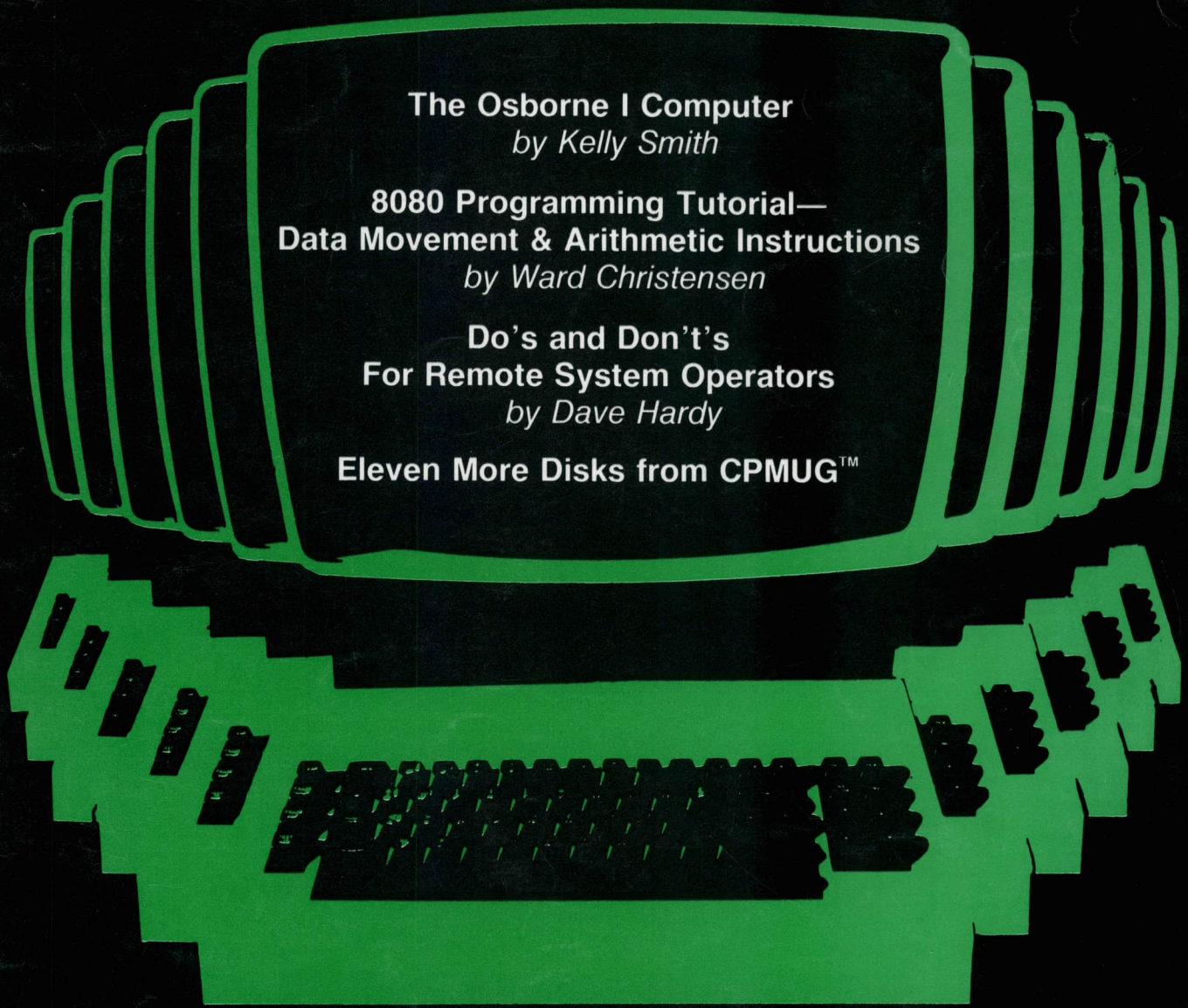
LIFELINES®

THE SOFTWARE MAGAZINE™

$2.50 December 1981 Volume II, No. 7 (ISSN 0279-2575, USPS 597-830)



The Osborne I Computer
by Kelly Smith

8080 Programming Tutorial—
Data Movement & Arithmetic Instructions
by Ward Christensen

Do's and Don't's
For Remote System Operators
by Dave Hardy

Eleven More Disks from CPMUG™

T/Maker II:™
it does a number on VisiCalc™!

T/Maker II does more. And does it on *your* computer. VisiCalc is a fine aid for the computation of numerical problems. But it does have two major limitations: it is available only for a small number of systems, and its use is limited strictly to numbers, not words. To overcome these substantial limitations, Lifeboat Associates introduces T/Maker II.

Unlike VisiCalc, T/Maker II is designed to run on most business computers with SB-80™ or other CP/M-80® compatible operating systems.

Available now for UNIX™. And soon there will be T/Maker II versions for RT-11™ and other systems as well.

Works with words as well as numbers. Like VisiCalc, T/Maker II reduces the manual tasks involved in computing and calculating financial documents. But since most business problems and reports involve words as well as numbers, T/Maker II also functions as a full-screen text editor for word processing.

T/Maker II is the most advanced aid for the analysis and presentation of numerical data and text material. In a matter of minutes, an entire document — including all edited text, all figures and all calculations — is created, reviewed on your screen and reported in printed form.

T/Maker II turns your small business computer into a powerful, sophisticated and convenient tool. A tool that will save you money, time and energy, and eliminate the need for costly time-sharing.

With T-Maker II you can easily perform an unlimited number of analytical and reporting tasks which integrate numerical and text processing. You'll find T/Maker II perfect for such things as:

- Financial Statements
- Statistics
- Profitability Reports
- Revenue and Expense Analyses
- Portfolio Evaluations
- Price Lists
- Rate Structures
- Expense Accounts
- Cash Flow Projections
- Checking Account Reconciliations

. . . and much, much more.

Easy to learn and use. You don't have to be a programmer to operate T/Maker II. Just follow T/Maker II's easily understood and ordered instructions, set up your data in rows and columns, define the relationships and T/Maker II will do the rest: it will perform the computations and formatting necessary to prepare your document. When you're finished you can analyze your report on your screen or store it on a diskette. Or, you can have the report printed with presentation quality.

And when any changes have to be made, simply enter the new figure or relationship and tell T/Maker II to adjust and recalculate all the new results.

Editing capabilities. As a full-screen editor for word processing, T/Maker II handles text up to 255 characters wide. It includes features like text formatting and justification, centered titles, a text buffer for block moves and repeated inserts, global search and replace commands for printing your letters, reports and documents. Wide documents are supported by horizontal scrolling.

Low cost. The cost of T/Maker II is miniscule when you consider all the time, energy and money it saves. T/Maker II is brought to you exclusively and supported completely by Lifeboat Associates, world's largest computer software publisher. For more information send us the coupon below.

Mail coupon to: Lifeboat Associates,
1651 Third Ave., NY, NY 10028. Or call (212) 860-0300.

☐ Please send me more information on T/Maker II.

☐ Please send me a free Lifeboat Catalog featuring over 200 programs.

Dealer, Distributor and OEM inquiries invited.

Name _____

Title _____

Company _____

Street _____

City _____

State _____ Zip _____

Software With Full Support

T/Maker II is a TM of P. Roizen.
SB-80 is a TM of Lifeboat Associates.
CP/M 80 is a reg. TM of Digital Research, Inc.
VisiCalc is a TM of Personal Software, Inc.
UNIX is a TM of Bell Laboratories.
RT-11 is a TM of Digital Equipment Corp.

As an example of what T/Maker II can do, see the chart below. The operator entered only the data shown in boldface. T/Maker II calculated and reported all the other values.

	— Actual —			Growth		Total	—Projected—		
	1978	1979	1980	Rate	Average	(000's)	1981	1982 *	1985
Item A	**42,323**	**51,891**	**65,123**	24.04	53,112	159.34	80,782	100,206	191,262
Item B	**45,671**	**46,128**	**49,088**	3.67	46,962	140.89	50,891	52,761	58,791
Total	87,994	98,019	114,211	13.93	100,075	300.22	131,673	152,966	250,053
% Item	48.10	52.94	57.02	8.88	52.69	158.1	61.35	65.51	76.49
% Item	51.90	47.06	42.98	-9.00	47.31	141.9	38.65	34.49	23.51
Total	100.00	100.00	100.00	—	100.00	300.0	100.00	100.00	100.00

***Two intervening years not shown.**

LIFEBOAT WORLDWIDE offers you the world's largest library of software. Contact your nearest dealer or Lifeboat.

Lifeboat Associates
1651 Third Ave.
New York, N.Y. 10028
Tel: (212) 860-0300
Telex: 640693 (LBSOFT NYK)
TWX: 710-581-2524 (LBSOFT NYK)

Lifeboat Inc.
OK Bldg., 5F
1-2-8, Shiba-Daimon
Minato-ku, Tokyo 105, Japan
Tel: 03-437-3901
Telex: 2423296 (LBJTYO)

Lifeboat Associates, Ltd.
PO Box 125
London WC2H 9LU, England
Tel: 01-836-9028
Telex: 893709 (LBSOFTG)

Lifeboat Associates, GmbH
Hinterbergstrasse
Postfach 251
6330 Cham, Switzerland
Tel: 042 36 8686
Telex: 865265 (MICO CH)

Intersoft GmbH
Schlossgartenweg 5
D-8045 Ismaning, W. Germany
Tel: 089-966-444
Telex: 5213643 (ISOFD)

Lifeboat Associates, SARL
10, Grande Rue Charles de Gaulle
92600 Asnieres, France
Tel: 1-733-08-04
Telex: 250303 (PUBLIC X PARIS)

Lifeboat Associates
World's foremost software source

LIFELINES®

December 1981

Volume II, No. 7(ISSN 0279-2575 USPS 597-830)

Contents

Opinion

Features

The CP/M™ Users Group

Software Notes

Product Status Reports

Miscellaneous

Lifelines (ISSN 0279-2575, USPS 597-830) is published monthly at a subscription price of $18 for twelve issues, when destined for the U.S., Canada, or Mexico, $40 when destined for any other country. Second-class postage paid at New York, New York. POSTMASTER, please send changes of address to Lifelines Publishing Corporation, 1651 Third Ave., New York, N.Y. 10028.

Editorial Comments

Microcomputers - The Domain of Giants

The microcomputer industry is unique in a great many respects and it seems to spawn an endless variety of innovations at an ever increasing rate. Companies such as Microsoft, Digital Research, Peachtree, Lifeboat Associates, etc., all began as entrepreneurial enterprises; each has played a very major role in the most rapidly advancing industry in the electronics field. This is evidenced, in part, by the large number of major corporations in the computer field which are profoundly influenced by the directions such relatively small companies have taken.

Equally important have been the contributions of individuals such as Kenneth Bowles. In 1974, Dr. Bowles was engaged in teaching an introductory programming course at the University of California at San Diego. In looking for a good approach to teaching languages, he hit upon the idea of using a language relatively unknown in the United States, called Pascal. As you recall, Pascal was developed by Niklaus Wirth in 1965.

Interestingly enough, Wirth also had been concerned about the importance of the student's proper introduction to his first language. This is reflected in his observation that the first language which a student encounters "profoundly influences his habits of thought and invention and that the disorder governing these languages directly imposes itself onto the programming style of the students". Unfortunately Wirth's initial attempt was something of an underwhelming success. The first Pascal was written in FORTRAN; it was regarded as a dismal failure until it was replaced by a second Pascal compiler - this time written in Pascal itself.

To the uninitiated the fact that Pascal could be written in itself seems to be a kind of Catch-22, since it's not immediately obvious how it comes into existence. Nonetheless the efforts of Niklaus Wirth culminated in 1974 in an extremely interesting publication entitled "Pascal User Manual and Report". This was subsequently revised and became what is now commonly referred to as "The Second Report"(1978). The interested reader will find this book a classic example, even if not read in its entirety, of how one properly defines a language.

As you recall 1974 was the eve of the microcomputer era and it is this fact, plus Dr. Bowles' interest in pedagogical tools, which resulted in a major contribution to microcomputer languages.

Bowles was clever enough to recognize that the key to rapid and widespread use of Pascal was its speedy implementation on a large number of machines. But this meant that a method must be found for adapting it easily to a wide variety of microcomputer architectures. The solution was to create a Pascal pseudo-compiler which produced code for a virtual machine. Thus the output of the compiler was a pseudo-code called P-code; the virtual machine was the P-machine. Since P-code interpreters could be relatively easily produced for different micros, this provided the mechanism which allowed a large number of widely differing hardware architectures to support the same compiler. So the task was reduced to one of generating the P-code interpreters for each architecture. The details are not important to this discussion. Most significant is the fact that a large number of microcomputers soon supported Pascal. This particular approach resulted in what is now referred to as UCSD Pascal.

Soon committees were hard at work on an officially sanctioned standard for Pascal. These proceedings are extremely interesting to observe, and should the reader have the opportunity to attend, the experience will be worthwhile.

In 1979, Softech entered into an agreement with the Regents of the University of California at San Diego to market UCSD Pascal on a worldwide basis. While the original system developed by Bowles and his associates supported only UCSD Pascal, Softech soon introduced BASIC and FORTRAN. This system, named the "P-System" by Softech, has evolved rapidly into a complete program development system; facilities include text editors, program libraries and file management utilities, as well as compilers and assemblers.

But perhaps the most interesting aspect of Pascal has been the portability of applications written in this language. Since a large number of machines support Pascal and new hardware architectures can be quickly provided with P-code interpreters, applications written for one generation of machines are quickly transported to later generation machines as well as across machines of the same generation.

It should be noted that the p-system was included in the environments provided for the IBM Personal Computer. Presumably its inclusion was specifically for the reasons I have discussed.

Thus applications written in Pascal for 8 bit microcomputers will quickly find their way into the 8086 environment.

The full ramifications of the contributions of Dr. Kenneth Bowles, his staff, students and many followers will probably not be fully known or understood for some time, but there can be no doubt that they have made an invaluable contribution to microcomputers. His many books and articles have fostered a whole new generation of applications programs, utilities and programmers. The intensity with which the various standards committees have pursued the details of the Pascal language definition are to a large extent directly attributable to the widespread acceptance and use that UCSD Pascal has enjoyed.

As with many pioneers, Dr. Bowles, having planted a seed which has not yet reached full fruition, has supplemented his activities in the area of Pascal with a concentrated effort to support yet another major software effort, ADA.

One wonders what he has up his sleeve this time. Meanwhile Softech carries the baton for Pascal. One would have to believe that Niklaus is pleased, and maybe, just maybe somewhere Pascal is watching these events with a twinkle in his eye.

Edward H. Currie

The Pipeline

Carl Warren

Goodies and things of interest

Since the introduction of the Osborne I, a number of companies have become interested in small. Specifically, Sony has the Typecorder designed for the executive who likes to type on airplanes, and Novation has come up with the Infone. This latter device is quite nice and sports more features than Dolly Parton.

But the real news is another desktop system similar in concept to the Osborne machine, but oh so much better in implementation. This new unit, dubbed the Attache, is from Otrona Corp, 2500 Central Ave, Boulder, CO 80301 (303) 444-2274.

This classy little system weighs in at under 20-lbs, fits in a half cubic foot and offers the following features:

- A Z-80A processor
- A 5-in. CRT, that supports an 80x24 display plus raster style dot graphics
- Two 180K byte drives
- A full-sized flip down keyboard
- 64K bytes of RAM
- A direct memory processor to relieve the main processor from I/O duties
- Two multi-protocol ports
- CP/M, WordStar, BASIC-80, UCSD Pascal, Valet an interrupt manager, and Charton a plotting software package.

If all of that isn't enough, the Otrona folks have included a clock/calendar, and a sound synthesizer. The unit looks very much like something from Hewlett-Packard, only it's cream colored rather than brown. But its designers all have HP backgrounds and proved it with this design.

Should you want to pack it around with you, Otrona offers a DC power option, and a battery and charger option, plus an accessory pouch for all the extra goodies.

Yep, as with the Osborne machine you can attach an optional full-sized monitor, and an Epson MX-80 printer.

If you're thinking that's a lot of stuff to come in a small box you're right. What's more important, however, is that the box is designed right and has upgrade built in. But don't expect to pick this machine up for cheap; it carries a $3750 price tag to start.

Those of you who like to attend the Consumer Electronics Show, held in Las Vegas in January and Chicago in June, may be aware of the noises being made over the exhibitors who offer pornographic video tapes. According to a few local sources more than the video makers may have something

(continued next page)

to scream about come this January.

Apparently, porno venders have been casting an evil eye towards personal computers with graphics capabilities, and reportedly this year will be offering some very, very graphic software in the form of tapes, disks, and ROM cartridges.

Although no one is really sure what the software will be like, bets are on that it may cause a renewed interest in prospective buyers who needed a reason to pick up a machine.

Rumor had it that the big Z himself may have had a hand in the development of this latest software, but when contacted, Zoso explained that to him SEX was nothing more than a number system with a radix of 2 designed for numerical control of parallel processing.

Should your interests be more to the pure demands of personal computing, you might do well to take another look towards the good folks at Microsoft. Either following or starting a new trend, the Bellevue WA gurus have come up with a program called TASC that will take a source code written in Applesoft BASIC programs and compile it into machine code. The new tool not only compiles the code, thus speeding up execution, but uses a compression scheme to eliminate size restrictions usually found with compilers.

Since many of you are always looking for a method of enhancing your hard copy output, you might want to consider looking at Okidata's Microline 83A dot-matrix printer. Plan on spending about $1200 for the system then another $100 for the graphics option that gives all points addressable graphics with a resolution of 60 x 66 dots per inch. Interestingly, you can hook this printer up to your Apple and use that grand Visiplot/Visitrend package to create some very nice hard copy graphics without worrying about writing new drivers. If you're interested in this product give Lex Pietraszkeiwicz a call at (609) 235-2600, or drop into your local computer store.

Here's something to think about. This past September, Wayne Green mentioned in his newsletter that someday you'd be able to squirt data across the line very quickly by using compression techniques thus saving phone time. Well guess what? You can already do it if you're inventive enough — cuz the tool already exists.

Green's idea was to have dictionaries that compressed words then uncompressed them on the other end. Good idea, huh? Course the problem of how to do it exists. What Green's idea implies is that you not only have this neat dictionary system, but a very powerful database manager to handle it. Well it turns out that if you're using Ashton-Tate's dBase II, you have all the tools you need.

First, using the Application Design Language (ADL), you can quickly develop a communication package with database handling attributes. This is possible, because you have the use of memory variables and an undocumented CALL function. Moreover, since dBase only goes up to A400 you can put the necessary modem drivers right at the top without disturbing any of the open program area.

Now since you have all that done, and can call any routine you want, add COMMAND files for the data handling and searching a dictionary. Yep, you can even build the dictionary by using ADL and employing some of the techniques Ward Christensen has used in his squeezing programs. There you got it, and probably something that will sell well.

Of course maybe Jane can be talked into running a contest to see who can come up with the best implementation and offer a prize of a T-shirt with Zoso's picture on it, which from what I have heard may be available soon for a really large amount of money.

Speaking of selling, this is something I've wanted to do for a long time. Many of you may not realize it but a lot of the really elegant software has come into your hands from Ward Christensen, and notice even though he charges $50 for the CBBS package it's probably priced $400 too low.

Ward has been sharing his expertise, which is awesome, for nothing but the sheer fun of it. If you've had the chance to peruse the code he writes then you know what a grand coder he is. Since this is the end of the year, I've decided that Ward is to be the first recipient of the Wilson T. Meyer award of excellence.

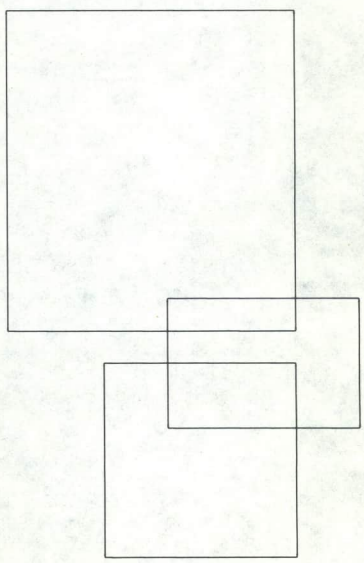
In case you don't know who Wilson T. Meyer is, he's a small pink bear that sits on a bookshelf in my office as a reminder that good things are sometimes pink.

Starting next year, I'll be giving out the Wilson T. Meyer award every three months to the individual or company that is the epitome of excellence or pink. If you have a candidate in mind drop a note to the Pipeline in care of the magazine.

In the meantime have happy holidays.

Renew

Did your subscription begin with the January issue? Then you had better heed the letters you've received about renewing your subscription. There's a lot coming up in the next year and if you don't call or write *immediately* you'll be missing some important information and some exciting software reviews. So start your new year with a new subscription.



Zoso

Dear Readers: Under better circumstances this month's column would have been chock full of the usual goodies, but we're going to have to forget them for the time being. The kind of mail which must get answered has been piling up again.

You may recall that two issues ago I volunteered a few unflattering comments about Supersoft's Disk Doctor and that Mr. Holland, the author of that program, replied angrily in the last issue. I will deal with Mr. Holland's letter first.

Dear Mr. Holland:

You mentioned that I would have received a free update if I had mailed in my registration card. Actually, someone else would have (and did). I am coming to realize that Supersoft, though not yet perfect, is getting better and could teach some other vendors a good deal about concerned and friendly product support. However to help an improving act improve even faster, I suggest that any policies which bind product support to registration cards should be seriously reevaluated. My experience with the Postal Service suggests that dropping things in a mailbox provides only marginally better odds than betting on the ponies. Really now, if a customer's check or credit card number clears, and if you ship them a package (via UPS!) which is received and signed for, why is any further paperwork necessary? I feel quite strongly about this because I have sent countless registration forms to various companies to no discernible purpose whatever. As far I've been able to tell, product registration cards, birthday cards, condolence cards, Sweet Sixteen cards; they're all about the same when it comes to qualifying for after-sale support.

Last but by no means least, I can't get used to the spelling 'discette'. Oh sure, there's no crime involved in spelling words any way you please, but to my ear 'diskette' sounds just like those things people shove into their computers and 'discette' sounds like something that Marie Antoinette and her pals would have tossed around the old courtyard in a giddy moment. Actually, now that I think of it, your preferred spelling sounds like lots more fun!

Regards,

Z.

Jim Tyson of Washington D.C. sent a very nice letter in which he asked some questions about 5 ¼" disk formats. To summarize:

Q: Are the differences in 5¼" formats due to hardware?
Z: Usually, but little is certain in the 'Disk dollhouse'.

Q: Are there any 5¼" standards?
Z: Not really. In fact there are insurmountable incompatibilities within different releases of the same manufacturer's operating systems. Intertec and Apple come immediately to mind.

Q: Will there be any standards?
Z: Almost assuredly not. With any luck at all, the technology will become obsolete before meaningful standards can be established. Newcomers to microcomputing may not remember the 'Kansas City Standard' (a truly wretched system, proposed a few years ago, by which hobbyists could swap pathetic, teeny programs stored on audio cassette). Guess what Zoso thinks will follow audio cassettes to the 'Binosaur' museum.

Q: Is any one format common to more than one machine?
Z: Yes, but only rarely. Lifeboat Associates offers more 5¼" formats than anyone else I know of. Examine a Lifeboat product list or catalog, and you will be able to see the few instances of cross-compatibility.

Q: Can a microcomputer using one format be programmed to write on a disk with another format?
Z: Yes, in some cases, but it is never easy! The time required to code most of these things is worth more than the cost of 8" drives attached to a second computer with whatever software will allow CP/M media limitations to be ignored. (BSTAM is one fine product which does this and more!)

Q: Is an upgrade to the operating system required so that a system using single sided single density disks could use double sided double density disks?
Z: Assuming the hardware can even support the change (and it's best if you don't assume this), the operating system will usually have to be rewritten.

I have stated on more than one occasion that I have no use whatever for 5¼" minifloppies. I still think that 'disklets' (my name for the nasty little playthings) and the matching toy drives are lamentable imitations of the real item. I know a lot of you disagree with me; Less power to you (literally)!! What else can I say?

Mr. Tyson, I honestly did like your letter, so if you'll just send me your shirt size and color preference c/o Lifelines, an official Zoso T shirt will be on its way to you.

