 CON1   LDA    XYDATA  Is it GOTOXY?
0469 B7                 0217        ORA    A
046A CA B0 04           0218        JZ     CON2    Yes, it is
                        0219 *
                        0220 *  GOTOXY makes "ESC","=",y,x to screen address
                        0221 *
046D FE 01              0222 GOTOXY CPI    1       ESCAPE has been received,
046F C2 7F 04           0223        JNZ    GOTO0   Now check for the "=" sign
0472 79                 0224        MOV    A,C
0473 FE 3D              0225        CPI    '='
0475 CA A5 04           0226        JZ     GOTO2   Yes, its the "=" sign
0478 AF                 0227        XRA    A       Error, abort GOTOXY procedure
0479 32 FB 05           0228        STA    XYDATA
047C C3 3D 04           0229        JMP    CONOUT  And output the Character
                        0230 *
047F FE 02              0231 GOTO0  CPI    2       Calculate the row value
0481 C2 8F 04           0232        JNZ    GOTO1
0484 79                 0233        MOV    A,C     Get the row value
0485 DE 20              0234        SBI    32      Sub 20H to get correct value
0487 E6 0F              0235        ANI    0FH     Make sure value in range
0489 32 09 C8           0236        STA    LINE
048C C3 A5 04           0237        JMP    GOTO2
                        0238 *
048F 79                 0239 GOTO1  MOV    A,C     Calculate column value
0490 DE 20              0240        SBI    32      Subtract the offset
0492 E6 3F              0241        ANI    3FH     Make sure value in range
0494 32 08 C8           0242        STA    NCHAR   Store the value
0497 CD 1C C1           0243        CALL   VDADD   Calculate screen address
049A 7E                 0244        MOV    A,M     Save character at cursor pos
049B F6 80              0245        ORI    80H     Put the cursor there
049D 77                 0246        MOV    M,A     This does it
                        0247 *
049E 3A FC 05           0248        LDA    TOGGLE  If print on, output CR/LF
04A1 B7                 0249        ORA    A
04A2 C4 11 05           0250        CNZ    PRNT0
                        0251 *
04A5 3A FB 05           0252 GOTO2  LDA    XYDATA  Increment pointer so we
04A8 3C                 0253        INR    A       know which argument to get.
04A9 E6 03              0254        ANI    3
                        0255 *
04AB 32 FB 05           0256 GOTO3  STA    XYDATA  Save the value
04AE AF                 0257        XRA    A
04AF C9                 0258        RET    .
                        0259 *
                        0260 *  All done with the GOTOXY procedure
                        0261 *
04B0 3A FC 05           0262 CON2   LDA    TOGGLE  If PRINTER: is on,
04B3 B7                 0263        ORA    A       send it the output
04B4 C2 FC 04           0264        JNZ    PRINT
                        0265 *
04B7 79                 0266        MOV    A,C
04B8 FE 0C              0267        CPI    CHFFD   Formfeed character?
04BA C2 CB 04           0268        JNZ    CON3    No, so go print character
04BD E5                 0269        PUSH   H       HL cannot be destroyed
04BE CD 1C C1           0270        CALL   VDADD
04C1 E5                 0271        PUSH   H       Save this screen address
04C2 CD F4 C0           0272        CALL   CLINE   Call erase
04C5 E1                 0273        POP    H       Get it back
04C6 36 A0              0274        MVI    M,0A0H  Put on the cursor
04C8 E1                 0275        POP    H
04C9 AF                 0276        XRA    A
04CA C9                 0277        RET    .
                        0278 *
04CB FE 7F              0279 CON3   CPI    CHDEL   Delete?, put cursor-left
```

```
04CD C2 D2 04      0280          JNZ     CON4
04D0 3E 01         0281          MVI     A,1
                   0282 *
04D2 FE 5F         0283 CON4     CPI     CHUND   allow underline
04D4 C2 E7 04      0284          JNZ     CON5
04D7 E5            0285          PUSH    H       Save these registers
04D8 C5            0286          PUSH    B
04D9 47            0287          MOV     B,A     Put the character in B
04DA CD 98 C0      0288          CALL    OCHAR   Output the character
04DD CD 1C C1      0289          CALL    VDADD   Now, put on the cursor
04E0 7E            0290          MOV     A,M     Do it now
04E1 F6 80         0291          ORI     80H
04E3 77            0292          MOV     M,A
04E4 C3 ED 04      0293          JMP     CON6    Exit gracefully
                   0294 *
04E7 E5            0295 CON5     PUSH    H       Save all registers,
04E8 C5            0296          PUSH    B
04E9 47            0297          MOV     B,A     Put the character in B
04EA CD 19 C0      0298          CALL    SOLOUT  Now, put to screen
                   0299 *
04ED C1            0300 CON6     POP     B       Restore the registers
04EE E1            0301          POP     H
04EF AF            0302          XRA     A
04F0 C9            0303          RET
                   0304 *
                   0305 *  CONSOLE: input status
                   0306 *
04F1 DB FA         0307 CONST    IN      CSTAT
04F3 2F            0308          CMA     .
