The Magazine For All Commodore Computer Users.

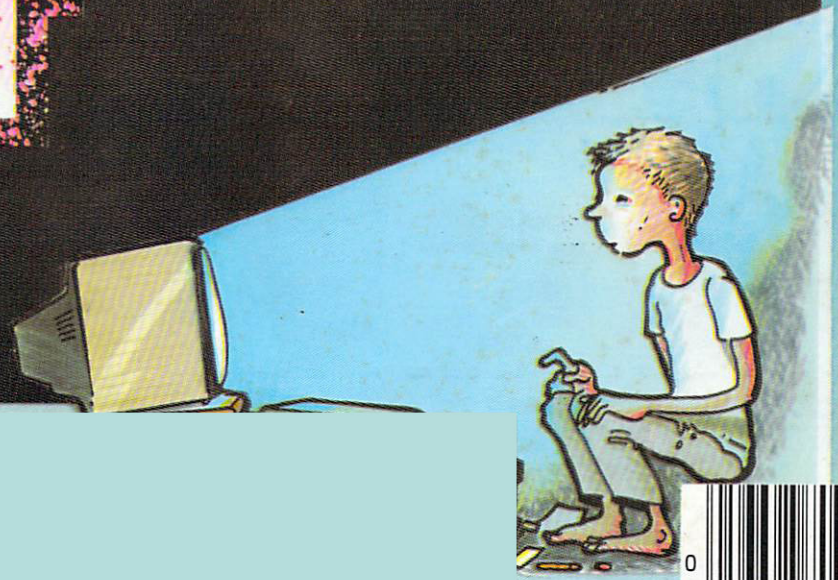
TPUG Magazine

$2.95
Issue No. 24

ORPHAN COMPUTERS
An Orphan Finds A Home
Getting started with Super-OS/9
24K VIC 20 Expansion
A do-it-yourself project
1571 Burst Mode
Part 2 of a series

C-16
PLUS/4
B-128
VIC 20
PET

Commodore's Orphan Computers



Bulk rate
U.S. postage paid
Buffalo, N.Y.
Permit #1575

0 56698 70976 24

ELECTRONICS 2001

AN AUTHORIZED COMMODORE SYSTEM DEALER SINCE 1979

- **New Products constantly arriving**
- **If a product is available We have it**
- **All hardware in stock at all times**
- **Full service**

THIS MONTH'S SPECIALS
PRICES IN EFFECT UNTIL JUNE 30, 1986

SHOEBOX
G/L
$89.95

OUBLIETTE
Oubliette
$14.95

SIDEWAYS FOR C-64
$39.95

IMAGE MAKER
MPS 802
HIGH RES.
PRINTING
$10.00

Compuserve Starter Kit
$39.95

PRINT SHOP
$39.95

NEWS ROOM
$49.95

PSI 5
TRADING CO.

$24.99

PaperClip
(the word processor)
$59.95
with SpellPack
$79.95

Kraft Joystick
$15.95

$36.

ELECTRONICS 2001 LTD.

Name _____

Address _____

_____

☐ Check or money order enclosed ☐ Visa ☐ MasterCard ☐ Amex

Acct.# _____ Exp. Date _____ Signature _____

5529 Yonge Street, Willowdale, Ontario M2N 5S3
Tel: (416) 229-2700

Note: All prices in Canadian Funds. Phone and mail orders welcome.
Ontario residents add 7% sales tax.
Add 5% for shipping (minimum charge $4.00)

10% discount for TPUG members on regular priced software, accessories and magazines.

TPUG Magazine

Publisher: Bruce Hampson
Editor: Nick Sullivan
Assistant Editors: Tim Grantham
Adam Herst
Production Manager: Astrid Kumas
Editorial/Production Assistant: Iwona Sukiennik
Advertising Sales: John Matheson
Cover Illustration: Thom K. Wu
Creative Direction (Cover): Steve MacDowall
Cover Scan and Assembly: LK Graphics
Typesetting: Noesis, Toronto
Printed in Canada by: Delta Web Graphics
Scarborough, Ontario

TPUG Magazine is published 10 times a year by TPUG Inc.

Correspondence: Send change of address and subscription enquiries to: TPUG Inc., Address Changes, 101 Duncan Mill Road, Suite G7, Don Mills, ON, Canada M3B 1Z3. TPUG Magazine welcomes freelance contributions on all aspects of Commodore computing. Contributions should be sent on disk, though accompanying hardcopy is welcome. Be sure to include return postage if you wish materials returned. Please indicate on the disk label which Commodore disk format and word processing program you have used. Payment for articles published is $30.00 per page if the author retains the copyright, and $40.00 per page if the copyright is assigned to TPUG Magazine. Payment is made on publication. All contributions are subject to editing for length and readability. Address editorial contributions and related correspondence to: The Editors, TPUG Magazine, 101 Duncan Mill Road, Suite G7, Don Mills, ON, Canada M3B 1Z3.

Circulation
Subscription: 12,000 Newsstand: 10,000
ISSN #0825-0367

Subscriptions to TPUG Magazine may be obtained by joining the Toronto PET Users' Group (TPUG) Inc.

Regular member (attends meetings)	$35.00 Cdn.
Student (full-time, attends meetings)	$25.00 Cdn.
Associate (Canada)	$25.00 Cdn.
Associate (U.S.)	$25.00 U.S.
	$30.00 Cdn.
Associate (Overseas — sea mail)	$35.00 U.S.
Associate (Overseas — air mail)	$45.00 U.S.

For further membership information, please contact:

TPUG Inc. Membership Information
101 Duncan Mill Road
Suite G7
Don Mills, ON
Canada M3B 1Z3

1552 Hertel Avenue
Suite 144
Buffalo, NY
USA 14216

TPUG Telephone Numbers
Business Office (416)-445-4524
Magazine Office (416)-445-4524
Advertising Sales (416)-445-4524

DIRECTORY

Inside Information

This month

In this issue we take a look at Commodore's orphans (and I don't mean Michael Tomczyk). The paths this industry has followed are strewn with the carcasses of abandoned computers. Commodore can legitimately be considered an old timer in the world of personal computers, and has contributed its share to this litter: many Commodore users are users of machines that are known collectively as *orphans*.

Owners of VIC 20s, C-16s and B-128s know all about what it means to use an orphaned computer. Software and information are next to impossible to obtain from regular outlets, and a stigma is attached to admitting that you use these computers. To make up for these obstacles, orphaned users are forced to group together for support and expend greater effort in learning their computers. If you want to use an orphaned computer then you had better be prepared to DIY (do it yourself).

In contrast, users of 'popular' computers, such as the C-64, enjoy extensive commercial support in the form of widely-available, sophisticated software. Using your computer is merely a matter of going to K-Mart or Toys-R-Us to pick up the latest package. The day will come, however, when the trip to TRU will reveal nothing but Atari ST and C-128 software. What position will 64 owners be in then?

Orphanism is definitely a state of mind. Whether these computers remain useful or are relegated to doorstop duty is entirely up to the user, as Jim Butterfield and Miklos Garamszeghy point out in their articles in this issue. A humorous look at the same topic is provided care of David Bradley. An evaluation of the B-128 by Arthur Klinger gives the lie to the idea that orphaned computers are necessarily technologically inferior.

The future of orphaned computers lies with the support given to them by users. New software and even hardware improvements are not unheard of. The machines continue to be productive, given this support, and can even be adopted into a new, current computer family. One of the most successful of these adoptions must surely be the implementation of OS-9 on the SuperPET computers. In his article, Robert Dray tells the story a newcomer to the Super-OS/9 system. Other rescues are also possible. Expanding the VIC by 24K goes a long way towards extending its usefulness and the procedure to effect this expansion is detailed in this issue by Ronald Byers.

The Amigan

Last month we ran a review of AmigaBASIC by Dick Barnes, whom we represented as the editor/publisher of the highly-respected *SuperPET Gazette*, one of the leading sources of information for the owners of that orphaned computer. Sadly, Dick has been forced to discontinue the *SuperPET Gazette* as of the July issue. In its place, however, he has launched *The Amigan*, a newsletter devoted to all aspects of the Amiga computer. We have seen the first issue, and it looks every bit as comprehensive and informative as its seminal predecessor. If you own an Amiga, this is a publication you won't want to miss. To subscribe from Canada or the States, send a cheque for $24.00 (US), made out to 'The Amigans'. The address is P.O. Box 411, Hatteras, North Carolina 27943, USA.

TPUG online

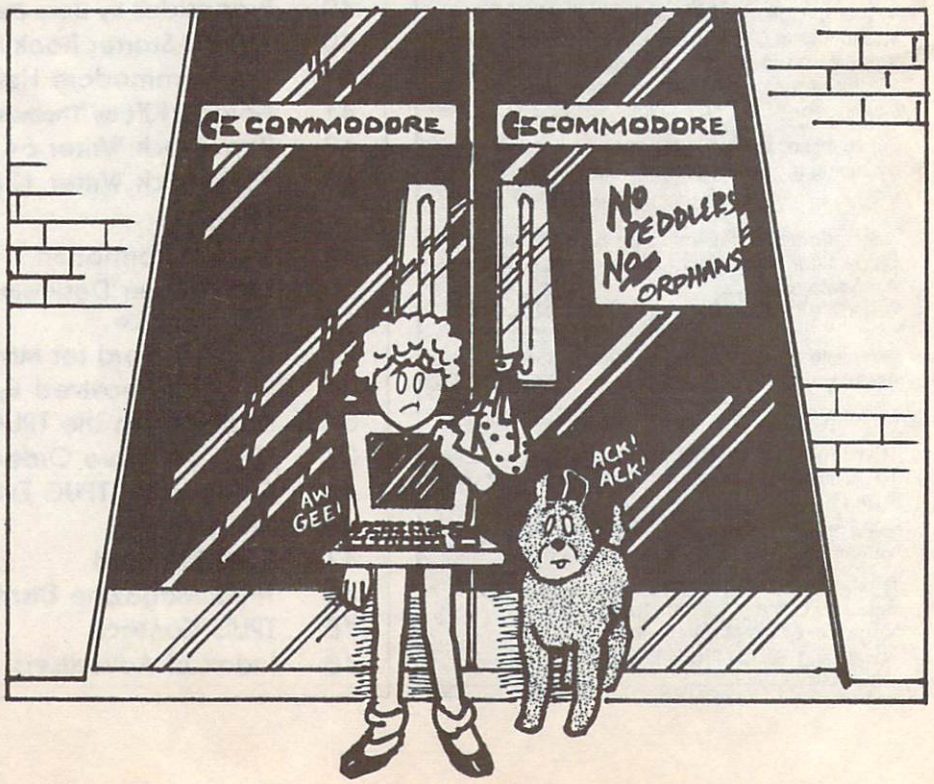
In keeping with its involvement in all aspects of Commodore computing, TPUG maintained a presence online on the CompuServe Information Service. This association has recently been terminated, and TPUG has moved to the friendlier pastures of the Delphi service. Along with the usual program libraries and information exchange, keep your eyes and modems open for *TPUG Magazine Online*. To reach the TPUG SIG, type **GR**oups at the Delphi Main prompt, then **FL**agship Commodore at the Groups prompt.

TPUG Magazine's online presence doesn't stop with Delphi, however. The new *TPUG Magazine* BBS is now up and running on our resident Amiga. Although our BBS is intended primarily for use by our authors, limited access is also available to readers. The number to dial is: (416) 445-0105. If you don't like what we are doing (and especially if you do), let us know how you feel. Line Noise letters and Answer Desk questions can also be directed to us through this BBS.

Still to come

Next month we take a look at one of the most popular types of simulation for microcomputers — flight simulators. This feature will cover available software, as well as some of the theory and programming that goes into these products.

The editors □



The Answer Desk

with Malcolm O'Brien

If you have a question for The Answer Desk, write to us at:

Answer Desk
TPUG Magazine
101 Duncan Mill Road, Suite G7
Don Mills, Ontario M3B 1Z3
Canada

ADM-31 terminal emulation

I have found an annoying bug in CP/M mode on the C-128. The cursor-left key uses the same control code (**CTRL-S**) as the ADM-31 code for suspending screen display. In programs like **Wordstar, dBase II** and especially **CalcStar** that use the cursor-left keys, cursor-left will sometimes cause the machine to lock up. The lock-up appears to be random and may have something to do with the program not disabling the interrupts before reading the keyboard. The computer can only be unlocked by immediately typing in **CTRL-Q**, the ADM-31 code to resume screen display. Has anyone else found this?

Miklos Garamszeghy,
Toronto, Ontario.

The following reply comes from Assistant Editor, Tim Grantham:

"It is my understanding that the upgrade to CP/M for the C-128 now drives the keyboard with interrupts, hopefully eliminating the problem. Adam Herst has brought in the upgrade as an addition to the TPUG library. The BIOS upgrade is available on CP/M disk (Z)AA. Give it a try and see what happens."

CP/M, as implemented on the C-128/1571 combination, is likely to open up a whole new world to CP/M and IBM users as well as to Commodore users. The following remarks from Miklos will serve to give you an idea of the possibilities:

"I also have programs to convert 1541 BASIC files into CP/M files and vice versa. I also have one for the IBM-PC which converts PC-DOS files into CP/M-86 format which can be read on the 1571. Using this you can exchange text, data and even program files (if you know what you are doing) between a C-128 and an IBM-PC. This is how I copied my CP/M software from 8-inch disk format to 1541 format using an IBM-PC as a middle step."

It would seem that Commodore has bounced back from the C-64 CP/M fiasco by providing C-128 users with the comparatively friendly operating environment of CP/M Plus — especially now that the BIOS has been upgraded.

Superpet APL listings

In addition to a dozen PETs, our school has a Commodore SuperPET on which we run Waterloo microAPL. Our system also comprises the Commodore 8250 dual disk drive, and Epson MX-80 F/T printer, and a Commodore 8023P tractor printer. I very much need to print the APL programs displayed on the SuperPET screen to paper but am unable to do so at present. I understand special software is necessary to form the APL characters on a dot-matrix printer. Can you suggest a source for such software? Is there a SuperPET users group? Can you suggest ways to accomplish such a task? Thank you for your help.

Mitchell Johnson,
Endicott, NY.

To answer your question, I consulted TPUG stalwart Tom Shevlin who also uses an MX-80 with APL. It is Tom's understanding that there were three versions of the MX-80. His is a modified version two, which includes an APL character generator. This character generator is an Epson product but it may not still be available. You'll need to make enquiry to Epson on that front.

Tom suggested two other possibilities. The first is to repeatedly redefine and print the MX-80's user-defined character. The second one is probably better and involves using the 8023P printer in graphics mode. Software for this purpose is available from the TPUG SuperPET library or from the International SuperPET Users Group. Here's the address of the latter:

Dick Barnes
ISPUG
P.O. Box 411
Hatteras, NC
27943

Membership in ISPUG is a very reasonable $15 (US). Their newsletter is a veritable gold mine for SuperPET users and Tom reports that they are beginning to give coverage to the Amiga as well.

Printer problem

I am using **Magic Desk 1 Type and File** and the **Traitex International 4.0** word processor with an Epson FX-85 printer and the CO64 interface board. The problem occurs when I want another copy of the same text, or if I use single sheets. At that time the printer stops at the end of a page and waits for the command to continue. When it does continue, the last few lines of the previous page are printed on the following page. Finally, the printer stops just two lines before the end. I pushed the ON-LINE button repeatedly but nothing happens. I have to turn the printer off and back on. Do I have to clear the buffer at the end of each page? This doesn't seem very practical.

Raoul A. Blouin,
Granby, PQ.

*I had similar strange results at first with my Homewriter. The reason was fairly simple: The paper-end detector was tripped before I finished printing my lines. This took the printer off-line and it wouldn't go back on-line with no paper available. Turning the printer off will ensure that it loses its memory (and your margins and tabs along with it). There are two solutions: The first is to set the paper-end detector DIP switch off. If you use this method, be careful — **do not print on the platen!** Printing on the platen will degrade its 'grabby' surface resulting in slippage and inconsistent line feeds.*

The second solution is better. On my Homewriter there is a DIP switch to select 11- or 12-inch paper. By selecting 12-inch you can print a full page (1-inch bottom margin) on an 11-inch page. This way, all stops in printing are controlled by the word processor. This is what we want to achieve.

I notice that Epson has suggested that you ensure that skip-over-perforation and sheet feeder modes are both off and that you should set the form length to 66 lines (this corresponds to an 11-inch page). Definitely defeat the skip and feeder modes but try setting form length to 77 lines (12-inch page). This should keep things neat and pretty. Thanks for sending print samples. These are always helpful.

Computer Orphans: Bargains or Paperweights?

by M. Garamszeghy

A computer orphan can be loosely defined as a machine which is no longer manufactured or supported by its original maker. Many people avoid them like the plague, but I rather like them. Consider this: *all* computers will eventually become orphaned. In fact, computer technology is such a fast growing area that computer hardware is often obsolete before it even leaves the factory!

Family history

Electronic computers are barely forty years old and desk-top microcomputers are not even into their teens yet. The first all-electronic calculating machine was built in the mid 1940's by J. Mauchly and J.P. Eckert of the University of Pennsylvania for the U.S. War Department to calculate ballistic trajectory tables. Dubbed ENIAC (for Electronic Numerical Integrator And Calculator), the device consisted of about 18,000 vacuum tubes, filled several large rooms and consumed enough electricity (120 kilowatts) to power a modern city block. ENIAC was externally programmed by connecting wires in certain patterns and was fed input on punched tape. It operated at speeds a thousand times faster than any mechanical calculating machine then in existence: 5000 instructions per second. By comparison, today's computers operate in MIPS (Millions of Instructions Per Second).

Despite the incredible achievement for its time, ENIAC's computing power was equivalent to less than a basic VIC 20! Computer technology has raced forward at an enormous pace with the birth of the transistor in the 1950's and large scale integrated circuits in the 1970's and 1980's. The course has not been an easy one, however. Many computers (all 'state of the art' in their day) and many computer companies were left for dead along the way. The current crop of 16-bit and 32-bit based micros (such as the Amiga, Atari ST, and IBM-RT) are probably as far ahead of the VIC 20 as the VIC 20 was ahead of ENIAC.