The next letter arrived courtesy (I think) of a gentleman in Lexington, Kentucky. He seems to think that I bludgeoned the mother tongue in an earlier column by having confused sociologists with social workers. His letter says in part:

"...I have not yet heard of a Sociologist having 'proteges', nor does the 'do-gooder' epithet fit the Sociologist well. I am certain that you mean to derogate the ever unpopular Social Worker and the nasty work he/she does, rather than the philosophical, fact-gathering, objective systems-oriented Sociologist. Sociologists are people like Durkheim, who reportedly was the first to use statistics to measure a society-wide phenomenon (Suicide); Comte, a rather mystical Frenchman who thought Sociology

(continued next page)

was the all-embracing Science of Sciences; and Talcott Parsons with his Structural-Functional analysis approach to social systems. These men have far more in common with Anthropologists than they do with Social Workers, and anyone not stupefied by his own prejudices and or educational indifference can easily perceive the difference."

"...but I must speak out when YOU, of all people blather on erroneously ...even when it is probably the Social Workers who are keeping the Great Unwashed from murdering us in our beds and taking away all our food and our cars and houses and children and computers."

"I always enjoy what you have to say; your wit and verve. However, you must not make any mistakes, or I will write to you again..."

WOW! Here we go:

Dear Sir,
I don't know where to start. For a guy from Kentucky you sure know a lot. Do you know for example what 'Raggler' is? It's what a Kentuckian puts in his gas tank, as in "Gimme tanka raggler". Just kidding, seriously.

I guess you told me, but I'm not sure what it was that you told me. My dictionary defines the word sociologist (which I choose not to capitalize) consistently with the context in which I used it. When I discussed sociologists, I was not referring to your pantheon of inconsequential pedagogues (of whom, till now, I was most mercifully unaware). I was referring to some of the modern breed, those who always bring up poverty and cultural deprivation as a stock justification for the behavior of many of those responsible for the wholesale stealing, maiming and killing which goes on right now. While on the topic, I don't rely on social workers to spare me the inconvenience of being murdered in my bed. Good door locks, a strategically located 12 gauge and two ill humored German Shepherds keep that necessity of [my continuing] life covered. The misguided social engineers, whom you so staunchly defend, by whatever name you wish to call them, are largely to blame for this. If I live another eighty years, I would never trust one of these speaking ostriches enough to let him/her help me cross the street safely.

As a technical nuance, I abhor the term 'Great Unwashed' and all the disdain and arrogance such words convey. When I have heard others refer to the 'Great Unwashed', they were always discussing citizens of Third World countries. If they were right (I don't know and I really don't care), the 'Great Unwashed' refers to people existing at or near starvation level (who are most unlikely to murder Americans who are at home, asleep in their beds).

I'm sorry you feel that I blather away erroneously, but what can I tell you; as I write this, it's becoming evident that if I want to remain a legend in the erroneous blather department I'd best train hard to stay ahead of some awesomely talented newcomers.

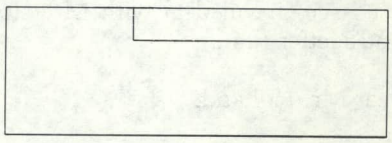
Almost too late to matter, you paid me a compliment. Thanks for that much.

Finally, you warned me to watch out lest I commit another faux-pas (as you define them), thus motivating you to write again. I'll try my level best to avoid this, but should I fail and incur further correspondence, PLEASE use a new ribbon. These are the only eyes I have!

So much for the letters. I'm sorry it took so long. Next time we meet, we'll have another contest, the best of what I missed this time and some new stuff too!

Happy Holidays,

Zoso



Holidays

The holiday season is fast upon us. You should consider gift subscriptions to *Lifelines/The Software Magazine* for your friends and relatives who are involved in microcomputing. As you probably realize from your own experience, the price of a subscription is small for the money *Lifelines* can save you in a year. Just send a check or credit card number and fill out the form below*. (Or call [212] 722-1700.) We'll send your gifted one a note to let them know of their good fortune, *and* we'll send you a free Zoso T-shirt. (Don't forget to tell us your size.)

Your name and address:

Name_____
Address_____
City _____ State_____ Zip _____

Shirt size _____
☐ Check enclosed
VISA or MasterCard Number
Expiration Date

_____

_____
Signature (if payment is by credit card)

The name and address of the gifted one:

Name_____
Address_____
City _____ State_____ Zip _____

*All orders must be prepaid by VISA, MasterCard or check. Checks must be in U.S. $, drawn on a U.S. bank. Subscription rates are $18 for twelve issues (one year) when the destination is the U.S., Canada, or Mexico. For subscriptions going to all other countries, the price is $40 for twelve issues.

A Subroutine to Check for Stack Overflow

Kelly Smith

Here is a simple subroutine to be CALLed in your applications program to check for a possible "stack overflow" condition. This subroutine might be especially helpful during the "debug stage" of your software development, when you may not be sure of your total stack requirements. You could make CALLs to "check$stack" from numerous places in your software as a monitor of stack allocation, and by using conditional assemblies, REMOVE all the CALLs when your debug is completed. Other applications, include stack-oriented languages such as STOIC, FORTH or PASCAL where some heavily "compute bound" applications programs could eventually "gobble up" memory and clobber the operating system.

I wrote a simple test program which you can use to verify the operation of "check$stack"; the exit on "blow up", resets the stack pointer to the "old stack" pointer, then displays 'Stack Over-Flow, Depth = nn' (where nn equals the "stack depth" in hexadecimal for up to 256 stack "levels"). Remember that the stack works *down* towards (typically) your applications program. Leave sufficient code between the "stack$end" and "oldstk" so (at worst) you can exit on stack overflow "gracefully". You clobber "oldstk", and all bets are off.

```
;
; This is the test for "check$stack"...with 'debug' false,
; the program will exit to CP/M with the stack overflow
;  message.
;
true    equ     -1      ; define true
false   equ     not true; define false
debug   equ     false   ; define debug (if true, makes sufficient stack)

        org     100h

        lxi     h,0     ; save "old" CP/M stack pointer
        dad     sp
        shld    oldstk
        lxi     sp,stack; set "local" stack
        call    test
        lhld    oldstk
        sphl
        ret
;
test:   call    test1
        call    check$stack
test1:  call    test2
        call    check$stack
test2:  call    test3
        call    check$stack
        ret
test3:  ret
;
; end of test on "check$stack", incorporate the following code
; for your particular application...with a little more thought,
; you could also display the address of the last CALL prior to
; the "stack overflow",  and thereby let your program tell you
;  WHERE it BLEW UP...I will leave that exercise for you.
;
check$stack:            ; check to see if stack pointer is below STACK$END
;
        push    h       ; save H$L Regs.
        lxi     h,-stack$end    ; won't work for STACK$END = 0000
        dad     sp
        pop     h       ; restore H$L Regs.
        rc              ; return if still o.k.
;
```

(continued next page)

```
;  come here on stack over-flow
;
        lxi     h,0     ; clear H&L Regs.
        dad     sp      ; stack pointer to H&L Regs.
        lxi     d,stack ; get original stack top address
        mov     a,e     ; do 16 bit subtract, to calculate "stack depth"
        sub     l
        mov     e,a
        mov     a,d
        sbb     h
        mov     d,a     ; 16 bit result in D&E Regs.
        lhld    oldstk  ; restore "old" CP/M stack pointer
        sphl
        push    d       ; save "stack depth"
        lxi     d,stack$overflow$message
        mvi     c,9     ; display string function
        call    5       ; let CP/M do the work...
        pop     d       ; recover "stack depth"
;
display$stack$depth:    ; display up to 256 deep stack digits
;
        mvi     c,2     ; display 2 digits
        push b ! push d ; save 2 digit count and "stack depth"
        mov     a,e     ; get hexadecimal digit
        rar ! rar ! rar ! rar   ; display high nibble first...
;
hexascii:               ; convert 1 digit hexadecimal to 1 digit ASCII
;
        ani     0fh     ; mask for low nibble position
        adi     90h     ; convert hex digit to ASCII digit
        daa
        aci     40h
        daa
        mov     e,a     ; pass to CP/M in E reg.
        mvi     c,2     ; display character function
        call    5       ; let CP/M do the work...
        pop d ! pop b   ; get "stack depth" and digit count
        dcr     c       ; debump digit count
        rz              ; return to CP/M, if both digits displayed
        push b ! push d ; not yet, so display second digit
        mov     a,e     ; get hexadecimal digit
        jmp     hexascii; display and exit to CP/M, next time thru...
;
stack$overflow$message: ; indicate stack overflow
;
        db      0dh,0ah,'Stack Over-Flow, Depth = $'
;
oldstk: ds      2       ; storage for "old" CP/M stack pointer
;
        ds      32      ; "dummy" program storage
;
stack$end       equ     $       ; stack end

        if      debug   ; if debug,
        ds      32      ; 16 level stack
        else            ; else,
        ds      2       ; 1 level stack
        endif

stack   equ     $       ; stack starts here
;
        end
```

After the Game – Part 2

Stephen Walton

GAME OVER

CHOOSE:

REVIEW

REPLAY

NEW PLAY

OTHER GAME

DEBRIEF ON SYSTEM

DEBRIEF LIVE

OTHER

Harris was in a large reclining chair, something like a La-Z-Boy or the kind sometimes used for blood donors. The words hung in front of him, on no visible screen. They weren't English, and the characters were unfamiliar, but he understood them.

The room reminded him of the studiedly spare, Bauhausian living room of a friend's summer place at Westhampton Beach. It was dark outside, but the sound of winter surf beyond a wall of plate glass was exactly right. The curious thing about the room was that it had no doors.

He reread the message suspended before him. "Debrief live," he said in whatever language it was, and the invisible screen vanished.

The woman walked through a wall. "Hello, darling," she said in the not-English. "Do you know me?"

She was naked, and better looking than any woman of whom he had intimate memories. "I can't say that I do. Something strange has happened to me."

She smiled. "The persona and its illusions cling for a while." She came to the chair and took his hand. "You'll be fine."

Harris noticed that he was naked too, and was beginning to become aroused. He felt his face flush. He squirmed in the chair and it moved so that he sat upright. "What *is* all this? I *died*."

"In the Game, dear." She released his hand, retreated to sit in a chair facing his.

"Are you here to — debrief me?"

She nodded. "As I've done many times before. As you've done for me, many other times."

"This is something customary for us?"

"Oh, of course. If the Game's been any good, at least."

"Then maybe you'd better get on with it." Harris found cigarettes and a tall drink on a table beside his chair; used them.

"Certainly, darling. You've just finished playing out a segment of a large, complex simulation." There was something pedantic about her tone — pedantic, but touched with dry amusement. "The simulation is that of an entire, if limited, world, with a complete internal history. You were one of its five billion or so inhabitants."

"I was alive," Harris said.

"Yes, dear. I'm sure you thought it was life, at the time. But it was really rather like the game you were playing on your friend's little computer. Yes, I was monitoring."

"You're talking about the interstellar-emperor game. But that. . ."

"The analogy holds. That little game was a simulation within a larger one — which you happened to think was life. But it *was* a simulation, a virtual universe."

"That's impossible!" Harris said. "I was *there* — waking, sleeping, working, screwing, going to the can — for forty-seven years."

The woman nodded briskly. "It took most of the afternoon. And of course you were *involved*. Just as when you played your friend's empire game. Wasn't that just a little like being *there*?"

"Yes, but it was just words and numbers on a screen. It wasn't real."

"It lacked something when compared to the larger life you knew. The analogy is good. The life you're still waking to is as much bigger than what you knew as Robin Harris as that was bigger than the numbers on a screen."

Harris drained his glass. The bodily sensations from the drink were intense, yet he knew he was thinking clearly. "Look, I had a wife and children, colleagues and friends, and a world of acquaintances and strangers. All those people — weren't they real?"

"Of course they were. Real enough, at any rate."

"What does *that* mean?"

The woman had lit a cigarette. She exhaled through her nose in an aggressive way. "Most of them were other players. At any time there are billions of us playing that game. And other billions playing other games."

Harris listened to the booming surf outside. It seemed that he had never heard it so distinctly before; it was distractingly attractive, and somehow terrifying. He shivered once, then spoke softly. "What about the other people? The ones who weren't players?"

A shrug. "Autonomous modules within the simulation. They had free will, and feelings just like yours. The ones that loved you *did* love you, and the ones that hated you hated you. Don't worry about 'real,' darling."

He stared at her, gawked at her beauty, and was glad that it had begun to seem familiar. He reached toward his empty glass and found it full again. He giggled. "You know,

(continued next page)

this really isn't what I'd expected. But I think I'm getting the idea."

She smiled. "Was it good for you?"

"Not bad, really not bad at all. A relatively quiet life, but there was —" He was stopped by a welling-up of pity, not for himself on the train, but for all the others, for their pains and disappointments and griefs, for players and modules together. It was a fullness of feeling the old Harris could never have permitted; now he let it rise and subside of itself, and waited for the tightness to leave his throat before he spoke. "Why," he finally asked, "why do we include so much suffering in a game?"

The woman laughed. "Because a game would be too dull to play, without it. Because it teaches something." She paused as though gauging his ability or willingness to hear what must come next. "And because the suffering in a game is so mild, so trivial, compared to what we have in real life."

Without knowing how, Harris knew she had told the truth. He also knew that his pity had not been wasted, would never be.

The woman rose from her chair and came toward him. "I think you're ready for the rest, now."

He nodded, then stood and kissed her. At that moment, their thoughts touched directly, the room and everything in it changed, and he was home.

Good game, darling?

He passed a tentacle across his high-winter broodmate's crop in a teasing but tender way. *Excellent. Some amazing textures. . . The limited forms of communication scarcely seemed a handicap, from inside.*

She oozed closer to him. *Mmmm. . . I saw that when I monitored. Are you ready to go back to work now?*

Work? He wanted to gromp with her, right here, right now, and he let her know it.

Darling, you've spent most of the day playing that game. She puffed out a pellicle to underscore her exasperation. *Your report to the tharlange is almost due, and we have a seizing to attend.*

He showed her the depth of his regret, lormed an orkel to express formal acceptance of chastisement. *You're right, of course. Let's just claj a few snabbas, and then —"*

* * *

It wasn't death, just a sudden and total cessation. For an unmeasurable period of time — an instant or an eon — it seemed that he existed without a body and in no universe. With equal abruptness, a familiar environment and his customary envelope of flesh returned to him.

Thorvald II, Emperor of the Thousand Suns, was in his cabin aboard the flagship of the Blue Fleet. Attending were his valet, himself a noble, and a young ensign.

Thorvald removed the gaming helm. "Yes?"

"The Ad-Admiral's compliments, Your Imperial Majesty," the ensign said with but the trace of a stutter, "and his apologies for the interruption. We've sighted a Freeper commerce raider, and your standing order was that —"

"Yes. Tell Freddy I'll be with him on the bridge in a moment. Armor?"

"Shouldn't be needed, sir."

"Very good." Thorvald and the ensign exchanged salutes and the young man left.

"Thorve," said the valet, who was also a physician, "any buzz from the game?"

Thorvald stood and stretched his arms over his head. "No. But it really was a nice one. I'll want to finish it later." He shrugged into the uniform tunic his valet held. "We'll probably have to give the designer some service points." *Games,* he thought. *If only real life were so simple!*

WordStar Release 3 Installation for TRS-80 Model II

James K. Korenthal

INTRODUCTION

Hooray for MicroPro! WordStar release 3.00 is now available, and it's terrific! Clearer menus, horizontal screen scrolling, text column manipulation (for multi-column documents), automatic linkage to SpellStar (and, of course, MailMerge), the works! All in all, a release worthy of the high standards that we've come to expect from MicroPro.

Release 3.00's INSTALL program supports many more terminal types than did its predecessors. One newly-supported terminal is the TRS-80 Model II, with various CP/M's. The remainder of this article will deal with special installation considerations for WordStar release 3.00, the TRS-80 Model II, and Lifeboat's CP/M.

Note for beginners: If you have this configuration, but don't know anything about terminal emulators for memory-mapped displays or about WordStar installation in general, you can skip to the section entitled **Installation**.

WHAT'S WRONG NOW?

I've installed WordStar releases 1.xx and 2.xx for a number of clients with Mod II's. They quickly became dissatisfied with the Lear-Siegler ADM-3A emulation that Lifeboat's CP/M provides. Although the emulation is true to the ADM-3A's function, and could easily be specified in WordStar's INSTALL program, three problems occurred when using WordStar in this way:

1. On-screen functions obtainable on the TRS-80 (but not on the ADM-3A) were not supported, reverse video being the most notable.
2. The Mod II arrow keys were not supported.
3. Performance was degraded, due to the second level of terminal emulation. (WordStar kept an internal screen image, passed information to an emulated ADM-3A, and then CP/M kept an additional screen image.

In order to solve these problems, I had to completely bypass INSTALL's terminal specification, and supply all the parameters directly for a memory-mapped display. That was quite a bit of work, and had to be redone each time a new WordStar release arrived.

You can imagine my relief, then, when I saw a selection for the TRS-80 Model II ("%") in the release 3.00 INSTALL program! However, that relief was short-lived. Although MicroPro's installation did solve problems 1 and 3 above, problem 2 still existed, and a new fault showed up: Upon exiting WordStar, the cursor was too large (by Lifeboat CP/M standards), and the CP/M prompt appeared in a "random" (that is, as far as the user is concerned) place on the screen, along with any junk WordStar happened to have on the screen at the time of exit.

HOW DO WE FIX IT?

A few simple patches to WordStar 3.00 will solve all these problems. I've also included a few things which are optional. The patches assume that you've already specified "%" as the terminal type in the INSTALL program.

Patch 1: Restore values for ADM-3A cursor positioning. This is necessary because, even if you've specified a memory-mapped display, WordStar uses the terminal emulator cursor positioning on entry and exit. In addition, SpellStar seems to use the terminal emulator in a few places, thus screwing up the display if you don't include this patch.

Patch 2: Intercept WordStar terminal initialization code. DON'T INSTALL UNLESS YOU'RE ALSO INSTALLING PATCH 5.

Patch 3: Intercept WordStar terminal de-initialization code. DON'T INSTALL UNLESS YOU'RE ALSO INSTALLING PATCH 6.

Patch 4: Eliminate cursor positioning delay and other function delays. This will give a slight performance improvement.

Patch 5: Terminal initialization. First call the initialization routine supplied by INSTALL, thus removing the Model II cursor. Then send a control-z to the CP/M terminal emulator, clearing the screen. This patch is optional, but I find a clear screen on WordStar entry much more pleasing than the scrolling that is normally done.

Patch 6: Terminal de-initialization. Call the de-initialization routine supplied by INSTALL (see Patch 7), thus restoring the cursor. Then clear the screen prior to exit. This one's optional, too. If you don't include it (but you've installed Patch 1), your WordStar screen will remain intact on exit, and you'll get the CP/M prompt on the bottom line. I find that non-technical users get confused when this happens, since they're not sure if they're in WordStar or at the CP/M command level.

Patch 7: Correct terminal de-initialization. This patch supplies the correct value to restore the cursor appropriate for Lifeboat's CP/M.

Patch 8: Change down arrow linkage. The hexadecimal code generated by the Model II's down arrow is 1F, which is interpreted by WordStar as a control-delete. This patch causes WordStar to interpret the down arrow like a con-

(continued next page)

trol-X, going down a line rather than deleting the previous character.

Patch 9: Add left, right, and up arrow linkages. These keys generate hex codes 1C, 1D, and 1E, respectively, codes which don't mean anything special to WordStar. We're adding linkages so these keys will be handled like control-S, control-D, and control-E, respectively. Note: Patches 8 and 9 still allow control-X, -S, -D, and -E to perform their normal functions.

An assembly listing of these patches (assembled using M80, with object code section slightly altered for clarity) follows:

```
                        ;WSPAT300.MAC - WordStar release 3.00 patches
                        ;Copyright (C) 1981  JEKCU, Inc.
                        ;                    c/o James E. Korenthal
                        ;                    175 West 12th Street
                        ;                    New York, N.Y. 10011
                        ;                    (212) 691-1459

                        ;For WordStar release 3.00 with TRS-80 Model II running under
                        ;Lifeboat CP/M, apply these patches after specifying terminal
                        ;type "%" in the WordStar INSTALL program.

                                .z80

                        ctlz    equ     x'1a'           ;control-z
                        esc     equ     x'1b'           ;escape
                        outch   equ     x'0106'         ;wordstar character output
                        clead1  equ     x'024a'         ;cursor positioning lead-in
                        linoff  equ     x'025e'         ;offset to add to line
                        inisub  equ     x'02a4'         ;terminal initialization
                        unisub  equ     x'02a7'         ;terminal de-initialization
                        delcus  equ     x'02ae'         ;delay after cursor set
                        morpat  equ     x'02e0'         ;wordstar user area
                        xtab    equ     x'0649'         ;editing command expansion
                        wsini   equ     x'336a'         ;old initialization routine
                        wsuni   equ     x'336d'         ;old de-initialization routine
                        wsuni2  equ     x'336e'         ;contains "cursor on" constant
                        crb     equ     x'6365'         ;"cursor back" routine
                        crf     equ     x'635b'         ;"cursor forward" routine
                        cru     equ     x'643e'         ;"cursor up" routine
                        crd     equ     x'6424'         ;"cursor down" routine

                                aseg

                        ;patch 1 - restore values for ADM-3A cursor positioning

                                org     clead1          ;lead-in is escape,"="
024A  02 1B 3D                  db      2,esc,'='

                                org     linoff          ;use space for both offsets
025E  20 20                     db      '  '

                        ;patch 2 - intercept wordstar terminal initialization code

                                org     inisub
02A4  C3 E0 02                  jp      jekini          ;point to our code

                        ;patch 3 - intercept wordstar terminal de-initialization code

                                org     unisub
02A7  C3 E9 02                  jp      jekuni          ;point to our code
```

```
                    ;patch 4 - eliminate cursor and function delays

                            org     delcus

02AE  00 00                 db      0,0             ;delay for 0 Msec

                            org     morpat          ;(user subroutine area)

                    ;patch 5 - terminal initialization

02E0  CD 6A 33      jekini: call    wsini           ;remove cursor
02E3  3E 1A         jekcom: ld      a,ctlz          ;clear screen
02E5  CD 06 01              call    outch
02E8  C9                    ret

                    ;patch 6 - terminal de-initialization

02E9  CD 6D 33      jekuni: call    wsuni           ;restore cursor
02EC  18 F5                 jr      jekcom          ;clear screen

                    ;patch 7 - correct wordstar terminal de-initialization

                            org     wsuni2          ;for lifeboat cp/m, cursor is
336E  68                    db      3*32+8          ; slow blink, start at line 8

                    ;patch 8 - change down arrow linkage

                            org     x'052f'         ;(no wordstar label)
052F  24 64                 dw      crd             ; (cursor down)

                    ;patch 9 - add left, right, and up arrow linkages

                            org     xtab
0649  1C 00                 db      x'1c',0         ;left arrow
064B  65 63                 dw      crb             ; (cursor back)
064D  1D 00                 db      x'1d',0         ;right arrow
064F  5B 63                 dw      crf             ; (cursor forward)
0651  1E 00                 db      x'1e',0         ;up arrow
0653  3E 64                 dw      cru             ; (cursor up)

                            end
```

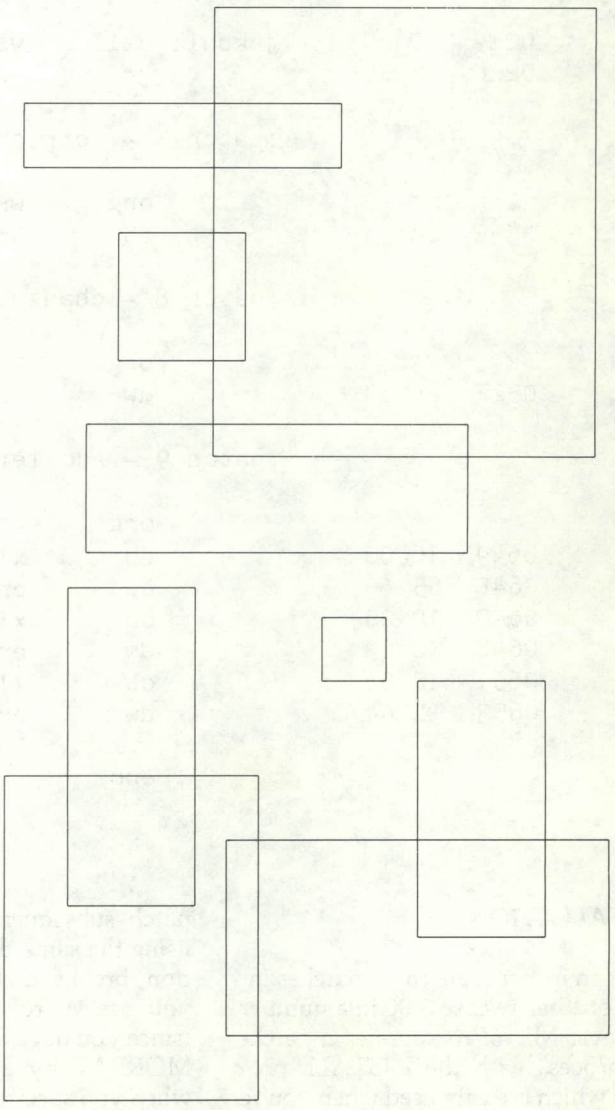
INSTALLATION

You can incorporate these patches into WordStar release 3.00 in a number of ways. MicroPro supplies a patching process with the INSTALL program which is easily used when you're making spot changes. However, as the number of patches grows, I find this method cumbersome. Admittedly, the INSTALL program allows you to specify some locations using WordStar labels, thus allowing you to patch subsequent WordStar releases using the same dialogue. This protection breaks down, however, when you use WordStar's MORPAT area (since you have to specify a linkage to MORPAT by absolute address), or when you specify an address not symbolically available in the INSTALL program. Since our patches both use MORPAT and specify absolute addresses, we might as well get them all in with absolute addresses, using DDT, SID, BUG, RAID, or any debugger you happen to have (or ZAP, if you're rich enough to afford it).