04F4 E6 01         0309          ANI     CRDYINP
04F6 3E 00         0310          MVI     A,FALSE
04F8 C8            0311          RZ      .       Return now if "not ready"
04F9 3E FF         0312          MVI     A,TRUE
                   0313 *
                   0314 *  REMOTE:  not implemented so do RETURN
                   0315 *
     04FB          0316 REMIN    EQU     $
04FB C9            0317 REMOUT   RET     .
                   0318 *
                   0319 *  PRINTER:  output (via control-p toggle)
                   0320 *
04FC 79            0321 PRINT    MOV     A,C     Get the character
04FD FE 0D         0322          CPI     CHCR
04FF CA 20 05      0323          JZ      PRNT2
0502 FE 0A         0324          CPI     CHLFE
0504 CA 20 05      0325          JZ      PRNT2
0507 FE 0E         0326          CPI     CHOME   Home?, print C/R, LF
0509 CA 11 05      0327          JZ      PRNT0
050C FE 0B         0328          CPI     CHCLR   Clear also
050E C2 1B 05      0329          JNZ     PRNT1
                   0330 *
0511 0E 0D         0331 PRNT0    MVI     C,CHCR  Print it
0513 CD FC 04      0332          CALL    PRINT
0516 0E 0A         0333          MVI     C,CHLFE
0518 C3 FC 04      0334          JMP     PRINT
                   0335 *
051B FE 20         0336 PRNT1    CPI     20H
051D DA 2E 05      0337          JC      ONLINE  Control char?, ignore it
                   0338 *
                   0339 *  PRINTER:  output (standard output routine)
                   0340 *
0520 DB F8         0341 PRNT2    IN      PSTAT
0522 E6 A0         0342          ANI     PRDYOUT
0524 FE A0         0343          CPI     PRDYOUT
0526 C2 20 05      0344          JNZ     PRNT2   Loop until ready
0529 79            0345          MOV     A,C
052A E6 7F         0346          ANI     ASCII
052C D3 F9         0347          OUT     PDATA
                   0348 *
052E AF            0349 ONLINE   XRA     A
052F C9            0350          RET
                   0351 *
0530 3E 09         0352 OFFLIN   MVI     A,NOTRDY
0532 C9            0353          RET
                   0354 *
0533 2A FA 05      0355 NSMSIZ   LHLD    MEMORY  Get memory size
0536 E3            0356          XTHL
0537 E9            0357          PCHL    Return now
                   0358 *
                   0359 *  Boot-up initialization
                   0360 *
0538 21 B3 05      0361 MACINT   LXI     H,MES1  Print start message
053B CD A0 05      0362          CALL    PRASC
                   0363 *
053E 21 FD 05      0364          LXI     H,ENDMARK  Start sizing at end
0541 3E AA         0365          MVI     A,0AAH  Test byte
0543 46            0366 SIZE     MOV     B,M     Save this memory location
0544 36 AA         0367          MVI     M,0AAH  Put in test byte
0546 BE            0368          CMP     M       OK? If not, ROM or no memory
0547 70            0369          MOV     M,B     Put the old value back
0548 23            0370          INX     H       Move to the next location
0549 CA 43 05      0371          JZ      SIZE
                   0372 *
054C 2B            0373          DCX     H       Point to last good location
054D 2B            0374          DCX     H
054E 7D            0375          MOV     A,L     Make HL an even number
054F E6 FE         0376          ANI     0FEH
0551 6F            0377          MOV     L,A
0552 EB            0378          XCHG    .       DE has last memory location
                   0379 *
0553 21 DC 05      0380          LXI     H,MES2  Print this message
0556 CD A0 05      0381          CALL    PRASC
0559 CD AB 05      0382          CALL    PRHEX   Now, print the value
055C 0E 20         0383          MVI     C,' '   Print a following blank
055E CD 3D 04      0384          CALL    CONOUT
                   0385 *
0561 CD 1C C1      0386          CALL    VDADD   Get the cursor address
0564 E5            0387          PUSH    H       Save this
0565 06 00         0388          MVI     B,0     So far, no characters
0567 CD 2D 04      0389 DIFSIZE  CALL    CONIN   Check for input
056A CA 67 05      0390          JZ      DIFSIZE  None yet
056D FE 0D         0391          CPI     CHCR    Carriage return?
056F CA 87 05      0392          JZ      DIF4    Terminate input
0572 FE 7F         0393          CPI     CHDEL   Delete?
0574 C2 80 05      0394          JNZ     DIF3    No, much be a hex number
                   0395 *
0577 78            0396          MOV     A,B     If no characters typed,
0578 B7            0397          ORA     A
0579 CA 67 05      0398          JZ      DIFSIZE  Then don't back up
057C 05            0399          DCR     B       Else adjust character count
057D 05            0400          DCR     B       (for later adjustment)
057E 3E 5F         0401          MVI     A,CHDEL-20H  Output back-up
                   0402 *
0580 04            0403 DIF3     INR     B
0581 CD E7 04      0404          CALL    CON5    allow delete
0584 C3 67 05      0405          JMP     DIFSIZE   Go get more input
                   0406 *
0587 CD 1C C1      0407 DIF4     CALL    VDADD   Get cursor location
058A 36 20         0408          MVI     M,' '   Erase the cursor
058C E1            0409          POP     H       Get start location of hex
058D 78            0410          MOV     A,B     Any numbers been entered?