Adopting an orphan

Does this obsolescence make the VIC 20 a useless paperweight? I think not. The benefits of purchasing any orphan computer can easily outweigh the perceived drawbacks. The chief advantage is the low cost of hardware and software for the machine. Most orphaned hardware is discounted to a small fraction of its original price because retailers want to get rid of it to make room for the 'new' stuff. When the VIC 20 was first released, it was a technological marvel: all that computing power, and colour too, for only a few hundred dollars. Several months later, Commodore announced that it was dropping the VIC 20 from its product line. Almost immediately, the price plummeted, with some going for about thirty dollars! (Oddly enough, they are now selling in the $50 to $100 range at some mass market retailers. Maybe demand for them is increasing again.) Expansion boards and other VIC 20 peripherals also dropped dramatically in price. If you can still find them, memory expansion modules are selling for less than the cost of the RAM chips you would need to build your own!

Software and documentation can be even cheaper. I recently bought new, unopened VIC 20 software that originally sold for over $40 for under a dollar from the discount bin at a local retailer. The programming guides for the VIC 20 are invaluable for beginners because they are written in a style that even the greenest of novices can understand. And although introductory manuals are usually machine specific, the basic computer concepts and jargon contained therein are usually applicable to any machine. (For comparison, witness the blank stare when a first time user tries to read the so-called 'introductory' manuals for the IBM-PC.)

If you want oodles of the latest software, then don't buy an orphan. Once a machine is orphaned, the commercial software publishers will drop it from their supported product line like a hot potato; ditto for third party hardware manufacturers. Fortunately, the VIC 20 can use much of the hardware designed for current machines — printers, modems, disk drives, and so on. Machines like the Texas Instruments 99/4A and the Coleco Adam were not so lucky: they are not just orphans, but "only children" as well, unable to share peripherals with more popular sibling models.

Some high quality software is still available for most orphans long after the manufacturer has pronounced the machine dead and buried. The best source of this software is often the libraries of user groups, such as TPUG. There is nobody more tenacious or blindly loyal than a group of avid computer users once their machines have been declared orphans by the powers that be. If there is a large enough ownership base, you are ensured of continuity of software. Most users enjoy writing new software for orphans simply because no one else may be doing it.

Happy and productive

I have seen VIC 20's and other orphaned computers put to very ingenious use because they were cheap and easy to operate. For example, the local outlet of a large department store chain uses VIC 20's and colour TV's to advertise in-store specials. The displays are bright, colourful and easy to read because of the large characters of the VIC 20. The messages are easy to program and update on a regular basis; it only takes a couple of BASIC **print** statements!

A factory I once toured, just outside Scranton, Pennsylvania, was using a half-dozen Sinclair ZX-81's to control its weather monitoring station. The task had been previously performed by the plant's minicomputer. As the engineering staff grew, there was no extra computer capacity to run the weather station. Adding more capacity would have cost several tens of thousands of dollars. With a few home brew adaptors, however, and some ZX-81's bought from a local jobber for about $10 each, one of the plant engineers created a better system than the one run by the minicomputer. The ZX-81 is a very compact computer (about the size of a paperback book) that can be stashed almost anywhere and can run

from batteries for long periods of time. This made it ideal for remote locations.

The above examples are perfect illustrations of a saying that a professor of mine was found of repeating: "You don't need a chauffeur-driven limousine to cross the street if you can do it on roller skates". In other words, you may not need the latest and greatest computer to get the job done.

Because of their low cost, orphans can be a great training tool for youngsters and adults alike. My VIC 20 had not seen much use since I got my C-128. At the same time, my mother-in-law and recently retired father-in-law felt that the electronic revolution was passing them by. They jumped at the chance to borrow my 'old' computer. Within weeks, 'my old computer' had become 'their computer'. Similarly, my brother recently bought the orphaned C-16 (it was very cheap he tells me). While he is more in tune with computer technology than many people I know, I wouldn't call him an expert. Now he is rapidly learning what can and cannot be done with a microcomputer. In addition, his four year old daughter is fascinated by it and plays with it constantly. VIC 20's are very popular with grade school computer classes for the same reasons that many 'serious' users despise them. The displays are bright and colourful, with large easy-to-read characters. The keyboard can be used with little trouble by most children and is virtually indestructible, a boon when working with young kids.

If your perception of computers comes from watching old Star Trek episodes, then you could be in for a shock when you try your first session with a real computer. Several people I know have bought expensive MS-DOS type computers. (Everyone has them, they tell me, so we bought one too). It took one of them several weeks and repeated phone calls to a frustrated salesman ("What is a boot disk?") to figure out how to start it up. The manuals almost require a Ph.D. in computer science to read; they are totally baffling to a novice. Consequently, my friends hardly ever use these wonderful machines. Now I ask you, who owns the bargain and who owns the paperweight?

Other bargains can be had in obsolete peripherals. I bought my printer, which many people would describe as a clunky old daisywheel, for less than a third of the price of comparable ones on the market at the time. How? It was obsolete because the 'standard' for daisywheels of that type (pun intended) was 18 characters per second with boldface, subscripts, and superscripts. Mine printed a mere 12 characters per second without the fancy print styles, none of which I needed: in other words, a perfectly useable letter-quality printer at a bargain price. It has served me well for many years, printing articles like this one. My monochrome video monitor (for the 80 column mode of my C-128) was bought at a surplus store for about $25. Admittedly, it had an open chassis with no cabinet (I built one out of an old black and white TV of the same size and some plywood), but with a 20 MHz bandwidth, it works as well as or better than 'new' ones selling for five times the price.

Foster roster

There are some serious considerations to make before deciding to buy an orphan computer. (Most of these factors apply equally to the latest models). The first is usefulness. Is the computer and its available software capable of doing what you want it to do? Is there reasonable room for growth without expensive hardware additions? Are you able to write your own programs or adapt programs from other machines when you can no longer find commercial ones?

The second consideration is hardware compatibility. Is expansion hardware still available? Will hardware built for similar machines work, with little or no modification? Can you build your own custom hardware? The VIC 20 can use most, but not all, of the hardware designed for the C-64 and C-128. However, anything that plugs into the expansion port is not compatible, either physically or electronically. Although physically compatible, the expansion port on the Plus/4 and C-16 is not electronically compatible with the one on the C-64. In addition, their tape port and joystick ports are different from the C-64 'standard', thus joysticks, tape drives and most printer interfaces require at least minor electrical modification before they will work. Most of the hardware for the newer PETs can also be used on the older ones as well.

The third consideration is the life history of the machine. Had it had a reasonably long and useful life before consignment to the orphanage? Or was it dead before it hit the market? If a machine has a large enough user base, both technical and moral support for it amongst users and user groups will continue for long after it has been dropped by the commercial concerns. Was the machine relatively bug free? Or was it dropped because it had more bugs than the local swamp?

The final consideration is price. Is the price significantly lower than that for 'new' machines of similar type? Do the replacement models represent a significant change? Or is the only difference a minor cosmetic one? (I am told that black computer equipment is out, and grey and tan are in.) Obviously, each decision is a very personal one and one person's ideal machine might be in someone else's junk pile.

In short, if you feel comfortable with an orphan, like its price and can get or write software to make it do what you want, or if you simply want to try your hand at computers, then by all means buy it. If after a while you discover that you need a more advanced machine, congratulations! Pass the orphan along to someone else or keep it for playing games. Your only loss is the small cost of the orphan, but in the mean time you have gained an enormous amount of hands-on experience.

On the other hand, if your main intent is to keep up with the Joneses' or to have a high-tech conversation piece, then buy that top of the line model. But remember, your top of the line machine will eventually become an orphan, and maybe sooner than you think. You may just be buying yourself an even more expensive paperweight. □

Orphans

by Jim Butterfield

There are many microcomputers which are orphans. They have lost their support environment, and while the machines themselves may continue to work, users feel — to a greater or lesser extent — 'cast adrift'.

It's nice to be in the mainstream. It's handy to have a dealer on every city block; to be able to visit a bookstore and find plenty of books about your machine; to have a large body of fellow-users to exchange information and programs with. But if your machine is one that was discontinued, or didn't catch on, you don't get this type of support. In that case, your computer is an orphan, and so are you; you'll need to make your way almost by yourself.

Most of the very first microcomputers — old-timers may remember brand names such as Scelbi, MITS Altair, and SOL — have lapsed into oblivion. More recent brands have had good user play — remember Osborne, Texas Instruments TI-99, and Sinclair? — only to be discontinued and fade from the scene. Even where a manufacturer is still in business, early purchasers may find themselves stranded. Remember the do-it-yourself Apple I, the Lisa, the TRS-80 model 2, or the Atari 400? Even the same model can change radically: a programmer would find a marked change from the original Apple II (with integer BASIC) to a contemporary Apple IIc; and the TRS-80 model 1 has little in common with the TRS-80 CoCo (except perhaps that both machines were available with at least two incompatible BASIC implementations). Even where the name's almost the same, you can be orphaned with a machine from the wrong generation.

Commodore is no exception. It has had a series of computers, many of which are no longer in production, and some of which are clearly orphans. Nonetheless, there's more in common across the line of machines that stretches from the PET 2001 (1976) to the Commodore 128 (1985) than is the case with many other manufacturers. All machines use essentially the same BASIC and the same machine language, although this does not ensure compatibility. The Amiga is a complete break with tradition, of course.

Let's look at some of the old machines that now may be classified as orphans because Commodore has discontinued them. They will run for a long time, and they will have many programs available. But... if they're not made, they're going to fade.

The original PET 2001 had a tiny keyboard, 8K of memory and a problem with the keyboard (unless you managed to change the old 011 ROM for a corrected 019). The cassette deck was built in; but a logic error meant that a disk couldn't be fitted — unless you switched your ROM set to the 'Upgrade' version. The keyboard had an odd upper/lower case reversal which enraged touch typists (if they weren't already mad about the tiny keyboard); and there was some screen snow. Still, a nice machine with screen editing, still capable of running many BASIC programs written today.

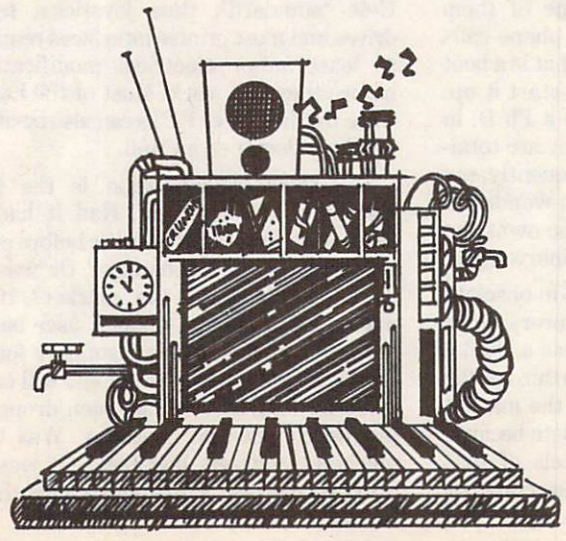
The 'Upgrade ROM' machine (sometimes called ROM 2.0 and sometimes BASIC 3) could use a disk drive and had a number of improvements. Then came the 4.0 machines. Here's where the PET/CBM hit its stride. Many versions were released, some with 40 column screens, some with 80; and good logic, including DOS commands. The whole family used the moderately fast IEEE-488 bus to connect to disk drive and printer.

Some special orphans: the Super PET, with dual processors (6502 and 6809), lots of extra RAM, and custom languages provided by Waterloo University; and the 8096, which also had extra RAM but mapped in a different way. A limited amount of specialized software emerged which exploited the extra capabilities of these machines, and they were wonderful — word processors that allowed a huge document to be written to RAM, and spreadsheets with massive capacity. A restyled 8096 later emerged as the 8296; in this case, software was included as part of the package.

The VIC 20, first of the colour machines, was bargain priced, had limited memory, but all in all was a pleasing machine — Commodore's first mass market computer.

All the above machines enjoyed success in the marketplace to a greater or lesser degree. All are now discontinued. But now we'll mention the hard cases: the machines that never got market acceptance.

The Plus/4 and its junior cousin, the Commodore 16, were nicely designed machines. The 16 had some serious limitations, but the Plus/4 was a joy to work with in many ways, and I still pine for features not seen on other machines. In particular, the ACIA chip made communications a snap. Wanna pour stuff off disk and onto a phone line? It's a clumsy job on the 64 or 128, but there's no interlock on the Plus/4 — you just move the data on through. It could have been the great bulletin board machine of all time. And the screen colours! Not until the



The ultimate orphan computer

Amiga was there anything like it. And it had most of the BASIC features that 128 users rave about.

But Commodore pitched these machines at a price that the marketplace didn't like, and fitted the Plus/4 with a built-in demonstrator program that made the machine look slow and stupid . . . and these machines faded away. They are still seen in discount houses, premium sales, and countries outside North America. But... if you own a Plus/4 or a Commodore 16, you own an orphan.

The B-128 was part of a serious plan by Commodore to design a series of new business machines. By the time they got the bugs cleared away (it took years), the machines were obsolescent; they contained too many chips for current market pricing. They made it to market in Europe (where they are known as the 700 series), but in North America they ended up being unloaded through discount houses (mainly Protecto Enterprises) at a bargain price — but also an orphan price. There was a marvellous colour cousin, the C machine, which was never available; but due to a curious set of happenings, a very few users managed to get their hands on them and refused to give them back when Commodore recalled them.

Not just computers are orphans, of course. We could discuss orphan disk drives, orphan printers, orphan modems, and even orphan cassette drives. Their status can be especially vexing: you can often still get them repaired, but it's hard to know which is more difficult, disposing of one or getting one.

Is an orphan all alone? Not by a long shot. There are special interest groups, program libraries, newsletters and other aids for users who might otherwise feel isolated. It's hard to do much at retailers: books, software and hardware don't stay in stock for long. But there are other sources and sometimes mail order houses do well supporting a product that might be classed as 'inactive'.

Then again, there are people who seek out orphans. They like to choose their own pathways and would be annoyed by a highly popular machine where it seems that everything has been done already. Such people often get a more rounded view of their computer system since they have to do everything themselves: hardware, programming, interfacing and repairs. They can feel that they are pioneering in their own area.

Of course, if you're the ultimate orphan — you have the only machine of its kind — you must do it all yourself. And when you do achieve success on a project, say in writing a program, you have nobody with whom to share the results. There's nobody to brag to, nobody to applaud... you must derive satisfaction simply from the knowledge that you've accomplished what you set out to do.

Sometimes it can be satisfying to be a 'small group' orphan. You get to know the users in your SIG much better than would be the case with the massive group that gathers around a highly successful machine.

So choose your own style. If the machine you are using is discontinued, you don't need to abandon it. Chances are there will be many years of discovery that can be spent on your computer and many people to share your findings with. And you can save money by not leaping to every new computer that comes along.

On the other hand, many of us can't bear to see others with a newer, shinier machine than the one we've got. You just have to leap in when the new machine arrives with those extra features.

But it's hard to give up the old computer. And some of us, determined to take on the new without giving up the old, end up with whole collections of computers. In that case, I suppose we're starting... an orphanage. □

Super-OS/9* Is Here

TPUG has implemented the popular 6809 operating system OS-9* on the SuperPET. Super-OS/9 greatly expands the software availability and the hardware capability of the SuperPET while preserving access to the Waterloo languages and programs.

The cost of Super-OS/9 to club members is $210 (Cdn) (plus $10 shipment/handling Ontario residents add 7% PST), which includes the cost of a hardware modification that will not affect the normal operation of your SuperPET, installation instructions and the operating system disks.

To obtain your copy please send your cheque or money order to:

TPUG
101 Duncan Mill Rd., Suite G-7
Don Mills, Ontario
Canada M3B 1Z3

What does Super-OS/9 offer?

- A true disk operating system with features found on UNIX* systems and on the AMIGA.
- Multi-tasking and multi-user capability.
- Hierarchical directory structure with time/date stamps.
- Programmable file access privileges for increased security.

Extensive software is available for OS-9, most of which runs on Super-OS/9.

Super-OS/9 V1.1 includes an assembler, editor, symbolic debugger, communication software and terminal emulation package. Available languages include BASIC09, CIS COBOL, Fortran 77, Pascal, Lisp, C and others. Application software include wordprocessors, spelling checkers, data bases and spread sheet programs.

TPUG has acquired public domain software and will assist users in the conversion of commercial software to Commodore format.

Portability and Expandability

- Super-OS/9 programs will run on all OS-9 based computers (like the CoCo).
- Super-OS/9 will support standard disk drives (IBM format) and the Hi-res graphics board (700 x 300 dots).
- Super-OS/9 software is C compatible with OS-9 68k and AT & T Unix system V.

For further information call TPUG Inc. at (416) 445-4524, ask for Bruce.

NOTE: If you own a 3 board SuperPET and wish to acquire Super-OS/9, please call TPUG before, ordering Super-OS/9, for info about a hardware fix to a design error in your SuperPET computer.