First of all, I'm assuming you've done a first-time installation of WordStar, using the INSTALL program and specifying "%" as your terminal type. (**Note:** If you're using CP/M Version 2.25a, answer "N" to the question "Does your CP/M leave the video board enabled?") That'll give you a file called WS.COM, which you can then modify using DDT, for example, as follows:

(continued next page)

```
A>stat ws.com

 Recs  Bytes  Ext Acc                <= 124 records, each 128 bytes,
  124    16k    1 R/W A:WS.COM          tells us that we must save
Bytes Remaining On A: nnnk             back 124/2 = 62 256-byte
                                       pages when we're finished
                                       applying our patches.

A>ddt ws.com
DDT VERS 2.2
NEXT  PC                  <= verify page count by noting that
3F00 0100                    x'3f' - 1 = x'3e', or decimal 62.
-s24a                     <= patch 1
024A 00 2
024B 1B
024C 3D .
-s25e
025E 00 20
025F 00 20
0260 00 .
-s2a4                     <= patch 2
02A4 C3
02A5 6A e0
02A6 33 2
02A7 C3                   <= patch 3
02A8 6D e9
02A9 33 2
02AA FF .
-s2ae                     <= patch 4
02AE 0A 0
02AF 05 0
02B0 FF .
-s2e0                     <= patch 5
02E0 00 cd
02E1 00 6a
02E2 00 33
02E3 00 3e
02E4 00 1a
02E5 00 cd
02E6 00 6
02E7 00 1
02E8 00 c9
02E9 00 cd                <= patch 6
02EA 00 6d
02EB 00 33
02EC 00 18
02ED 00 f5
02EE 00 .
-s336e                    <= patch 7
336E 65 68
336F 01 .
-s52f                     <= patch 8
052F C3 24
0530 67 64
0531 07 .
-s649                     <= patch 9
```



```
wordstar table 2


0649 00 1c
064A 00
064B 00 65
064C 00 63
064D 00 1d
064E 00
064F 00 5b
0650 00 63
0651 00 1e
0652 00
0653 00 3e
0654 00 64
0655 00 .
-g0

A>save 62 ws.com     <= save back newly-patched WordStar, or
                        play it safe with A>save 62 ws2.com
                        and testing ws2 before erasing ws.com.
```

You'll find that the small time investment needed to install these patches will greatly enhance your enjoyment of WordStar release 3.00. Happy word (star) processing!

Hot Off The Press:: I've just switched to CP/M 2.25a and discovered a potential problem with NEC Spinwriter 5510 installation. It seems you can't get away with specifying, "No communications protocol" to WordStar, directing the printer to the CP/M "LST:" device, and telling Lifeboat Associates' CONFIG program to use ETX/ACK for serial port B. If you try this, the printer will hang up periodically. A workable solution seems to be to specify to INSTALL that you want to use ETX/ACK Protocol, and that the printer is connected to the TRS-80 Model II serial port (INSTALL will assume port B). Both of these specifications are easily made via the INSTALL menus.

Jim Korenthal is President of JEKCU, Inc., a new software development firm in New York City. Send correspondence to him in care of Lifelines, *1651 Third Ave., New York, N.Y. 10028.*

Indexing Utilities: FABS and MAGSAM

Steve Patchen

FABS VERSION 2.4
by: COMPUTER CONTROL SYSTEMS
298 21st Terrace S.E.
Largo, FL 33541
Price: about $195

MAGSAM VERSION III & IV
by: MICRO APPLICATIONS GROUP
7300 Caldis Ave.
Van Nuys, CA 90146
Price: about $145 and $295

FABS and MAGSAM are both indexing utilities which are intended to be used along with a programming language in creating application for CP/M systems. FABS occupies 11K of RAM in addition to the other application routines. MAGSAM requires 6K for both CBASIC versions and must be run under at least a 32K system. For a full discussion of indexes see chapter 10 of FUNDAMENTALS OF DATA STRUCTURES by Ellis Horowitz and Sartaj Sahni; published by Computer Science Press, Woodland Hills, Calif. 1976. For the purpose of comparing MAGSAM and FABS I will briefly discuss the differences in the two indexing models used by them and point out the inherent limitations and capabilities of each model. In addition, I will compare details of the two utilities relating to setting them up for use, to interfacing to programming languages and to the functions provided by each utility.

FABS uses what is called a B-Tree index structure while MAGSAM uses a hashed index structure. Basically an index is used to provide a quick access to information stored in a computer. The index uses some scheme of pointing to the information which avoids having to do a sequential search of all the information in storage in order to find what you are looking for. The idea is to look at the minimum number of locations to find the information or to discover that it isn't there. The CP/M directory is an example of an index. It stores information which enables programs to find files stored on the magnetic storage media associated with the computer. These two indexing utilities provide indexing for information stored within data files by providing a means to find individual data records within a data file quickly. The information is usually given a name called a key. This key is a unique identifier for the information. It is stored in a structure which is designed to allocate a location for this key which is somehow related to its name value. Since the key itself is used to determine the value of its location the number of characters in the key frequently has a strong effect upon the resultant index structure. That is, the size of the index file may be dependent upon the length of the key name used or the number of keys which can be stored in an index may be dependent upon the length of the key. A hashed index is a structure which uses an algorithm to convert the key name value to a fixed length physical pointer of a sequential location in the index in which are stored the pointers to the actual information. Since this usually involves reducing the actual number of characters in the key down to some number of a predetermined range there is a possibility of two different keys producing the same resultant physical index location. A hashed index therefore has an overflow location associated with each index location to contain the keys which happen to produce the same index value. These overflow locations are searched sequentially. Figure 1 depicts a hashed index structure.

(continued next page)

A B-Tree index usually uses the ASCII sorting sequence of the characters in the key to determine the location to put the key in the index, but its actual location is also influenced by the order of entry of the keys and thus what is already in the index. The B-Tree uses what is called a balanced tree structure to store its keys. Figure 2 depicts a B-Tree. This essentially means that the length of the search path through the tree to the null point which tells you that the key you are looking for is not in the tree is the same for all keys. The length of the path increases by one step for each doubling of the index storage size. This offers the advantage that all accesses to information have a maximum which is constant for a given index size. In contrast, the hashed index can become lopsided and build up a large number of overflows from one or more index locations. These overflow locations have to be searched sequentially and thus can add substantial delays to locating some information. Cleaning up this imbalance requires a special routine to rebuild the index and incorporate the overflows into the primary structure. The B-Tree in contrast must do its reorganization at each insertion into the index. The algorithm for locating the keys usually only involves collation comparisons and key moving; hashing usually requires multiplication and division, which are time consuming. The net effect is that if the hashed index is balanced the access times and insertion times for the two structures will probably be about the same.

A key is inserted into the tree index by entering the top bucket and comparing the key to those already there. If the key is between keys in the bucket and there is room, it is deposited there. If it is greater then any key in the bucket it travels down to the next level to the right. If it is less then any key in the bucket it goes to the next level bucket to the left. It continues in this manner down the tree until the last level. If the bucket is not full it is left there. When a bucket becomes full keys must be redistributed in some manner to make room or a new level of buckets must be added at the bottom.

I had no problems getting FABS running. Everything worked as specified in the manual. MAGSAM IV required re-linking and at first I didn't have the proper version of Microsoft's L-80 to link it. After relinking I encountered a little confusion trying to do the experiments in section 5 of the manual. They were written for the version III and had some notes as to the differences at the beginning of the section but each experiment did not remind me of the different behavior expected of me at the appropriate points. Once I figured out my problem I had no further trouble.

FABS has a program which creates a test file of 1000 records for you to play with and which allows you to get some feel for the responsiveness to a random search. MAGSAM's test program only allowed hand entry of records and thus the number of records never became large enough for me to feel that I was testing its ability to locate records quickly.

FABS is a relocatable machine language overlay with 6 different entry points to accommodate 6 different languages: CBASIC2, SBASIC, MBASIC, PL/I-80, FORTRAN-80, and PASCAL/MT. It will accommodate both version of Microsoft's interpreter and the compiler. MAGSAM is available for CBASIC2, MBASIC, BASCOM, and Micropolis BASIC. A separate implementation must be ordered for each, however. Version III of MAGSAM comes as basic source code while version IV is a relocatable machine module.

MAGSAM uses eleven variables to pass and receive parameters between the module and the application program. FABS uses one buffer for two way parameter passing and one error register location. The functions each utility provides are similar. Each has functions to create new indexes and to open existing indexes. FABS can open up to six at once while MAGSAM can have up to 20 open at once. Each allows you to insert and delete keys. FABS re-uses deleted keys for new inserts if any are available before adding to the file length. MAGSAM requires you to stop and use a special program to recover space occupied by deleted records. Each has both an exact match search and a generic search function but they work differently. MAGSAM requires the generic key to be of the exact key length and returns the record equal to or greater than the search key while FABS accepts a truncated key and returns the first record match that much of the key. Each allows a search for the next indexed record but only FABS has a previous record function. FABS also has a first and a last indexed record function. FABS allows you to find out how many records are in the file and how many deleted records remain un-reused. Both FABS and MAGSAM have a way to write a series of inserts into the index which save time from the individual insert function. MAGSAM has a write load function which requires that the key be entered in collating sequence. FABS normally writes each insertion to mass storage before returning to the application program but it has a build function which does not save the memory map constructed until a write build function is executed. This function does not place any restrictions upon the key order of insertion but it does require that the write be performed before attempting any searching or closing. Both utilities accept a key and a function and return the sequential record number. The application program must then do a random read or write to get or store this record as required.

The FABS manual contains a table which can be used to determine the total number of keys which you can expect to be able to index for a given index length. The nature of the B-Tree structure and some limit on file size causes this to be a somewhat imprecise number as explained in the manual. The table implies a practical limit to key length of about 50 but 100 characters are probably possible. MAGSAM allows up to 110 characters in its keys. Other limits may be imposed by the computer language being used with either utility.

Either one of these utilities would enhance the performance of most applications. The six different entry points for FABS suggest the possibility of having data files and indexes which can be used by programs written in different languages. The file data structures may have to be adjusted to be accessible by the different languages, however. Overall, I would judge FABS to be the most versatile and easiest of the two utilities to use.

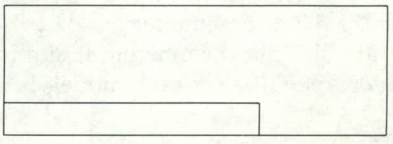


Figure 1

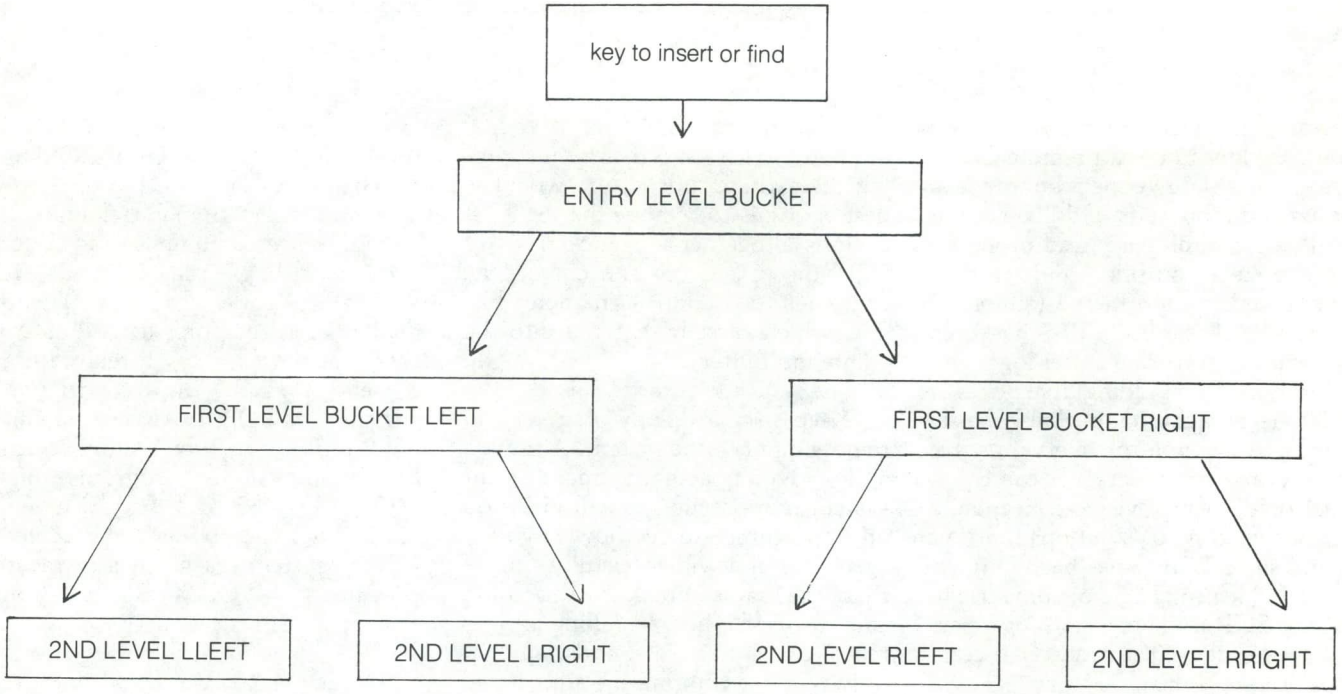
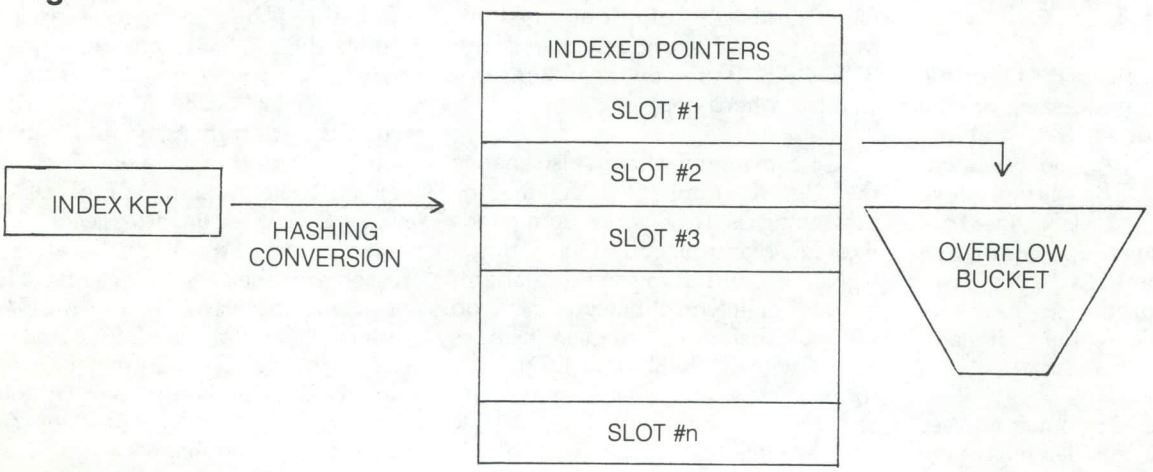
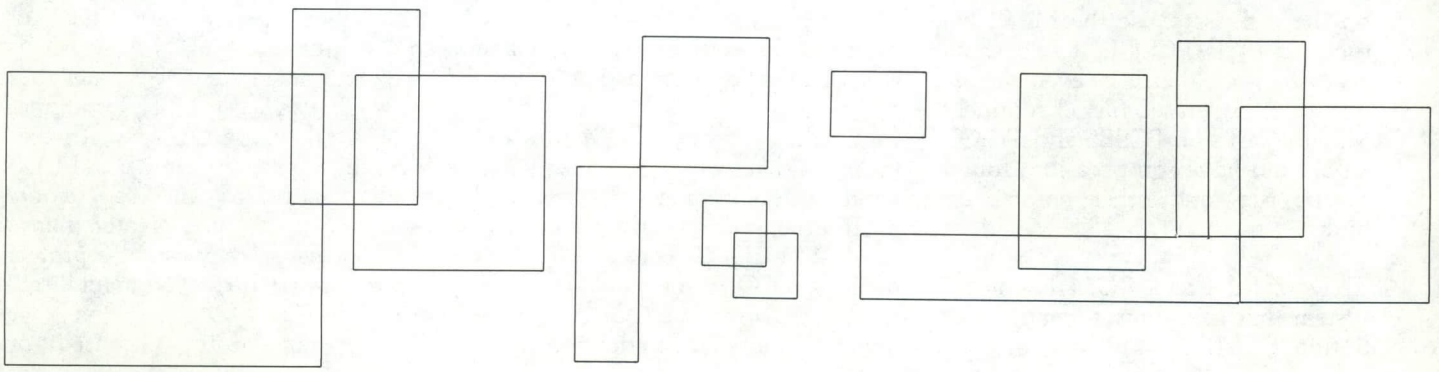


Figure 2





Remote System Do's and Don'ts

Dave Hardy

Doesn't it really burn you when someone logs into your remote CP/M system (that you've spent hundreds of hours and thousands of dollars to put on-line for public use) and promptly tries to steal you blind and crash or ruin your system? In the (almost) 2 years that Technical CBBS has been on line, I've compiled a user log about six feet high (no kidding, five boxes of 3000 sheets each) that probably has at least one example of every possible thing that a system crasher can try to steal or destroy. I've been keeping a list of the "top-10" solutions that I've found since TCBBS has been in operation, which might be of some use to new SYSOP s. There is nothing amazing in the file - it's mostly just common sense, but it is very easy to forget these ideas. I know that from many painful experiences.

SYSOPs, here are some things you can do to stop a potential system crasher:

☐ Keep a CRCK file for ALL of the .COM files that you leave on-line (And also any other files that get loaded into the TPA and executed, like MINICBBS.OBJ). If you have any password files, CRCK those too. For obvious reasons, don't leave the CRCK file on-line (CRCK.ASM is a program that produces a cyclic-redundancy-check value for any specified files).

☐ Use MDIR.COM frequently to see what goodies the invader may have left you in certain user areas. **Note**: MDIR.COM will NOT find files that are "hidden" in the "user areas" above 20H! The best way to see everything on the disk is to use the MAP (M) function of DU.COM. It will show *everything* on the disk, even the remains of any erased files. I routinely Map the disks on TCBBS and SYSOP CBBS and have, on occasion, found special files and other no-no's on both.

☐ Don't leave any .COM files on the system that can allow a remote user to find .SYS files. Most directory programs (and also WHATSNEW) allow anybody to list .SYS files by just typing an extra character or two. The best action is to remove the .SYS list options altogether (A quick fix is to DDT the .COM file and change the character to a control character like ^C which can't be entered into the command buffer).

☐ Keep a hard-copy log of all remote input to the system. Although a log won't actually make your system more secure, it will give you an opportunity to see how anybody "gets in", and will, hopefully, insure that the same break-in procedure can't be used twice. Installing a log is really easier than it sounds, since it only requires printing the stuff typed by the remote user, not the stuff typed by the system. An inexpensive (i.e. cheap) printer is perfect, since you don't need letter-quality type to see who's been interfering with your $3000-and-up computer system. Many BYE programs (like BYE69.ASM) already include the option for a hard-copy log.

☐ Of course, you should also change the CP/M commands. Again, the best thing is to *remove* commands like ERA and SAVE (Don't forget REName!), but if you're not that ambitious, or if you think you can't do without them, just change them as usual, with SYSGEN and DDT. Try to pick new commands that aren't easy to guess, although it's impossible to guarantee that no one will be able to figure them out in time (I have a listing from TCBBS log where someone spent about 8 hours trying to find one of the commands). If you want to eliminate a command, you can imbed a control character into the command word and make it impossible to use.

☐ Don't leave any .COM files out that would allow a remote user to examine or modify memory, or to load a .HEX file. It is perfectly safe to leave out ASM.COM, because it can't make a .COM file, but to leave LOAD.COM or L80.COM out is to invite a remote user to download his favorite debugger to see what he can do. BASIC.COM and DDT.COM are also bad news, since both could allow a remote user to make changes in memory. Even a compiler can be left safely on-line, as long as its associated loader program is not available. Also, don't leave out any files that would allow a remote user to send a .COM file over to your system. XMODEM.COM checks for .COM files and won't allow them, but many other programs, like MODEM.COM and BSTAM, will allow *any* file to be sent or received. Once a system crasher has a way to download a .COM file to your system, all is lost.

☐ In CP/M 2.x, an illegal drive request might also change the current user area! In other words, a remote caller who is logged into A: user 0 could type "Q:" and end up on A: user 1! Digital Research doesn't think of this as a bug, because in an unmodified CP/M system, a disk select error will cause a PERMANENT BDOS error. The problem arises when the user changes his BIOS to allow a warm-boot on a disk select error, instead of a permanent BDOS error. CP/M doesn't reset the user/drive byte properly. That's the reason for the strange results. This problem can be fixed in your BIOS by properly handling a SELDSK error, but if you don't have the source for your BIOS, you could be in trouble. Another way to protect yourself against this problem is to keep "private" stuff in user 5 or 16-32. Strangely enough, all other user areas can be entered with an illegal drive code. Putting things in user 5 will make them pretty safe, and, of course, putting things in user areas 16-32 will make them even safer, but the CCP can't get YOU into those areas, so their use is a bit restricted. Most BYE programs have a MAXUSER equate that will keep remote users out of any area greater than a preset value, so they can also protect you to a certain extent from an illegal drive select.