058E B7            0411          ORA     A
058F CA 9B 05      0412          JZ      DIF5    If not, use calculated value
0592 EB            0413          XCHG    .       Otherwise,
0593 CD 40 C3      0414          CALL    SHEX    Convert
0596 7D            0415          MOV     A,L     Make sure the value is even
0597 E6 FE         0416          ANI     0FEH
0599 6F            0417          MOV     L,A
059A EB            0418          XCHG    .       Put the value in DE
                   0419 *
```

```
059B EB            0420 DIF5   XCHG    .       This puts DE in HL
059C 22 FA 05      0421        SHLD    MEMORY  Save it for later
059F C9            0422        RET
                   0423 *
                   0424 *  Message print routine
                   0425 *
05A0 AF            0426 PRASC  XRA     A       This tests end-of-msg
05A1 BE            0427        CMP     M       At end?
05A2 C8            0428        RZ      .       If so, return immediately
05A3 4E            0429        MOV     C,M     Get the character
05A4 CD 3D 04      0430        CALL    CONOUT  Send it out
05A7 23            0431        INX     H       Bump pointer to next
05A8 C3 A1 05      0432        JMP     PRASC+1  keep looping
                   0433 *
                   0434 *  PRHEX print value of DE as ASCII
                   0435 *
05AB 7A            0436 PRHEX  MOV     A,D     Print D
05AC CD 0B C4      0437        CALL    HEOUT
05AF 7B            0438        MOV     A,E     Print E
05B0 C3 0B C4      0439        JMP     HEOUT
                   0440 *
                   0441 *  Messages and data storage
                   0442 *
05B3 0B            0443 MES1   DB      CHCLR
05B4 55 43 53 44   0444        ASC     'UCSD Pascal for Solos'
     20 50 61 73
     63 61 6C 20
     66 6F 72 20
     53 6F 6C 6F
     73
05C9 0D 0A         0445        DB      CHCR,CHLFE,CHLFE
05CC 53 69 7A 69   0446        ASC     'Sizing memory'
     6E 67 20 6D
     65 6D 6F 72
     79
05D9 0D 0A         0447        DB      CHCR,CHLFE,0
                   0448 *
05DC 4D 65 6D 6F   0449 MES2   ASCZ    'Memory available to address: '
     72 79 20 61
     76 61 69 6C
     61 62 6C 65
     20 74 6F 20
     61 64 64 72
     65 73 73 3A
     20 00
05FA 00 00         0450 MEMORY DW      0
05FC 00            0451 XYDATA DB      0
05FD 00            0452 TOGGLE DB      0
                   0453 *
     05FE          0454 ENDMARK EQU    $
                   0455 *
                   0456        END
```

EXTENSIONS FOR SOFTWARE # 1: FILE RENUMBERING

Software #1 is a complete Editor-Assembler-Monitor package which was distributed by Processor Technology. It was produced in at least two versions that I know of. The earliest version was a true self-contained system that included drivers for a terminal interface. This version was distributed with both source and object code back in the days when PTCo did not yet make a complete computer system. A later version was distributed as a CUTS tape (object only). This one took advantage of many features of the CUTER/SOLOS monitor, including the tape interface. I suspect that ALS-8 may be basically an enhancement of Software #1--in any case they both use the same kind of file structure, in which lines are numbered and each line is preceded by a byte count.

I obtained the original version of Software #1 some time ago, before I decided to make my homebrew system into a pseudo-Sol. Because it fit into 4K of RAM, could be adapted for ROM, and was well documented I made much use of the package, modified it extensively, and learned how it worked internally. Many of my modifications would be hard to apply to the more well-known cassette version, but I have rewritten a few of them to apply to this version. I would like to make a few of these available to PROTEUS members.

The following program renumbers a Software #1 source file. The first line is numbered 0010 and succeeding lines are spaced 10 numbers apart. This is a big step toward making Software #1 easy to use for serious program development.

The listing that follows has an odd origin because it resides in my system immediately below a 6502 cross-assembler that sits in the 1800H to 1FFFH region and. This cross-assembler is a parasite on Software #1 which uses every possible subroutine in the PTCo package. I didn't want to tamper with Software #1 itself, so I let it continue to use the area from 0F60H up for the assembler symbol table. This is why my extensions are crowded into the top of the 1st 8K of memory. Since Software #1 is a memory-to-memory assembler it is unlikely that any programs to be assembled will be so large they will require more than 1K for the symbol table.