Expand your VIC

by Ron Byers

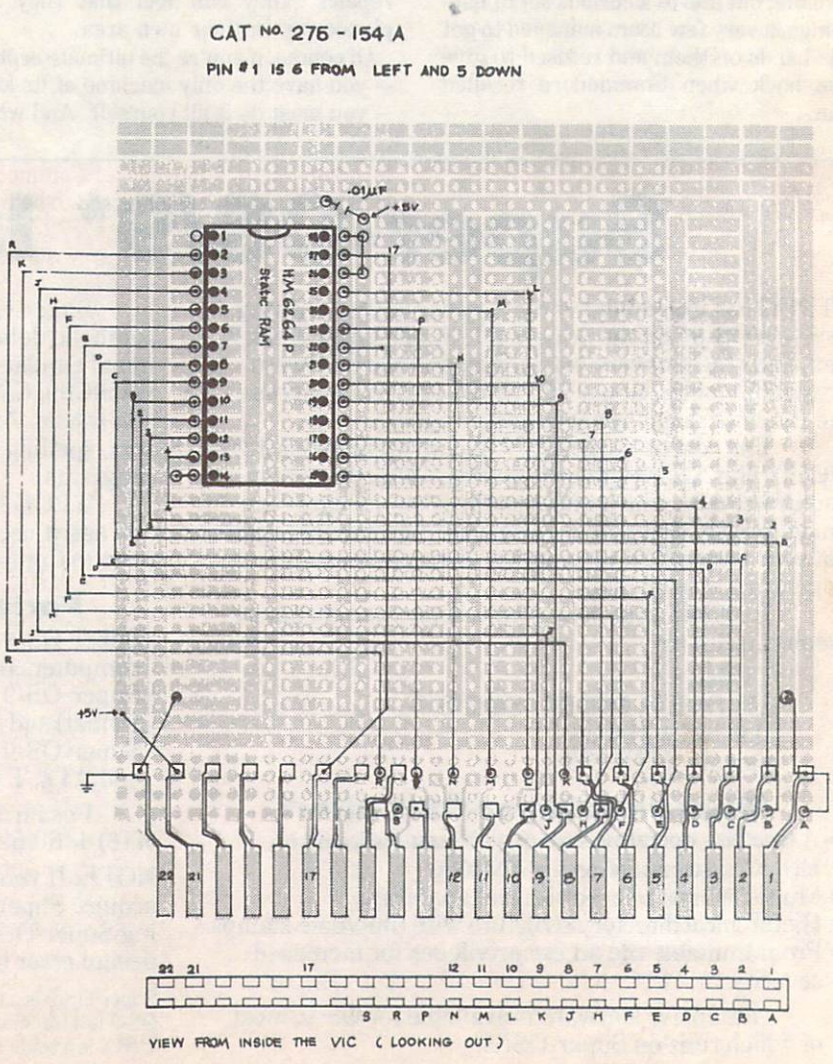
In these days of mega-byte memories and hard drives, turning on your VIC 20 and seeing only 3583 bytes free is enough to give one an inferiority complex. If seeing 28159 bytes free would make you feel better, the obvious answer is the purchase of a 24K memory expander. Unfortunately, you may have to look for a long time or be very lucky to find one for sale at any price. Also, of course, the price wouldn't have to be very high before you would decide that a second-hand C-64 would be a wiser purchase. Before you give up in despair, however, consider the possibility of building your own memory expander.

An amateur radio magazine called *73 Magazine* (Issue #292, Jan. 1985) had an excellent article by Gary P. Brefini that gives all the schematic diagrams and information necessary to do the job. An inter-library loan could probably make this article available to you. The things I learned from building two of these, the diagrams I made in order to translate the schematics into something I could follow, and the technique of piggy-backing chips should make it possible for even the neophyte hardware hacker to do a brain transplant on the VIC.

The hardware needed for this project may be purchased from Radio Shack, with the exception of the static RAM, which may be obtained from Jameco Electronics, 1355 Shoreway Rd., Belmont, CA 94002 (phone 415-592-8097). (It is part #6264P-15 and is priced at $4.49 U.S. in their last catalogue.) If you want to expand to 24K of additional memory, you will need three of these. When you make your trip to the 'Shack' to pick up the circuit board (276-154-A) and a .01 microfarad capacitor and some wire, be sure to get a 28-pin socket too. Use of the socket will allow you to do all of the wiring and checking while you are waiting for the chips to arrive. Note that the wire you use must be very small in diameter: small enough (with insulation on) to pass through the holes in the circuit board. Colour-coded wire makes the job much easier as well.

The diagram shows the placement of the components. As you can see, some of the wires from the RAM chip socket go to the lettered pads on the bottom side of the edge connector and others go to the numbered pads on the top side. It may make soldering easier if you feed the wires going to the top pads up from below. You must use a fine-tipped, low heat iron, and a fine, low temperature solder for best results. Notice that the chip socket is placed with pin 1 in the hole that is the sixth from the left and the fifth from the top of the circuit board, as shown in the diagram. This placement will give one hole on either side of the socket for the attachment of wires. The next row of holes on the right (viewed from the top) may be used as the +5V bus. Run a wire from this to pad 21. The second row to the left of the socket is the ground strip. Connect this to pad 22. The third row of holes on the right is connected to ground also. Pins 14 and 22 on the IC socket are connected to ground in this way, and pins 26 and 28 are similarly routed to the +5V on edge connector pad 21. The .01 capacitor is connected between +5V and ground.

Before working with the circuit board, go over it with fine emery paper or steel wool to make the contacts clean and shiny. Run and solder all of the wires as shown in the diagram and then carefully check each one with an ohm meter or tester to see that there is continuity between each pin from the socket to the appropriate pad on the edge connector. Note that the lettered pads *do not* strictly follow alphabetical order! There *should not* be continuity between any two pins from the socket except between pins 14 and 22, and between 26 and 28. The



solder connections are very close together so care must be taken to prevent solder bridges. If there are any, the ohm meter will find them and they must be removed before you insert the chip and power-up. Extra time taken at this stage will prevent problems later on. Pads A, 1, and 22 on the edge connector should be shorted together, but make sure that there is no continuity between pads 21 and 22.

When all of the wiring and soldering has been completed and checked, carefully insert a static RAM chip into the socket. Turn the VIC off and push the edge connector into the cartridge port. The port is slightly larger than the edge connector (at least mine was), so be sure to center the board so that there is the same space on both sides. Hold your breath and turn on the VIC. If all is well, you should be greeted with a message which says 11775 bytes free. If not, turn off the VIC, reposition the board and give it another chance.

Although it may take you several hours to wire an expander for 8K, it should only take a few minutes to add two more 8K blocks if you have two additional static RAM chips. Power-down the VIC and remove the circuit board. (If you wish to check the integrity of the other RAM chips you could put them, one at a time, in the place of the original chip in the socket.) Find pin 20 on the second chip and gently bend it out to 90 degrees and then place it directly, pin-for-pin, on top of the first. Carefully tack-solder each pin on the second chip to the first (except for pin 20). Run a wire from pin 20 on chip 2 to pad 11 on the edge connector. You have just added another 8K of RAM. Try it as you did the first and you should see 19967 bytes free.

Let's go for 28159 bytes free. Bend out pin 20 on a third chip and piggyback it on top of chip two. Tack solder to the pins below as before. Connect a wire from pin 20 to pad 12. That should do it . . . 24K!

The *73 Magazine* article mentioned above gives a short program to check each memory location in the added 24K. Generally speaking, if you can load a long program which uses the expanded memory and if it lists alright, all is well. If you use the original article, note that there is an error in their diagram. Pad 7 should not be grounded.

You will find that this expander, along with the PET emulator program available from TPUG, will make your VIC 20 considerably more useful. I find that with the expander, a 1600 modem and a 40 Column terminal program, the VIC can telecommunicate with the best of them. □

Orphan update

by David Bradley

The world of Commodore computers is filled with what is kindly called orphaned equipment. After years of experience with this equipment, David Bradley assures us that there is still life after abandonment, and offers the following helpful hints and information.

Rehabilitate your orphan

Tired of hearing your machines are obsolescent? I'm here to tell you that Commodore equipment can be put to good use for years after other machines have been relegated to dusty cupboards — or worse.

- A 1525 printer can be used to keep a car from rolling. Just place it by one of the tires and your worries are over.
- A 1541 can be used to assist you in starting a small restaurant business. No, not to keep track of your inventory or do your accounting, but to cook your food, as well as make toast.
- All of your old, burned out Commodore 64 power supplies can be used to help you with your boating. Tie them all together or put them in a strong bag of some sort and use them as an anchor for your small to medium-sized water craft.
- An old PET of any model can be used as a stylish hood scoop for your car.
- Old Vic 20's are a perfect thing to put into time capsules. That way future generations can look back and see just how barbaric the 1980's were.
- Commodore joysticks have absolutely no use at all.

New Commodore products

Commodore has just announced that the Commodore 64 will soon be able to operate a modem at 19.2 Kilobaud. Up until recently, the C-64 was limited to 1200 (and even that was pushing it), but Commodore has discovered a fix for this. It seems that the chips that handle the user/RS232 ports of the C-64 (two 6526's) were located too far from the video chip (6567). Apparently, the distance from the chips slowed things down so much that it severely limited the communications speed of the computer. The answer to this problem was to redesign the circuit board so that the chips in question reside right next to each other. Another technological triumph for the company we have all come to love so much!

Another development that should come as no surprise to all those that know and love Commodore is the introduction of a special interface for the 1526/802 type printers that will allow them to work 100% with **Print Shop, Newsroom** and all those other terrific, graphic-type programs. It can be installed quite easily. After you have plugged in the interface to 120 VAC, all you have to do is take the serial cable that is included with the interface and plug it into the unused serial port on your 1526/802 as well as one of the two serial ports on the interface. Add a suitable amount of paper into the interface and check to ensure that the ribbon (included at no extra cost with the interface) is properly inserted. The final step is to turn your 1526/802 printer off and then load and run the graphics programs that you could never make use of before. To return to normal text mode, turn the interface off and apply power to the 1526/802 printer again. By the way, to order this fine Commodore technological advancement, be sure to call your local Commodore dealer and ask him/her about the MPS-801.

On the technical scene

Commodore has finally sent out an official release dealing with the poor picture quality that most people experience when they attempt to hook their C-64 to a television set. It seems that the Commodore 64 gives off so much radiation of various sorts, that it makes a clear picture on a TV impossible, unless you hook it up to a TV in the house or apartment next to yours. Apartments directly above or below will produce an improved display but will not be quite as clean as one next door. This is due to the fact that a lot of the radiation that the 64 produces is known as vertical radiation. Rural users are, as Commodore so eloquently puts it, SOL (whatever that means). If you are not on good terms with neighbours directly beside you, you can try neighbours two or three doors away, but Commodore did not test this. Their dedicated technical person does not get along with the people that live farther than one door away from him. Some times I guess relatives can be like that. □

Protecto's B128/8050/4023 system

by Arthur R. Klinger

Protecto Enterprises has provided several thousand users with an excellent computer system comprising the 128 kilobyte B-128 computer, the 1 megabyte single-sided 8050 dual-disk drive, the 80 column 4023 dot-matrix printer, a monochrome monitor, manuals and a disk of training and utility programs — all at the bargain price of 795 dollars (US). They also make available most of the programs needed by serious users, including a word processor, a data base, spreadsheet, general ledger, inventory and telecommunications programs and others. These are well-known, full-featured programs worth several times more than their present price of fifty dollars each.

Most of this hardware and software has been reviewed individually in various detail in publications serving the serious PET/CBM user. This review touches only lightly on each item, pointing out a few features, developments and problems that may not be widely known.

The B-128 computer

The B-128 is a beautifully-packaged 128 kilobyte, 80 column machine having an internal power supply, 96 key keyboard, a 6509 processor running at a 2 MHz clock speed (twice the traditional 1 MHz speeds), an industry standard RS232 serial interface as well as the high-speed IEEE-488 instrumentation parallel bus, a cartridge slot, memory expansion bus, the famous 6581 SID (Sound Interface Device) with audio output, an expanded BASIC 4.0+ with **print using**, **bload** and **bsave**, an internal ASCII/PET ASCII conversion, and other features. About 90 dollars worth of parts can bring internal memory up to 256 kilobytes. The numeric keypad is possibly the most convenient on the market, with ., +, -, *,/, entry, clear entry and double-zero entry all conveniently with the ten digits under the operator's one hand. All graphics characters are available from the keyboard, and are shown on the front sides of the keys. Engineers, scientists and technically inclined users would particularly like such things as the powerful IEEE-488 (GPIB) instrumentation bus, the quick 2 MHz computation speed, the SID 'waveform generator', the complete numeric keypad, a key dedicated to the pi symbol and pi constant, and a very good 9-digit floating-point BASIC.

During the first few weeks of use, I discovered a few minor flaws in the B-128's operation. The top of the case just above the internal power supply gets quite warm, although no problems have yet surfaced. Unless sound-making routines are written carefully, the internal speaker emits a continuous low-volume but irritating sound. The SID chip has the capability of handling analog signals and digital switch closures (the paddle and joystick control lines as found in the C-64 and C-128 computers). Many business and technical users need these ports for sketchpads, plotters, scanners and other analog resistance or voltage monitoring. In the B-128, however, these lines have not been brought to the outside world via joystick connectors or other means.

The B-128 has a dedicated key to switch from upper case/graphics mode to upper case/lower case mode. Although it is a business machine, the machine powers up in graphics mode, and graphics characters are displayed on the screen with no spacing between lines (correct for graphics use). Unfortunately, the zero spacing is maintained when the machine is switched to business mode, which makes the screen appear cramped, and allows descenders to touch upper case or tall letters in the line just below. The user can add a single pixel space between lines by poking an 8 to register 9 of the 6845 video controller chip, but it seems that this action should have been included in the ROM routine that services the graphics/lowercase key.

A check of the B128's horizontal line graphics shows that the graphics characters shown on the **c** in graphics mode erroneously prints the graphics symbol shown for the shifted **v**, and vice-versa. Considering the logical layout of the graphics set on the keyboard, it appears that the character ROM, and not the keyboard labelling, is in error. A similar check of the vertical line graphic characters and bar-graph characters shows that some of the left-most and right-most line characters are 2 and possibly 3 pixels wide, contrary to the one-pixel widths expected by the user and created by the traditional PET and CBM machines. Somehow, the position of these line graphics characters seem also to be slightly offset from the position they would normally be expected to occupy. These idiosyncrasies can be seen when the characters are printed to the screen beside or below each other.

The 8050 disk drive

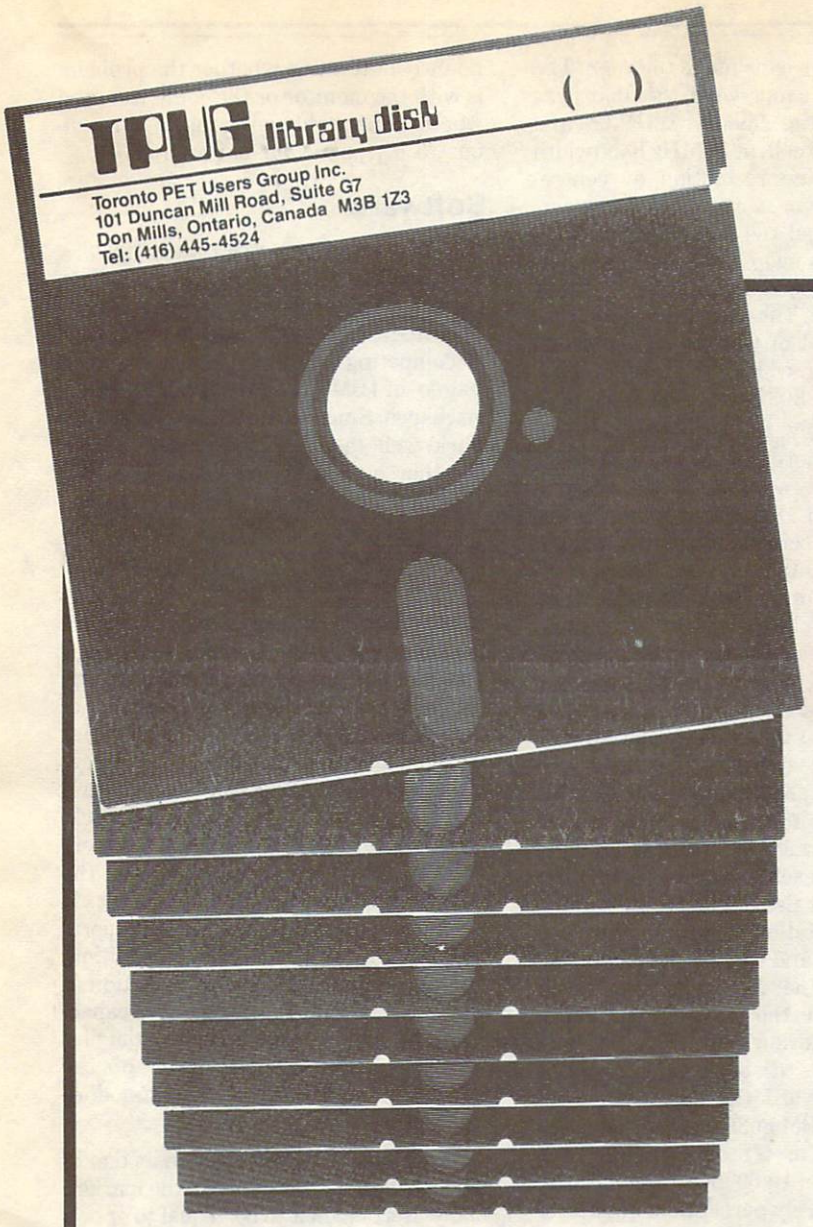
The 8050 dual-floppy disk unit uses 100 track-per-inch single-head drives with a storage capacity of 533,248 bytes on 77 tracks on one side of each of the two disks. Its DOS (2.5) has several features not available on the 175 kilobyte 4040 and 1541 single drives popular with PET and C-64 users. The 8050 drives transfer information over the IEEE-488 bus (only), at a rate that is roughly 4 to 6 times faster than that for the 1541's serial bus. Another version of this drive, the 8250, writes on both sides of both disks, providing a total of more than 2 megabytes of storage on two inexpensive floppies. The 8250 DOS (2.7) allows one random access file to occupy the entire 2.12 megabytes of disk storage.

In spite of their high 'quad' density of information storage, 8050 drives have a reputation for excellent reliability and accuracy. My 3 year old 8050, which uses Micropolis drives, has been used to format at least 3000 disks and to store or copy a few thousand programs, often using the cheapest of single-density blank disks. Yet I have encountered only one 'bad disk', and no other problems or errors have ever been traced to the drives or DOS. The moment the doors are closed on a fresh installed disk, the drives begin turning to insure solid, precise centering, and the disk ID is checked. This eliminates the primary sources of read-write errors in the older 2040/4040 drives.