☐ You can protect licensed or

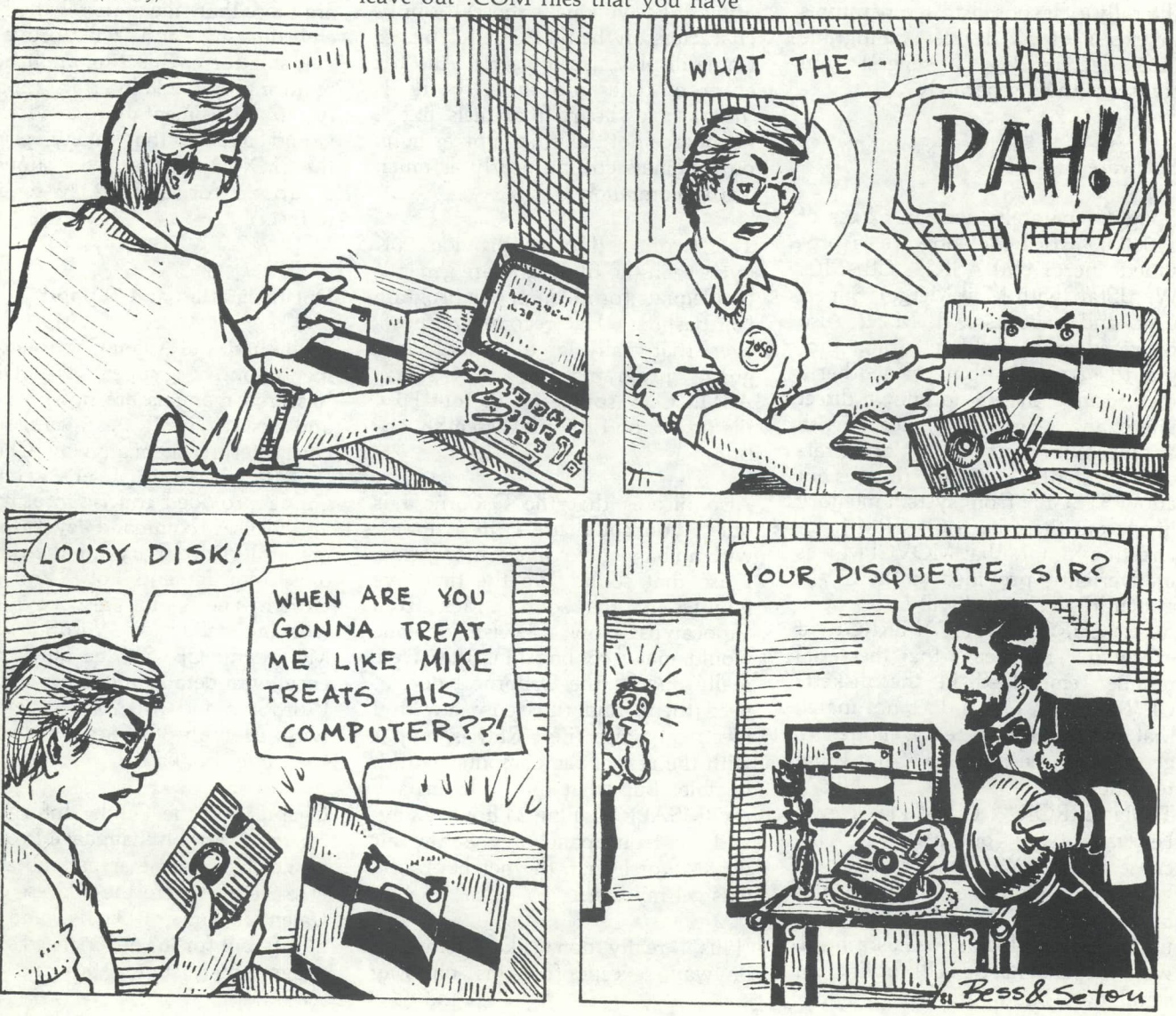
private software by keeping it in an inaccessible user area, and using a short loader program like Keith Petersen's SECURITY.ASM. This really works, and makes the SYSOP feel good when he sees in the log that some invader who thinks he has just successfully stolen MINICBBS has actually just stolen a short loader program.

☐ Probably the biggest security problem is *incredible stupidity*. It is rumored that some SYSOPs have actually done really dumb things like leave PIP.COM or MODEM.COM or FORMAT.COM (shiver...) out in a public user area! If you absolutely have to leave one of these (potentially) nasty little programs on your system, put it in a user area that can't be accessed remotely (or at least a non-public area) and rename it to a .OBJ file. Then even if someone gets into the user area with the program, he can't run it (.OBJ).

☐Don't leave your system or CP/M passwords anywhere on the system. Use TAG.COM to make sure that someone can't XMODEM your BYE.COM program and other goodies. Don't leave a SYSGEN image (CPM56.COM) laying around either, since it could be downloaded and DDT'ed to find the system commands. Also, don't leave your system PW's on another system in a private message to a friend thinking that his message system is private, because it probably isn't. I'm not being paranoid, everybody *really is* trying to break into my system...

☐ Watch out for booby-trapped .COM files! If someone sends down an .OBJ file suggesting that you leave it out on your system, be sure to check that file for any hidden functions that may allow someone to break into your system later. One way to prevent this would be to only leave out .COM files that you have assembled from SOURCE files. In any case, be suspicious of any files left you for public use that don't have the source with them. A good programmer could hide a secret function in a .COM file so well that it could only be found with a great deal of difficulty. In addition, an unknown .COM file might also have many other terrible hidden functions (See SIG/M Volume 18, File 18.9: BANZAI.ASM for some ideas) that could even destroy other files on the system's disks, so be careful.

Some of these precautions may seem like a pain in the neck, but the more prevention you have, the less you have to worry about the one in a thousand callers who wants to misuse your system. *No system* is absolutely secure, but with these suggestions you should be able to run a system that is almost secure, which isn't really that bad.



The Osborne I Computer

Kelly Smith

Well, I did it, bought (Wife: "but why do you need another computer???") an Osborne I, with an out-the-door price of $1901.70! And to answer my wife's "why?", there are many reasons. They follow.

Transportability

Throw the little sucker (gently) into the back of your car, and take it anywhere you can find 110 VAC (or use the battery pack, 4 hours running time). But a word of caution here: the Osborne I may take the heat in the trunk of your car, but those 5 inch floppy diskettes, neatly tucked into two carrier slots, curl up and die (literally) if exposed to temperatures in excess of 52 degrees Centigrade (125 degrees F.); I suggest you transport them separately.

Software

Bought separately, well in excess of $1000.00 worth of software is provided here: MBASIC, CBASIC, WordStar (with Mail Merge), SuperCalc, and, of course, CP/M 2.2. Also provided are a diskette format program (sorry folks, but yet another 5 inch format; there's no known direct interchange capability except with another Osborne I), a [sort of bastardized] SYSGEN program. (It does not allow SYSGEN from system memory image put there by MOVCPM!). Interesting to note that MOVCPM was inadvertently provided on the CP/M system diskette, and will inform you of "SYCHRONIZATION ERROR" if executed. This means that the serial number embedded in the diskette CP/M system image does not match that of MOVCPM...easy enough to get around the problem (See: 'What to to about CP/M's "SYCHRONIZATION ERROR"', CP/M-Net News, February 1981), but it's useless because of the SYSGEN program provided (an additional note on this later). Finally, a disk-to-disk copy utility is provided as well as a hardware dependent SETUP utility for serial port Baud rate selection (300 1200 Baud), CRT screen width selection (52 to 128 columns), and real-time calendar/clock set-up (which may be read by an applications program, nifty!).

Hardware facilities

A (quasi) modem interface, RS-232C (serial) interface, and IEEE-488 (not full implementation, but adequate) bus interface round out the means of getting to the "real world". There is also a video interface; it is unfortunately not usable with just any video monitor - you must use the 12" monitor (purchased separately) provided by Osborne Computer Corporation. It lets you make those itty-bitty characters on the built in CRT bigger (same display, just bigger!). And for entry of data into the Osborne I, there's a keyboard that feels like a keyboard, not the usual junk as hung on terminals costing nearly as much as this computer!

The two 5" floppies provide 90K bytes each (no files), and are generally adequate for medium size program "bit-bashing". I do recommend however, that really large assembly language programs use the '.BAZ' ASM.COM (source on B:, put HEX file on A:, no PRN file) assembly option.

Also nice is that the Osborne I is quiet; you could use it in a library with no hassle! Not like some systems I use that sound like the first five minutes of the sound track from 'Apocalypse Now' - I wish someone would make cooling fans that were really quiet - the Osborne I doesn't need them, and runs cool anyway! Electrical noise (i.e., RFI) is on-par with the rest of the computer world, terrible! But what the heck, usually my IMSAI is running 24 hours a day, and its front panel sprays the air waves worse than the neighborhood cats get my van.

(Plus I really don't like Channel 2 anyway, except for this one-way love affair with Melody Rogers [Two On the Town] that I have...Osborne I goes off promptly at 7:30! Does she know I exist? Can a computer phreaque [and pervert] find true love with this fleeting nymph of the boob-tube? Will firmware finally meet software (and wow, does she look soft!...and cuddly, and...?)

Gads! I lost track, where was I? Oh yes, as with any other computer of the plastic-wrap variety, TV watching is messed up. But then for me, watching WordStar reformat a paragraph is more fun than watching 'Laverne and Shirley' anyway.

To sum-up, the interfaces provided are more than adequate for the home environment, and will prove invaluable for engineering applications requiring IEEE-488 interface capability to instrumentation devices. It would appear that many printers (i.e., MX-80, Anadex, etc.) should be a 'drop-in' for the RS-232C serial interface.

Documentation and Support

As with all the Adam Osborne documentation, it's super! The Digital Research manuals are not provided, and just as well; the average user can't interpret them anyway. Chapter 6 of the 'Osborne I User's Reference Guide' provided concise information at the user 'command level' that is [generally] necessary for the day-to-day requirements of "knowing" CP/M. The 'Guide' also covers the working details of WordStar/MailMerge and CBASIC as quick reference, with detailed documentation of WordStar and MBASIC covered in two [superbly Osborne re-written] soft-cover books.

Chapter 8 of the 'Guide: 'Information for the Assembly Language Program' is sketchy in some areas (and also in error; e.g., where the address of the Calendar/Time-of-Day is...and a few more), but for the experienced system programmer, should pose no prob-

lem. If you want to do anything fancy, however, get the manufacturer's data sheets for the 6850 ACIA (serial/modem port), 6821 PIA (IEEE-488 port), and 1793 FD Controller/Formatter. If you don't know what a ACIA and PIA are, you are not experienced! Interesting that the IEEE-488 interface is covered in detail (including assembly language I/O examples); that's probably to clarify that it is not full implementation. The video display software interface gets a good working over, with a 'mixed bag' of Control Code and Escape Sequences for almost all the graphic applications you could desire. Watch out for the (Table 8-1) graphics characters table; at first glance at the table, and then at the CRT, you would swear that it was wrong (confused dummy me at first!). It's just that the printed table is set with surrounding 'black' on 'white' characters, just looks strange 'till you get your head screwed on: you can't put 'white' characters on 'white' paper!

Support

In this day and computer (?) age support seems to be lacking with so many small computer companies, but not with Osborne! As with any new product, there are a few wrinkles that need to be worked-out: the N-Key rollover doesn't; a quirk with the keyboard alpha-lock; no CP/M (a la Intel) I/O Byte implementation. But listen to this: the dealer that sold you the Osborne I will receive new keyboards, PROMs, whatever...and update your computer for free!!! Considering the already inexpensive price for the computer and software, you get real factory support and back-up at no additional cost. I am impressed, considering what I have paid (and no doubt will continue to pay) for updates to software packages; i.e., for only $150 more, I can get MP/M II updated from my [less than bug free] MP/M 1.1...or Pascal/Z 4.0 for only $50 more, when I've already paid over $100 to fix PAST bugs! Anyway, a welcome change from past experience is in store for the computer consumer who buys from Osborne.

Additionally, addenda are due (also free) to the manuals provided, as well as the long awaited release of SuperCalc.

Finally, to complete the icing-on-the-cake, someone on the customer relations phone knows what he's talking about, is friendly and courteous (this is Dave Lopez). If you need to talk with 'the man' (Adam Osborne), he actually will return your call if he's not in. Also, Ken Jacobsen is the guy to talk to about the Osborne Approved Software List (if you are a software author), and Annette Truesdale will provide information (software listings) and details of the 'innards' if you send a requesting letter stating "why and what for". I was told that schematics may be made available also, but no policy decision to do so has been made yet (I have visions of dropping a Seagate '5 Inch Winch' into that place where the B: disk currently sits).

A little more about the documentation: they didn't tell everything! Bet you didn't know that you could cold-boot the system off of the B: disk (B: becomes 'logical A:', A: becomes 'logical B:'). Here's how:

Enter 'doublequote' (the keyboard character ") and then carriage return...simple!

Want to run 'built-in' diagnostics? Then just enter Control-D with CAPS LOCK off (and after pressing RESET), and you will be treated to the following display:

```
ROM DIAGNOSTICs
Select Test:
 A,B Boot Sys
   D isk
   K ey-vdt
   M emory
   R ead in test

: — «— diagnostic prompt and cursor, waiting for command
```

You have the option of just booting the CP/M system off of disk A: (or B:!), a (primitive) disk test with 'D', a keyboard/video display test (including generation of graphic characters from the keyboard) with 'K', a memory test (sort of...) with 'M', or read data in from the RS-232C serial port (used by Osborne technicians when floppies are dead I suspect). Here are some details (as I have been able to discover them!):

D isk — Entering 'D' prompts with "Select (A,B):"; select enter A (or B), and the diskette is quickly verified for CRC errors; you should then see the display "ERR cnt - 0000".

Any disk errors detected during a second pass are displayed as:

Dsk ERR - (sts trk sec)- FE 03,4C 0A,00

A bit cryptic perhaps, but status for the 1793 chip is displayed, as well as Track and Sector number information.

K ey-vdt — Entering 'K' prompts with "Enter Keys (till ^Z):"; just enter keyboard characters, and confirm their proper display on the video monitor. Graphics characters may be displayed by entering ESC, then 'g' (lowercase G), and then CTRL (Control) with the associated keys for graphics characters.

M emory — Entering 'M' displays "Memory Test", and loops continuously (until RESET) while displaying a single incrementing ASCII/graphic character in the upper-right portion of the video monitor. This appears to be just a simple 'read-ryte/write-complement/read/restore-byte' test, and is only good for 'hard' RAM failures; but considering it's an undocumented 'freebie', what the heck!

R eadin test — Entering 'R' prompts with "Start input: Len="; a serial input device (such as a modem) connected to the RS-232C port will be read. If the device is not ready, the Osborne I bounces back to the system sign-on prompt. At the completion of a block read, the length of the record is displayed (in Hex) after the "Len=" display. How it's actually used, and with what real protocol

(continued next page)

beats the devil out of me, as I have yet to figure out where in the system the PROM resides! It remains resistant to any and all attempts to find it. I will fill you in on the details, as soon as I find (and disassemble) it, but for now...

For those of you that want to go 'stomping' through the Osborne I computer system PROMs (they are 'shadowed' in the normal CP/M system environment) and want to discover further internal details of the computer (for instance; the real BIOS jump vectors are at address 100H and selectively accessed by the secondary jump table at EA00H!), use this short routine to: (1) turn off low memory RAM and turn on the system PROM; (2) move the PROM data to address 8000H (for examination); and (3) turn low memory RAM back on and PROMs off, to restore the system to normal. The 'GET-PROM' program is as follows:

Edit and Assemble, load the '.HEX' file with DDT.COM, execute with a 'G4000 ‹cr› '. Re-execute DDT.COM, and Dump or List (or even Move it to address 100H, exit DDT and "SAVE 16 PROMS.COM ‹cr› for disassembly using ZESOURCE.COM), and maybe YOU will discover further surprises in the Osborne I!

For now, I will wait on any detailed 'exploration' until the new system PROMs come...and give you further details on this interesting (and inexpensive) computer.

If asked "what else would you want?" on the Osborne I, I would first ask for a valid copy of MOVCPM.COM along with a real SYSGEN.COM and configurable BIOS.ASM. I think that for applications programmers (and even the occasional 'hacker'), this is essential. Second, if you intend to write Z80 assembly language code (this is a Z80 computer!), ZSID.COM, MAC.COM and it's associated library files would be a nice 'extra' to complement the already super software offered with this computer...finally, that vision of a 20M byte Seagate 'Winch'...Oooo, Ahhhh!

To sum-up, this could be the best thing that's happened to microcomputing since the Altair. And with today's inflation its probably cheaper than the 1975 Altair price, as well as the "Best Bargain for Your Buck"!

Index Available

On December 15th, a complete index of *Lifelines* articles will be available for $2.50, including material covering our contents through this December 1981 issue. issue. Updates to this index will be made every three months.

All orders should be pre-paid, by check, MasterCard, or VISA. Checks must be in U.S.$, drawn on a U.S. bank. Write for your index or call (212) 722-1700.

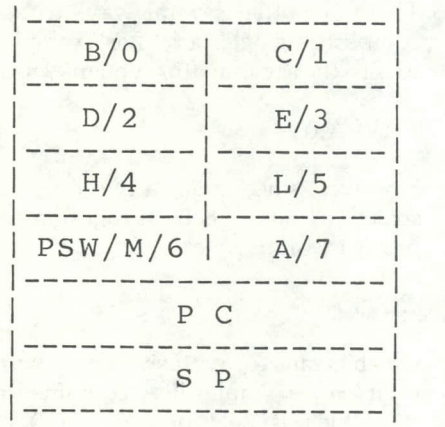
```
        org   4000h      ; put above system PROM area

        di                ; disable interrupts or it won't work
        xra   a           ; turn on PROMs
        out   00h
        sta   0ef08h      ; let system know we are in PROM
        lxi   b,1000h     ; make counter for 4K bytes to move
        lxi   h,0000h     ; and move data starting from address 0
        lxi   d,8000h     ; and put it in high RAM at address 8000
move:   mov   a,m         ; get a byte of PROM
        stax  d           ; save it in RAM
        inx   h           ; bump PROM pointer
        inx   d           ; bump RAM pointer
        dcx   b           ; de-bump byte count to move
        mov   a,b         ; check if all data moved
        ora   c
        jnz   move        ; loop until all 4K PROM moved to RAM
        mvi   a,1         ; turn off PROMs
        out   1
        sta   0ef08h      ; let system know we are in RAM
        ei                ; enable interrupts
        jmp   0000h       ; warm-boot CP/M

        end
```

8080 Programming Tutorial Data Movement Instructions & Arithmetic Instructions

Ward Christensen

In this section of the tutorial, I'll cover the data movement instructions of the 8080 microprocessor.

Figure 1 will help picture the movement instructions.



PSW = Program Status Word
A = Accumulator
M = Memory as addressed by HL
PC = Program Counter
SP = Stack pointer

The numbers shown are the values the registers take on in the 8080 instructions that reference them.

THE DATA MOVEMENT INSTRUCTIONS

MOV

The simplest form of data movement is from one register to another. The MOV instruction does this. It moves 8 bits at a time. The syntax of the MOV instruction is:

MOV destination,source

Destination and source may be any of A, B, C, D, E, H, L, or M. You may not explicitly move data to the PSW. The special case of:

"MOV M,M"

is not legal. The "bit pattern" this instruction would generate means to HALT the processor, and actually has its own OP CODE: HLT.

An example, "MOV A,B", will move into A, the value currently in B. I specifically worded it that way, rather than the more obvious "move B to A" to emphasize the *order* in which the registers appear in the operand of a MOV instruction.

The register which is shown in Figure 1 as PSW/M/6 deserves special attention. Use of register "M" is quite different from registers A, B, C, D, E, H, and L. "M" refers to the contents of MEMORY, as addressed by the register pair HL. Thus:

MOV A,M

moves to A, from memory. So, if the HL registers contain the address 1234 in hexadecimal (usually referred to the way the assembler accepts it: 1234H), then whatever value is in the memory at that location is moved into the A register.

LDA and STA

It is fastest to use HL as a pointer, and M as a register when moving data to or from memory. Why? Because the instruction is only 1 BYTE long, and the 8080 doesn't waste much time fetching it.

It is not always convenient to have the address you want to reference in HL. The LDA (LoaD Accumulator) and STA (STore Accumulator) instructions move data between memory and the A register, but have the address in the instruction itself, rather than in the HL register. The syntax of the LDA and STA instructions are:

LDA address
STA address

For example: LDA VALUE1 or STA VALUE2.

If you need to move one byte of data from one place in memory to another, then coding LDA address1 then STA address2 is the most practical way. Or, if HL points to one of the fields (say address1) you could fetch the data with a MOV: MOV A,M then use STA to store the data: STA address2.

(Aside: I have used the phrases "pointer" or "pointed to by" or "points to". If you knew what I mean, skip the next paragraph.)

If I code "MOV A,M", I load into A, the memory byte location addressed by the HL register pair. However, I will not use the "formal" term "addressed" any more, but will say "HL points to" the location, or "the location pointed to by HL".

I mentioned LDA and STA are practical for only single byte moves. You might ask "how do I move a block of data?" Read on...

LDAX and STAX

When moving data between a register and memory, HL is not the only register pair which can be used as a pointer to memory. The BC and DE register pairs may also be used.

The LDAX (LoaD Accumulator using indeX) and STAX (STore Accumulator using indeX) accomplish this function. The syntax for the LDAX and STAX instructions are:

LDAX xreg
STAX xreg

where "xreg" is either B or D. NOTE that even though you might think of register *pair* BC or DE, all instructions using them simply use the first letter: B or D.

You can now begin to see how to put some of these instructions together to make a program.

Suppose DE points to one place in

(continued next page)

memory, and HL points to another, and you want to move 3 bytes from where DE points, to where HL points. Code:

```
LDAX D
MOV M,A
```

3 times, incrementing registers DE and HL each time, so they point to the next value.

In future articles I'll cover the increment instructions, as well as getting into looping, which will allow us to move an arbitrary number of bytes (rather than the 1 or 3 which I have shown in examples so far).

MVI and LXI

At times, you know that you want a specific character or value moved into a register or register pair. Thus there is a need for instructions that move data *from the instruction itself*, not from another register or memory. Note that these are not *required* instructions, since you *could* address a memory location to get the data. However, then you need 2 bytes (16 bits) of address in the instruction to point to the data, and 1 or 2 bytes (8 or 16 bits) of actual data somewhere else in memory. With an "immediate" instruction, you save the 16 bit pointer, since the data is right in the instruction.

This type of instruction is called an "immediate" instruction, because the data is moved "immediately" from the instruction to the register or register pair.

To move an 8 bit value into a register, use the MVI instruction. The syntax of the MVI instruction is:

```
MVI register,data
```

(such as MVI A,5 which moves a 5 into register A). Applicable registers are A, B, C, D, E, H, L, or M.

The data moved may take on any 8 bit value, such as a decimal number from 0 to 255, a hexadecimal value from 00H to 0FFH, or an ASCII character such as 'X'.

On occasion, you might want to move PART of a 16 bit value into an 8-bit register. Or, you may want to move a negative value, such as:

```
MVI A,-1 ;«===
```

(Won't work.) The reason this won't work on all assemblers, is that they treat "-1" as a 16-bit value, namely 0FFFFH. The assembler "objects" to "throwing away" the first two FFs.

A way around that is to ensure that the assembler sees it as a value which *will* fit in 8 bits, i.e. in which the "high order" 8 bits are all zero.

One technique is to "figure out" what the correct value is:

```
MVI A,0FFH ;means -1
```

However, there are times when either (1) you want to use some value that isn't so easy to calculate, or (2) you have an "unknown" value, such as wanting to load the low byte of an address.

If you were to code the following:

```
MVI A,-1 AND 0FFH
```

the assembler would "see" 00FFH as the value, and wouldn't "object". Similarly, if you have a label "FOO", and you want to load the "low byte" of its address into A, code:

```
MVI A,FOO AND 0FFH
```

Let's look at that in more detail: Suppose FOO is at 126CH. Expanding this to binary, and setting up for the AND 0FFH:

```
0001 0010 0110 1100 for 126CH
0000 0000 1111 1111 for 0FFH
———————————
0000 0000 0110 1100 "anded"
```

Thus the AND 0FFH ensured that the low order 8-bits of the address FOO were used, and the high order 8-bits were all zero.

The value loaded into a register may also be a single character as in:

```
MVI A,'5'
```

Note here the difference between 5 and '5'. The first is a numeric value, equivalent to any of the following:

```
character: (not printable)
decimal:   5
octal:     005
hex:       05
binary:    0000 0101
```

and the second is a character value, equivalent to any of the following

```
character: '5'
decimal:   53
octal:     065
hex:       35
binary:    0011 0101
```

There are often several ways to accomplish a single task. For example, to zero the accumulator, you may use:

```
MVI A,0
```

However, since this is a two byte instruction, most 8080 programmers prefer the more "elegant":

```
XRA A
```

which means to "exclusive-or" A with itself. For example, if A contained a '4', which is 34H or 0011 0100 in binary, exclusive-oring it with itself gives:

```
0011 0100 exclusive-ored to
0011 0100 gives:
———
0000 0000 Zero!
```

This was just a "flavor" of a logical instruction - I'll go into them in detail after the arithmetic instructions section of the tutorial.

Earlier I discussed the use of HL as a pointer to memory, but did not discuss how the value is loaded into HL. Frequently you want to load HL with a specific value, such as the address of a location in memory, for example, the address of the first byte of a string of characters to be printed. The LXI instruction does this. It can be used to load either BC, DE, HL, or the stack pointer (SP) with a 16 bit value. The syntax of the LXI instruction is:

```
LXI register pair,value
```

(such as LXI H,5000). It is possible to load character using LXI, but it is very infrequently done. The data value is more commonly a number, or address.

For example, to clear the BC register pair to 0 you LXI B,0. The data value loaded with the LXI instruction can be treated either as a positive number, from 0 to 65535, or as a signed number, ranging from -32768, to +32767. It may be expressed in hex, as in "LXI H,5CH", or as a label, such as "LXI H,FCB". NOTE this last example is NOT loading the value STORED at "FCB", but rather the ADDRESS of FCB.

The LHLD instruction (explained next) would be used to load the data AT the label FCB.

LHLD and SHLD

The HL register pair is very useful, partly because of its role as the address pointer for the special "M" register mentioned previously, but also because it can do 16 bit arithmetic adds (which will be covered in a future article), and a 1-instruction shift left 1 bit.