Albert S. Woodhull
RFD 2
Amherst, Mass., 01002

```
FILE
NUMB  2000  2AAF

ASSM 1754 3000


1754                0010 ;NUMB
1754                0020 ;TO RENUMBER SOURCE FILE
1754                0030 ;      A.S.WOODHULL
1754                0040 ;      7/VII/79
1754                0050 ;     edited 25 Jan 81
1754                0060 ;
1754                0070 ;TEST FOR CURRENT FILE
1754 CD 00 18       0080 NUMB   CALL CFTST
1757                0090 ;
1757                0100 ;1ST CHECK FOR LESS THAN 999 LINES
1757 11 00 00       0110 NUM    LXI D,0
175A 2A 58 0D       0120        LHLD BOFP
175D                0130 ;
175D                0140 ;GET COUNT,TEST FOR EOF
175D 7E             0150 CNT    MOV A,M
175E FE 01          0160        CPI 1
1760 CA 79 17       0170        JZ DONUM
1763                0180 ;
1763                0190 ;POINT TO NEXT COUNT
1763 85             0200        ADD L
1764 6F             0210        MOV L,A
1765 7C             0220        MOV A,H
1766 CE 00          0230        ACI 0
1768 67             0240        MOV H,A
1769                0250 ;COUNT THAT LINE
1769 13             0260        INX D
176A                0270 ;LESS THAN 999 LINES?
176A 7A             0280        MOV A,D
176B FE 03          0290        CPI 3
176D DA 5D 17       0300        JC CNT
1770 7B             0310        MOV A,E
1771 FE E8          0320        CPI 0E8H
1773 DA 5D 17       0330        JC CNT
1776                0340 ;QUIT IF TOO MANY LINES
1776 C3 F1 17       0350        JMP ABORT
1779                0360 ;
1779                0370 ;INITIALIZE LINE NO,GET SOF
```

```
1779 21 30 30            0380 DONUM  LXI H,3030H
177C 22 BF 0D            0390        SHLD LNLO
177F 22 BD 0D            0400        SHLD LNHI
1782 2A 58 0D            0410        LHLD BOFP
1785 EB                  0420        XCHG
1786                     0430 ;
1786 1A                  0440 NLOOP  LDAX D          ;GET LINE LENGTH
1787 FE 01               0450        CPI 1           ;CK FOR END OF FILE
1789 CA E2 17            0460        JZ EOF
178C                     0470 ;POINT TO LSD OF LINE NO.
178C 13                  0480        INX D
178D 13                  0490        INX D
178E 13                  0500        INX D
178F 13                  0510        INX D
1790                     0520 ;REPLACE OLD LINE NUMBERS
1790 3E 30               0530        MVI A,30H       ;ASCII ZERO
1792 12                  0540        STAX D
1793 3A BF 0D            0550        LDA LNLO        ;GET 10'S DIGIT
1796 3C                  0560        INR A
1797 FE 3A               0570        CPI 3AH         ;DECIMAL CY TEST
1799 D2 A7 17            0580        JNC CY10
179C 1B                  0590        DCX D
179D 12                  0600        STAX D
179E 32 BF 0D            0610        STA LNLO
17A1 3A BE 0D            0620        LDA LNHI+1      ;GET 100'S DIGIT
17A4 C3 B7 17            0630        JMP DOC
17A7                     0640 ;IF CARRY OUT OF 10'S DO NEXT
17A7 3E 30               0650 CY10   MVI A,30H
17A9 1B                  0660        DCX D
17AA 12                  0670        STAX D
17AB 32 BF 0D            0680        STA LNLO
17AE 3A BE 0D            0690        LDA LNHI+1      ;GET 100'S
17B1 3C                  0700        INR A
17B2 FE 3A               0710        CPI 3AH         ;TST FOR DEC CARRY
17B4 D2 C2 17            0720        JNC CYC
17B7 1B                  0730 DOC    DCX D           ;STORE 100'S
17B8 12                  0740        STAX D
17B9 32 BE 0D            0750        STA LNHI+1
17BC 3A BD 0D            0760        LDA LNHI        ;GET 1000'S
17BF C3 D2 17            0770        JMP DOK
17C2                     0780 ;DO NEXT IF CARRY OUT OF 100'S
17C2 3E 30               0790 CYC    MVI A,30H
17C4 1B                  0800        DCX D
17C5 12                  0810        STAX D
17C6 32 BE 0D            0820        STA LNHI+1
17C9 3A BD 0D            0830        LDA LNHI        ;GET 1000'S
17CC 3C                  0840        INR A
17CD FE 3A               0850        CPI 3AH
17CF D2 F1 17            0860        JNC ABORT
17D2 1B                  0870 DOK    DCX D           ;STORE 1000'S
17D3 12                  0880        STAX D
17D4 32 BD 0D            0890        STA LNHI
17D7                     0900 ;
17D7                     0910 ;NOW COMPUTE START OF NEXT LINE
17D7 1B                  0920 NEXT   DCX D
17D8 1A                  0930        LDAX D
17D9 83                  0940        ADD E
17DA 5F                  0950        MOV E,A
17DB 7A                  0960        MOV A,D
17DC CE 00               0970        ACI 0
17DE 57                  0980        MOV D,A
17DF C3 86 17            0990        JMP NLOOP