Some users have reported problems when the 8050s are moved or transported, and have blamed it on such things as flexing of the frame of the drives. In my case, this so-called 'problem' has always been cured by simply commanding the drive to 'initialize', which repositions the head at its proper 'home' position, re-establishing the relationship of head position to track position.

Many or most of the earlier 8050s were

Continued overleaf...



TPUG LIBRARY DISKS

10 months of the latest, fabulous public domain software only

$79.95 CANADIAN
US orders US$59.95

Subscribe now to start receiving the TPUG disk of the month

You **SAVE 20%** or more by using this offer as a TPUG member

THIS OFFER VALID FOR PAID UP TPUG MEMBERS ONLY

PLEASE PRINT

Name: _____ Membership # _____

Address: _____

_____

Type of Payment

☐ Cheque
☐ Mastercard
☐ Visa
☐ Money Order

Amount $ _____
Currency ☐ Can. ☐ US
Credit Card # _____
Expiry Date _____

Signature _____

Type of Computer

☐ C 64
☐ VIC 20
☐ PET ☐ 4040 ☐ 8050
☐ SUPERPET (10 disks) ☐ 4040 ☐ 8050
☐ C128 (1541 only)
☐ MS/DOS
☐ AMIGA (3½") (7 disks) ☐

shipped with Tandon drives having hinged 'trap-door' access similar to those on 4040 Shugart-built drives. The recent 8050s from Protecto come equipped with MPI (Micro-Peripherals) model 101 drives having doors similar to those of the Tandon drives. However, the MPI units have heavy cast-aluminum frames and are thought by some to be better than the Tandon drives. A pleasing discovery was the fact that the recent 8050s from Protecto contain all of the ROM and control electronics of the bigger dual-sided 8250. One writer, R. Dale Connelly, stated in the *JCCUG Newsletter* that MPI model 102 dual-sided drives can be purchased for about $100 each, and installed in place of the original drives with 2 jumper changes to yield a full-blown 2 megabyte 8250. However, I have not been able to locate any MPI 102s, get a reply from MPI, or find directions for the conversion.

The 4023 printer

The 80-column model 4023 dot-matrix printer has been popular for some time, and is quite well known by PET and CBM users. It is very much like the popular Epson MX-70 or MX-80 printers and can use the Epson ribbon cartridge, but supports the full Commodore set of ASCII characters with all graphic and cursor control characters. It uses a heavy-duty, jewelled Shinwa print head with a life expectancy of 50 million characters. It prints bi-directionally at about 45 80-character lines per minute, or 124 20-character lines per minute. It will print user downloaded characters, format numbers and lists through an excellent **print using**-style formatting capability, and has incrementally variable line spacing. It receives information over the IEEE-488 parallel bus only (it will not work with the C-64 serial bus without an interface). It sprocket feeds the paper and takes standard width pin-fed fan-fold paper. The machine seems well made and reliable, and prints fully-formed characters that look nicer than those from my larger, more expensive Commodore model 8023 dot-matrix printer.

The Xtron monitor

The purchaser has a choice of monitors and a choice of green or amber screens. By paying 20 dollars more than the 795 dollar package price, I obtained the Xtron 12-inch AG12X green-screen monitor with an integral 'tilt-and-swivel' stand and 'super-high' resolution. The monitor's styling is modern, and the colour and styling match the B-128 so well that it almost seems made to order. The manual that came with the unit was labelled for the 'model 12HP39' and specified an excellent 20MHz bandwidth and an 800-line resolution at centre-screen. However, a company representative states that the AA12X/AG12X actually have an even better resolution of 1000 lines at centre-screen, and 800 lines at screen edge. The monitor is quite comparable to that of Commodore's superb 12-inch screen, which is an integral part of their CBM 8032 computer.

Unfortunately, the advantage of high resolution is more than offset by the fact that the display occupies only a relatively small area (6.3 by 7.5 inch) in the centre of the 12-inch screen! Even after adjusting the horizontal and vertical size controls for maximum size, the measured *diagonal* of a full-screen display was only 9.6 inches. Nearly an inch of margin at the top and bottom, and nearly 1.4 inches of margin at each side, were left unused. This meant that the display occupied only 47 square inches, or about half of the exposed screen area! Representatives of Xtron and Protecto shared my opinion that the fault was with the B128, and the Protecto representative said that other monitors were also not being utilized fully. Experimentation with the B-128 video chip registers and the Xtron controls indicates that the Xtron monitor could probably scan the full screen width, although resolution falls off slightly and 'pincushion' distortion increase noticeably toward the edges. In comparison, my CBM 8032 screen is adjusted to within 1/2 or 5/8 of an inch of the screen edge, or 10.65 inch diagonal size, with only a barely perceptible deterioration in resolution or distortion.

When the video controller chip of the B-128 was poked to add another pixel line of space beetween each displayed line, the Xtron screen was utilized to a satisfactory degree in the vertical direction. However, I could find no way to obtain a similar spreading of the display horizontally. The problem results in characters that are uncomfortably small for prolonged, comfortable reading. The characters also appear to be somewhat awkwardly 'tall and thin'. For this reason alone, I find myself going back to my comfortable 8032 for prolonged word processing or programming — almost negating the very reason for the B-128 purchase! Two other distant B-128 users did not notice the problem until it was brought to their notice by telephone, so the shrunken screen may not be as noticeable on other monitors. I have yet to try other monitors on my B-128, to more confidently determine whether the problem is with the monitor or the computer, and whether the problem is general or is unique to my particular units.

Software

The key programs for the B-128 are **Superscript II**, **Superbase**, and **Calc Result**. All three are popularized as smooth, fast, powerful programs capable of competing with the best in the 'big league' of IBM PC and other high-priced machiines. **Superscript II** and **Superbase** work well together, and are available together as **Super Office**. Although I haven't used the manuals much, they are in full size three-ring binders and appear to be clear, complete, concise, and reasonably well indexed.

Superscript II has all the standard features: document chaining to unlimited lengths, search and replace, headings and footings, decimal tabs (alignment of decimals), disk formatting, printer lines per inch and built-in mailing list merge. It has a 32,000 word dictionary with a capacity of 200,000 words, and a background printing mode that allows letters and reports to be printed out while the user is typing in another document. It also does calculations for financial reports and so on, with row and column addition, multiple column, addition, calculation of numbers within text, and number capacity to 20 digits. It stores in sequential files instead of the program files of the popular **WordPro** series of Commodore word processors.

Superbase has wide acclaim as one of the best data base systems on the market. Many users claim it to be "equal to or better than **dBase II** but smoother, faster and easier to operate". It has a built-in programming language, user-definable screen formats with up to four screens, calculation capabilities, sorting on any field, and comprehensive search and selection capabilities. It handles 15 files per database, unlimited records per file, 127 fields or 1108 characters per record, and 255 characters of text per field. It supports printouts to 255 columns.

Calc Result is a 'three-dimensional' spreadsheet containing up to 32 pages of electronic spreadsheets of 64 by 256 lines per page, all of which are interrelatable. Four pages can be viewed on the screen at once. Formulas are protected from accidental erasure. The program handles conditional functions such as **if-then-else** and **or-and-not**. Numbers can be translated into charts on the screen or printer; mathematical functions such as sine, cosine and logarithms are sup-

ported; and statistical functions can compute minimum, maximum, mean, standard deviation, and so on. Commands like Replicate, Copy, Move, Goto and others make **Calc Result** easy to work with. The user can decide on the printed layout, with control over the printing order of columns and the number of times the column will appear, location of printout, and storage of the layout. Help screens ease learning and use.

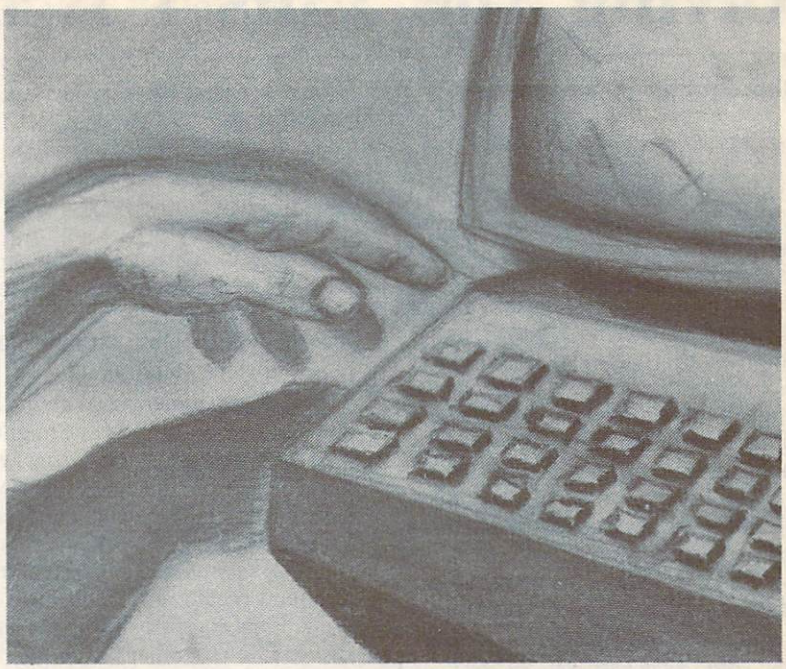
So far I have used only **Superscript II**. People familiar with **Wordpro** will find **Superscript** easy to learn and in general quite satisfying. However, in the form provided, it has serious bugs with its cursor controls, insert/delete functions and speed. The 'bell' is sounded near each right margin, but the speaker then emits a low-volume but irritating sound that varies or pulses when the cursor controls are used. It is occasionally too slow in accepting input, omitting letters during short bursts of familiar words. Worse yet, it frequently repeats letters, putting 2 or 3 characters when only one is needed. While cursoring right or left, the cursor pauses arbitrarily every 5 to 15 character spaces, slowing progress and making it difficult to stop precisely at desired locations. With more than a few paragraphs, pushing text apart with the shift-insert key is agonizingly slow and erratic, with action sometimes taking place well after the user gives up and releases the key.

These problems may all involve interrupt functions, the internal servicing routines that are executed by the machine 60 times per second. Whatever their source, they are severe enough to have forced me to set aside **Superscript II** until a fix is available.

Summary

This article was written with two objectives in mind: to alert people to the existence of a very capable and complete computer system available at an extraordinarily low price, and to call for assistance on fixing the few bugs or otherwise enhancing this fascinating but somewhat 'orphaned' system. At least 20,000 B128s have been released for distribution in the USA, and most of them are apparently already in use in homes and offices. With the number of users, experts, authors and sources of information now available, there is no doubt that such problems as those mentioned above will be (and are being) quickly corrected. The B-128 and its associated hardware and software makes for a highly capable system, and should make a lot of people happy. □

TPUG PROGRAMMING CONTEST



TPUG is once again offering you the opportunity to reduce the costs of your hobby. The Librarians Committee of TPUG is sponsoring a programming contest as a means to encourage you to submit your programs to the library. The winner of this contest will be selected at random from the names of the submitters of all programs accepted by the librarians from the submissions received between the first publication date of this notice and Friday, October 31, 1986. The more programs you submit, the greater your chance of winning.

RULES

- Submissions must be received on or before the deadline.
- Submissions must be on diskette (VIC programs may be submitted on cassette — two copies, please).
- Submissions must be original material.
- Submissions can be for any Commodore machine.
- Submissions should indicate that they are contest submissions.
- All submissions become the property of TPUG.
- TPUG general policy of returning a disk of your choice on acceptance remains in effect for all submissions.
- Unaccepted disks will be returned.
- Freeware submissions will not be accepted for contest consideration.
- Submitter's name must be included in a comment statement at the start of the program as well as on the front of the disk.
- First, second and third prizes will be awarded consisting of 100, 50, 25 blank disks respectively or 25, 10, 5 disks (respectively) from the TPUG libraries.

The Librarians Committee

Solving the Plus/4 RS232 mystery

by Jim Grubbs

Innovation is a wonderful thing. That's what they tell me, anyway. In order to make a better product, changes must be made. I remember the first time I tried to make my C-64 go 'beep'. It didn't seem much to ask. Little did I know that it would take a thorough understanding of the SID (Sound Interface Device) chip to accomplish this simple task. Along came the Plus/4 and this time Commodore made 'beeping' even simpler than it was on a VIC 20. Things were looking up — until I got ready to put my Plus/4 into telecomputing service.

Everything *looked* good. The pseudo-RS232 port was there. *RUN Magazine* even said that the port was ninety-nine per cent compatible with VIC 20 and C-64 accessories. In a casual aside, they also said that little information was available on the RS232 port. This statement should have made bells ring and red lights flash in my head!

It was late the first night I brought the Plus/4 home. Throwing caution to the wind, I reached for my trusty VIC modem. It just didn't seem to want to fit. I was certain that the lateness of the hour and my fatigue from learning about a new machine was causing me to hallucinate. I redoubled my efforts.

I wasn't hallucinating — the cases on the 1600 and 1650 modems were simply too big to fit in the access hole provided on the back of the Plus/4. Dissappointed, I headed off to bed, cursing the Commodore gods and their 'innovation'.

In a calmer moment the next day, I carefully removed the plastic case from the 1600 to expose the bare ciruit board. I figured that perhaps with its shell removed, the modem would fit in the Plus/4. My thinking was correct, and the first rung on what would turn out to be a long ladder was successfully mounted. Next, to telecompute!

There is an old saying about the word 'assume' that's not appropriate for a family magazine, but it applied to my experiments. After carefully typing in the sample modem program in the Plus/4 manual, I tried accessing our local bulletin board. Once again, failure met my attempts: no receive, no transmit. Back to the drawing board!

I'm somewhat of an expert on RS232 communications with the VIC 20 and C-64, so when I took a good look at the **open** statement in the Plus/4 program, I thought I had found the problem. On the VIC 20 and C-64, a simple **open 2,2,3,chr$(6)** will put you in business at 300 baud communications with standard protocol. The Plus/4 statement looks like this: **open 2,2,3,chr$(22)+chr$(5)**. Not quite the same thing, is it?

Those of you unfamiliar with the RS232 port on the VIC 20 and C-64 might want to go back and review the 'Gateway to the World' articles in the June and December 1984 issues of *RUN*. Briefly, there are two registers in your computer that control RS232 parameters — the control and command registers. First glance at the layout of the control register in the Plus/4 showed it to be similar to the VIC 20 and C-64. A careful look at bit number four, though, revealed something new. When this bit is set to a one, the baud rate generator, the 'heart' of the communications hardware, determines the receive rate. All standard data speeds are available in the Plus/4. If this bit is *not* set, an external source can be used to control the receive speed. If the bit isn't set, the receiver does not work!

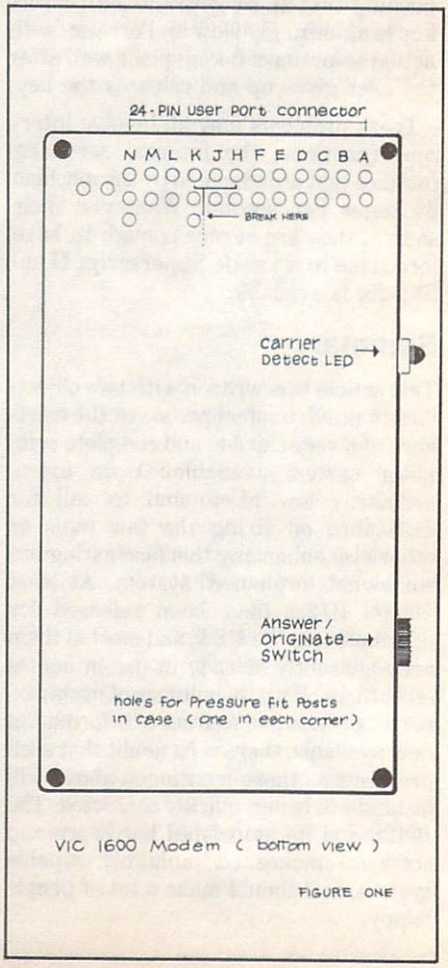
If bit number 4 in the control register is to be set, a 16 must be poked into this register. Just like our old trusty VIC 20 and C-64, a 6 is poked into the register to set the unit for operation at 300 baud. When we put the values together we come up with **chr$(22)**. Mystery number two solved!

The **chr$(5)** is what sets the command register. Unlike the VIC 20 and C-64, where all values in the command register are optional, some value must be placed here. Two bits are critical for normal operation.

Bit zero (the first bit) disables the receiver if cleared. If this bit is set to 1, the receiver is all ready to go and the DTR line (Data Terminal Ready) is brought low. The values in bits 2 and 3 determine the status of the transmitter and enable the transmitter interrupt. To set the value of bit 2 to a 1, we **poke** a 4. Once again, by adding the two values together we end up with **chr$(4+1)** or **chr$(5)**. Mystery number three was now solved, but why wouldn't anything work?

I went on a safari inside my Plus/4 and discovered that a 6551 UART (Universal Asynchronous Receive Transmit) chip lives there. All you need to know is that this chip is what does the RS232 communicating for you. In other machines, UART-type things have been accomplished by the VIA and CIA chips. Studying some technical information about this chip led me to believe that something coming from the VIC modem was telling the 6551 to hold off on receiving and transmitting. In data communications, this is known as handshaking. If the proper signals aren't present, nothing happens.

Before taking drastic action on the 1600, I decided to see if I could make the RS232 port receive and transmit at all. For experimentation purposes, I used a very simple homemade modem I had developed for another purpose. Sure enough, sending a signal into the Plus/4 resulted in printing on the screen. Typing on the keyboard caused the transmit



VIC 1600 Modem (bottom view)

FIGURE ONE

tones to shift just as they should.

One connection at a time, I started duplicating the signals present on the VIC modem. It was a tedious process and not one I recommend for beginners. By the time I got to pin H on the 1600 I had struck gold!

Pin H on the user is assigned as the DCD or Data Carrier Detect signal. When the VIC 1600 detects the carrier tone coming through the line, it drives the voltage level on this pin low. That's great on the C-64 and the VIC 20, but guess what? The Plus/4 thinks that this means the carrier has disappeared and turns off the receiver. Now I was getting somewhere.