Therefore, you frequently want to load or store the contents of HL in memory. For this, use the LHLD (Load H and L Directly) and SHLD (Store H and L Directly) instructions. The syntax of these instructions is:

LHLD address
SHLD address

(for example, LHLD VALUE1). The value in memory is in a slightly odd format. Those of you familiar with other computers are likely familiar with the idea that the most significant part of a data value is stored first, the least significant last. (This is the same way we think of ordinary decimal numbers: 123 has the "most significant" part, i.e. "100" first, and least significant, "3", last.)

"High order first" is not the case with the 8080. Probably for some reason such as minimizing hardware on the 8080 chip, the low order byte of data is loaded or stored in memory first.

Thus if the value 1234H is in HL (12H in H, 34H in L), a SHLD instruction would result in 34H stored in the first byte referenced, and 12H stored in the next. This is not extremely important to know, but is necessary if patching a program, or assembling by hand.

XCHG

Since the "M" register only refers to the contents of memory as addressed by HL, it is frequently useful to be able to "swap" or "exchange" the contents of DE, with that of HL. The XCHG (eXCHanGe) instruction, which is coded with no operands, exchanges the contents of DE with the contents of HL.

A typical usage, is:

```
XCHG        ;SWAP DE, HL
SHLD FOO    ;STORE HL at FOO
XCHG        ;PUT THINGS BACK
```

SPHL

This instruction moves the contents of HL to the stack pointer, "SP". It will be covered in more detail in a future tutorial dealing exclusively with STACK related instructions.

XTHL

Most programs get by without this instruction, but it is a data movement instruction, and therefore deserves attention in this section of the tutorial. XTHL exchanges the top value on the stack, with the contents of HL. I'll bring up this instruction in future articles when I discuss the use of the stack. Note that this does not exchange HL with the *value* of the stack pointer, but rather with the top value *on* the stack. This is an easy mistake to make.

Actually, the most common usage of XTHL in programming which you will see in The CP/M Users Group, is to "waste time". Looking at the INTEL 8080 book, I find that XTHL is the LONGEST executing instruction the 8080 has.

For example, Sam Singer, in his DFOCO (Double density format and copy program) on Volume 38 of the CPMUG library, uses a loop containing two XTHL's to "waste time" waiting for the floppy disk head to seek to track 0 after issuing a "home" command. (Note that since XTHL swaps the top of the stack with HL, it is usually done two at a time, when used as a time delay.

There are a "few" other ways of moving data, which will be covered in future sections of the tutorial. For example, "moving" data to an output port, or "moving" the stack pointer (SP) to the HL register, which must indirectly be done, by loading HL with 0, and ADDING the stack pointer.

ARITHMETIC INSTRUCTIONS

Microcomputers are quite capable of doing arithmetic. Most of you have seen BASIC programs which handle floating point numbers quite easily. In truth, the most simple arithmetic in BASIC, such as "A=A+1" results in the execution of thousands of instructions "beneath the covers".

Typically, a microcomputer is able to do little more than one byte addition and subtraction, two byte addition, and sometimes (but not in the 8080) two byte subtraction.

As the 16-bit processors become more prevalent, we will have machines at our disposal which do 16-bit multiplication and division of integers.

Floating point processing is still relegated to either fairly large and complex subroutines, or to special purpose "math chips" - integrated circuits which do only specialized math operations.

I have yet to see an assembler program which executes floating point operations, except as part of a higher level language compiler or interpreter.

8080 STATUS BITS

Before I get into the instructions, we should look at the "flag" or "status" bits which are set when an arithmetic instruction is executed.

You saw this byte of flag bits when I discussed the 8080 architecture. The byte was the PSW, or Program Status Word, and may be seen pictorially as:

(continued next page)

S	Z	0	A	0	P	1	C

SYMBOL MEANING

S Sign: on if negative
Z Zero: on if results of previous op = 0
0 Unused bit, set to 0
A Auxiliary carry: used in decimal arithmetic
P Parity bit: on if even
1 Unused bit, set to 1
C Carry: on if carry.

These bits are set or reset, by arithmetic, logical, or compare instructions.

I'll cover how each specific instruction type affects them in the appropriate sections of the tutorial. For now I'll concentrate on how arithmetic instructions affect them.

Here's a detailed explanation of the bits:

The "S" or sign bit, is set by arithmetic instructions if the result has the sign bit, or highest bit, on. If you want to handle the value as signed, then the S bit on in the PSW means the result was negative. Recall that an 8-bit value may be considered to be either a positive value from 0 to 255, or a signed value, from -128 to 127.

The "Z" or zero bit, is set if the previous arithmetic instruction resulted in a zero result.

Another use of the Z bit is to test a register for zero, after decrementing it. This applies to 8-bit registers only. To test a 16-bit register for zero requires several instructions.

The "A" or auxiliary carry bit, is meaningful only when the byte being operated upon is considered to be two decimal digits, instead of an 8-bit binary number. For example, adding 23H and 19H, results in 3CH. However, if we thought of 23H and 19H as decimal 23 and 19, i.e. having each 4-bit half of the 8-bit value thought of as a decimal digit, then we would prefer an answer of 42.

The "A" bit holds the "carry" out of the low four bits, if considered to be a decimal number. For example, the addition of the digit 9 and 3 in the above example would set the auxiliary carry.

The DAA (Decimal Adjust for Addition) is used to make the binary number back into the correct decimal one.

The "P" or parity bit, keeps track of whether an odd or even number of bits were on in the result of the last arithmetic operation. This is not frequently used. A notable exception, was in the early Microsoft BASIC (then called MITS BASIC). They did lots of "hack" coding tricks, partly because it was "fun" to do, and partly to conserve space and time.

The Z-80 uses this bit as an "overflow" bit, when executing arithmetic operations, and as a parity bit when executing logical (AND, OR, etc) operations. The 8080 *always* treats it as a parity bit. It was this distinction, in the early MITS BASIC, that made it not run on the Z-80.

The "C" or carry bit is set when the results of an arithmetic operation exceeds the capacity of the register to hold the number. For example, adding 5 to 255 in the accumulator, results in an answer of 4, with the carry bit set.

The carry bit is also set by subtraction operations which resulted in the need to "borrow" a bit.

THE ARITHMETIC INSTRUCTIONS

INR and DCR

INR increments ("adds 1 to") an 8-bit register, and DCR decrements ("subtracts 1 from") an 8-bit register. The syntax is:

INR reg
DCR reg

For example, INR A. Applicable registers are: A, B, C, D, E, H, L, or M. All PSW bits except carry are set by these instructions.

INX and DCX

INX is used to increment and DCX to decrement, a register PAIR. The syntax is:

INX reg pair
DCX reg pair

For example, INX B. Applicable registers are: B, D, H, or SP. Note that *no* PSW *bits* are set by these instructions.

At first, I was "saddened" that I couldn't affect the PSW by INX and DCX. Then, as I began to use the instruction, I found it was frequently to increment or decrement a register pair to point to the next (or previous) byte somewhere in memory. I was counting on, say, ZERO or CARRY bit to be set for some reason *other* than using INX and DCX, so *didn't* want INX or DCX to change it. So it seems reasonable that INX and DCX don't change the PSW.

The INR, DCR, INX, and DCX instructions were pretty simple. Let's take one step up, to where you may add or subtract something other than just 1...

ADI and SUI

ADI and SUI are used to add or subtract an "immediate" 1-byte value to the accumulator. The syntax is:

ADI value
SUI value

"Value" may be any non-negative 8-bit value, such as a decimal or hex number, or an ASCII character.

Similar to many 8080 instructions, the accumulator is not explicitly referenced. However, it is where the addition or subtraction occurs.

To perform simple additions or subtractions on other registers requires either multiple INR or DCR (but only if you wanted to add or subtract 1 or 2 or 3), or temporarily moving the value to the A register first:

```
MOV   A,B      ;B is now in A
ADI   5        ;ADD 5 to it
MOV   B,A      ;move it back
```

Of course, this clobbers the value in A. I could have saved A in, say, C, such as:

```
MOV   C,A      ;save A
MOV   A,B      ;B is now in A
ADI   5        ;ADD 5 to it
MOV   B,A      ;move it back
MOV   A,C      ;restore A
```

I'll show you other ways of saving A later.

You'll note the above lines of code do *not* reflect good comments, since they merely tell *what* the "op codes" are doing, not "what purpose" the instructions fill. For example, "save A" is OK, but for "ADD 5 to it", you would instead say something like "Add 5 to the line length kept in B, to account for the ZIP code". That may be considered to be *too* long for a comment, so you might shorten it to "add 5 for ZIP CODE".

ACI and SBI

These instructions stand for Add with Carry Immediate, and Subtract with Borrow Immediate. Remember earlier when I talked about the Carry bit? Well, now you'll get a chance to USE it. The syntax is:

```
ACI value
SBI value
```

"Value" may be any non-negative 8-bit value, such as a decimal or hex number, or an ASCII character.

Suppose you have a value in the BC register pair, and want to add 1234 hex (I'll call that just 1234H from now on). I could:

```
MOV A,C          ;GET LOW
ADI 34H          ;ADD IN 34
MOV C,A          ;SAVE IT BACK
MOV A,B          ;GET HIGH
ADI 12           ;ADD IN 12
MOV B,A          ;MOVE IT BACK
```

That look OK to you? Well, it *will* work, but not for all data. Suppose BC had the value 238AH in it. Adding 1234H to 238AH, gives 35BEH, which is the correct value. However, if BC had 17F3H in it, the answer would be 2927H, not 2A27H which is the correct answer. Why? Because I didn't account for the CARRY from the addition of 34H to F3H: the carry from the low byte to the high byte.

Here's where the "ACI" instruction fits in. Just like "ADI", it will allow adding the 12H, but it has the added benefit of *adding in the carry*! Thus, the 12H + 17H will result in 2AH, the correct answer, because it added one for the *carry* adding 12H to 17H.

SBI works similarly, but instead of seeing that carry is added, it sees that 1 is subtracted, i.e. that "borrow" is taken care of. The carry bit is used, but after you do a subtraction operation, the carry bit is thought of as a "borrow" bit.

To repeat my example, but this time subtract 1234H from what is in BC, I would:

```
MOV A,C          ;GET LOW VALUE
SUI 34H          ;SUBTRACT 34
MOV C,A          ;SAVE IT BACK
MOV A,B          ;GET HIGH VALUE
SBI 12           ;SUBTRACT 12
MOV B,A          ;MOVE IT BACK
```

ADD and SUB

ADD and SUB are used to add or subtract a value to/from the accumulator. These instructions work on 8-bit values just like the immediate instructions above. However, now the data can come from another register, or from memory. (Remember, register "M" is a special one, and refers to the memory pointed to by the HL register.) The format of the ADD and SUB instructions is:

```
ADD reg
SUB reg
```

where "reg" is one of: A, B, C, D, E H, L, or M.

Bear with me for a few more instructions, I'll get to some useful examples

ADC and SBB

ADC and SBB are used like ADD and SUB, but just like ACI and SBI, they take the carry bit into account. The format of the ADC and SUB instructions is:

```
ADC reg
SBB reg
```

where "reg" is one of: A, B, C, D, E, H, L, or M.

Let's tie in several of the instructions which you have learned about, and write a program to add two values in memory together. Let's make the numbers 3 bytes long.

Suppose the first value is at 100H, and the second value is at 200H, and you want to add these together, storing the result in 200H.

Step one is to get some registers pointing to the data:

```
LXI  D,102H    ;POINT TO LOW
                ;ORDER VALUE
LXI H,202H    ;POINT TO LOW
                ;ORDER VALUE
```

You'll note I pointed to the third byte, not the first. This is because I am assuming the first byte to be the most significant, and the third byte to be the least significant byte.

I now have to load a byte of the first value, and add in the second:

```
LDAX D     ;GET BYTE AT 102H
ADD  M     ;ADD BYTE AT 202H
STAX D     ;STORE SUM BACK
           ;  AT 102H
```

That takes care of the low order byte. I now must point to the "middle" byte:

```
DCX  D     ;BACK UP TO 101H
DCX  H     ;BACK UP TO 201H
```

Then add in the next two bytes. *Note* this time (since you might have gotten a carry from the low-order addition) that you have to use ADC (add with carry) to be sure the carry is accounted for:

```
LDAX D     ;GET BYTE AT 101H
ADC  M     ;ADD BYTE AT 201H
           ;  WITH CARRY
STAX D     ;STORE RESULTS
           ;  BACK
```

Then repeat the DCXs, and repeat the above 3 lines of code again. When this is done, 100H-102H will contain the sum of 200H-202H + the original value in 100H-102H. See *Listing 1* for what it looks like all put together.

I've thrown in a few comments to get you used to seeing them, in hopes that when you write programs, you too will comment them for the sake of readability when you give it to someone else (or even go back to look at it yourself after 6 months).

(continued next page)

In this example, I placed the high-order digits of the number at the lower addresses. This is for "familiarity", i.e. 100 means "one" hundred, "no" tens, and "no" ones. In actuality, it is quite arbitrary where the high or low order is stored, leftmost or rightmost. In the 8080, you learned that the LHLD and SHLD (load HL direct, store HL direct) instructions dealt with data low order byte first.

Later, you'll learn how to code a LOOP, such that you could generalize this program to add from 1 to perhaps 255 byte long numbers, not just 3. The program will actually be smaller, but just slightly more complex to understand, since it uses instructions I haven't covered.

DAA

The DAA instruction, Decimal Adjust for Addition, allows the 8080 to efficiently handle packed decimal addition.

Packed decimal numbers have two decimal digits per byte. Each digit occupies 4 bits. For example, the decimal number 37, shown in binary, is:

```
00110111
```

All of the addition instructions in the 8080, perform BINARY arithmetic.

To add packed decimal numbers, you first add them in binary, then use DAA to "fix them up". Here is an example, with the contents of the accumulator showing how the instructions work. This is actually condensed output from running the three instructions under DDT, the Digital Research Dynamic Debugging Tool that comes with CP/M:

```
E0 A=00 P=0100 MVI  A,23
E0 A=23 P=0102 ADI  19
E1 A=3C P=0104 DAA
E1 A=42 P=0105 ...
```

En represents the state of the "A" (auxiliary carry), bit of the PSW. It is not called "A" under DDT because "A" means the Accumulator.

A=nn shows the contents of the accumulator.

P=nnnn is the Program Counter, i.e. where we are currently executing instructions.

Follow the instructions: first I moved a 23H into the accumulator. (Note that DDT assumes HEX, so I didn't type 23H). I then "added immediate", a 19. Since the ADI instruction took both operands as binary, not decimal, it added 19H + 23H, giving 3CH.

Note that the auxiliary carry bit came on *after* the addition: (see the E1?)

The DAA instruction then "said": "If the value of the right 4 bits of the result is greater than 9 (it is: it is "C" hex, or 12), then add 6 to it. Thus, 3CH + 6, gives 42H. 42 is the decimal sum of 19 and 23, so you see how by "treating" the operands as decimal numbers, performing a "binary" addition, but then "fixing things up" with DAA, you have in effect, performed decimal arithmetic!

The 8080 does not have a "decimal adjust for subtraction" instruction. Several instructions are necessary to accomplish this.

That's it for the 8-bit arithmetic instructions. Now I will go over the one 8080 16 bit arithmetic instruction, and how to combine instructions to do 16 bit subtraction.

DAD

DAD performs a 16 bit addition. Specifically, the contents of register pair BC, DE, or HL, or the contents of the stack pointer (SP) may be added to HL. The format of the DAD instruction is:

```
DAD rp
```
rp may be B, D, H, or SP.

Unlike the previous arithmetic instructions which affected all of the PSW bits, DAD affects only the carry bit. To reinforce that thought by putting it another way: You *cannot* use the PSW to test for zero, minus, or parity after doing a DAD, as DAD doesn't change these status bits.

You can, however, code two instructions to test the value of HL, for example to see if it is zero. Here's how:

```
MOV  A,H ;get high answer
ORA  L   ;combine with low
```

I jumped ahead there a bit, with the ORA instruction, which will be covered later. In brief, it "OR"s bits from the L register, with the bits in A. Thus, *only* if *no* bits were on in EITHER H (which I moved to A) or L (which I ORed with A), will the zero indicator be set.

To test if the results of DAD is minus:

```
MOV  A,H ;get high answer
ORA  A   ;set minusto
         ;value in H
         ;     (now in A)
```

The sign status bit (S bit of the PSW) is now set to the value in HL. Note this does not properly reflect zero, since the zero status bit would be set only based on H, and L might or might not be zero.

Note that I only looked at the H register when determining if HL was negative. This is because in a binary number, only 1 bit, the leftmost, is used for the sign bit.

Next time, more on the uses and "tricks" of "DAD", then the "logical" instructions - for ANDing, ORing, etc.

If you have any comments on this tutorial — any suggestions, complaints, or questions, please write to me:

Ward Christensen
% Lifelines
1651 Third Avenue
New York, N.Y. 10028

(See next page for Listing 1)

```
;
;ROUTINE TO ADD 2 3-DIGIT NUMBERS,
;STORED AT 100H AND 200H, WITH
;THE ANSWER PLACED AT 200H
;
        LXI     D,102H  ;POINT TO LOW ORDER VALUE
        LXI     H,202H  ;POINT TO HIGH ORDER VALUE
;
        LDAX    D       ;GET BYTE AT 102H
        ADD     M       ;ADD BYTE AT 202H
        STAX    D       ;STORE SUM BACK
;
        DCX     D       ;BACK UP TO 101H
        DCX     H       ;BACK UP TO 201H
;
        LDAX    D       ;GET BYTE AT 101H
        ADC     M       ;ADD BYTE AT 201H
                        ;       WITH CARRY
        STAX    D       ;STORE RESULTS BACK
;
        DCX     D       ;BACK UP TO 100H
        DCX     H       ;BACK UP TO 200H
;
        LDAX    D       ;GET BYTE AT 100H
        ADC     M       ;ADD BYTE AT 200H
                        ;       WITH CARRY
        STAX    D       ;STORE RESULTS BACK
```

LISTING 1

Notes on dBASE II

Michael Olfe

Editor's Note: We thank Michael Olfe for his contributions this month and *last* month, when he graciously shared his notes on dBASE II with us.

I have found the following bugs (or features) in dBASE II:

1 .The addendum to the manual states that in the command "DISPLAY FILES LIKE" you may use "*" and "?" to specify ambiguous filenames, as in CP/M-80. But "LIST FILES LIKE S*.DBF" incorrectly return "NO FILE".

2 .A "$" in a picture clause causes the number entered in response to get to be always set at zero; e.g., if num is a 12 digit number with 2 decimal places

@ 0,0 say "Number " get num picture "$9999999.99"

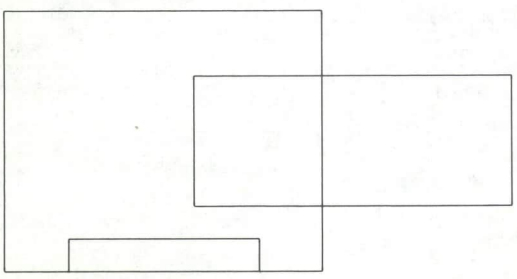
will always set num to 0.00, no matter what number is entered.

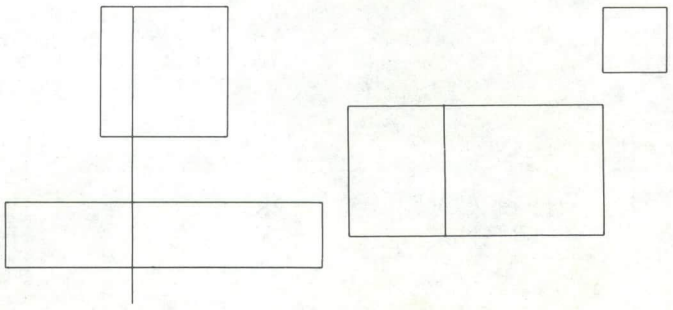
3 ."STR(number,length,decimal-places)" returns asterisks if length is less than the position of the first digit, and returns the number right-justified in the field if the length is greater than the number of digits. Whether this is a bug or a feature is a matter for discussion.

5 .dBASE II does very little error-checking and can therefore be very unforgiving:

STORE dd+'HI to dd

where dd is some string variable and you forget the closing quote, will either hang the system or reboot.





The CP/M Users Group: Volumes 65-75 Catalogues and Abstracts

The catalogs and abstracts are presented here pretty much as they exist on the volumes. Many of the SIG/M-originated CPMUG disks do not have space for abstracts, and that's why you won't find the abstracts here. Volumes 71 through 74 first came from the Pascal/Z Users Group, and Charlie Foster, their Director, wrote the abstracts for these volumes. CPMUG thanks The SIG/M Users Group for their contributions.

CPMUG Volume 65

a) MITS to CP/M file conversion system
b) HELP file system
c) related system support programs
d) FIG-FORTH 1.1 system

		-CATALOG.065	CONTENTS OF CPMUG VOLUME 65
		UGFORM.LIB	SUBMITTAL FORM
		SIG/M.LIB	SUBMITTAL FORM
VOL.#	NAME	SIZE	COMMENTS
65.1	MITSCNVT.ASM	28K	MITS to CP/M file conversion
65.2	MITSCNVT.COM	7K	
65.3	MITSCNVT.CPM	3K	
65.4	MITSCNVT.DOC	7K	
65.5	MITSCNVT.HEX	8K	
65.6	ABORTSUB.COM	2K	HELP file system
65.7	HELP.ASM	13K	
65.8	SYSLIB.HLP	37K	
65.9	SYSLIB.REL	9K	
65.10	HELP.COM	2K	
65.11	MASM.SUB	1K	
65.12	PHELP.COM	2K	
65.13	PHELP.MAC	6K	
65.14	HELP.DOC	4K	
65.15	SD-2/6.ASM	18K	Updated SUPER DIRECTORY display
65.16	SD.COM	2K	
65.17	ENTAB.ASM	6K	Space suppression program
65.18	ENTAB.COM	1K	
65.19	UTIL.FOR	2K	FORTRAN array handling subroutine
65.20	LOOPBAK1.BAS	2K	PMMI loop back test
65.21	FORTH11.ASM	40K	FIG-FORTH version 1.1
65.22	FORTH11.COM	23K	
65.23	FORTH11.DOC	4K	

CPMUG Volume 66

HELP file system on major system level software

		-CATALOG.066	CONTENTS OF CPMUG VOLUME 66
		SIG/M.LIB	SUBMITTAL FORM
		UGFORM.LIB	SUBMITTAL FORM
VOL.#	NAME	SIZE	COMMENTS
66.1	HELP.COM	2K	HELP file system describing
66.2	HELP.HLP	7K	HELP file system
66.3	CPM.HLP	31K	CP/M 1.4

66.4	CPM2.HLP	37K	CP/M 2.2
66.5	ASM.HLP	4K	CP/M 1.4 ASM (assembler)
66.6	ASM2.HLP	4K	CP/M 2.2 ASM (assembler)
66.7	MAC.HLP	8K	CP/M MAC (macro assembler)
66.8	MBASIC.HLP	21K	Microsoft BASIC
66.9	CBASIC.HLP	14K	CBASIC
66.10	EBASIC.HLP	12K	BASIC-E
66.11	CBASIC2.HLP	22K	CBASIC-2
66.12	MASM.HLP	8K	MACRO-80 (M80)
66.13	ALGOLM.HLP	13K	ALGOL-M
66.14	C.HLP	17K	BDS 'C'
66.15	FORTRAN.HLP	7K	Microsoft FORTRAN
66.16	PASCAL.HLP	10K	PASCAL/MT

CPMUG Volume 67

Documentation catalog of CPMUG volumes 1-42 and SIG/M volumes 1-3 as published by the NYACC. (Note: SIG/M volumes 1 and 2 are available as CPMUG volumes 55 and 56, and CPMUG volume 57 corresponds to SIG/M volume 11, which is the new, improved version of SIG/M volume 3).

SIG/M.LIB	SUBMITTAL FORM
UGFORM.LIB	SUBMITTAL FORM

This volume contains the various *.DOC, ABSTRACT.*, READ.ME, etc files as contained in the CPMUG and SIG/M volumes. This was compiled and published by the NYACC to facilitate ready reference of public domain software for the microhobbyists. SIG/M volume 12 is the corresponding cross reference to this release.