17E2                     1000 ;
17E2                     1010 ;WHEN DONE PUT NEW MAXLINE IN DIR
17E2 2A BD 0D            1020 EOF    LHLD LNHI
17E5 22 5C 0D            1030        SHLD MAXL
17E8 2A BF 0D            1040        LHLD LNLO
17EB 22 5E 0D            1050        SHLD MAXL+2
17EE C3 16 00            1060        JMP RDY
17F1                     1070 ;
17F1                     1080 ;ONLY ALLOW 999 LINES
17F1 21 FA 17            1090 ABORT  LXI H,MSG
17F4 CD 2A 02            1100        CALL SCRN
17F7 C3 16 00            1110        JMP RDY
17FA                     1120 ;
17FA 41 42               1130 MSG    DW 'BA'
17FC 4F 52               1140        DW 'RO'
17FE 54                  1150        DB 'T'
17FF 0D                  1160        DB 0DH
1800                     1170 ;
1800                     1180 ;TEST FOR CURRENT FILE
1800 2A 58 0D            1190 CFTST  LHLD BOFP
1803 7C                  1200        MOV A,H
1804 B5                  1210        ORA L
1805 CA 60 04            1220        JZ NFERR
1808 C9                  1230        RET
1809                     1240 ;
1809                     1250 ;EQUATES FOR CASSETTE VERS 1.0 OF SOFT1
1809                     1260 RDY    EQU 0016H       ;RE-ENTER SOFT1
1809                     1270 SCRN   EQU 022AH       ;STRING PRINTER
1809                     1280 NFERR  EQU 0460H       ;NO CURRENT FILE ERROR
1809                     1290 ;
1809                     1300 ;STORAGE LOCS ARE IN ASCII BUFFER AREA
1809                     1310 LNHI   EQU 0DBDH
1809                     1320 LNLO   EQU 0DBFH
1809                     1330 ;
1809                     1340 ;FILE PARAMETERS
1809                     1350 FILE0  EQU 0D53H       ;CURRENT FILE NAME
1809                     1360 BOFP   EQU FILE0+5     ;POINTER TO START
1809                     1370 MAXL   EQU FILE0+9     ;MAX LINE NO.
```

WRITE AND IMAGE

Allen T. Fincher
Suffolk, VA

Simple stated, WRITE performs the opposite of READ (see Bill Blomgren's letter on page 16 of Vol. 3, #3). WRITE puts the contents of the specified area of memory into the given file verbatim; that is, without any additional bytes added to the written data.

When a file is written using IMAGE, the first two bytes written into the file are the length of the segment, the next two bytes are the load address of the segment, and then the data within the segment is written. If a start address is included, it is written last. This is the information printed on the console when the 'EXTRACT filename' command is used.

A program cannot be written on a disk and later executed by just typing its file name if it was put on disk using WRITE, because there would be no segment length, load address, or execution address included in the file for PTDOS to use. I am not referring to a program that was read off a disk using READ, but instead, to a new file.

Also, an existing image file cannot be put into memory using READ and then written to disk using IMAGE and have it run when subsequently loaded by typing its file name. Instead, it must be written out to disk using WRITE to that the proper segment length and load address which are also in memory along with the program code will be the only ones in the file.

Note the example of the IMAGE command at the bottom of page 2-25 of the PTDOS manual. It shows a file that will have several areas of memory contained within that file, all of which will be loaded with their respective data or code when the file is reloaded by typing its name. This can be done because the load length and load address of each segment is recorded in the file along with the data for the segments when IMAGEd.

LETTERS TO THE EDITOR:

..BELLS ON MY SOL

I have always envied terminals with an audible BELL which responds to control-G. Now my SOL has the makings of that feature.

Radio Shack's solid state buzzer 273-060 sells for $2.99 and will operate directly from the SOL parallel port! The loudness is about right too. Just connect the black lead to pin 1 or 2 of an appropriate connector and connect the red lead to a data output bit (Pins 18-25). The buzzer can be turned on by:

```
MVI  A,0FFh     ;turns on all 8 bits
OUT  0FDh
```

and can be turned off by:

```
MVI  A,0        ;turns off all 8 bits
OUT  0FDh
```

Of course you'll need more sophisticated code if you are using other parallel output bits for other things.

Now when remote users of my system want to get my attention they run a CP/M program called BEEP. I still haven't figured out a useful way to use control-G because CP/M converts it to two letters (^G) before it echos it to my CP/M console output routine.

Also, here are my dues for 1981. Keep up the good work.