By breaking the printed circuit foil on the 1600 leading to pin H, I forced the DCD signal to float high. It's a bit crude, and you lose the ability to know when the carrier is present, but it does allow reception.

This time I had it for sure, right? Not quite. There was still one hurdle to jump.

On the 1600 modem, pin H is connected directly to pin K, the CTS or Clear To Send signal. Once again, on the 1600 when the carrier is present, the CTS line is driven low. That's all right with the VIC 20 and C-64, but the Plus/4 thinks this means it is *not* clear to send. So, breaking the connection to this pin returned the signal to the proper state and everything now worked.

The necessary modifications are illustrated in figure 1. This is definitely not a project for the weak at heart, nor is it recommended for people not familiar with working on circuit boards. Further, it is a short term solution only, since use of the DCD and CTS lines is effectively lost. This has particular implications for uploading and downloading some programs.

Rest assured, the 6551 UART in the Plus/4 opens up the capability for sophisticated data communications. There are even some indications in the memory map that Commodore had something up their sleeves for special communications software/hardware. Has anyone figured out what the 'Kennedy' routine does yet?

Like the SID, the UART opens up new avenues for computer programmers. And just like the SID, it will take some time to learn how to properly address this chip. The *Programmer's Reference Guide* should be some help when it finally appears. Data communications is an important aspect of computing. The Plus/4 appears to be designed to make maximum use of telecomputing possibilities. High speed, synchronous communications with full error checking is just one. In the meantime, isn't it fun to beat Commodore at their own game?

Step by step instructions

- Carefully remove the 1600 modem from its case. It is held in place by four pressure-fit posts, one located in each corner.
- After removing the circuit board from the case, align it so that it matches the diagram in Figure 1.
- Locate the double row of solder connections located at the top of the board. Count carefully and locate the foil that runs between pins H and K.
- Note the vertical foil that connects to the foil between H and K. It runs between pins J and K.
- Using an X-acto or similar knife, carefully cut the trace you just located. Use caution to insure that only this foil is broken.

The modem is now ready to use with the Plus/4! *Do not attempt to put it back in its case*, since the enclosed 1600 modem will not fit into the opening on the Plus/4. Make sure the unit is right-side-up before you insert it. Operation of the 1600 will be identical to before. □

I Was Blind, But Now I Can C

by Robert W. Dray

Have you ever felt that the devil sent us computers to punish us for all the times we did the right and honourable thing?

I own a SuperPET, but at work I use the new ICON computer with its Unix-like operating system. Recently, I was informed that I was to teach the C programming language beginning in January 1986. I think my superiors selected C because it was not one of the many languages I could practice at home on the SuperPET. Never fear, TPUG to the rescue with Super-OS/9, another Unix-like system, for the SuperPET.

When I first heard that TPUG was offering OS-9 for the SuperPET, I was elated. I would now have an operating system similar to the one on the ICON, and I could get a C compiler for it. With no hesitation, I sent the cheque, and some time later I received a phone call to come into town and pick up a parcel.

With great excitement I opened the package containing several books, some disks and a cute little circuit board. On reading the instructions, I learned to my horror that I would have to take a soldering iron to my SuperPET. How could I violate a long-time friend that way? Nevertheless, after 24 hours of studying the diagrams and wondering whether or not I was capable of such delicate surgery, I opened the lid and started.

The instructions were fairly clear, and I eventually reached the point where they said to turn on the computer, and run the test program. The test didn't work.

You can imagine the sinking feeling in the pit of my stomach. Had I killed the patient? What was this act of foolishness going to cost me to have repaired?

I pulled the parts back out and checked all the pins and solder joints; everything looked okay to me, so I put it back together. This time it passed the test, and I had OS-9 running on my own computer.

Unlike Commodore's own operating system, which is burned into ROM chips, OS-9 is disk-based. If you wish to change Commodore's BASIC 2.0 to BASIC 4.0, for example, you have to remove some chips and replace them with new ones. To modify a disk-based operating system, you simply put the new information on the disk.

Disk-based systems are easily personalized. If you don't like the opening message on power up, you can easily change it. If you are very weird, you even can change the names of the commands so that **dog** instead of **dir** will produce a directory of the disk. You could fix it so that nobody would be able to use your system because only you know the commands.

With Super-OS/9 up, my next task was to get C, so I ordered it from TPUG. After some initial problems (my order got misplaced), I finally received the package of two disks and a book.

The book, like other computer-related books, assumes you know far more than you actually do. There I was with two C disks plus one operating system disk. Now, my 4040 disk drive has only two slots and, any way you figure it, three disks can't fit into two slots! After some reading and a lot of frustration, I noticed that there were two versions of the C compiler. The one you use depends on which version of OS-9 you are running. I could put aside the disk for Level 2 OS-9 systems, and use the one with the program **cc1**. Now I was down to two disks and two drive slots. But which goes where?

OS-9 was meant for very large disk-based systems. A single Commodore 173K diskette can have a very long directory if the individual programs are short. You can imagine how long the directory would be if the disk could hold 10 megabytes. To get around this problem, Unix-like systems create a tree structure of directories and subdirectories. Each directory or subdirectory can contain files or subdirectories. This enables you to organize the contents of your disk so that, for example, all the files related to one job are in the same directory. This system makes makes it much simpler to deal with crowded disks.

One of these directories is called **cmds**, and this is where the OS-9 system goes to find out how to perform any of the commands you give it. Well, each of the two disks, the C compiler and the operating system disk, had a **cmds** directory. With a flash of insight, I figured that when using the compiler, I would not need the OS/9 disk, since the compiler disk had its own **cmds** directory. Thus, the compiler disk goes into drive 0.

The problem of where to stuff these disks required only three days to solve. (Nearing the third day, my guesses as to where to stuff them were becoming increasingly imaginative.) The next problem was to determine where to place the C program I wanted to compile. Since I didn't need drive 1 for anything else, I decided to create a program and store it there.

When using the tree structure of directories, the directory in which you are located is called your working or data directory. You move from one (sub)directory to another with the command **chd xxxx**, where 'xxxx' is the name of the directory you wish to enter. If the directory is many layers down in this tree structure, you can specify the complete path, starting with the drive number. For example, you may wish to go from a directory on drive 0 to one called **sam** on drive 1. You would use the command:

chd /d1/school/chemistry/sam.

In addition to the working directory, there is another directory called the execution directory. This is the directory you tell the operating system to search to find out what a given command means. When you first power up, this execution directory is automatically set as the **cmds** directory on drive 0. Now, wouldn't you think that placing the compiler disk in drive 0 with a **cmds** directory on it, would enable the system to find the commands. No way, Jose! You've no idea how I have come to hate the message **error #216**.

Eventually I realized that my normally intelligent machine might not be so gifted after the radical brain surgery I had performed, and I decided to tell it to change its execution directory to **cmds** on drive 0, by using the command **chx /d0/cmds**. It worked! Once you have changed the disk in drive 0, OS-9 apparently can't find the new one until you tell it where to look.

The time had come: I moved to the directory called **c.prgs** in drive 1 that contained my C program (with **chd /d1/c.prgs**). The compiler was in drive 0, so I used **chx /d0/cmds** to inform the operating system where the the commands were to be found. I then gave the command **cc1 test.c** to start compiling my program. The disk drive started to whir, and a message appeared indicating

that the compiler had started. Slowly, other messages appeared on the screen as various parts of the compilation process were completed. Finally the last step was under way as the link message appeared.

This compiling process was *slow* — ten minutes or so — but it was working! Then, suddenly, a new message: **linker fatal... unable to produce output file... error #004**. I quickly grabbed my list of error messages, only to find that there *was* no error #004... I had had better moments in my life.

C source programs always end with the suffix **.c**. The compiled program has the same, but without the suffix. Looking around, I noticed a program called **test** in the **cmds** directory on drive 0, but there was nothing in it. For the next few days, I tried every thing I could think of, and the only thing I noticed was that the computer was trying to put the final compiled program in the **cmds** directory on drive 0, rather than in the directory containing the original program on drive 1.

Eventually, after several calls to TPUG, I reached Gerry Gold, who suggested I come out to a SuperPET meeting. Reluctantly admitting defeat, I made the journey.

At the meeting, Avy Moise told me that the compiler disk was full and that there was no room on it for the output file, hence the error message. The secret is to redirect the final output from its normal default destination of **/d0/cmds** to drive 1 (in a directory called **c.prgs**, in this case) with the command:

```
cc1 test.c -f=/d1/c.prgs/test
```

The gods smiled on me: the compile worked. I had written and compiled my first C program on my own computer, and it took less than six months.

At the SuperPet meeting, someone suggested a way to speed up the process by creating and using a ramdisk. In many computers, you can tell the computer that a portion of its RAM (random access memory) is a disk, which can be formatted and used just like any other disk. When you use the ramdisk, the data transfer is internal to the computer, and so is much faster. In the course of compiling a C program, many temporary files are created as the compiler gradually changes your source code into machine language. If it could write these files internally on a ramdisk, the compiling process would be much faster.

To create the ramdisk, you first ask for a directory of the ramdisk with the command **dir /dram**. This produces an error message, since the disk doesn't yet exist. You then format the ramdisk with **format /dram**. This prints some data on the screen and asks a question. Answer "y", and when it asks for the name of the disk, you simply give any name that you might give for any other disk.

At this point, I moved to the directory containing my C program, and copied the program to the ramdisk. I then used **chd /dram** to move into the ramdisk as my working directory and gave the command to compile the program. This time the compiling process went much faster, requiring only two or three minutes. I directed the final output back to the **c.prgs** directory in drive 1.

It has been a long and frustrating trip, but I try to tell myself that it was just one of life's little tests to allow me to prove once again that people can be the masters of their machines — if they are not driven insane first. □

A layman's guide to burst mode

by M. Garamszeghy

Part 2: Burst read

In part 1 of this series on the 1571 burst mode (*TPUG Magazine, Issue 23*), we examined the various burst mode commands and how to access them. This month we shall cover the basics of the burst read data transfer protocol.

Burst mode data transfer is very fast, up to 3800 bps (bytes per second). This is considerably faster than the normal 1571 'fast' mode of about 1600 bps and the 1541 (or 1571 slow mode) rate of about 350 bps. (When you take into account the 'overhead', such as disk initialization, track to track jump time and sector seeking time, the average burst mode speed is about 2200 bps, while the average value for 1571 fast mode is about 1100 bps). During normal data transfers (i.e. those using **print#, get#, load, save**) between a C-128 and a 1571 or 1541 drive, a significant amount of time is taken up by the convoluted Kernel routines that must be followed for each byte to be sent.

Burst mode eliminates some of the inefficiency by sending data in blocks (up to 1k bytes long) based on a much simpler, and therefore much faster, hardware handshake. Data are exchanged directly between the 1571 and the data register of the Complex Interface Adaptor #1 chip (CIA 1), based on a simple signal from the normal serial bus controller (on the CIA 2).

There are four simple steps to performing a burst mode read operation. These are:

- Send the appropriate command string to access a burst mode read;
- Initialize the CIA chips;
- Read the data;
- Restore the default I/O devices.

The first step can be performed by either machine language or BASIC statements. A typical BASIC statement may be as follows:

```
open15,8,15,"u0"+chr$(10)
```

This particular example will ask the 1571 to use the *Query Disk Format* command to analyze the format of side 0, track 0 of a disk. The equivalent in simplified ML (machine language) would look like:

```
ldx #$00
stx #ff00
lda #$0f
ldx #$08
ldy #$0f
jsr $ffba    ;setlfs
lda #$00
jsr $ffbd    ;setnam
jsr $ffc0    ;open
ldx #$0f
jsr $ffc9    ;chkout
lda #$55
jsr $ffd2    ;chrout
lda #$30
jsr $ffd2    ;chrout
lda $#0a
jsr $ffd2    ;chrout
jsr $ffcc    ;clrchn
```

The first two instructions are the ML equivalent of BASIC's **bank 15** statement. This instruction should be used in ML whenever you want to call Kernal routines because they are all located in BANK 15. The Kernal jump table addresses are all the same as other Commodore computers. (The C-128 also has several new Kernal routines, only one of which is of interest for burst mode. The Spin/Spout routine, which is used for burst writes, will be discussed in the next installment.) The bank switching is handled automatically by the **open** statement in BASIC. If there is a possibility that the disk drive connected to the C-128 is not a 1571 or that the 1571 has been set to 1541 mode, you can test bit 6 of the *fast serial flag* (RAM location $0a1c, decimal 2588). If this bit is set after an **open** operation (in either BASIC or ML), then the drive is a *fast* device (i.e. a 1571 in fast mode).

The second step is to initialize the interrupt register of CIA 1 and to tell the serial bus that you are ready to receive data. It is easiest to do this with machine language:

```
sei
bit $dc0d
lda $dd00
eor #$10
sta $dd00
```

If this procedure is being called as a subroutine, from either BASIC or ML, then you will need to add an **rts** after the last instruction to return to the calling program. The first instruction disables the normal processor interupts, such as keyboard scanning. This has the effect of increasing the amount of time that the hardware can dedicate to data transfer and eliminating the trapping of keyboard presses which may cause errors during data transfers. The bit instruction is used to reset the Interrupt Control Register (ICR) of CIA 1. The final three instructions toggle the state of the Acknowledge and Ready For Data (ARFD) line, which is used as a clock during the burst transfer. This is a signal to the system that we are ready to receive data.

The third step is to read the burst data. The basic subroutine for reading burst data bytes is quite simple:

```
        lda #$08
wait    bit $dc0d
        beq wait
        lda $dd00
        eor #$10
        sta $dd00
        lda $dc0c
        rts
```

The first three instructions create a wait loop until bit 3 of CIA 1 is ICR is turned on. This condition indicates that a byte is ready to be received. The next three instructions toggle the state of the ARFD line, causing a data byte to be transferred. The final two instructions read the data byte from the CIA 1 data register and return to the calling program. To store this byte, an indexed **sta** instruction similar to:

```
sta ($fa),y
```

is normally used (assuming zero page locations $fa and $fb contain the low and high bytes respectively of the data buffer, and the y register used as an index). The read subroutine is often called from an indexed loop, especially when reading blocks of data. It is important that you keep track of the number of bytes transferred and that your indexing method can handle the number of bytes involved. A summary of annotated assembly language routines needed to read each of the burst mode commands is given in Table 1. These routines can be entered directly on the C-128 with its built in monitor by replacing the labels (e.g. wait, next) with absolute addresses.

The most convenient location for the ML is the cassette and RS232 buffers beginning at $0b00 (decimal 2816). The combined buffer space gives you 768 bytes for ML. (Remember that the number of bytes transferred for a sector read is 1 + the number of bytes per sector. MFM format disks may have sector sizes of 128, 256, 512, or 1024 bytes per sector. *Fast load* GCR sectors have 254 bytes per sector. The number of bytes transferred for other burst mode commands depends on the command. See Part 1 of the burst mode article for a complete description of each command.) In order to use the I/O and Kernal routines, the C-128 must be set for bank 15. Unfortunately, this also limits the maximum size of a data buffer to 8k bytes (bank 0 RAM below $4000 is visible in bank 15 also.) This limitation can be overcome by playing with the Memory Management Unit (MMU) configuration register ($ff00 — all banks) to switch between bank 0 and bank 15 on the fly. Your machine code must be in an area visible to both banks (i.e. below $4000, in a location such as the cassette buffer) for this method to work. The indexed **sta** instruction mentioned above should be replaced with:

```
ldx #$3f
stx $ff00
sta ($fa),y
ldx #$00
stx $ff00
```

The first two instructions set the C-128 to bank 0. The data byte is then stored in the correct bank 0 location. The last two instructions switch back to bank 15. This simple technique allows you to use about 60k of bank 0 as a data buffer. There is no need to protect the unused RAM in bank 0 from being overwritten by variables (they are in bank 1), but don't forget to start your buffer above any ML or BASIC program that may be occupying bank 0 .

The final step, after all of the data have been transferred, processed, and stored, is to close the disk channel and restore the default I/O. In ML, this is done with:

```
cli
jsr $ffcc    ;clrchn
```

It is very important to include the **cli** instruction. This re-enables the processor interrupts that were turned off by the initial **sei** instruction in step 2.

That, in a nutshell, is how to read data in burst mode. You will see that the longer the file, the greater the advantage of burst mode. The machine language portion is poked into the the cassette buffer beginning at RAM address $0b00 (decimal 2816). One final note about burst fast load. I have found by experience that the only way to get consistent performance from this command is to append the DOS wild card character * to the end of the filename. I suspect that the somewhat quirky (what else is new?) 1571 operating system expects to have a full 16 character filename (i.e. padded with shifted spaces) for the burst fast load when it searches the directory for your file. Using the DOS wild card eliminates this problem. Fortunately, fast load is the only burst command which uses a filename. In the next installment, we shall examine burst mode writing. □

More programs overleaf

Summary Of Assembly Language Burst Mode Read Routines

General Read-a-burst-byte Routine (used by all subroutines below)

```
read1  lda #$08
wait   bit $dc0d    ;wait for bit 3
       beq wait     ; of CIA#1 ICR
read2  lda $dd00    ;toggle clock
       eor #$10
       sta $dd00
       lda $dc0c    ;get data byte
       rts
```

Note: Before using any of the following routines, you must load zero page locations $fa and $fb with the low and high bytes of the start of your data buffer and call the appropriate burst mode command.