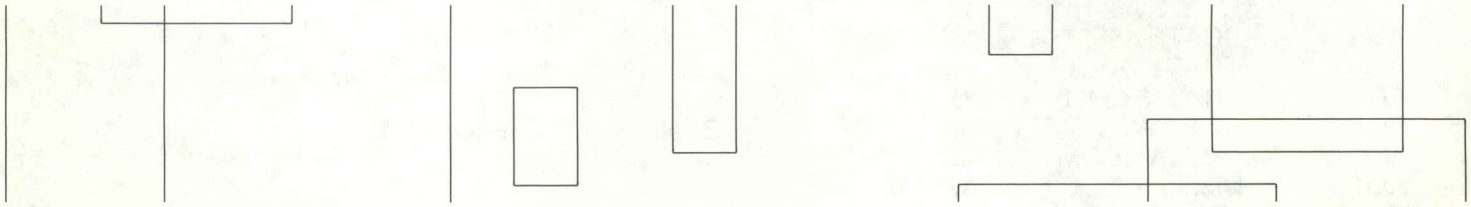
CATALOG.nnn refers to a CPMUG release volume nnn.
SIGMLOG.nnn refers to a SIG/M release volume nnn.

CPMUG Volume 68

miscellaneous CP/M utilities

-CATALOG.068	CONTENTS OF CPMUG VOLUME 068	
SIG/M.LIB	SUBMITTAL FORM	
UGFORM.LIB	SUBMITTAL FORM	

VOL.#	NAME	SIZE	COMMENTS
68.1	BYE67.ASM	39K	Remote console program for PMMI
68.2	BYE67.DOC	12K	
68.3	MLIST42.ASM	12K	Multiple file list utility
68.4	DU-V75.OBJ	6K	Disk utility system
68.5	DU-V75.DOC	5K	
68.6	DU-V75.ASM	40K	
68.7	FINDBD42.ASM	33K	Creates file of badspots
68.8	APLMODEM.ASM	36K	CP/M file transfer for Apple 2 using DC Hayes Micromodem 2
68.9	COMAND.LIB	7K	Console string processor
68.10	COMBINE.ASM	7K	Merges multiple files
68.11	FILE-XT2.ASM	8K	System disk utility display
68.12	MNEMON21.ASM	28K	Multi-pass CP/M memory tester



(continued next page)

CPMUG Volume 69

Miscellaneous CP/M utilities

	-CATALOG.069	CONTENTS OF CPMUG VOLUME 69
	-CATALOG.ACK	Acknowledgement file
	SIG/M.LIB	SUBMITTAL FORM
	UGFORM.LIB	SUBMITTAL FORM

VOL.#	NAME	SIZE	COMMENTS
69.1	SCRAMBLE.DOC	2K	Command used to encode a CP/M file
69.2	SCRAMBLE.ASM	6K	
69.3	SORTV.DOC	2K	Sort program for variable length records
69.4	SORTV-12.ASM	13K	
69.5	TAG2.ASM	6K	Set/reset display the "no copy" flag
69.6	MNEXEC.COM	3K	MicroNet executive
69.7	MNOVRLAY.ASM	4K	Overlay for MicroNet executive
69.8	WHICH/1.ASM	4K	Returns size and version of CP/M
69.9	STATUS.ASM	8K	Present various systems information under 2.2
69.10	NEWQCAT.ASM	5K	Quick catalog routine
69.11	WORM8/8.ASM	7K	Memory test
69.12	TFX12/18.ASM	21K	CP/M to CP/M file transfer utility
69.13	XMODEM41.ASM	29K	Remote CP/M to CP/M file transfer
69.14	MOVPATCH.ASM	3K	Modifies MOVCPM for remote access
69.15	21BIOS.ASM	48K	New BIOS for CP/M 2.0
69.16	21BIOS.DOC	3K	
69.17	21BOOT.ASM	4K	
69.18	MACRO.LIB	18K	
69.19	NPGEN.ASM	9K	
69.20	SPCLMAC.LIB	4K	
69.21	XDIR.ASM	13K	Updated CP/M directory display
69.22	XDIR.COM	2K	

CPMUG Volume 70

Miscellaneous CP/M utilities.

	-CATALOG.070	CONTENTS OF CPMUG VOLUME 70.
	-CATALOG.ACK	Volume 70 acknowledgement file.
	SIG/M.LIB	SUBMITTAL FORM
	UGFORM.LIB	SUBMITTAL FORM

VOL.#	NAME	SIZE	COMMENTS
70.1	2411DUMP.ASM	18K	iCOM microfloppy utility
70.2	2411DUMP.COM	3K	
70.3	MFMACRO.LIB	17K	
70.4	3812DUMP.ASM	21K	iCOM floppy utility
70.5	3812DUMP.COM	4K	
70.6	DDMACRO.LIB	18K	
70.7	AREACODE.ASM	14K	Region and state area code search
70.8	AREACODE.COM	7K	
70.9	BANZAI.ASM	9K	Copyright notice protect
70.10	CAT2.ASM	6K	Master catalog system for CP/M 2.X
70.11	FMAP3.ASM	7K	
70.12	UCAT2.ASM	8K	
70.13	CHAT15.ASM	6K	Chat with local remote CP/M operator
70.14	CHAT15.COM	1K	
70.15	DISPLAYP.ASM	3K	Display facilities of ED.COM
70.16	DISPLAY.COM	2K	
70.17	DISPLAY.DOC	3K	
70.18	FILE-EXT.ASM	7K	Display disk directory including hex data

70.19	FILE-EXT.COM	1K	
70.20	FILE-XT2.ASM	8K	Same as FILE-EXT.ASM with 2.X compatibility
70.21	FILE-XT2.COM	1K	
70.22	LOOK.ASM	7K	Searches for 1-9 byte sequence in memory
70.23	LOOK.COM	1K	
70.24	MACTIME.ASM	7K	Patch for real time clock in MAC.COM
70.25	ASMTIME.ASM	7K	Patch for real time clock in ASM.COM
70.26	STATTIME.ASM	6K	Patch for real time clock in STAT.COM
70.27	SAP.ASM	6K	Updated directory sort for CP/M 2.X
70.28	SAP.COM	1K	
70.29	XLOOK.ASM	13K	Disk/examine/modify utility

CPMUG Volume 71

Miscellaneous Pascal Z programs. Original materials from Pascal Z User Group volume 1.

-CATALOG.071	CONTENTS OF CPMUG VOLUME 71	
-CATALOG.ACK	Acknowledgement file	
ABSTRACT.071	Comments file	
CRCKFILE.071	CRC of volume 71	
SIG/M.LIB	SUBMITTAL FORM	
UGFORM.LIB	SUBMITTAL FORM	

VOL.#	NAME	SIZE	COMMENTS
71.1	AUTOBOOT.ASM	5K	Autoboot on CP/M cold start
71.2	AUTOBOOT.COM	1K	
71.3	LINEARP.PAS	11K	Simplex algorithm to minimize
71.4	LINEARP.COM	21K	a cost function to constraints
71.5	VARIANT.PAS	1K	Demo for variant records
71.6	VARIANT.COM	5K	
71.7	REVERSE.PAS	2K	Demo for linked lists
71.8	REVERSE.COM	6K	
71.9	EDITFILE.PAS	9K	Adapted from S-100 Microsystems
71.10	EDITFILE.COM	13K	
71.11	RT.PAS	4K	Demo program for non-text files
71.12	RT.COM	8K	
71.13	STARS.PAS	6K	Game
71.14	STARS.COM	7K	
71.15	ADDN.PAS	1K	Simple demo to add two numbers
71.16	ADDN.COM	6K	
71.17	ZMNEMONS.DOC	13K	Programming aid
71.18	TRIAN.PAS	1K	Demo on FOR loops
71.19	TRIAN.COM	4K	
71.20	CONCHAR.PAS	6K	Utility for command line input
71.21	SCAN2X.PAS	5K	File READ evaluation
71.22	SCAN2X.COM	8K	
71.23	STRDEMO.PAS	14K	Demo on string functions
71.24	STRLIB.DOC	4K	Part of STRDEMO.PAS
71.25	LONG.PAS	3K	Demo to string words together
71.26	LONG.COM	7K	
71.27	MAKEREL.DOC	8K	Convert REL. files from function blks
71.28	LIOS.ASM	7K	Novice utility
71.29	CONVERT.PAS	1K	Convert gas in liters
71.30	CONVERT.COM	6K	
71.31	COMPARE.!!!	11K	Compare source code files from UCSD
71.32	DUMP.ASM	23K	Expanded CP/M DUMP program
71.33	LSTR.PAS	2K	Generates a line of various length

(continued next page)

Since this is the very first disk distributed by the Z-users group, it seems appropriate that I give you a little history of just how this all started.

When I started in the microcomputer arena, two years ago, I didn't know anything about anything. But I am a compulsive reader and spent much, much time reading what the experts had to say. I then took their advice, pocketbook permitting. That meant that I was going to have a S-100 bus, Z-80, dual 8" floppies, 64k of memory, all running under CP/M. Then came languages. At first, I went along with BASIC. However, it didn't really turn me on so I just collected and drifted from one language to another. The experts said the greatest thing around was Pascal. So when UCSD Pascal was offered to our local Computer club I chipped in and got a copy. Out of forty members who got that Pascal, only three of us got it up and running. None of us used it after we got it up and running. Mostly, because we were all CP/M oriented. So we gave up on Pascal. Meanwhile, I was staying in touch with various friends around the country and one of them told me about Ithaca Intersystems' Pascal/Z. It was only version #1 but I loved it from the very first. I liked the idea, and in spite of the early bugs, I had a ball with it. So I bought version 2.0 and liked that even better.

Many programs later I was a confirmed Pascal freak. So much so that when the SF Fair rolled around, I looked up Steve Edleman. He was the guiding light for Ithaca Intersystems and we seemed to have some common interests. To make a short story, shorter, Steve took a couple of my suggestions to heart and in May of 1980 announced to the world that Ithaca Intersystems would actively support a Pascal/Z user group. I offered to direct it and keep it running, so we made a deal.

The intent of this group is to assist Pascal/Z, Z80 and Z8000 software dissemination. That way, we'll all get more programs to play around with and have fun too.

For the stranger who happens to pick up this disk, Pascal/Z requires a Z80 CPU, CPM and 56k of usable memory. Most everything else is up to the owner of the system. I intend to publish a flyer bimonthly so the bugs, fixes and any other interesting items can be passed on. Donations are certainly needed. I will try to edit and test all programs sent, give the author plenty of credit and spread his program all over the world. Its like a chain letter, you send in one, and get a hundred back. So don't be lazy, send it in. Someone, somewhere will be glad you did. And you will too.

CPMUG Volume 72

Original materials from Pascal Z User Group volume 2

PCE System Monitor

-CATALOG.072		CONTENTS OF CPMUG VOLUME 72
SIG/M.LIB		SUBMITTAL FORM
UGFORM.LIB		SUBMITTAL FORM
CRCKFILE.072		CRC of CPMUG Volume 72
ABSTRACT.072		comments on PCE System Monitor

VOL.#	NAME	SIZE	COMMENTS
72.1	SYSMON.DOC	103K	Complete Manual of System Monitor
72.2	SYSMONA.Z80	2K	Central subsystem module
72.3	ASPM1.Z80	11K	Command subsystem, module one
72.4	ASPM2.Z80	12K	Command subsystem, module two
72.5	VECTOR.Z80	3K	Intel I/O standard vector subsystem
72.6	CONSOL.Z80	14K	Console subsystem
72.7	DISK.Z80	5K	Floppy disk subsystem, it ties CPM to the monitor so that both operate as an integrated whole
72.8	CASS.Z80	8K	Cassette subsystem, controls the Dajen/Teletek UCRI, includes Zapple compatile (RI) & (PO)
72.9	BMGEN.Z80	1K	Bit map generator
72.10	BMGEN.COM	1K	
72.11	LOADER.Z80	10K	Static loader for system
72.12	LOADER.COM	3K	
72.13	SYS.COM	16K	Sys Monitor boot-up program
72.14	ONE.HEX	14K	Used with bringing up bit map
72.15	ZERO.HEX	14K	Used with bringing up bit map
72.16	COPYTT.Z80	1K	Sample command, similiar to a COM file, this one is a Disk subsystem test utility
72.17	COPYTT.SMC	1K	

I was very fortunate this month. I was able to talk a company named PCE SYSTEMS into donating this complete disk of Z80 software. They had paid a programmer to write it exclusively for them but their microcomputer products evolved so rapidly that by the time it was ready, they didn't need it anymore. Nowadays they boot directly into CP/M. So we lucked out, we now have the complete source and the manual for it, all on this disk. You can bet, I'll be trying to get other manufacturers to do the same thing with their outmoded software.

This disk is so large however that I couldn't edit/verify to see if this stuff works but I have been assured by PCE SYSTEMS that it does. So whoever gets it up and running I would appreciate it if they would drop me a line outlining their views on it.

CPMUG Volume 73

Miscellaneous Pascal Z utilities. Original materials from Pascal Z User Group volume 3.

		-CATALOG.073	CONTENTS OF CPMUG VOLUME 73
		ABSTRACT.073	COMMENTS ON CPMUG VOLUME 73
		CRCKFILE.073	CRC OF CPMUG VOLUME 73
VOL.#	NAME	SIZE	COMMENTS
73.1	AUTHOR.PAS	10K	General purpose "keyword in text"
73.2	AUTHOR.COM	11K	
73.2a	ADDRESS	1K	
73.2b	BYTE.MAG	1K	
73.2c	S100.MAG	1K	
73.3	CALC.PAS	12K	Calculator mode program
73.4	CALC.COM	16K	
73.5	CPLOT.PAS	2K	Simple banner program
73.6	CPLOT.COM	5K	
73.7	DELAY.PAS	1K	General Pascal Z utilities
73.8	DELAY.REL	1K	
73.9	DELAY.SRC	1K	
73.10	INPORT.REL	1K	
73.11	INPORT.SRC	1K	
73.12	OUTPORT.REL	1K	
73.13	OUTPORT.SRC	1K	
73.14	KEYIN.REL	1K	
73.15	KEYIN.SRC	1K	
73.16	NAD.PAS	10K	Permuted keyword index program
73.17	NAD.COM		11K
73.18	TDIABLO.MAC	6K	Driver for Diablo w/Teletek FDC-1
73.19	TDIABLO.COM	1K	
73.20	DIABLO.Z80	5K	Driver for Diablo w/SD Systems S100
73.21	DIABLO.COM	1K	
73.22	RANDOM.PAS	2K	Fibonacci random number generator
73.23	RANDOM.REL	1K	
73.24	RANDOM.SRC	4K	
73.25	REBOOT.COM	1K	Rebooting desired file
73.26	STRLIB.LIB	12K	Pascal Z library
73.27	WUMPUS.PAS	12K	Wumpus game in Pascal Z
73.28	WUMPUS.COM	15K	
73.29	WUMPUS.DOC	4K	
73.30	CAVE0	1K	
73.31	CAVE1	1K	
73.32	CAVE2	1K	
73.33	CAVE4	1K	
73.34	CAVE5	1K	
73.35	ENTRY.PAS	6K	Creating SRC files
73.36	ENTRY.COM	9K	
73.37	ENTRY.DOC	6K	
73.38	PEEK.PAS	2K	Peek and Poke in Pascal Z
73.39	POWERI.PAS	1K	Demo program on powers of numbers
73.40	POWERI.COM	7K	
73.41	POWERI.REL	2K	

(continued next page)

73.42	RDR.PAS	7K	An alpha-numeric numbers conversion program
73.43	DU.Z80	29K	Updated disk utility using Z80 code
73.44	DU.COM	4K	
73.45	DU.DOC	1K	

Things have been going so fast that the disks have been rolling out before any feed-back has been received. So as of now I don't know if anyone likes what we got or not. But as long as you folks will send me stuff I will edit it and publish it. It looks like volume #4 should be a little slower. But I have some people I haven't asked yet for donations, so who knows what I'll be able to drag out of the woodwork.

I was going to have a Read.Me file on each disk but I found that unless I had a lot to say it was just not needed. So unless something special comes along I'll just stick to this format.

AUTHOR.PAS/COM–A general purpose "keyword in context" program. Includes samples.

CALC.PAS–Here is your built in calculator adapted to Pascal/Z. The number crunchers among us should take this and expand it to its limit. But it's got good potential as is, now just patch it sothe results go to either the printer or the disk and then you'll have a permanent record.

CPLOT.PAS/COM–A simple banner idea but useful for simple designs, just to be different.

DELAY.PAS/REL/SRC, INPORT.REL/SRC, OUTPORT.REL/SRC, KEYIN.REL/SRC–Ray Penley tells me I goofed by not including these in with the volume 1.So here they are and let me know of any other goofs.

NAD.PAS/COM–A general purpose "Permuted keyword index" program. A good start but needs to be upgraded to become classy.

TDIABLO.MAC/COM–Driver for Diablo which works for the Teletek FDC-1 board.

DIABLO.Z80/COM–Driver for Diablo which works for the SD System S-100.

RANDOM.PAS/COM–This random generator implements the Fibonacci series approach.

REBOOT.COM–In volume #1 I included a Autoboot program. This is an example of ideas breeding ideas. Tim Oleseo saw it and said there is a better way. In this one you type—-
REBOOT yourfilesdesire cr
and that's it. To remove an entry—
REBOOT cr
Simple huh!

STRLIB.LIB–The begining of a library, some good stuff.

WUMPUS.PAS/COM/DOC–Nothing need to be said about this CAVES game but it is Pascal 3.0 and also it has Caves that can be added to, so for you game freaks, get hot and put your ideas in some strange caves.

ENTRY.PAS/COM/DOC–Tutorial on how to make external SRC files with examples. Needed by all of us beginners.

PEEK.PAS–A peek and poke in Pascal/Z yet. Who knows you might need it.

POWERI.PAS/REL/COM–A demo, but useful program for powers of numbers.

RDR.PAS–Alpha-numeric numbers conversion program.

CPMUG Volume 74

Miscellaneous Pascal Z utilities. Original material from Pascal Z User Group volume 4.

-CATALOG.074	CONTENTS OF CPMUG VOLUME 74
UGFORM.LIB	SUBMITTAL FORM
ABSTRACT.074	Comments
CRCKFILE.074	CRC of Volume 74

VOL.#	NAME	SIZE	COMMENTS
74.1	CFIO.LIB	2K	Program to open files
74.2	CONCORD.COM	10K	Program that builds an alphabetical
74.3	CONCORD.PAS	11K	listing of distinct words
74.4	COSINE.PAS	2K	Cosine program
74.5	COSINE.REL	2K	
74.6	COSINE.SRC	6K	
74.7	GEN5.COM	8K	Demo on accessing CP/M files
74.8	GEN5.PAS	5K	
74.9	GRAPH1.COM	10K	Demo on creating a graphic representation
74.10	GRAPH1.PAS	1K	of a function.
74.11	ISORTV1.COM	6K	Insertion sort with linked list
74.12	ISORTV1.PAS	4K	
74.13	LINENOS.COM	8K	Keep track of text lines
74.14	LINENOS.PAS	4K	
74.15	POINT.COM	6K	Demo on the use of pointers
74.16	POINT.PAS	2K	
74.17	RANDOM.MAC	4K	"Professional" random number generator
74.18	RANDOM.PAS	1K	Simple random number generator
74.19	SINCOS.SRC	5K	Sine/cosine utility
74.20	SINCOS.REL	1K	
74.21	STRLIB.DOC	5K	Ray Penley's latest updated string lib.
74.74	STRLIB.LIB	13K	
74.23	TRIG4.COM	9K	Simple Demo program that builds a
74.24	TRIG4.PAS	2K	short Trig chart
74.25	XREFG2.COM	13K	A binary tree type cross-ref generator
74.26	XREFG2.PAS	18K	
74.27	ZPTEX.COM	8K	Very simple text formatter
74.28	ZPTEX.PAS	8K	
74.29	LESSON4.	5K	
74.30	RECIPE.COM	12K	Recipe program
74.31	RECIPE.PAS	17K	
74.32	RECIPE.MST	1K	
74.33	RCPDAT.YYY	1K	
74.34	FCLOSE.COM	8K	Three different ways to close a file
74.35	FCLOSE.PAS	4K	in Pascal Z.

Frankly, I thought that this disk would take a little while to get together. But Ray Penley, Bob Harsch and a mild donation by me brought this volume together so fast I almost got overloaded. The utility level is increasing also. There are some very good programs on this disk that are of professional quality. It seems that we are a success.

CFIO.LIB–Add this to your library. It is a program to help you open files, really nice. By the way, the CF means Connect Files.

CONCORD.COM, .PAS, .CCD–This a program that builds an alphabetical listing of all the distinct words which appear in a text file. (Useful for a programmer in locating garbage variables in a program). Makes heavy use of pointers and the data string. Some good ideas here.

COSINE.PAS,. REL, .SRC–Before the bug got fixed, this was our subsituted Cosine program. It is now new and improved.

GEN5.COM, .PAS–A demo on accessing CP/M files and reading them back. Instructional for the novice.

GRAPH1.COM, .PAS–A demo on creating a graphic representation of a function.

ISORTV1.COM, .PAS–An insertion sort with linked list.This program can be easily adapted to sort single characters, integer numbers, real numbers, months or any item that can be ordered.

LINENOS.COM, .PAS–Many times it is difficult to keep track of text lines. This program will number your lines for you so it will be easy to count or find.

POINT.COM, .PAS–A demo on the use of pointers taken from page 49 of the Pascal/Z manual.

RANDOM.MAC–A professional random number generator for your library. Two years of testing went into this one, should be good.

(continued next page)

RANDOM.PAS Something simple Bob threw in.

SINCOS.SRC, .REL–A debugged sine/cosine that works.

STRLIB.DOC, .LIB–Ray Penley's latest updated string lib.

TRIG4.COM, .PAS–A simple Demo program that builds a short Trig chart.

XREFG2.COM, .PAS–This is a binary tree type cross-ref generator. Very useful and very instructive.

CPMUG Volume 75

MBASIC disassembler
Date Routines
Miscellaneous utilities
Original materials from Pascal Z Users Group volume 5

-CATALOG.075	Contents of CPMUG Volume 75
ABSTRACT.075	Comments
CRCKFILE.075	CRC of CPMUG Volume 75
SIG/M.LIB	SUBMITTAL FORM
UGFORM.LIB	SUBMITTAL FORM

VOL.#	NAME	COMMENTS
75.1	EXPO.PAS	Demo on the use of exponents
75.2	EXPO.COM	
75.3	COMPARE.DOC	UCSD program Compare documentation
75.4	DISASMB.DOC	Disassembler in Microsoft BASIC

Related modules

8085	LST	8085	TAB	XREF	BAS
ZILOG	LST	ZILOG	TAB	XREF	SUB
INTEL	LST	INTEL	TAB	TDL	LST
TDL	TAB	LST8085	BAS	LSTINTEL	BAS
LSTTDL	BAS	LSTZILOG	BAS	TAB8085	BAS
TABINTEL	BAS	TABTDL	BAS	TABZILOG	BAS
DISASMB	BAS				

75.5. DATE.DOC — Date routines

Related modules

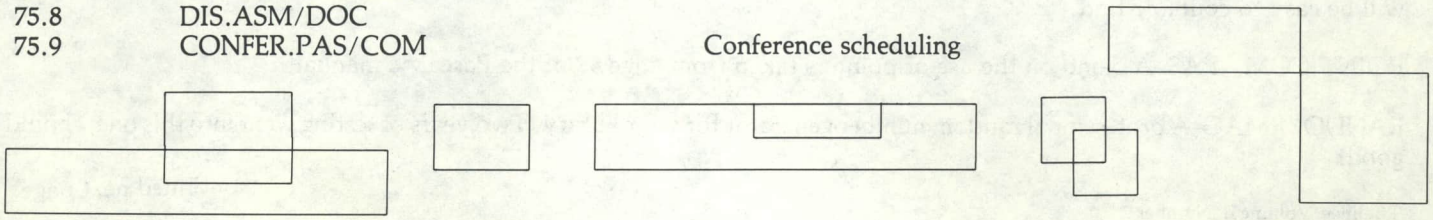
PROMT	LIB	MAKEDATE	LIB	RMAKEDA	LIB
BRKDATE	LIB	DASTRLON	LIB	DASTRSH	LIB
DASTRFIX	LIB	DATE	PAS	DATEFUNC LIB	

75.6 MISC.DOC — Miscellaneous routines

Related modules

GETINT	LIB	CAPCHAR	LIB	PADSTR	LIB
CAPSTR	LIB	DEPAD	LIB	STRVAL	LIB

75.7 HANOI COM/PAS
75.8 DIS.ASM/DOC
75.9 CONFER.PAS/COM — Conference scheduling



CPMUG Volume 75

I am starting to get feedback from the membership now and this disk is the beginning. I am very happy to pass on most anything that anyone writes. I only ask that it work. If your program has special requirements be sure and include that information. At first, I was going to only include source on these disks so that I could get more items on them. But, so many people are either just starting out and don't have enough knowledge or can't afford everything that may be necessary. So that is why I put both source and running programs on these disks. That way the advanced can modify to their hearts' content and yet the novice can use the programs.