Sincerely,

Dick Greenlaw 2/1/81

Dick Greenlaw

..PRODUCT REVIEW 18 March 1981

Since I find product reviews one of the most useful features in Proteus/News here are three brief ones that reflect my experience over the past two years.

1. Paper Tiger Printer. I replaced my old TTY with an IDS 440 and after eighteen months of moderately heavy use it continues to perform flawlessly. It is connected to the parallel port so it prints at maximum speed but can be used with the serial port if preferred. The chief reason I selected an IDS unit was to get the graphics option since I do lots of plotting. The technique is quite clever; the seven dots of the print head are arranged in a vertical column and the binary bit configuration of ones and zeros controls which pin head needles are fired. Programming graphics is like using machine language so it's tedious but the results are accurate and repeatable.

Incidently, I donated the ASR-33 to a deaf group; they have a great need for TTY's in working condition.

2. Central Data Dynamic RAM. Another good product; install and forget. I wanted to add the top 32k to my system to accomodate ALS8. The excellent CD manual uses as an example,complete with diagrams, the Sol with SOLOS occupying the C000 block to demonstrate their deselect feature using mini-jumps so even a duffer like me couldn't get it wrong.

3. Exatron Stringy Floppy. This time a real fiasco. Their literature looked good. I was especially pleased to find that they used Proc.Tech. ECBasic with modified I/O. I won't go into the gory details but delivery promises were broken time after time and when the unit was finally shipped, it didn't work. EXATRON's suggestion was that I replace all 64k of dynamic memory with static memory. That didn't sound very cost-effective and I declined. Not recommended.

Len Kalish
580 S.San Vicente Blvd.#3
Los Angeles,CA 90048

..QUESTIONS ON SOL MODS, PRODUCT REVIEW

I have several questions for you or other PROTEUS members. First, has anyone upgraded their SOLs to the 4 Mhz 8080 CPU and if so, how was that done. Second, I see on my SOL PC board a video expantion interface. What is that for? The only reference to the interface is in the parts list and in the X drawings. Third and last of the questions, how could I make the SOL jump on reset to E800 rather than to C000? It would be much more convenient for me to have my North Star boot up on reset.

I recently completed the NOISEMAKER II programmable sound generator kit from Ackerman Digital Systems. The two AY-3-8910 programmable sound generators each have 3 12-bit tone generators, 1 4-bit amplitude control for each of the 3 tone sources, 1 5-bit noise generator, and 1 16-bit envelope generator. Also, each AY-3-8910 has 2 I/O ports for user applications plus, the tone generators may be used as D/A convertors rather than for sound production. Overall, the kit was fairly simple to make, and to my surprise, it worked the first time. The variety of sounds that might be produced with the 6 tone generators is fantastic as is the quality of tone. The board is a standard S-100 and it has a breadboard area where addition of on-board memory could be accomplished. Except for the vague instructions on how to address the individual PSG (programmable sound generator) and the minimal number of examples of sound generation, I would have to give give this project an EXCELLANT rating. For more information pertaining to the PSG, I found the data manual for the chip to very informative. Also, the data manual gives many examples for sound generation and the more complex musical compositions. The tones may be produced between the low of 30.5 Hz to the high of 125 kHz with a 2 mHz input clock. My computer now plays music!!

Sincerely,

ANDREW R. BOND
Box 233
Graton, CA. 95444
(707) 823-1232

...ON HELIOS IN A Z-80 SYSTEM

A while ago, Joe Maguire was asking about running a Helios on a 4 MHz Z-80 system. I have been doing this for a while and I'll be glad to correspond with anyone having problems with such a combination.

Tom Quinn, Route 2, Box 234K, Eatonville, WA 98328.

..HELP NEEDED ON MICROPOLIS DISK 1053 11

Is there anyone out there or do you know how to put P.T.s' Game programs and Basic on my Micropolis Disk 1053 11 to operate under CPM. I'm a complete novice but do really enjoy learning about my computer,so if someone can help please make it with simple stupid instructions. Thank you.

Bruce G. Diller
18651 E. Gallarno Drive
Covina, CA. 91722
(213) 966-0710

...ON USING BOTH SIDES OF A FLOPPY DISKETTE 2/1/81

I am now using both sides (one at a time) of many of my diskettes even though I have single-sided drives. It certainly saves a lot of money and space.

No, I didn't fork over $12 for a punch. I was encouraged by reports it is easier than that, and it is. The following procedure is for 5" diskettes, but should be about the same for 8" diskettes.

To convert regular minidiskettes to flip-over dual-sided diskettes (for use on single sided drives) I use an ordinary paper punch which makes a hole about the size of the sector hole window already present. I got a punch at an Ace hardware for less than a dollar with a plastic crap catcher. It's the pliers type, but with a mild curve to the handle so I don't have to bend the diskette too much to get in through the center hole.

I protect the diskette surface by using a clean work surface and by inserting half an index card into the center hole so it is always between the punch and the magnetic media.