Single Byte Read (used for Inquire Disk, Inquire Status and Read Sector Interleave)

```
       ldy #0       ;reset pointer
       sei          ;disable interrupts
       bit $dc0d    ;clear CIA#1 ICR
       jsr read2    ;signal when ready
       jsr read1    ;read byte
       sta ($fa),y  ;store byte
       cli          ;restore interrupts
       jmp $ffcc    ;clear I/O channels
```

Multi-Byte Read (for Query Disk Format)

```
       ldy #0
       sei
       bit $dc0d
       jsr read2
       jsr read1
       jsr store
       cmp #2
       bcc exit     ;GCR disk
       and #$0e
       cmp #0
       bne exit     ;MFM error
       jsr read1    ;read status byte
       jsr store
       and #$0e
       bne exit     ;MFM error
       jsr read1    ;# sectors/track
       jsr store
       jsr read1    ;logical track #
       jsr store
       jsr read1    ;minimum sector #
       jsr store
       jsr read1    ;maximum sector #
       jsr store
       jsr read1    ;CP/M interleave
       jsr store
exit   cli
       jmp $ffcc
store  sta ($fa),y  ;store byte
       iny          ;increment pointer
       rts
```

```
Read N Sectors Of Data
(128 byte MFM sectors)

       ldx #(number of sectors)
       stx $fc
       ldx #0       ;# sectors read
       stx $fd
       sei
       bit $dc0d
       jsr read2
next2  ldy #0
       jsr read1    ;read status byte
       and #$0e
       cmp #0
       bne end      ;end if error
next1  jsr read1
       ldx #$3f     ;goto bank 0
       stx $ff00
       sta ($fa),y
       ldx #0       ;goto bank 15
       stx $ff00
       iny
       cmp #$80     ;end of sector?
       bne next1    ;get next byte
       ldx $fd
       inx
       cpx $fc      ;last sector?
       beq end
       stx $fd
       tya
       clc
       adc #$80     ;inc pntr 128 bytes
       bcc next2    ;read next sector
       inc $fb
       jmp next2
end    cli
       jmp ffcc



Read N Sectors Of Data
(256 byte GCR or 256*n byte MFM sectors)

       ldx #(number of sectors)
       stx $fc
       ldx #0
       stx $fd
       ldx #(sector size/256)
       stx $fe
       stx $ff
       sei
       ldy #0
       bit $dc0d
       jsr read2
next2  jsr read1
       and #$0e
       cmp #0
       bne end      ;end if error
next1  jsr read1
       ldx #$3f
       stx $ff00
       sta ($fa),y
       ldx #0
       stx $ff00
       iny
       cpy #0
       bne next1
       ldx $fe
       dex
       stx $fe
       inc $fb
       cpx #0       ;end of sector?
       ldx $ff
       stx $fe
       bne next1
       ldx $fd
       inx
       cpx $fc      ;last sector?
       bne next2
end    cli
       jmp $ffcc



Fast Load Entire File
(254 byte GCR sectors)

       sei
       bit $dc0c
       jsr read2
next2  jsr read1
       sta $fc
       cmp #2
       bcs last     ;last sector in file
       ldy #0
next   jsr read1
       ldx #$3f
       stx $ff00
       sta ($fa),y
       ldx #0
       stx $ff00
       iny
       cpy #$fe     ;only 254 data bytes
       bne next
       tya
       clc
       adc $fa
       sta $fa
       bcc next2
       inc $fb
       jmp next2
last   jsr read1    ;get # bytes in last
       sta $fc
       ldy #0
next3  jsr read1
       ldx #$3f
       stx $ff00
       sta ($fa),y
       ldx #0
       stx $ff00
       iny
       cpy $fc      ;last byte?
       bne next3
       cli
       jmp $ffcc
```

The 1526/MPS 802 Printer

by Ranjan Bose

The last year or so has seen many new peripheral devices being introduced for the Commodore 64. Printer manufacturers have not lagged behind. The latest development has been the introduction of Commodore compatibles such as the Gemini-SGC model, specifically for the Commodore 64.

The prices of Commodore printers have been dropping steadily, and the 1526 (now marketed as the MPS 802 with few changes), which sold for as high as five hundred dollars in 1983, can now be purchased for about three hundred. The 1526 is no match for most other non-Commodore dot matrix printers, which support a myriad of functions like underlining, superscripts and subscripts, italics, multiple fonts, pitches and bit mapped graphics. However, if you are satisfied with a fairly decent character font and excellent tabular formatting and do not need to do a lot of custom-designed graphics, and if you want all this without burning holes in your pocket, by all means buy a 1526 printer.

The earlier 1526s had several problems, such as blocking the serial bus (specially with the VIC 20), and incompatibility with several word processors and other programs. Commodore has since produced a new version of the 1526 ROM to make the machine compatible with their Plus/4 and C-16 computers. This version is called 07C (part # 325341-08; the older version was 05.) To check which version you have, switch your printer on while keeping the paper advance button pressed. The printer will initiate a self-test and report the version number on the top line. To stop the test, turn the printer off.

The 1526 allows the creation of one dot-addressable graphic character. To do this, draw an 8 by 8 matrix on paper, and darken the cells required by the character to be created. Think of each vertical column of 8 cells as the column of 8 printing pins in the print-head. The lowest cell has a value of 1, the next one higher up is 2, then 4, 8, 16, 32, 64 and 128 (the topmost cell). The value for one entire column is the sum of the values of the darkened cells. The values range from 0 (a blank column) to 255 (a dark column). A whole character is represented by 8 such values, each representing a column.

To define this new character for the printer, open a file with a secondary address of 5 (**open 1,4,5**), then create a string by concatenating the eight column totals you have calculated (**a$ = chr$(val1) + . . . chr$(val8)**), and print the string to the open file (**print#1,a$**). A custom character defined in this way can be referenced from then on as **chr$(254)**.

If you need more than one such character, there are two things to bear in mind. If your line containing the custom character is longer than 80 characters and is printed over two lines (wrapping), you cannot redefine another character once the wrapping has occurred — the earlier character gets printed in spite of the redefinition. Secondly, if you want to print two custom characters on the same line, you have to execute **chr$(141)** (carriage return without line feed) before you can define subsequent characters. The trick is to define a character, print it, print **chr$(141)**, tab the print-head to the next print position (since the carriage has returned to column 1), and repeat. This procedure requires you to keep track of what column you have to print to next; tabbing to that column is achieved by sending the column number after a **chr$(16)**. Both the following commands will print at column 51:

```
print#1,chr$(16)chr$(5)chr$
  (1)"hi"
print#1,chr$(16)"51"
```

Now comes the time to let you in on a couple of great undocumented Commodore mysteries. The manual says that while defining a character you should send a string of 8 characters. If you are using a version-05 1526, or are defining only one character, you are fine. But if you have a 07 ROM in your printer and wish to define multiple characters, you have to send *nine* characters, or you will get terminator errors and no custom characters. The ninth byte is not printed and can be anything. I usually stick with the safe non-printing zero.

The implications are clear. If you want to work with a foreign character set, the 1526 will be impossibly slow. Nor is the machine ideally suited for high-resolution screen dumps, although an excellent public domain program for that purpose by Ajay Jindal can be found in the TPUG library. What the 1526 is really good at is formatted tabular output (right or decimal justified for numbers and left justified for alphabet) for applications like business reports

For this you must define a string defining the output fields to be used. This string uses special codes to denote different types of formatting: **A** for string variables, **$** for the dollar sign and amounts, **9** for numbers, the period for decimal alignment, and so on. The format is activated by printing the string to a file with a secondary address of 2. Unformatted data that is sent to a file with a secondary address of 1 will be formatted according to definition before being printed.

You can use the paging option to get reports or program listings that do not run over the seams between two sheets of fanfold paper. This is done by printing **chr$(147)** to an open printer file. Printing a **chr$(19)** rolls the paper up to a fresh page. You can control line spacing in increments of 1/216th of an inch, and can vary the number of lines per page. Files with secondary addresses of 6 and 3 let you accomplish these feats.

The 1526 also supports enhanced printing. Characters can be printed in normal width, double width, triple width or quadruple width, depending on the number of **chr$(14)** codes sent before the character (including custom-defined characters). You can print reverse field characters (white on black background) by sandwiching a string between **chr$(18)** and **chr$(146)** codes. □

Fun with function keys

by M. Garamszeghy

Some computer operating systems (such as CP/M and MS-DOS) allow you to create a disk file containing a series of common key strokes or commands that can be executed without having to type them in from the keyboard each time they are used. This is different from the execution of a regular BASIC or machine language program in that it is generally used for immediate mode, 'housekeeping' commands to set up default input/output configurations, or to specify the sequencing or job queuing of other program files. With MS-DOS, this is called a DOS 'batch file' (denoted by the file name extension **.bat** on the disk directory). The equivalent in CP/M is a 'submit file' (**.sub** on the directory). Some sophisticated programs, such as **Lotus 1-2-3**, incorporate a similar feature. When operating from within such a program, the process is usually called a *keyboard macro*. Whatever you choose to call it, this handy feature allows you to store a series of frequently-used keystrokes or commands, and execute them on demand by entering a single keystroke or command.

Experienced CP/M users will know that the **submit** feature is available on the C-128 when operating in CP/M mode. What most people don't know, however, is that BASIC 3.5 (on the Plus/4 and C-16) and BASIC 7.0 (on the C-128 in native mode) also incorporate a similar, though slightly less sophisticated, feature. The trick lies in the creative use of the function keys.

BASIC 3.5 and 7.0 have an intrinsic command called **key**, which allows you to assign a string of alphanumeric characters to a given function key. The string of characters will be recalled, printed on the screen and executed (if applicable) each time the key is pressed. The syntax is: **key n, "string"** for a string constant enclosed in quotes; **key n, (a$)** for a string variable; or **key** to list the current key definitions.

In the first two cases, **n** is a numeric constant or variable corresponding to the function key number (1 to 8) being defined. There are two more programmable keys on the C-128. These are the **help** key and the **shift-run/stop** key combination. Unfortunately, the definitions for these keys cannot be changed with the **key** command. More on this later.

Normally, the function keys are defined with single commands. Unlike other versions of BASIC that have a similar **key** command (such as IBM BASIC), you are not restricted to entering just one command per function key. In fact, the only restriction on function key definitions is that the total string length for all ten keys combined cannot exceed 246 characters on the C-128 (slightly less in BASIC 3.5), with a maximum of 128 characters for any key. If all ten keys are defined, this is equivalent to an average of 25 characters per key — nearly double the maximum of 15 per key allowed by IBM BASIC. If only four keys are defined, the average size is 61 characters. Using all ten programmable keys, you can have the equivalent of eleven BASIC programs in RAM at a time!

Of what practical use is this capability? Let me give you a few examples. To **list** a program on a printer, you would normally type in something like:

```
open 4,4:cmd4:list
print#4:close4
```

If you are doing a lot of listing while debugging a new program, you can define a function key to do the same task with a statement like:

```
key1,"open4,4:cmd4:list"+ch
  r$(13)+"print#4:close4"+c
  hr$(13)
```

Now each time you want to list a program, you need only press **f1**. The rest is automatic. The printer file is even properly closed after the listing. You can list a disk directory to your printer with a slight variation of the above sequence:

```
key2,"load"+chr$(34)+"$"+ch
  r$(34)+",8"+chr$(13)+"ope
  n4,4:cmd4:list"+chr$(13)+
  "print#4:close4"+chr$(13)
```

These examples include two special characters (**chr$(13)** and **chr$(34)**). The first is a **return**, which must be included in the key definition if you want the command to executed when you press the function key. As shown in the examples, the command string can be broken up into more than one logical line by inserting more than one **chr$(13)**. The second special character is the double-quote, which must be used to delimit file names and so on instead of literal quote marks.

In short, any series of BASIC commands that can be entered from the keyboard in direct mode can be assigned to a function key, allowing you to store several mini-programs in memory independently of a major BASIC or machine language program. If you wish to abort a command sequence once it has started, the usual **run-stop/restore** key combination will terminate any key commands in progress and return you to immediate mode. Error conditions cannot normally be detected. The computer will print an error message and attempt to execute the next statement if there is one.

One of the main advantages of using batch command files and keyboard macros is that the sequence of commands can be stored in a disk file for later recall and use. This is also true for Commodore function key definitions. On the C-128, key definitions are stored in RAM locations 4096 to 4352 in bank 0. They can be saved to disk with the following simple command:

```
bsave"filename",b0,p4096 to
  p4352
```

To retrieve the definitions, type:

```
bload"filename",b0
```

With the Plus/4 and C-16, which lack the **bsave** and **bload** commands, the easiest way to save and load the definitions is from within the machine language monitor. Enter the monitor with the **monitor** statement. To save the definitions, type:

```
s "filename",08,055F,05E7
```

To load the definitions, type:

```
l "filename"
```

Function key definitions can also be used with many BASIC extensions available for the VIC 20 and C-64, such as the **Programmers' Aid Cartridge**. The length and sophistication of the definition string depends on the BASIC extension being used. Most, however, will allow you to assign more than one command to a given key. In addition, if you know where the key definition buffer is located, you can usually save the new definitions for future use.

The **help** and **shift-run/stop** keys on the C-128 are a bit more difficult to program. The **Keyedit** program in the box

accompanying this article, is a BASIC 7.0 program that allows you to redefine all the programmable keys on the C-128. The program also allows you to load and save the key definitions in disk files for later use. **Keyedit** is simple to use by following the prompts and menus displayed on the screen. It can be used with either a 40 or an 80 column display. The routine for redefining the **help** and **shift-run/stop** keys can easily be adapted to your own programs if desired. □

```
10 trap260:cr$=chr$(13):print" function key editor by
   m. garamszeghy" cr$ "  options:":s=0
20 print" 1: load key defs"cr$"2: list keys"cr$"3: ed
   it a key"cr$ "4: save key defs"cr$"5: quit":input"
   <2 spaces>selection";s
30 ifs<1then10:elseifs>4thenend:elseonsgoto40,60,100,
   230
40 gosub90:directory:fl$="":input" filename to load";
   fl$:iffl$=""then10
50 bload(fl$),b0,p4096:ifdsthen250:else10
60 print"   current key definitions:  ":key:l=0:fori=
   4096to4103: l=l+peek(i):next:s=peek(4104):h=peek(4
   105)
70 lo=4106+l:hi=4105+l+s:gosub280:print" s-run/stop,"
   x$:lo=hi+1: hi=4105+l+s+h:gosub280:print"help,"x$:
   print"  total key definition length "l+s+h
80 print"  press a key to continue":getkeya$:goto10
90 print" ":char,5,24,"or press [return] for main men
   u ":return
100 gosub90:print"edit key definition"cr$"  1: f1",,"
    2: f2"cr$" 3: f3",,"4: f4"cr$" 5: f5",,"6: f6"cr$
    " 7: f7",,"8: f8":s=0
110 print" 9: s-run/stop","10: help":input"  select a
    <space>key";s:ifs<1ors>10then10
120 print" ":char,2,21,"press [esc] to keep old defin
    ition or": char,2,23,"[shift]-[return] to keep ne
    w definition "
130 print"   key #";s;"  current definition:  " :nd$=
    ""
140 l=0:fori=1tos:l=l+peek(4094+i):next:lo=4106+l:hi=
    4105+l+peek(4095+s): gosub280:printx$cr$"  new de
    finition >>  ";
150 getkeya$:ifa$=chr$(27)then100:elseifa$=chr$(141)t
    hen190
160 ifa$<>chr$(13)anda$<>chr$(34)thenprint""a$" ";:nd
    $=nd$+a$:goto150
170 ifa$=chr$(13)thenprint" +chr$(13)+ ";:nd$=nd$+a$:
    goto150
180 ifa$=chr$(34)thenprint" +chr$(34)+ ";:nd$=nd$+a$:
    goto150
190 ifs<9thenkey(s),nd$:goto100
200 x=peek(4095+s):poke4095+s,len(nd$):ifs=10then for
    i=1tolen(nd$):poke4105+l+i,asc(mid$(nd$,i,1)):nex
    t:goto100
210 a$="":le=l+x+4105:fori=le+1tole+peek(4105):a$=a$+
    chr$(peek(i)):next
220 nd$=nd$+a$:fori=1tolen(nd$):pokel+i+4105,asc(mid$
    (nd$,i,1)):next:goto100
230 gosub90:print"  file last loaded  ";fl$:fs$="": i
    nput"  filename to save";fs$:iffs$=""then10
240 bsave(fs$),b0,p4096top4352:ifdsthen250:else10
250 ifds=63theninput"  file exists..overwrite y/n";o$
    :ifo$<>"y"then10:else270
260 print"   disk error:"cr$cr$ds$:resume80
270 open15,8,15,"s0:"+fs$:dclose:goto240
280 x$=chr$(34):fori=lotohi:ifpeek(i)=13then x$=x$+ch
    r$(34)+"+chr$(13)+"+chr$(34):goto300
290 ifpeek(i)=34thenx$=x$+chr$(34)+"+chr$(34)+"+chr$(
    34): elsex$=x$+chr$(peek(i))
```

Electronic Phone Book

1) Insert your COMAL disk in drive*.
2) Type LOAD "C64 COMAL*",8
3) Type RUN (starts COMAL)
4) Type AUTO (turn on auto line#'s)
5) Enter the program lines shown below (COMAL indents lines for you)
6) Hit RETURN key twice when done
7) Type RUN

```
e=enter  f=find  l=list
f
What name? COMAL
COMAL Users Group    608-222-4432
```

```
0010 dim name$ of 20, phone$ of 12
0020 dim disk$ of 2
0030 black:=0; white:=1; yellow:=7
0040 background black
0050 repeat
0060   pencolor white
0070   print "e=enter  f=find  l=list"
0080   case key$ of
0090   when "e","E"
0100     enter'name
0110   when "f","F"
0120     input "What name?": name$
0130     find'name(name$)
0140   when "l","L"
0150     find'name("")
0160   otherwise
0170     print chr$(147) //clearscreen
0180   endcase
0190 until true=false //forever
0200 //
0210 proc enter'name
0220   input "Enter name : ": name$
0230   input "Enter phone: ": phone$
0240   if name$>"" then add'to'file
0250 endproc enter'name
0260 //
0270 proc add'to'file
0280   open file 2,"phone.dat",append
0290   disk$:=status$
0300   if disk$<>"00" then
0310     close // data file not found
0320     open file 2,"phone.dat",write
0330   endif
0340   write file 2: name$,phone$
0350   close
0360 endproc add'to'file
0370 //
0380 proc find'name(search$)
0390   zone 21 // set auto tab to 21
0400   pencolor yellow
0410   open file 2,"phone.dat",read
0420   while not eof(2) do
0430     read file 2: name$,phone$
0440     if search$ in name$ then
0450       print name$,phone$
0460     endif
0470   endwhile
0480   close
0490   print "Hit <return> when ready"
0500   while key$<>chr$(13) do null
0510 endproc find'name
```

* If you don't have COMAL yet, order a Programmer's Paradise Package-$19.95. It includes the complete COMAL system plus over 400 pages of information. Add $5 more to get our 20 interactive lesson Tutorial Disk. Add $2 shipping. Visa/MC or US funds check accepted. Send to:

COMAL Users Group USA
6041 Monona Drive, Room 109
Madison, WI 53716
phone 608-222-4432

Amiga Dispatches

by Tim Grantham

I have always viewed IBM-PC compatibility for the Amiga with some bemusement. It seems so incongruous — like hitching an ox to a Maserati. Yet recent attempts to encourage my friends and students to consider an Amiga for their first purchase were met with the same response, with depressing regularity: "Yes, but does it run **Lotus 1-2-3**?" Being able to say 'yes' did not make me feel any better — if the Amiga can't sell on its own merits, perhaps we will have to accept that our technoshocked society can't absorb any more whiz-bang electronics. (Frankly, I believe this sort of PC tunnel vision is the result of an elaborate conspiracy by IBM, John Dvorak, and Jerry Falwell to discourage innovative technology and deprive me of my right to bare arms.)