Kenneth Kuller of Eagan, Maine submitted these programs.

EXPO.PAS/COM–He wanted a benchmark program so he wrote this. It is a good Demo on the use of exponents.

COMPARE.DOC–Ken feels, as I do, that credit should be given when due. On disk #1 [CPMUG Vol. 71] there is a UCSD program called Compare that needed to be converted to Pascal/Z. I didn't receive the .DOC file with the program but Ken found it. So here it is and the proper credits are included.

This next large group was submitted by Scott Custin of
Washington, DC.

DISASMB.DOC–Read this first because there are a lot of small programs associated with this disassembler. It is in Microsoft BASIC, Version 5.1, which I would not normally use; but since it does disassemble TDL plus Z80 plus 8080 plus 8085 I figured someone, somewhere would be tickled to get this.

8085	LST	8085	TAB	XREF	BAS
ZILOG	LST	ZILOG	TAB	XREF	SUB
INTEL	LST	INTEL	TAB	TDL	LST
TDL	TAB	LST8085	BAS	LSTINTEL	BAS
LSTTDL	BAS	LSTZILOG	BAS	TAB8085	BAS
TABINTEL	BAS	TABTDL	BAS	TABZILOG	BAS
DISASMB	BAS				

DATE.DOC–Again, read this first since Scott has included several programs, all dealing with dates. These routines treat a date as one of a series of consecutive integers. The concept is similar to Julian dates, used by a number of "BIG" computer programs, and has a number of advantages over storing the month, day and year separately.

PROMT	LIB	MAKEDATE	LIB	RMAKEDA	LIB
BRKDATE	LIB	DASTRLON	LIB	DASTRSH	LIB
DASTRFIX	LIB	DATE	PAS	DATEFUNC	LIB

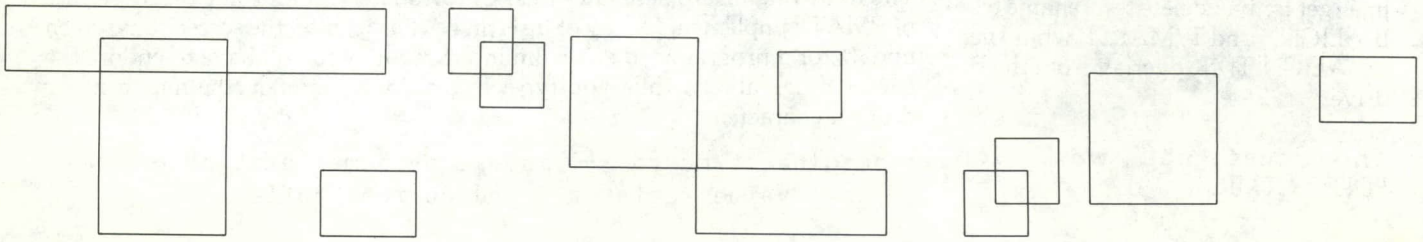
MISC.DOC–Yep, read this first. Scott tossed in some odds and ends. You have to give Scott credit, he sure has been creative. Keep this up folks, and we will have a excellent library.

GETINT	LIB	CAPCHAR	LIB	PADSTR	LIB
CAPSTR	LIB	DEPAD	LIB	STRVAL	LIB

HANOI.COM,.PAS–I included this not for its game value but it is an excellent Demo on recursive procedures. If you can see how this works (it's deep) you'll certainly understand local versus global variables.

DIS.ASM, .DOC–This was given to me for our TDL use. However, the three or four assemblers I had available would not assemble it without massive errors. So if anyone does make a COM file that works good be sure and send me a copy. Let me know what assembler you used also.

CONFER.PAS, .COM–Just to show you how long Ray has been chipping away at Pascal/Z, I thought I would include this old one. It does have some useful ideas and can be used as is. It sets up a schedule for a conference.



Bugs

COBOL-80

Any ACCEPT after CALLing a subroutine will cause incoherent results. The authors say that this causes the first four bytes of the CRT driver to be gobbled up. The CRT driver should be begun with four NOP's.

MailMerge
Version 3.0

A bug occurs in operation of the ".av" command. MailMerge command files which contain ".av" will cause WordStar to stop while printing a page and ask the value, when the page should be finished first.

The fix is to avoid ".av" ; it should not be used in files which are called by other files for repetitive processing. Use the page option in WordStar to make it pause after each page. When asked "PAUSE after printing each page?", answer "Y" (yes).

T/MAKER II Version 2.3.2 On Small Capacity Disks

Michael Olfe

Users of T/MAKER who have disk capacities of less than 100K may have to put some of the .TMK overlays on another disk drive. This will cause T/MAKER no difficulty if, when the function is invoked, its name is preceded by a drive designation. Unfortunately this very desirable feature is not mentioned in the manual.

For example, if ALIGN.TMK were on drive B:, and all the other T/MAKER files were on drive A:, the user could construct and execute the command line "get tst.bak delete test rename test B:ALIGN", and T/MAKER would get the ALIGN.TMK overlay from the B: drive.

This does not work for EDITOR.TMK.

Special Sauce and a Sesame Bun OR Macros for PMATE

Michael Olfe

If you're holding onto that big Macro that makes PMATE look like Complete EMACS, or the one that translates 8008 code into Ada source, send them in to the Macro of the Month contest. Until then, the following crumbs will have to suffice.

I used Wordmaster and WordStar for a long time before meeting PMATE, so naturally remapped the PMATE keyboard to look like Wordmaster/WordStar for cursor motion and deletions. This made it even more essential to have a help function, since virtually all the keys were remapped and I could no longer use the manual as a reference for "instant" commands. This macro reads the PMATE.HLP file, which is a list of control keys and what they do (ideally it is 21 lines long so that it fills the screen), into the text current edit buffer and pauses to allow you to read it. On a keystroke the help text is deleted from the buffer and you can go on editing. Users of Wordmaster will feel quite at home.

```
;  help macro  -- just invoke by name , no parameters
;  uses value register 0 and current buffer

 txipmate.hlp$#v0ts^w$gPress key to continue$#@0d
```

Two other macros have proved very useful strip comments and count characters. The stripping macro in the manual does not delete spaces preceding a semicolon or blank lines. The macro below will do this, and typically reduces a heavily-commented source file to one-third its former size. The counting macro is for matching opening and closing parentheses, braces, brackets, and begin-end pairs.

```
; strip macro -- invoke by name, no parameters

a[es;$@e_-m$-s^N   $mki
$@t=0]a[es

$@e_-2mk]
agDone$

; count macro -- if macro is permanent macro x,
;  and you are counting ´}´, invoke .x}$

alqa 0v0 [es^AA$va0@e_](@0-1)
gTotal # is above. Press key.$@0
```

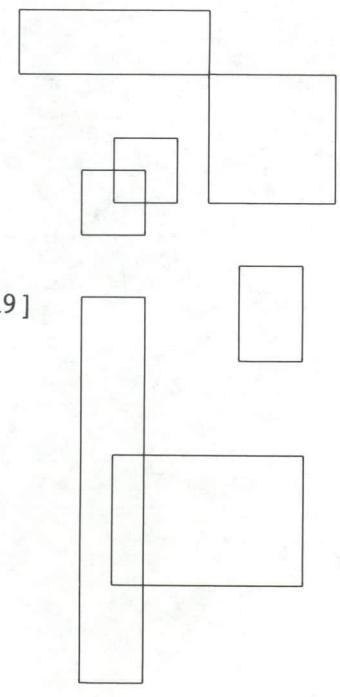
The UCSD Pascal editor has a feature which is very useful for writing structured programs, called auto-indent. A space at the beginning of a line sets the left margin one space to the right, and a delete moves it one space to the left. The following macro simulates this. Unfortunately, until some future release of PMATE implements a way of inserting a character on the screen sans screen update, or a programmed exit from insert mode, you will have to endure the cursor flying about while you type. Surprisingly, though, the macro never drops a character.

```
; autoindent macro  -- invoke by name, exit by escape
;  uses value register 9 and current buffer
```

```
0v9
[g$@k>32&(@k<127){@ki^}
@k=32{@x=@9{va9}@ki^}
@k=127{-d@x<@9{-1va9}^}
@k=13{@ki@9qx^}
@k=9{@x=@9{va9^}@ki^}
@k=27%]
```

It sometimes happens when you're writing a program that you need to look at memory. No need to exit the editor and load DDT. Use this macro.

```
; Hex/ascii dump macro -- Part 1 -- Resides in buffer 2
;                       -- invoke .2XXXX$ where XXXX is start
;                            adress in hex
;       Uses: buffers 1 and 2 for macros, 8 for scratch, 9 for results
;             variable 8 for last adress to dump, 9 for adress counter
;
; We modify the text value being loaded into the 9 register in buffer 1:
; it's easier this way than to do the arithmetic

lqa ble a c0000$^AA$.1
; restore begin adress and radix, display results

bleac^AA$0000$qob9ea
; Part 2 of hexdump macro -- resides in buffer 1
;
; initialize the counters -- register 8 is initialized to dump 32 bytes
;   change it to what you want, but be prepared to wait for large dumps

16qi0000v9@9+20v8qi

; initialize buffer, radix,  and tabs. Write header.

b9kb8kb9e16qoyk3ye

i      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F$
; display only positive numbers

0q-

; this loop does all the work

[@9/16@r=0[b8gb8k@9=@8%i
$@9\i: $]@@<16[i0$]@@\i$9i$b8e(@@<32)!(@@>126)[i.$][@@i$]b9eva9]
```



Finally, for the odd occasion when you may want to edit or print a text containing macros with WordStar, this macro will create a text file from a macro file.

```
; Translate macro  -- if macro is in buffer 1, and MACROS.PMA
;                       is a disk file containing PMATE macros,
;                       invoke by .1MACROS.PMA$
;       This macro takes a macro file with '.PMA' extension,
;       translates it to printable format,
;       and writes it back out to disk with extension .TXT
;
; Uses :
;   Buffer 9 and 0 for scratch

; read it in

lqa bk be i^Aa$ b9k b9e xi^A@0$ be -cpma$txt$ b9e
```

(continued next page)

(PMATE Macro continued from previous page)

```
; translate it
a[@t<32[@t=13^@t=9^@t=10^@t=27[36r][i^$@t+64r]]m@t=0]

; write new file

xo^A@0$bte
```

T/MAKER II Tip

Many people ask, "Can I use the data in a file from another source?"

Yes, and here is an example of how it can be done.

Your income statement, as prepared by the BOSS, Peachtree or other accounting package, is on your disk as a file named INCOME.ST.

Type: TMAKER G INCOME.ST E

This is what appears on your screen:

```
                    ABC Corporation
                  INCOME  STATEMENT
             For the year ended December 31, 1981

Sales                                          $900,000
Cost of goods sold:
   Beginning inventory          $ 50,000
   Add: Purchases                700,000
                                --------
      Total                     $750,000
   Less: Ending inventory         60,000
Cost of goods sold                             690,000
                                               -------
Gross profit                                   $210,000
Operating expenses:
   Depreciation                                 20,000
   Other                                        80,000
                                               -------
      Total operating expenses                 100,000
                                               -------
Income before taxes                            110,000
Income tax expense                              20,000
                                               -------
Net income                                      90,000
```

```
                    ABC Corporation
                  INCOME  STATEMENT
             For the year ended December 31, 1981
ex                                  9,999,999      9,999,999
zv

  +    Sales                                          900,000
       Cost of goods sold:
+         Beginning inventory          50,000
+         Add: Purchases              700,000
                                      --------
=+           Total                    750,000
 -        Less: Ending inventory       60,000
ex                                    ,,,,,,,,      9,999,999
jcl                                      +                =
 =-    Cost of goods sold                             690,000
                                                      -------
  =+   Gross profit                                   210,000
       Operating expenses:
+         Depreciation                                 20,000
+         Other                                        80,000
                                                      -------
=  -        Total operating expenses                 100,000
                                                      -------
   =+    Income before taxes                          110,000
    -    Income tax expense                            20,000
                                                      -------
    =    Net income                                    90,000
```

Now just add your T/MAKER II Mask (**bold face**).

Now all you have to do is change your variables and COMPUTE.

Operating Systems

Description	Version

These operating systems are available from Lifeboat Associates, except where otherwise mentioned.

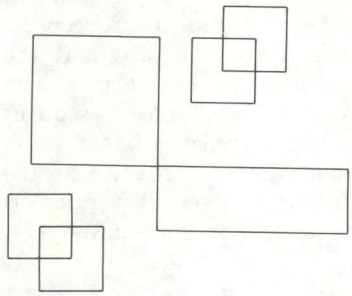
CP/M-80 for:

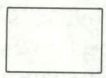
Apple II w/Microsoft BASIC	2.20B
Datapoint 1550/2150 DD/SS	2.2
Datapoint 1550/2150 DD/DS	2.2
Datapoint 1550/2150 DD/SS w/CYN	2.2
Datapoint 1550/2150 DD/DS w/CYN	2.2
Durango F-85	2.23
Heath H8 w/H17 Disk	1.43
Heath/Zenith H89	2.2
iCOM 3812	1.42
iCOM 3712 w/Altair Console	1.42
iCOM 3712 w/IMSAI Console	1.42
iCOM Microfloppy (# 2411)	1.41
iCOM 4511/Pertec D3000 Hard Disk	2.22
Intel MDS Single Density	1.4
Intel MDS Single Density	2.2
Intel MDS 800/230 Double Density	2.2
MITS Altair FD400, 510, 3202 Disk	1.41
MITS Altair FD400, 510, 3202 Disk	2.2
Micropolis Mod I - All Consoles	1.411
Micropolis Mod II - All Consoles	1.411
Micropolis Mod I	2.20B
Micropolis Mod II	2.20B
Compal Micropolis Mod II	1.4
Exidy Sorcerer Micropolis Mod I	1.42
Exidy Sorcerer Micropolis Mod II	1.42
Vector MZ Micropolis Mod II	1.411
Versatile 3B Micropolis Mod I	1.411
Versatile 4 Micropolis Mod II	1.411
Horizon North Star SD	1.41
Mostek MDX STD Bus	2.2
Ohio Scientific C3	2.24
Ohio Scientific C3-B/74	2.24B
Ohio Scientific C3-C'(Prime)/36	2.24B
Ohio Scientific C3-D/10	2.24A
Ohio Scientific C3-C	**2.24A**
Sol North Star SD	1.41
North Star SD IMSAI SIO Console	1.41
North Star SD MITS SIO Console	1.41
North Star SD	2.23A
North Star DD	1.45
North Star DD/QD	2.23A
Processor Technology Helios II	1.41
by Lifeboat/TRS-80 5 ¼"(Mod I)	1.41
by Lifeboat/TRS-80 Mod II	**2.25B**
by Cybernetics/TRS-80 Mod II	2.25

Hard Disk Modules

Description	Version
Corvus Module	2.1
APPLE-Corvus Module	2.1A
KONAN Phoenix Drive	1.8
Micropolis Microdisk	1.92
Pertec D3000/iCOM 4511	1.6
Tarbell Module	1.5
OSI CD-74 for OSI C3-B	1.2
OSI CD-36 for OSI C3-C'	1.2
SA-100A for OSI C3-D	1.2

New Products and New Versions appear in **boldface**.





U.S. POSTAL SERVICE

STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION

(Required by 39 U.S.C. 3685)

1. TITLE OF PUBLICATION	A. PUBLICATION NO.	2. DATE OF FILING
Lifelines	0 2 7 9 2 5 7 5	11.17.81
3. FREQUENCY OF ISSUE	**A. NO. OF ISSUES PUBLISHED ANNUALLY**	**B. ANNUAL SUBSCRIPTION PRICE**
monthly	12	$18 US, $40 foreign

4. COMPLETE MAILING ADDRESS OF KNOWN OFFICE OF PUBLICATION *(Street, City, County, State and ZIP Code) (Not printers)*

1651 3rd Ave., New York, NY 10028

5. COMPLETE MAILING ADDRESS OF THE HEADQUARTERS OR GENERAL BUSINESS OFFICES OF THE PUBLISHERS *(Not printers)*

1651 3rd Ave., New York, N.Y. 10028

6. FULL NAMES AND COMPLETE MAILING ADDRESS OF PUBLISHER, EDITOR, AND MANAGING EDITOR *(This item MUST NOT be blank)*

PUBLISHER *(Name and Complete Mailing Address)*

Anthony R. Gold, 1651 3rd Ave., New York, N.Y. 10028

EDITOR *(Name and Complete Mailing Address)*

Edward H. Currie, 1651 3rd Ave., New York, N.Y. 10028

MANAGING EDITOR *(Name and Complete Mailing Address)*

Jane V. Mellin, 1651 3rd Ave., New York, N.Y. 10028

7. OWNER *(If owned by a corporation, its name and address must be stated and also immediately thereunder the names and addresses of stockholders owning or holding 1 percent or more of total amount of stock. If not owned by a corporation, the names and addresses of the individual owners must be given. If owned by a partnership or other unincorporated firm, its name and address, as well as that of each individual must be given. If the publication is published by a nonprofit organization, its name and address must be stated.) (Item must be completed)*

FULL NAME	COMPLETE MAILING ADDRESS
Intersoft Corporation	1651 3rd Ave., New York, N.Y. 10028
Anthony R. Gold	1651 3rd Ave., New York, N.Y. 10028
Larry B. Alkoff	1651 3rd Ave., New York, N.Y. 10028

8. KNOWN BONDHOLDERS, MORTGAGEES, AND OTHER SECURITY HOLDERS OWNING OR HOLDING 1 PERCENT OR MORE OF TOTAL AMOUNT OF BONDS, MORTGAGES OR OTHER SECURITIES *(If there are none, so state)*

FULL NAME	COMPLETE MAILING ADDRESS

9. FOR COMPLETION BY NONPROFIT ORGANIZATIONS AUTHORIZED TO MAIL AT SPECIAL RATES *(Section 411.3, DMM only)*

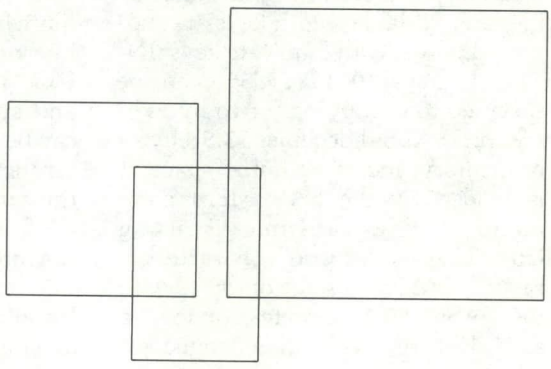
The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes *(Check one)*

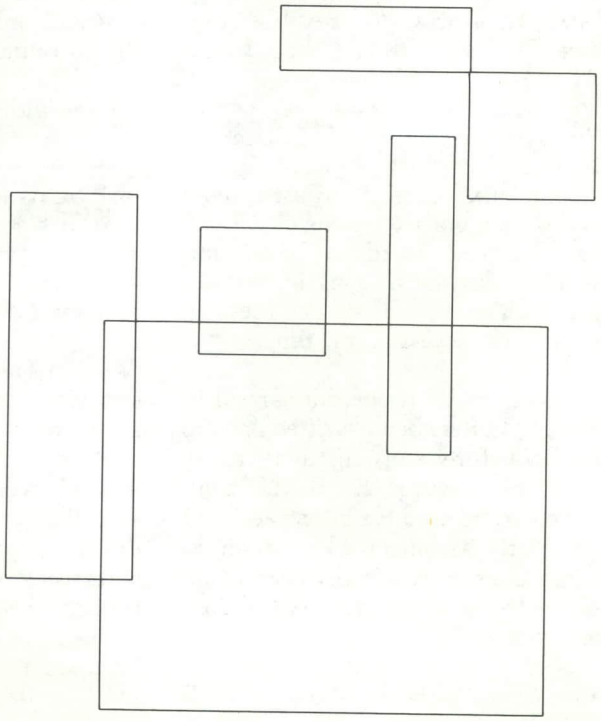
☐ (1) HAS NOT CHANGED DURING PRECEDING 12 MONTHS ☐ (2) HAS CHANGED DURING PRECEDING 12 MONTHS *(If changed, publisher must submit explanation of change with this statement.)*

10. EXTENT AND NATURE OF CIRCULATION	AVERAGE NO. COPIES EACH ISSUE DURING PRECEDING 12 MONTHS	ACTUAL NO. COPIES OF SINGLE ISSUE PUBLISHED NEAREST TO FILING DATE
A. TOTAL NO. COPIES *(Net Press Run)*	7,350	10,000
B. PAID CIRCULATION 1. SALES THROUGH DEALERS AND CARRIERS, STREET VENDORS AND COUNTER SALES	2,900	3,550
2. MAIL SUBSCRIPTION	4,350	6,228
C. TOTAL PAID CIRCULATION *(Sum of 10B1 and 10B2)*	7,250	9,778
D. FREE DISTRIBUTION BY MAIL, CARRIER OR OTHER MEANS SAMPLES, COMPLIMENTARY, AND OTHER FREE COPIES	75	200
E. TOTAL DISTRIBUTION *(Sum of C and D)*	7,325	9,978
F. COPIES NOT DISTRIBUTED 1. OFFICE USE, LEFT OVER, UNACCOUNTED, SPOILED AFTER PRINTING	25	22
2. RETURN FROM NEWS AGENTS		
G. TOTAL *(Sum of E, F1 and 2 - should equal net press run shown in A)*	7,350	10,000

11. **I certify that the statements made by me above are correct and complete** SIGNATURE AND TITLE OF EDITOR, PUBLISHER, BUSINESS MANAGER, OR OWNER

PS Form 3526 July 1981 *(See instruction on reverse)* *(Page 1)*





New Products

This software is available from its authors, from computer stores, from software distributors and publishers.

FABS II
Computer Control Systems, Inc.

FABS (Fast Access BTree Structure) II is a data management and application tool designed to provide rapid access to large data files (up to 65K records, depending upon key size and the number of primary keys). Data record keys are maintained in a key-sequential, multipath, balanced tree structure. Nodes are not re-read into the buffers if that particular node is already present from some past operation.

Multiple primary keys in the same key file, and variable length keys are supported, along with duplicate keys. INTEGER keys (0 to 65535) can be specified to occupy only two bytes in the tree. FABS II occupies 13.5K bytes of memory, including buffer space; it is loaded by the SAVEMEM command during run-time if using CBASIC2, loaded into a character array if S-BASIC is used, combined with the BASIC-80 Interpreter, or loaded as a .REL file with other languages. (Host languages include CBASIC2, S-BASIC, BASIC-80, BASIC Compiler, PL/I-80, FORTRAN-80, Pascal MT+.)

This package is intended to direct the data file, while operating independently from it. Past-deleted record pointers are retained by FABS II so that those records can be reclaimed. This is handled in a last in, first out fashion. As many as six key files can be open for access at one time.

Normally FABS II operations result in changed buffers being written back to the disk before returning to the calling program. However, the BUILD command can be used for a long series of key inserts, as when the keys of an existing data file are initially inserted into the tree when a data file is converted to FABS II.

A relocation program allows FABS II to be adapted to various memory sizes. In addition to relocating the program, this program also creates a BASIC INCLUDE file which can be included in CBASIC2 or S-BASIC to provide symbolic reference to all of the FABS II calling addresses.

FABS II requires 48K of workspace for the host language and FABS II; disk capacity required is dependent on the file size of the host.

IBIOS
Miken Optical Company

IBIOS is an interactive BIOS for CP/M-80, designed to lock the user into a running program without allowing interruptions. The only executable commands are those conforming to the provisions of the program, with respect to function, syntax, and time of issuance.

With this software, which does not require interrupt hardware, the user can interrupt a program that performs I/O. The IBIOS command functions and syntax are user-definable. They can be executed from any program environment. IBIOS is transparent to the currently running program and to CP/M-80. It loads automatically and fits into CP/M-80's BIOS space.

Installation requires a knowledge of assembly language and CP/M-80 system alteration procedures. IBIOS is useful only on systems where the BIOS routines can be modified. It is available in the form of source code listings with command examples.

New Versions

Microstat
Version 2.04

Version 2 includes a more flexible file structure; longer file names; new programs for moments about the mean, skewness, kurtosis, and stepwise multiple regression; ability to declare each file's precision; shorter code size due to a printer toggle; multi-field sorts using Shell-Metzner; no pre-sort for scatterplots.