Procedure: make a template using carbon paper if necessary. You want a sturdy but thin square outline of a diskette envelope with the sector hole, guide notches and write protect notch carefully marked. The center hole is not needed. Punch and cut out the two holes. The guide marks are just to help distinguish the two sides.

Allign the template on the back side of a diskette with the guide notch marks on the marked (read hole) edge of the diskette. Mark the new sector hole and the write protect notch. A red ball point pen will work. Turn the diskette with template over and put the template on top with the same orientation. Mark the new sector hole on this side.

Put the protective paper or card into the center hole on either side, centered on the new sector hole location. Bend the diskette carefully to allow the punch to get to the new sector hole location on that side via the center hole. There is a stretchy plastic liner as well as the cardboard. They are not fused together, so be sure to get both in the punch jaws. The plastic prefers to stretch, so punch several times in the same place and turn the punch while it is closed to get good cutting action without loosing allignment.

Then do the same on the other side and punch the write protect hole on the edge. A semi-circle worked ok with my Micropolis drives. The punch is slightly wider than Memorex's square holes. That is great, because my drive sometimes misreads the Memorex hole with disastrous results if I don't notice the l.e.d.

Format while you are doing the next diskette.

I see only one disadvantage: you could want files from both sides at the same time! Because of that, I am using this technique mostly for archival storage and inactive diskettes.

Dick Greenlaw
251 Colony Ct.
Gahanna, Ohio 43230

EVERYTHING YOU EVER WANTED TO KNOW ABOUT THE USE OF THE H-L REGISTER PAIR IN THE SOLOS/CUTER EXECUTION OPERATION BUT DIDN'T KNOW YOU WANTED TO KNOW

In Michael McKelvey's article in Vol. 3, #4 about transferring CUTS cassette files to PTDOS image files brought up a question I've seen raised in past issues but about which little has been explained. The reason the H-L register pair setting is important is that when the EXEC command is given in SOLOS/CUTER, the H-L register pair is loaded with the starting address of SOLOS or CUTER, whichever is being used. So in SOLOS, H-L will contain C000H when leaving SOLOS after performing the EXEC command.

The reason for this is mainly for running programs using CUTER. Because CUTER can run just about anywhere in the computer's ram area, programs written would not know where the CUTER I/O vectors were unless that program's I/O calls were changed. So what is done in SOLOS/CUTER compatible programs is that when first entered, the H-L register pair is saved in memory. Then when an I/O operation is to be performed, the following routine would find the proper SOLOS/CUTER vector location. Let's say that a keyboard read from SINP is to be performed. Then a CALL to the following will find where to go:

```
GETKEY: LHLD STARTADDR  *SOLOS/CUTER start addr saved on entry
        PUSH D          *Preserve D-E
        LXI  D,19H      *Offset to SINP
        DAD  D          *Make H-L point to SINP
        POP  D          *Restore D-E
        PCHL            *Jump to SINP, use its return
```

Or the actual call operands within the program may be over-written during initialization to point to the various I/O vectors as follows:

```
        LDHL STARTADDR  *Was stored on entry to program
        LXI  D,19H      *Offset to SINP
        DAD  D
        SHLD GETCH+1    *Over-write CALL addr value
        .
        .
        .
        .
GETCH:  CALL 0000H
```

I hope this helps clear the confusion over the H-L register problem.

Allen T. Fincher
Suffolk, VA

..TARBELL ON SOL

...If anyone in the area is interested, I'm adding modified Tarbell single density controller boards to SOLS. After they are added, the SOL will run PTDOS or CP/M merely by pressing the right key on boot. It makes the SOL more versatile. Also will have a SOL 2E parallel printer driver running under CP/M soon. My home phone is now (213) 345-3662.

Regards,

Jack Kinney

UNCLASSIFIED ADS

Do YOU have any SOL/HELIOS programs which the other members of Proteus could benefit from? Help others from having to re-invent the wheel! Send your disks to Chuck Athey, Helios Disk Librarian
5571 Shorehaven Circle
Livermore, Ca 94550

To those who donated programs for the latest H-7 library disk my thanks. If I inadvertently messed your programs up please let me know, keep them coming! Chuck Athey (415) 449-8337

---

1 Diablo 1620 KSR for sale, very good condition. $1400 or best offer. Call Chuck Athey @ 415-449-8337 or 5571 Shorehaven Circle, Livermore, Ca 94550

For sale: FMT and MACRO as described in 'Software Tools', both with major inhancements, ie. Table of Contents generation in FMT..... Either will compile using the small C compiler by Ron Cain or BDS-C. 8080 Assembly version also available. Either for $30.00 on Helios disk or $25.00 on Cuts Tape. CP/M IBM formatted disk also available. $10.00 discount for both. Contact Chuck Athey Sunrise Computer, (415) 449-8337, 5571 Shorehaven Circle, Livermore, Ca 94550.

FOR SALE: SOL 20 Rev. E . Excellent condition with North Star Dick Controller and Two SA-400 drives and assorthed games and software (including CPM and Whatsit). $1800. or best offer.