But whatever I might believe doesn't change the fact that Amiga dealers can't keep the 5¼ inch drives in stock; that orders for the Amiga have jumped (along with CBM's stock) since the announcement of the Sidecar, the hardware PC emulator; and that Important Media Persons who have until now ignored or poohpoohed the Amiga's capabilities have suddenly taken notice and deigned to confer legitimacy upon it. Jerry Pournelle, reporting on the Amiga/Sidecar combination in *InfoWorld*, was so taken with it that he nearly dropped his Kaypro: "The next time somebody tells you the computer revolution is over, tell them they're nuts." John Dvorak, also writing in *Infoworld*, went right crazy and said that the new Amiga software was "pretty impressive stuff."

So I schlepped over to CBM Canada, where the folks in Dealer Support kindly demonstrated a prototype version of the Sidecar. It works as promised, folks. I watched PC **Flight Simulator** run in one window, while several Amiga tasks ran in others, albeit more slowly than usual. I was told that the Sidecar, which is about half the size of a PC-10 (Commodore's IBM clone) and plugs into the expansion port of the Amiga, will come with 512K RAM, one 5¼ inch disk drive, and one expansion slot for any PC-compatible card (not three, as shown at Atlanta Comdex). It will sell for about 800 dollars (Cdn.), offers complete, full-speed emulation and should be available in the fall. When I questioned the reduction in the number of expansion slots, I was quoted costs and told not to be so ungrateful.

To be fair, CBM has never claimed that the Amiga would replace the IBM PC — the emulation was intended to act only as a bridge for those using PCs in other situations. Yet CBM obviously regards promotion of the Sidecar as a grim marketing necessity. In these conservative times, they're probably right. But nobody is gonna make *me* learn MS-DOS; AmigaDOS is irritating enough to one used to the noble eccentricities of Commodore DOS.

Cardco is rumoured to be developing a C-64 emulator, and another company is working on an Apple II+ and IIe emulator. However, there appears to be no truth to the rumour that the Ranger will be sold with a Cray XMP emulator.

Software news

During a trip to my local Amiga dealer the other day, I was very impressed by the range and quality of Amiga software already available. I was more impressed by the absence of *vapourware*. Products that have been announced are, with few exceptions, appearing as promised.

A number of database management programs for the Amiga have made their appearance. Transtime Technologies Corp. have ported their **Datamat** relational DBMS from MS-DOS and Unix. This very powerful program makes use of the Amiga's multitasking, does stats and graphs, and has dBASE III format files. Micro-Systems Software's **Organize!** also stores its files in the dBASE III format, thought it is not truly relational. Also available is **Mi Amiga File**, $99 US from Softwood Co.

Language news... TDI's Modula-2 compiler continues to get favourable response, particularly because of TDI's vigourous customer support and prompt bug-fixes... COMAL is apparently being ported to the Amiga by Unicomal in Sweden... ABSoft has released Fortran 77, a compiler for $299 US.... Claudio Nieder and his colleagues at Rutgers University are making available a public domain version of the new single-pass Modula-2 compiler by Niklaus Wirth... Lattice's 3.04 version of their C compiler is now in beta-test... APL Plus for MS-DOS machines will run under the **Transformer**, as will PC-DOS 3.2 for 3½ inch disk drives...

After some complaints from customers, MSS have rewritten the manuals for **Online!** and **Analyze!**, and added indexes. I hope they plan to do the same for **BBS-PC!**... Precision Software's famous **Superscript**, available in versions for almost every Commodore machine, is being ported to the Amiga... The **VIP Professional** spreadsheet is very similar to **Lotus 1-2-3**. At 400K of code, however, only 40K of workspace is left over! If you have a hard-disk, there is not even enough memory free to load the demo and instructional programs. The program is not copy-protected... **Instant Music** from EA (Electronic Arts) is scheduled for a June release... **The Music Studio** from Activision ($49.95 US) is not getting a favourable response from computer music aficionados. Steve Bennett reports that while it does have MIDI capability, it only provides a MIDI-OUT. It cannot handle slurs, sampled sounds, cannot make global key or tempo changes in a piece of music, does not use the IFF standard for its sound and music files and, to add insult to injury, is copy-protected. It *can* handle triplets...

Aegis Developement is offering a deal to purchasers of **Aegis Draw**: a special price on **Aegis Pro Draw** worth more than the price of **Draw** (which should be available as you read this). **Pro Draw** should be available this summer... Look for the debut of *Envisions*, another high-tech Amiga magazine. For a substantial $80 US, you get four issues a year that include fancy graphics, a disk, and an audio cassette... The official Amiga technical manuals are finally being shipped by Addison-Wesley and all should be available by the time you read this.

AmigaDOS upgrade

The 1.2 versions of Kickstart and Workbench are now in beta-test and reportedly offer significant improvements, including support for the 5¼ inch drives. In fact, Dave Haynie of CBM engineering reports that 1.2 has a facility to define new devices, one of which is a disk drive. All that needs to be done is to specify in a simple ASCII file the number of tracks, sides, sectors per track, et cetera. Other disk improvements will apparently include faster random disk i/o,

a RAMdisk you can invoke from Workbench, and a search path feature for program startup and execution.

Other features include: auto-configuration; circle and ellipse drawing routines in the Kernel; a 100 per cent improvement in the speed of the areafill routine; recovery from guru errors by the cancellation of the task running at the time of the error; and windows that scroll as the scroll gadget is moved (not after).

Less good news is that the new version will cost somewhere between 25 and 50 dollars.

Speaking of enhancements, CBM is expected to announce the Amiga 2000 (not to be confused with the Ranger), the next machine in the promised family of Amiga products. This is rumoured to include a 20 meg hard-drive, 2 meg RAM, an 8086 co-processor, a 3½ inch drive, and a 5¼ inch drive, all built in.

Hardware news

Some mechanical bug reports: John Mesiavech reports that the MicroForge hard-drives prevent the use of the PRT: device. Pretty serious, if true... If you ever find that your disk drive(s) don't work on your own Amiga but will work on another CPU, have the 256K add-on memory checked. For some reason, problems with this unit sometimes only show up in the operation of the disk drives... The manual for the external 3½ inch drives warns that you cannot add more than one additional drive to the Amiga without an external power-supply. Some users have ignored the warning without penalty. In some cases, no modification was necessary. Some have chained three drives by modifying the power pins on the connectors, a simple operation apparently. We'll keep you posted... CBM is currently repairing Amigas the same way they repair C-64s: they swap the motherboard with a new one...

The Star Gemini SG10 works well with the Epson setting... Zenith makes a monitor with high-persistence phosphors intended particularly for graphics in interlaced modes such as that used by the Amiga. It's called the ZVM-136 and retails for a mere 1199 dollars here in Canada.

Studio Amiga

One of the most interesting, and potentially most productive ways to use the Amiga is in broadcasting. A documentary aired on PBS called *Trade Wars* (about the economic summit recently held in Japan) used graphics and subtitles that had been created entirely with an Amiga (featured prominently with the hosts of the program). J. Eric Chard tells of using **Aegis Animator** to create animated graphics for the television station where he works. Broadcast equipment requires an interlaced picture, but **Animator** is not intended for this mode. Chard simply runs the PD program **Setlace**, written by Bob Pariseau of C-A, before firing up Animator, and *voila!*, an interlaced picture is produced. The output from the Amiga is then run through a time-base corrector, and a video processor to adjust some phase shift, and a perfectly acceptable broadcast-quality signal is generated.

Amigans dispatched: In a late-breaking story, CBM has announced a layoff (read 'dismissal') of some 20 employees at C-A, including Bob Pariseau, who was in charge of software development and a tremendous asset to the company, in my opinion. Another 120 employees at the CBM West Chester HQ were also given pink slips. Jay Miner, general manager of C-A, and the designer of the Amiga's custom chips, announced that the layoffs were primarily in engineering support and documentation, and that CBM will continue its development and support of the Amiga. □

ESC G 2

by Adam Herst

It is the beginning of May as I write this; the long wait for the end of the Canadian winter is over. I wish I could say the same about the long wait for the release of the 1750 RAM expansion. From the south, along with the warm weather, comes news of 1750 sightings at computer dealers throughout the United States. The Canadian variety must hibernate longer, because no such sightings have been reported here.

C-128 link

A couple of months ago I made mention of a new product for the C-128 that had come into the office for beta-testing. The **128-link**, manufactured by Rich-Hill Telecom, is a multipurpose interface for the C-128: 'multipurpose' because it provides for the connection of both IEEE interface drives and parallel interface printers. The package consists of an interface that plugs into the expansion port (a modified version of RTC's **64-link**) and a ROM chip replacement mounted on a circuit board.

As you might have guessed from that last description, this interface requires a bit of installation. The manual includes diagrams and instructions, but both could be better than they are now. Nonetheless, I was able to perform the installation in about half-an-hour with no problems, and I'm a far cry from being a hardware hacker. It goes without saying (which is why I'm saying it anyway) that this installation voids your warranty with Commodore. Since this only lasts for 90 days (my toaster has a warranty that lasts longer than that — a promising subject for a future column), this shouldn't bother most of us.

Using the **128-link** is easy. Its there when you turn on the computer, whether you are in 64, 128 or CP/M mode. It can access both serial and IEEE drives in the same session and can alternatively access a serial or a parallel interface printer. At the start of a session you designate one of the two buses (serial or IEEE) as primary. This bus can access up to four drives, as usual. The other bus defaults to the alternative form and allows access to a single drive. This drive can take on any drive number that is not in use on the primary bus, and is designated by you at the start of the session.

The **128-link** was pressed into service immediately. The work-horses in the *TPUG Magazine* office are PETS and 4040 or 8050 drives. These were connected to my trusty 128 with no problems. The Mannesmann/Tally printer was the next test. NLQ in pica print has never been easier. Software compatablilty is a bit more of a problem. While the **128-link** does not interfere with CP/M mode (in fact, I was even able to get CP/M to boot from a hard-disk!), it does seem to cause problems with a limited number of programs. Most of these programs involved direct disk access, consequently incompatability was not unexpected. Other programs developed gremlins. None of these programs, however, became unusable.

The best feature of the **128-link** is the responsiveness of the manufacturer. Suggestions about ways to improve **128-link**'s performance were both solicited and acted upon. Good products and continued support go a long way towards extending the usefulness of any computer. It is comforting — and hardly surprising — to find out that the C-128 is generating that quality of interest.

CP/M Renaissance

If the promise of 'all that great CP/M software' was one of your reasons for buying a C-128, you've probably been more than disappointed to learn that most of the 'great stuff' (that is, the 'great stuff' you can find) costs as much, if not more, than your computer. The most frequent complaint I've heard is 'But **WordStar** costs $500!'. If rumors prove true that may no longer be the case. A special C-128 version of **WordStar**, priced at $99 dollars, is supposedly in the works. ESC G 2 will keep you posted on the latest flashes.

While good news in itself, a $99 **Wordstar** foreshadows a greater trend. Even in the face of the classification of CP/M Plus as a 'mature product' by DRI (mature products are the orphans of software!), the release of the C-128 may help this operating system to flourish rather than perish. On top of the hordes of new hackers (and Commodore hackers are a breed apart), the large installed base of these computers in the hands of consumers accustomed to low-cost software is helping to drive down prices — prices that had forced high-qualtiy, versatile products to stagnate. The inclusion of the C-128 as an option in the installation menu of many CP/M products (many of which come in under the $99 mark) is evidence of the impact that the C-128 is having.

Learning to crawl

From the feedback I've been getting, both at the magazine and through the C-128 meeting, it appears that CP/M mode remains a mystery to many C-128 users. This is hardly surprising, given the disparities between it and Commodore DOS. To compound the problem, CP/M documentation in the *C-128 Users Guide* is decidely sparse. While many good books are available detailing the use of CP/M, few of them include in-depth discussions of CP/M Plus (most were written when CP/M 2.2 held sway), and even fewer (read none) detail the implementation of CP/M on the C-128.

This void has just been filled by a recently released book from Abacus Software. The latest addition to their line of C-128 books, *The C-128 CP/M Users Guide* does an admirable job of introducing CP/M on the 128. Targeted at the Commodore user without CP/M experience, this book starts from square one. It details characteristics unique to the 128, as well as providing a more advanced look at the CP/M commands and the Z80 ROM in the C-128. You will undoubtedly advance beyond the contents of this book fairly rapidly but, to my mind, this speaks well for the layout and pace of this *introductory* guide. If you're just getting started, this one is a must.

Because many C-128 users are just getting started, either in computing in general or CP/M in particular, there is a lot to learn. Please let me know of the direction you would like to see this column take, as well as the scope of the topics we should cover.

Footnote

As promised, honourable mention goes to Dr. M. Tucker Brawner of Savannah, Georgia for correctly identifying the source of this column's name as being "...derived from the CP/M mode screen attribute command **<esc> G2** which initiates 'blinking' display. Ergo... Flash". Congratulations, Doc, good to see someone's on their toes. □

THE COCKROACH TURBO-ROM

A plug-in replacement ROM for the 64, 128 and SX64

- **Fast LOAD, SAVE, VERIFY, FORMAT**
- Innocent SAVE with REPLACE
- Screen dump during execution
- DOS Wedge:

disk status; non-destructive directory with wild cards; scratch, rename, initialize, validate; clear directory and free BAM; change disk device number; @P/@: open 4,4:cmd4/print #4: close4; load with SHIFT/RUN-STOP; cold start, unnew; compatible with VIC switch (LOGO loads in 35 secs); has no tape or RS232 routines but don't worry; you get two ROMS in one - TURBO plus original kernal - toggle between them

$60. CDN. $45 U.S.

Cheque, money order, VISA or MASTERCARD

kobetek systems limited
1007 Commercial Street
New Minas, N.S. B4N 3C4
(902) 678-9800

Ask Someone Who Knows

If you enjoy **Jim Strasma's** many books, and his articles in this and other magazines, you'll be glad he also edits his own highly-acclaimed computer magazine, now in its sixth year of continuous publication. Written just for owners of Commodore's many computers, each **Midnite Software Gazette** contains hundreds of brief, honest reviews.

Midnite also features timely Commodore news, hints and articles, all organized for instant reference, and never a wasted word. Whether you are just beginning or a long-time hobbyist, each issue will help you and your computer to work together effectively.

A six issue annual subscription is $23. To subscribe, or request a sample issue, just write:

MIDNITE SOFTWARE GAZETTE
P.O. Box 1747
Champaign, IL 61820

You'll be glad you did!

ATTENTION COMMODORE 64/128 USERS !!!
INTRODUCING THE RAINBOW KERNAL ROM

We can now offer you an improved Kernal ROM that replaces the stock ROM (U4) in the C64 or C128 ! This replacement has the following features and commands :

BUILT IN:
1. TEN DOS WEDGE COMMAND!
2. FUNCTION KEYS ASSIGNED TO :
F1 ...DIRECTORY (won't erase memory)
F2...SCREED DUMP F3...LISTS PROGRAM
F4...RUN PROGRAM IN MEMORY F5...ENABLE ML MONITOR IF IN MEMORY...
F6...CLEAR SCREEN
F7...DELETE LINE FROM CURSOR TO END.
F8...SYS TO ADDRESS (DEFINED BY =S)
3. DEFAULT COLORS TO YOUR CHOICE!
4. DEFAULT DEVICE CHANGED TO DEVICE 8!
5. 100% COMPATIBLE WITH ORGINAL ROM!
6. 100% COMPATIBLE WITH EPYX FASTLOAD!
7. ALTERNATE FUNCTION KEYS
8. PERSONALIZED POWER-ON MESSAGE
9. RESET,UN-NEW,DEVICE 9 OPTION,DISABLE OPTION
10. + 20 MORE CONVENIENT FEATURE!
11. INSTRUCTIONS FOR INSTALLATION AND COMMANDS INCLUDED (older model C64's with a soldered U4 chip require de-soldering.)

INTRODUCTORY PRICE $ 30.00 US
40.00 CDN-C64-VER
35.00 US
45.00 CDN-C64-MODE (128)

128 MODE VERSION COMING SOON!!!
MONEY ORDER, MASTER CARD,VISA ACCEPTED
PHONE: (604) 792-3437 FOR INFO OR ORDER
...2-9:00 PM.PACIFIC TIME
RAINBOW ELECTRONICS 45421 AIRPORT RD.
CHLLIWACK B.C. CANADA V2P 7K3

Tune your Computer into • MIDI •

If you own a Commodore, Atari ST, or Apple, then you may want to turn it on to the world of Music with a keyboard and interface. We stock the most popular MIDI interfaces (starting at $49.95) plus we have a limited offer on the CASIO **CZ101** Digital MIDI Synth. for only **$269** It's easy to turn your computer into a player piano* with the QRS Player Piano Roll Disks that feature music from 1900's to today (from Scott Joplin to Madonna). Interfaces with a 6 song music disk start as low as **$49.95** and the demo has Gershwin playing Rhapsody in Blue.