Release 2.04 corrected a bug involving DATOP.BAS and the artificial limitation of 52 variables. The suppression of treatment means was repaired in ANOVA.BAS and RANOVA.BAS.

PLAN80
Version 2.1

The maximum screen width is now 132 characters, rather than the former 80 characters. The following bugs have been fixed in this new release:

1 - There was a limit on maximum model size; now model of the size indicated by :MODELSIZE may be run.
2 - The shift function can be used now to refer to the same row or column in which answers are to be assigned. Calculations proceed from left to right and top to bottom. Therefore, only negative shift factors may be used in statements like the following: ROWX = ROWX(-1)*1.08.
3 - The :PRINT command now permits only part of a report to be printed, as in: :PRINT (ROW12,ROW33,COL6,COL11).
(Read as one line.)
4 - The :GET statement now operates properly when a file on disk has more columns than are currently recognized by a model in memory.
5 - Highlighting now works properly on terminals using screen attributes.
6 - Additional characters in the graph mode make it easier to control the display on terminals without the highlighting function.

CONTROL RDBMS
Version 5.5A

This new version of this OASIS-compatible package includes such enhancements as more English-like command sentence structure, increased report generation facilities. New subtotal capabilities allow break-on and totalling functions to be performed more easily; heading and footing features are implemented for report generation. Automatic line wrap permits easier reading of long records. 40% more user instruction lines have been added in this release, and a new manual has been written.

XASM Cross Assemblers

XASM05 Version 1.05
The following bugs have been fixed with this new version:

1 - An AT sign ("@") or percent sign ("%") used in an illegal context in the source file sometimes caused the assembler to hang up.
2 - Illegal digits in binary, octal, and decimal numbers were not detected. In particular, if the trailing "H" was omitted from a hex number, the numeric value was incorrect but no error was reported.
3 - Some assembly language statements (e.g. DW) did not report an error if one of the operands was undefined, unless it appeared last in the operand list.
4 - No PRN file was created when the assembler command line specified the "D" switch along with a switch which turned off the listing (i.e., "O" or "X"). In consequence, it was impossible e to get a file listing only those lines containing errors.
5 - When using INCLUD files under early versions of CP/M-80(1.3 or 1.4) and CDOS, garbage was sometimes inserted into the listing and object files.

XASM09 Version 1.07
The bugs noted above have been repaired in this version; additionally, another problem was handled in version 1.06, which received limited distribution. Instructions using indexed addressing with a constant offset were incorrectly assembled when the offset was in the range -32..-17 or 16..31.

XASM18 Version 1.41
The problems mentioned under XASM05 are now corrected. Two enhancements have also been added. The machine registers are now predefined as user symbols. New symbols can be EQUated to registers and used in place of register names as instruction operands. This is in response to user demand for compatibility with library source code supplied by RCA.

The new mnemonics INP and OUT accept register operands and generate the same instructions as INP1..INP7 and OUT1..OUT7, respectively.

XASM48 Version 1.62
The bugs mentioned under XASM05 are now remedied. A new feature of this product is the record-type field of "01", applied to the end record in object files generated by XASM48. This reflects Intel's revised definition of the HEX file format.

XASM51 Version 1.09
The bugs already noted in the section on XASM05 have been fixed. Three other improvements/fixes have been made:

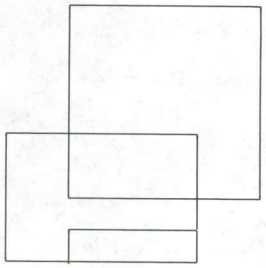
1 - Certain instructions (SET, CLR, CPL) left garbage on the assembly-time stack. As a result, the assembler crashed if enough of these instruction were encountered. This was a rare occurrence, but has been repaired.

2 - The predefined symbol EXTI0 is correctly rendered now, instead of being expressed as EXTIO.
3 - The end record in object files generated by this product now has a record-type field of "01", reflecting Intel's revised definition of the HEX file format.

XASM65 Version 1.97
XASM68 Version 2.00
See XASM05.

XASMF8 Version 1.04
The bug fixes listed under XASM05 are included; the opcode generated for the BR7 instruction has been corrected. (It was 87H and now is 8FH.)

XASM400 Version 1.03
See XASM05. All fixes except number 2 apply to this cross-assembler as well.



Are you looking for a special gift to suit a computer freak you're near and dear to? See page 6.

You probably need our special reference tool, an index to all past *Lifelines* articles. See page 22.

Change of Address

Please notify us immediately if you move. Use the form below. In the section marked "Old Address", affix your *Lifelines* mailing label — or write out your old address exactly as it appears on the label. This will help the Lifelines Circulation Department to expedite your request.

New Address:

NAME

COMPANY

STREET ADDRESS

CITY STATE

ZIP CODE

Old Address:

NAME

COMPANY

STREET ADDRESS

CITY STATE

ZIP CODE

VERSION LIST

November 15, 1981

The listed software is available from the authors, computer stores distributors, and publishers. Except in the cases noted, all software requires CP/M-80, SB-80, or compatible operating systems.

New Products and **new versions** are listed in boldface.

S Standard Version
M Modified Version
OS Operating System
P Processor
MR Memory Required

Product	S	M	P	MR	
ACCESS-80	1.0		8080/Z80	54K	
Accounts Payable/Cybernetics	3.1		Z80	64K	Needs RM/COBOL
Accounts Payable/MC	1.0		8080/Z80	56K	For CP/M 2.2
Accounts Payable/Structured Sys	1.3B		8080	52K	w/It Works run time pkg.
Accounts Payable/Peachtree	07-13-80			48K	Needs BASIC-80 4.51
Accounting Plus			8080/Z80	64K	
Accounts Receivable/Cybernetics	3.1		Z80	64K	Needs RM/COBOL
Accounts Receivable/MC	1.0		8080/Z80	56K	CP/M 2.2
Accounts Receivable/Peachtree	07-13-80		8080	48K	Needs BASIC-80 4.51
Accounts Receivable/Structured Sys	1.4C		8080	56K	w/It Works run time pkg.
Address Management System	1.0		8080		Requires 2 drives
ALDS TRSDOS		3.40	8080	32K	Needs TRSDOS. TRSDOS Macro-80
ALGOL 60	4.8C		8080	24K	
ANALYST	2.0		8080	52K	Needs CBASIC2,QSORT/ULTRASORT
APL/V80	3.2		Z80	48K	Needs APL terminal
Apartment Management (Cornwall)	1.0	1.0	8080		Needs CBASIC2
ASM/XITAN	3.11		Z80		
Automated Patient History	1.2		8080	48K	
BASIC Compiler	5.3	5.3	8080	48K	
BASIC-80 Interpreter	5.21	5.21	8080	40K	w/Vers. 4.51,5.21
BASIC Utility Disk	2.0	2.0	8080	48K	
BOSS Financial Accounting System	1.08		8080	48K	Needs 2/3- drives w/min 200k each, & 132-col. printer
BOSS Demo	1.08		8080	48K	
BSTAM Communication System	4.5	4.5	8080	32K	
BDS C Compiler	1.44	1.44T	8080	32K	w/'C' book
Whitesmiths' C Compiler	2.0		8080	60K	
BSTMS	1.2	1.2	8080	24K	
BUG / uBUG Debuggers	2.03		Z80	24K	
CBASIC2 Compiler	2.08		8080	32K	w/CRUN(2,204P, & 238)
CBS Applications Builder	1.3		8080	48K	Needs no support language
CIS COBOL Compiler	4.4,1		8080	48K	
CIS COBOL Compact	3.46	3.46	8080	32K	
FORMS 1 CIS COBOL Form Generator	1.06	1.06	8080		
FORMS 2 CIS COBOL Form Generator	1.1,6a	1.16	8080		
Interface for Mits Q70 Printer					CP/M 1.41 or 2.XX
COBOL-80 Compiler	4.01	4.01	8080	48K	
COBOL-80 PLUS M/SORT	4.01		8080	48K	
CONDOR II	**2.06**		**8080**	**48K**	
CREAM (Real Estate Acct'ng)	2.3		8080	64K	CBASIC needed
Crosstalk	1.4		Z80		
DATASTAR Information Manager	1.101		8080	48K	
Datebook	2.03		8080	48K	Needs 80x24 terminal
dBASE-II	2.02A		8080	48K	
dBASE-II Demo	2.02A		8080	48K	
Dental Managememt System 8000	8.7A		8080	48K	Needs CBASIC
Dental Management System 9000	1.07		8080	48K	Needs CBASIC
DESPOOL Print Spooler	1.1A		8080		
DISILOG Z80 Disassembler	4.0	4.0	Z80		Zilog mnemonics
DISTEL Z80/8080 Disassembler	4.0		8080/Z80		Intel mnemonics,TDL extensions
EDIT Text Editor	2.06		Z80		
EDIT-80 Text Editor	2.02	2.02	8080		
FABS	2.4A		8080	32K	
FABS II	**4.07**		**8080/Z80**	**48K**	
FILETRAN	1.20			32K	1-way TRS-80 Mod I,TRSDOS to Mod I CP/M
FILETRAN	1.4			32K	Needs TRSDOS. 2-way TRS-80 Mod I,TRSDOS & Mod I CP/M
FILETRAN	1.5			32K	1-way TRS-80 Mod II,TRSDOS to Mod II CP/M
Financial Modeling System	2.0			48K	
Floating Point FORTH	2		8080/Z80	28K	
Floating Point FORTH	3		8080/Z80	28K	
FORTRAN-80 Compiler	3.43	3.43	8080	36K	
FORTRAN Package	3.40				Needs TRSDOS
FPL 56K Vers.	2.51		8080	56K	
FPL 48K Vers.	2.51		8080	48K	
General Ledger/Cybernetics	1.3C		Z80	48K	Needs RM/COBOL
General Ledger/MC	1.0		8080/Z80	56K	Needs CP/M 2.2 or MP/M
General Ledger/Peachtree	07-13-80		8080	48K	Needs BASIC-80 4.51
General Ledger/Structured Sys	1.4C		8080	52K	w/It Works Package
General Ledger II/CPaids	1.1		8080	48K	Needs BASIC-80 4.51
GLECTOR Accounting System	2.02		8080	56K	Use w/CBASIC2,Selector III

VERSION LIST

Product	S	M	P	MR	
GLECTOR IV Accounting System	1.0		8080		Needs Selector IV
HDBS	1.05A		+	52K	
IBM/CPM	1.1		8080		
Insurance Agency System 9000	1.06		8080		Needs CBASIC
Integrated Acctg Sys/Gen'l Ledger			8080	48K	Needed for 3 pkgs. below
Integrated Acctg Sys/Accts Pyble			8080	48K	
Integrated Acctg Sys/Accts Rcvble			8080	48K	
Integrated Acctg Sys/Payroll			8080	48K	
Interchange			Z80	32K	
Inventory/MicroConsultants	5.3		8080/Z80	56K	Needs CP/M 2.2
Inventory/Peachtree	07-13-80		8080	48K	Needs BASIC-80 4.51
Inventory/Structured Sys	1.0C		8080	52K	w/It Works Package
Job Cost Control System/MC	1.0		8080/Z80	56K	Requires CP/M 2.2
JRT Pascal System	1.4		8080	56K	
LETTERIGHT Text Editor	1.1B		8080	52K	
LINKER			Z80		
MAC	2.0A		8080	20K	
MACRO-80 Macro Assembler Package	3.43	3.43	8080/Z80		
Magic Typewriter	3		Z80	48K	
Magic Wand	1.11		8080	32K	
MAGSAM III	4.2		8080	32K	
MAGSAM IV	1.1		8080	32K	Needs CBASIC
MAILING ADDRESS Mail List System	07-13-80		8080	48K	
Mail-Merge	3.0		8080		
Master Tax	1.0-80		8080	48K	
Matchmaker			8080	32K	
MDBS	1.05A		+	48K	
MDBS-DRS	1.02		+	52K	
MDBS-QRS	1.0		+	52K	
MDBS-RTL	1.0		+	52K	
MDBS-PKG			+	52K	w/all above MDBS products
Microspell	4.21		8080	48K	Needs 15 K
Medical Management System 8000	8.7a		8080		Needs CBASIC
Medical Management System 9000	**1.07**		**8080**		**Needs CBASIC**
Microcosm			Z80		CP/M 2.X or MP/M
Microspell	4.3		8080	48K	Needs 150K/drive
Mince	2.6		8080	48K	
Mince Demo	2.6		8080	48K	
Mini-Warehouse Mngmt. Sys.	5.5		8080	48K	Needs CBASIC
Money Maestro		1.1	8080/Z80	48K	CP/M 1.4 or 2.2
MP/M-II	2.0		8080	48K	Needs MP/M
MSORT	1.01		8080	48K	
Microstat	**2.04**		**8080**	**48K**	**Needs BASIC-80, 5.03 or later**
Mu LISP-80/Mu STAR Compiler	2.10	2.12	8080		
Mu SIMP / Mu MATH Package	2.10		8080		muMATH 80
NAD Mail List System	3.0D		8080	48K	
Nevada COBOL	2.0		8080	32K	
Order Entry w/Inventory/Cybernetics			Z80		Needs RM/COBOL
Panel	2.2			44K	Also for MP/M
PAS-3 Medical	1.77		8080	56K	Needs 132-col. printer & CBASIC
PAS-3 Dental	1.63		8080	56K	Needs 132-col. printer & CBASIC
PASM Assembler	1.02		Z80		
Pascal/M	4.02		8080	56K	
PASCAL/MT Compiler	3.2		8080	32K	
PASCAL/MT+ w/SPP	5.25		8080	52K	Also has SuperBr'n & 32K ver., Needs 200K/drive
PASCAL/Z Compiler	4.0		8080	56K	
Payroll/Cybernetics, Inc.			Z80		Needs RM/COBOL
Payroll/Peachtree	07-13-81		8080	48K	Needs BASIC-80 4.51
Payroll/Structured Sys	1.0E		8080	60K	w/It Works run time pkg.
PEARL SD	3.01		8080	56K	w/CBASIC2,Ultrasort II
PLAN80 Financial Package	2.0		8080	56K	Z80/8080
PL/I-80	1.3		8080	48K	
PLINK Linking Loader	3.28		Z80	24K	
PLINK-II Linking Loader	1.10A		Z80	48K	
PMATE	3.02		8080	32K	
PRISM/ADS	2.0.1		8080	56K	Needs CBASIC, 2.06 or later & 180K/drive
PRISM/IMS	2.0.1		8080	56K	Needs CBASIC, 2.06 or later & 180K/drive
PRISM/LMS	2.0.1		8080	56K	Needs CBASIC, 2.06 or later & 180K/drive
POSTMASTER Mail List System	3.4	3.4	8080	48K	
Professional Time Acctg	3.11a		8080	48K	Needs CBASIC2
Programmer's Apprentice			8080/Z80	56K	needs Basic-80
Property Management Program (AMC)	**4.2**		**Z80**	**48K**	**Needs CBASIC 2.07¢, CP/M-80 2.0¢**
Property Management System	07-13-80		8080		Needs BASIC-80 4.51
Property Manager	1.0		8080	48K	Needs CBASIC
PSORT	2.0		8080		
QSORT Sort Program	2.0		8080	48K	

(continued next page)

VERSION LIST

Product	S	M	P	MR	
Real Estate Acquisition Programs	2.1		8080	56K	Needs CBASIC
Remote	3.01		Z80		
Residential Prop. Mngemt. Sys.	1.0		Z80	48K	
RM/COBOL Compiler	1.3C		8080	48K	w/Cybernetics CP/M 2
RAID	**5.0.2**	**5.0.2**	**8080**	**28K**	
RAID w/FPP	**5.0.2**	**5.0.2**	**8080**	**40K**	
RECLAIM Disk Verification Program	2.1		8080	16K	
SBASIC	5.4		8080	48K	
Scribble	1.3		8080		
SELECTOR-III-C2 Data Manager	3.24	3.24	8080	48K	Needs CBASIC
SELECTOR-IV	**2.15**		**8080**	**52K**	**Needs CBASIC**
Shortax	1.2		Z80	48K	TRSDOS,MDOS too, needs BASIC-80 5.0
SID Symbolic Debugger	1.4		8080		N/A-Superbr'n
SMAL/80 Programming System	3.0		8080		For CP/M 1.x
Spellguard	2.0		8080/Z80	32K	Needs Word Processing Program
Standard Tax	1.0		8080	48K	Needs BASIC-80 4.51
STATPAK	1.2	1.2	8080		NeedsBASIC-80 4.2 or above
STIFF UPPER LISP	**2.5**		**8080**	**48K**	
STRING BIT FORTRAN Routines	1.02	1.02	8080		
STRING/80 bit FORTRAN Routines	1.22		8080		
STRING/80 bit Source	1.22		8080		
SUPER SORT I Sort Package	1.5		8080		Max. record=4096 bytes
SELECT			**8080/Z80**	**40K**	
T/MAKER II	2.3.2		8080	48K	Avail. for CDOS
T/MAKER II DEMO	2.2.1		8080	48K	
TEX Text Formatter	2.1		8080	36K	
TEXTWRITER-III	3.6	3.6	8080	32K	
TINY C Interpreter	800102C		8080		
TINY C-II Compiler	800201		8080		
TRS-80 Customization Disk	**1.3B**		**8080**		
ULTRASORT II	4.1A		8080	48K	
Lifeboat Unlock	1.3		8080		Use w/BASIC-80 5.2 or above
VISAM	2.1		8080	48K	
Wiremaster			Z80		Needs 180K/drive
Wordindex	3.0		8080	48K	Needs WordStar
Wordmaster	1.07A		8080	40K	
WordStar	3.0		8080	48K	
WordStar w/MailMerge	3.0		8080	48K	
WordStar Customization Notes	3.0		8080		
XASM-05 Cross Assembler	**1.05**		**8080**	**48K**	
XASM-09 Cross Assembler	**1.07**		**8080**	**48K**	
XASM-51 Cross Assembler	**1.09**		**8080**	**48K**	
XASM-F8 Cross Assembler	**1.04**		**8080**	**48K**	
XASM-400 Cross Assembler	**1.03**		**8080**	**48K**	
XASM-18 Cross Assembler	**1.41**		**8080**		
XASM-48 Cross Assembler	**1.62**		**8080**		
XASM-65 Cross Assembler	**1.97**		**8080**		
XASM-68 Cross Assembler	**2.00**		**8080**		
XMACRO-86 Cross Assembler	3.40		8080		
XYBASIC Extended Interpreter	2.11		8080		
XYBASIC Extended Disk Interpreter	2.11		8080		
XYBASIC Extended Compiler	2.0		8080		
XYBASIC Extended Romable	2.1		8080		
XYBASIC Integer Interpreter	1.7		8080		
XYBASIC Integer Compiler	2.0		8080		
XYBASIC Integer Romable	1.7		8080		
ZAP-80	1.4		8080		Needs 50K/drive
Z80 Development Package	3.5		Z80		N/A-Magnolia,Superbr'n,mod.CP/M
ZDM/ZDMZ Debugger	1.2/2.0		Z80		For N'Star,Apple,IBM 8"
ZDT Z80 Debugger	1.41	1.41	Z80		N/A-Superbr'n,mod.CP/M
ZSID Z80 Debugger	1.4A		Z80		N/A-Superbr'n,mod.CP/M

+These products are available in Z80 or 8080, in the following host language: BASCOM, COBOL-80, FORTRAN-80, PASCAL/M, PASCAL/Z, CIS-COBOL, CBASIC, PL/I-80, BASIC-80 4.51, and BASIC-80 5.xx.

WIN AN IBM* PERSONAL COMPUTER FROM THE NEW MAGAZINE THAT FEATURES THEM.

ANNOUNCING PC™ THE INDEPENDENT GUIDE TO IBM PERSONAL COMPUTERS.

If you're interested in the new IBM Personal Computer, you'll be *very* interested in new *PC* magazine.

Packed with vital information, *PC* will help you keep up with the ever-expanding variety of uses for IBM Personal Computers. *PC* will provide hundreds of helpful tips on adapting Personal Computers for your own needs, and keep you up-to-date on all new developments affecting IBM "PCs" and the people who use them.

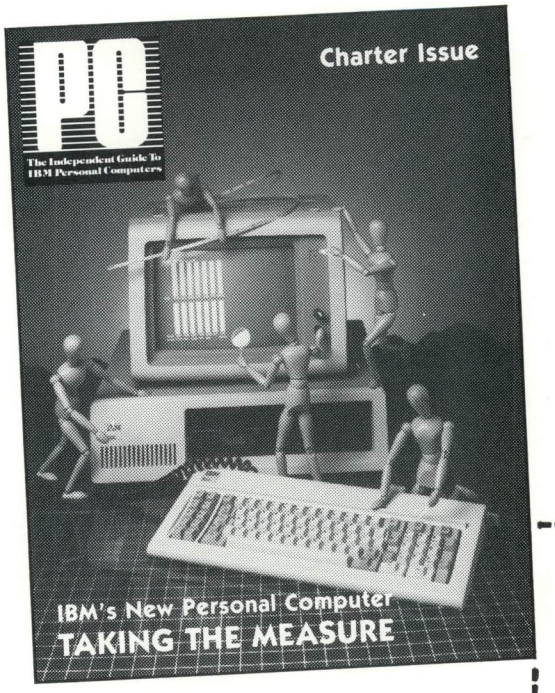
PC even has a special department called PC-Lab™ to evaluate hardware, software and supplies that can be used with IBM Personal Computers—helping you choose those that best suit your needs.

When IBM introduced its Personal Computer, an IBM executive said, "If you could choose one word to describe this system, it would be 'quality.' " That will also be the guiding light for *PC* magazine—in its easy-to-understand writing, its colorful design, and particularly in its devotion to you, the reader.

PC's premiere issue will be out in January, and no one who wants to know more about IBM Personal Computers should be without it—or any of the informative issues to follow. So if you're interested in IBM Personal Computers, be a *PC* Charter Subscriber.

OUR NO-RISK TRIAL OFFER

PC's special offer for Charter Subscribers is the first six issues for just $12, a $6 saving from the single-copy price. And if you're not satisfied with your first issue for any reason, you can cancel and get a refund. Simply write "I want a refund" on the mailing label of your first copy, send it back to us within 10 days, and *PC* will refund your payment in full plus 20¢ to pay for your stamp.



ANNOUNCING THE PC "PC" GIVEAWAY.

PC wants to know who's interested in the new IBM "PC," so we're giving one away to help gauge public interest. You could be the one to win our prize—an IBM Personal Computer system unit (16K) with keyboard and color/graphics monitor adapter. (Available IBM products of equal value may be substituted if you prefer.)

You don't have to subscribe to enter our Giveaway drawing; just fill out and mail the "Giveaway" end of the double coupon below. But why not use the other half of the coupon to enter your subscription at the same time. After all, with our no-risk trial offer, you can't lose. And you'll be sure not to miss out on *PC's* information-filled Premiere Issue.

GIVEAWAY RULES

☐ Mail entries postpaid to the address on the entry blank. Entries must be postmarked before midnight, March 1, 1982. Drawing will be held, and the winner notified by April 1, 1982. Prize delivery date will be subject to IBM product availabilities. ☐ Entries must be on an official entry form or its exact copy. ☐ You may enter as many times as you wish, but each entry must be mailed in a separate envelope. ☐ Employees of Software Communications, Inc., its affiliates, dealers, distributers, advertising agencies and media not eligible. ☐ Void where prohibited, taxed or restricted by law.

SUBSCRIBE TO PC NOW, AT NO RISK.

☐ **YES I want to be a Charter Subscriber to PC.** My charge information or check for $12 made out to **PC is enclosed.**

PC
1239 21st Avenue
San Francisco, CA 94122

Name _____

Address: _____

_____

City _____

State_____Zip_____

☐ Visa ☐ MasterCard ☐ Check Enclosed

Acct. # _____

Exp. Date _____ Bank # (MC only) _____

Signature _____

ENTER THE PC "PC" GIVEAWAY.

☐ **YES Enter my name in PC's drawing to give away an IBM Personal Computer.**

Area Code/Phone# _____

I now have: ☐ an IBM Personal Computer
☐ another personal computer
☐ no personal computer

*PC is an independent journal, not affiliated in any way with International Business Machines Corporation. IBM is a registered trademark of International Business Machines Corporation. *PC* and PC-Lab are trademarks of Software Communications, Inc.

HELIX®
1651 Third Avenue / New York, N.Y. 10028

Second Class Postage Paid
At New York, N.Y.