Mike Erickson, Alphanetics Engineering Consulting
P.O. Box 597
Forestville, CA 95436 (707)887-7237

FOR SALE: SOL 20 Less keyboard - no extra memory, but otherwise complete. $300.00
Dennis Polito
2411 Lincoln Ave.
Belmont, CA 94002 (415)592-5319 or 595-3949(answer. machine)

FOR SALE: Helios II controller and formater boards, $300.00 or best offer. (Drive not inluded - use as backup boards). PTC 16KRA memory board $100.00 32K static Dytron memory board (See Solus News June 1978) $300.00.
Ron Parsons
9001 Laurel Grove Dr,
Austin, TX 78758 (512)836-2514

COMPLETE SOL WITH NORTHSTAR FOR SALE: Sol, 32K Northstar minifloppy, Integral Data Systems printer. $1200 or best offer.
BobTyler
2329 Thompson Pl.
Santa Clara, CA 95050 (408)244-4457

Feb. 9, 1981

Stan Sokolow
1690 Woodside Road, Suite 219
Redwood City, CA 94061

Dear Stan,

Attached is an article for PROTEUS. This one deals with a set of assembly language subroutines for use with MICROPOLIS BASIC. Some were designed when I converted several ECBASIC programs from the Proteus Cassette Library. The current set was done after a request from Paul Beauvais of Oxnard for a way to provide cursor controls which would facilitate his transfer of a business program from ECBASIC to MICROPOLIS BASIC. My thanks to him for the inspiration and the push. Of course, a copy will go to Lewis for inclusion in the Cassette Library.

Most recently my efforts have been bent towards moving (and enhancing) SOLOS to higher memory. I choose to add six new commands:

1. FILL addr1 to addr2 with hex byte.
2. MOVE from addr1 through addr2 to addr3 and up.
3. COMPARE and display mismatches from addr1 through addr2 with data at addr3 and up.
4. CS is cold start (MICROPOLIS cold boot at F400H).
5. CW is CP/M warm start at 0H.
6. MW is MICROPOLIS warm start at 4E7H.

I also put in a routine to initialize my CROMEMCO TU-ART on SOL power up (not strickly needed, but good practice).

The expanded SOLOS is now located at E000H in a pair of 2708's. The overflow (about 150 bytes) is in another 2708 at F000H and is located on a SOLID STATE MUSIC SYSTEMS PR-1. I decided on E000H for the new SOLOS instead of the more common F000H since it saved moving the MICROPOLIS controller from F400H. The only change to the SOL-20 was to bend U22-9 out so it did not enter the socket and then solder a jumper from U22-9 & U22-12.

The addition of a GODBOUT RAM-XX 24K static memory to go with my 32K DYNABYTE, gives me 56K of RAM plus SOL RAM & room for about 3.5K of EPROM (as yet unassigned). Now my Digital Research PL/I-80 has room to compile and link at least 30K source files.

I had just finished these changes when my copy of PROTEUS Vol. 3 No.5/6 arrived with Bob Stek's letter in it. I found the same problem with ERAS1 (although in my case I could fix it by changing the CFI 0D0H to CFI 0F0H). However, Bob didn't mention one other problem, namely, a typo in the CP/M source file. The error is in the GTBYT routine and prevents correct byte-mode cassette operation. The second MOV M,A should be a MOV A,M.

Note that four bytes of code from C037 to C03A are marked as part of the start-up routine. For the life of me, I can't find a need for it in SOLOS. Maybe it is needed for CUTER compatibility. I have left it out of my extended version of SOLOS with no ill effects yet!!

Yours truly,

MELVIN M. DALTON
7826 WEST 80TH STREET
PLAYA DEL REY, CA 90291

35/each

20

T A B L E O F C O N T E N T S

N E W P H O N E N U M B E R F O R P R O T E U S

Write it down. (415) 368-2300. That's your direct line to Proteus. We were using my dental office phone before, but the volume of calls was too much to bear. So please don't use any other number for calling us, but (415) 368-2300. Jane, our executive secretary, will answer the phone every weekday. If there is no answer, we are out. This way you won't have to pay for a long distance call, only to get my answering service.
The best time to call to speak to me is Wednesday [redacted] P.M Pacific time.

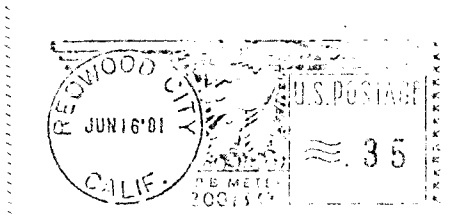
--Stan.

P R O T E U S / N E W S

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, USA, telephone (415) 368-2300.

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

From:
Proteus
1690 Woodside Road, Suite 219
Redwood City, California 94061
USA



Joe Maguire
PO Box 3742 DT
Anchorage, AK

99510