We have sequencer software, cables, CASIO and KORG keyboards, music disks, MIDI interfaces, plus more. Write or call for more information.

CALL - 201 - 838-9027
or write us at
MICRO-W. DISTRIBUTING INC
1342B Route 23 Butler, N.J. 07405

DEALER INQUIRES INVITED
*REQUIRES MIDI KEYBOARD

Micro Processes

From 'String Thing' to 'Translate Thing'

by Phil Kemp

As a means of speedily reading lines of data from disk files, Jim Butterfield's **String Thing** has found use in many a BASIC program. A small modification adds another useful function — code translation during file input.

As a user of **Speedscript** (the popular word processing program from *COMPUTE!'s GAZETTE*), I need to incorporate in text files some data from 'normal' files (BASIC source listings, for example). Now, **Speedscript** uses files of type PRG, and stores the text characters as 'screen code' (1=a, 2=b, 65=A, and so on). So, I need to take any file containing text stored in Commodore's 'half-ASCII' code, and copy the data to a PRG file in screen code.

This can, of course, be done in BASIC. Listing 1 shows a straightforward program using a translate table — an array of integer numbers such that, if *i* is the code of an input character and *j* is the code we want output for that character, then **j=t(i)**. In my case, I want each input 'a' (code 65) to be output as code 1, so **t(65)=1**.

This is not difficult. It is, however, very slow, mainly because of the use of **get** to read each input file character. The program in listing 1 takes nearly a minute to process an eight-block disk file.

We can speed the process up by using **String Thing** (available on several TPUG disks). Then we can read whole lines from the input file into a string variable **a$**. In this case, for each line read in, we need a **for...next** loop containing a character translation statement of the form:

```
print#N,chr$(t(asc(mid(a$,i,1))));
```

where **t** is the 'translate table' as before. The BASIC statement shown is complex, and runs slowly. There is a better, faster way available with little effort.

If we look at the short machine code routine that **String Thing** uses to input a line of data, we see that each character is read into the 6502 microprocessor's A (accumulator) register, then stored in the BASIC string variable **a$** in order. If, instead of storing the input byte, we transfer the value to the Y register, then use it as an index to a translate table, we have our code translation. The price? Two extra machine instructions. Now, both input and output operations can be done a whole line at a time, and are fast.

Listing 2 shows the completed translate program. Listing 3 shows a machine language monitor (**Supermon**) view of the new 'input' routine; this is identical to **String Thing** except where noted.

Listing 1 -- Translation in BASIC

```
10 dim t(255): rem allocate translate
   table space
20 rem now fill table
30 for j=0 to 7
40 for k=j*32 to k+31
50 v=32: if j=1 then v=k
60 if j=2 or j=6 then v=(k and 31)-64*
   (j=6)
70 t(k)=v
80 next k,j
90 t(13)=31: t(192)=32
100 input"input file";e$
110 input"output file";f$
120 f$=f$+",p,w"
130 open1,8,2,e$:open5,8,5,f$
140 get#1,a$: rem get bytes in turn,
150 rem and output the translated code
160 ifst=0thenprint#5,chr$(t(asc(a$+ch
    r$(0))));:goto140
170 close 1:print#5:close5
```

Listing 3 -- the INPUT#1 Routine

This code is poked into the tape buffer area, starting at location $0380 (decimal 896).

```
    0380 a0 02    ldy #$02     ;Copy 1st
    0382 b1 2d    lda ($2d),y  ;entry in
    0384 99 89 00 sta $0089,y  ;BASIC's
    0387 c8       iny          ;variable
    0388 c0 06    cpy #$06     ;table.
    038a d0 f6    bne $0382
    038c a2 01    ldx #$01     ;File #1
    038e 20 c6 ff jsr $ffc6    ;CHKIN
    0391 20 e4 ff jsr $ffe4    ;Get byte
    0394 c9 0d    cmp #$0d
*   0396 f0 12    beq $03aa    ;Line end
**  0398 a8       tay          ;Use byte
**  0399 b1 37    lda ($37),y  ;as index
    039b a4 8e    ldy $8e      ;Store to
    039d 91 8c    sta ($8c),y  ;string
    039f c8       iny          ;Store
    03a0 84 8e    sty $8e      ;length
    03a2 c4 8b    cpy $8b      ;Full?
    03a4 f0 04    beq $03aa    ;Y, exit
    03a6 a5 90    lda $90      ;Test I/O
*   03a8 f0 e7    beq $0391    ;Okay
    03aa 4c cc ff jmp $ffcc    ;CLRCHN
```

The code is identical to String Thing, except for the lines flagged:

** These two instructions added.

* These two branches change to accommodate the added instructions.

Listing 2 -- Translate Thing Completed

```
10 rem copy seq file to pgm file with
20 rem character translation by table
30 rem     based on 'string thing'
40 rem **    by jim butterfield     **
50 :
60 rem make room for table
70 poke56,peek(56)-1:clr
80 rem string must be first variable
90 a$="abcdefghijklmnopq"
100 a$=a$+a$+a$+a$+a$
110 a$=a$+a$+a$
120 rem above sets string to max (255)
130 i=peek(56)*256+peek(55):rem start
    of table
140 :
150 rem set up translate table to
160 rem translate cbm ascii to screen
170 rem codes used by speedscript etc
180 for j=0 to 7
190 for k=j*32 to k+31
200 v=32: if j=1 then v=k
210 if j=2 or j=6 then v=(k and 31)-64
    *(j=6)
220 poke i+k,v
230 next k,j
240 poke i+13,31: poke i+192,32
250 :
260 rem **   start  string thing   **
270 rem a$ was first variable defined
280 rem machine code for string thing
290 data 160,2,177,45,153,137,0,200
300 data 192,6,208,246,162,1,32,198
310 data 255,32,228,255,201,13,240
320 data 18
330 data 168,177,55:rem use table
340 data 164,142,145
350 data 140,200,132,142,196,139,240,4
360 data 165,144,240,231,76,204,255
370 rem poke input code to tape buffer
380 forj=896to940:readx:pokej,x:t=t+x:
    next
390 ift<>6120+168+177+55thenstop:rem b
    ad checksum
400 rem end of string thing setup
410 :
420 input"input file";e$
430 input"output file";f$
440 f$=f$+",p,w"
450 open1,8,2,e$:open5,8,5,f$
460 :
470 rem next sys same as 'input#1,a$'
480 sys896
490 rem l=size of input (could be 0)
500 l=peek(142)
510 if st=0 then print#5,left$(a$,l);c
    hr$(31);:goto 480
520 :
530 close 1:print#5:close5
540 poke56,peek(56)+1:rem release tabl
    e memory
```

Other code translations can be achieved simply by changing the values used for the translate table. The only oddity here is in line 780. For most applications, we would not want to change the end-of-line character (**RETURN**). In that case, line 780 can be replaced by the simpler:

```
780 if st=0 then print#5, left$(a$,1): go
    to750
```

A working character code translation program can be had simply by typing the code shown in listing 2. A review of the 'input' code shown in listing 3 may suggest other useful variations; character code translation is not the only function easily done in machine code.

Further, **String Thing** is not the only program that can be adapted to serve new purposes. I suggest that, for the price of a little thought, many programs can be easily extended in function. As a side benefit, the exercise also usually extends our knowledge. □

The MSD: A Reappraisal

by Sean Rooney

It's early September, 1984. Just received: a brand new, MSD SD-2 Super Disk Drive. Methodically you unpack, set up, turn on and tune in. Backing up disks was never easier. It was the best of times.

It was also the worst of times, and since then I've been forced to replace my MSD-2. Why, you ask? Because it's not as great as its proponents claimed it to be. Designed for the C-64, C-128, VIC 20, Plus/4 or PET, it is indeed a fine drive — when it's working. My MSD-2 started to do strange things about ten months into its first year.

At first the MSD's problems seemed to be innocent enough (as with icebergs only 90 per cent of the upcoming catastrophe was visible). It would start its initialization procedure on power-up normally, but about a third of the way through it would lock up (and stay that way). Turning it off and on repeatedly would eventually fix it.

It used to be that I could run disk-intensive software like COMAL or **The Print Shop** with no problem. Now they crash my drive. The MSD also frequently crashes during a backup, a disk collect, a load, or just about anything else. These crashes are not all the same. Sometimes, it will go unnoticed until I find that the MSD will not catalog when my C-128 tells it to. At other times, effects include things like having all the lights flash like crazy in totally random patterns — spectacular but nerve-wracking.

Seeking informed advice, I talked with (TPUG's) Louise Redgers about the problems that other MSD users have had. The MSD-2 seems to have a flaky (read cheap) power supply. Mine probably has this fault and a loose ROM socket to boot. I say this because tapping the casing with my fingers will sometimes cause the drive to work again.

After experiencing these misfortunes I've decideed to replace the MSD-2 with a more reliable disk drive (CBM-4040 with Quick-Silver Interface, or 2 1571s).

Potential users be warned. The MSD-2 may be much more useful than the 1541, but only if it works. □

Expansion Port Extender

by Richard N. Dawson

If you find it awkward to plug in your expansion cartridges, here's a neat, inexpensive solution.

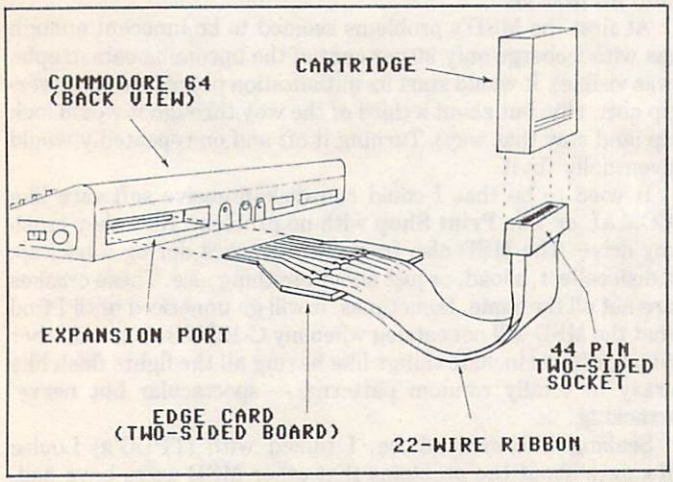
In an effort to hide all those ugly wires coming out the back and sides of the C-64 (for the datasette, disk drive, monitor, power cord and so on), I built in a custom panel that butts up against the back of the keyboard. The good news was that I succeeded in hiding all those cables, but the bad news was, I made it very awkward and difficult to plug in program cartridges such as **Music Composer**, **Magic Desk** and the new **COMAL Cartridge**. I fantasized some type of an extension system and searched to see what was available commercially. I found nothing but large, rigid, multisocketed and expensive systems.

So I set out to build my own Expansion Port Extender. An acquaintance told me how to make a 22-pin edge card by etching a two-sided copper clad circuit board. So I went off to my local Radio Shack store, and after a lot of consideration and examining products, I purchased the following items:

- 50 pin card edge connector (Part 276-1566, $4.95)
- Copper clad boards (Part 276-1591A, $1.98)
- Printed circuit board (Part 276-1576, $8.95)
- 25 cond ribbon cable (Part 272-772, $3.59)

Procedure

First, I had to customize the 50-pin socket by cutting it down to a 44-pin socket (22 per side). I used a fine-tooth hacksaw and carefully cut off the end three pin sockets. I also had to shave off



approximately 1/8 inch from the other end. I found it helpful to use an existing cartridge to make the measurement marks on the socket. I have heard that a ready-made 22 pin socket can be purchased for about twenty dollars (Cdn.), but I was unable to locate a vendor.

Next, I measured an existing edge card from one of my cartridges and cut the circuit board down to size. I found that using a large pair of tin snips worked well and did not rip up the copper surface. Since I am not an expert at soldering, I designed the edge card to widen the pin circuits so I could more easily solder wires to points that were more spread out. Again using an existing cartridge, I measured and marked the exact location of each pin on the edge of the circuit board. I found the general instructions in the etching kit enlightening (I was a first-time etcher). Using the special pen, I carefully drew each pin location and then extended each into a widening circuit. I followed the kit instructions regarding the time to bathe the board in the etching fluids and presto! — my new edge card was created.

Next, I carefully stripped each wire of the 22-wire ribbon and soldered all the points. I was advised not to use very much heat, which might damage the copper on the board. When finished soldering, I cleaned all points and surfaces, and then, point-by-point, tested each separate circuit to see if there were any shorts between pins, or broken circuits. Since all checked out okay, I then plugged the new edge card into the new socket, being careful not to push the cartridge in upside-down. I turned on the power, and there it was, the program cartridge!

Finally, I applied a coat of liquid plastic to the new edge card, making sure I kept the plastic away from the edge pins. I finished up the project by making a small but strong stand for the new socket. I wanted it to withstand the pressure of forcing on a cartridge.

One last hint: make a little cap out of plastic to place over your new socket to keep out dust and (more important) staples or paper clips, all of which can cause a short circuit and damage your computer.

I now have a new location to plug in my cartridges. It has proven to be far more 'user-friendly' for my children to use.

Note to readers: any hardware project, carries with it the risk of damage to your computer if improperly carried out. Please do ***not*** *attempt the project described in this article unless you are sure you know what you are doing. -Ed.* □

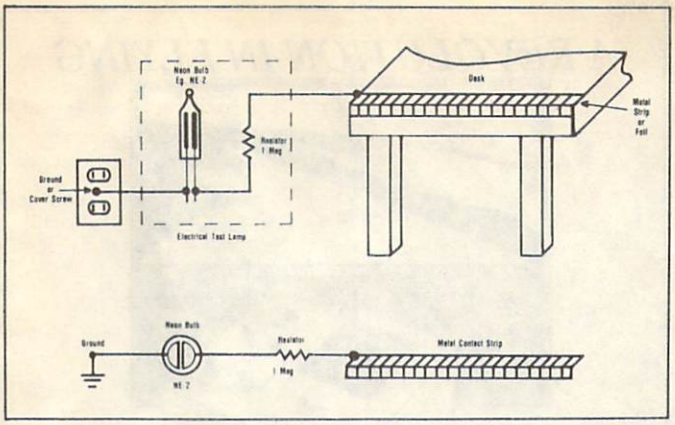
Reducing Static Shock

by James M. Ardovitch

Static electrical shock can be hazardous to your computer. It can cause loss of data, lock up the computer so that you have to power down and start again or, worse yet, cause permanent damage, making it necessary to pay a repair bill. Luckily, there is a way to minimize the hazard of static shock.

At your local hardware store or lumber yard you can obtain a length of stainless steel of the type used for counter edging, stair nosing and so on. This comes in various sizes and shapes to fit any computer station decor. Attach this stainless steel strip to the front edge of your desk, printer stand, or even the doorway to your room — whichever place offers a convenient location to touch with your hands before touching any electronic equipment.

Onto this strip of stainless steel attach a length of wire (#16 or larger), with a one megohm resistor in series. Attach this to ground — a cold water pipe or the screw in the electrical outlet box (if this box is electrically connected to ground through conduit or bx). For even more visual notice of static discharge, connect a neon bulb in this series arrangement. For this purpose you can also use an electrical outlet tester (one that has two leads



with a neon bulb and dropping resistor) to ground the metal strip. The resistor is necessary to prevent you getting a good zap of static charge; also a pulse isn't generated on the computer. □

Cartridge COMAL Emulation With COMAL 0.14

by Victor Gough

A number of members have asked how to use some of the programs written with the cartridge version of COMAL on their disk version, which lacks some of the cartridge's commands. As more of the advanced COMAL programers acquire the cartridge, this problem is increasing.

COMAL is an extensible lanquage. This means that you can add new commands to the language as procedures! Let's take as an example the command **VAL**, which is in the cartridge version but not the disk version.

VAL (X$) returns a value corresponding to the numeric equivalent of the string ordinand (for example, **VAL ("5") = 5**), or 0 if the string is not an ordinand. This command may be replaced in the disk version by the following procedure:

```
FUNC val (a$) CLOSED
s:= 0
y:= "." IN a$
l:= LEN (a$)
IF y <> 0 THEN y:l-y+1
FOR x:=1 TO l DO
 IF a$(x) IN "1234567890" THEN
  s:=s+(ORD(a$(x))-48)+(10EXP (l-x-y))
 ELSE
  y:y = y-1
 ENDIF
ENDFOR x
RETURN s
ENDFUNC val
```

Notice that the procedure is closed, so all the variables are local and have a separate identity outside of the procedure. You may not want the procedure cluttering your program listing since it is supposed to act like a command. You can hide it by renumbering it above 10,000.

This procedure and many others can be found on disk (K)B6. Further information about the procedures on this disk can be found in Kevin Quiggle's *The COMAL Library of Functions and Procedures*, which is available from the COMAL Users Group, USA. □

Merging Program Files

by M. Garamszeghy

In my BASIC programming, I use a library of standard subroutines, or program modules, to perform such tasks as selectively reading a disk directory, bubble sorting, disk file handling, screen formatting and graphics. Much of the initial code for a new program can be produced by combining some of these standard subroutines. Customization and optimization are relatively minor tasks compared to retyping the same old subroutines every time you want to use them.

The whole concept of combining several subroutines or program files into a single program usually depends on the existence of a BASIC command called **merge**, which performs this task automatically. Although Commodore DOS supports the combining of two or more data files (**append** and **concat** to name two methods), a **merge** command for BASIC programs is not available as a built-in command, even in the C-128's powerful BASIC 7.0. (One of the few sore points I have about my new C-128 is its lack of a **merge** command. Commodore could have easily implemented this command in BASIC 7.0 but for reasons unknown to mere mortals decided not to. After all, a **merge** command is far more useful than two separate commands (**directory** and **catalog**) for displaying the disk directory). **Merge** is available, however, with several BASIC extensions such as the Programmers' Aid Cartridges for the VIC 20 and C-64.

Despite this, it is very simple to combine any number of program modules into a single BASIC program. A very simple method that I recently developed for my C-128 follows, using the programmable function keys. This short BASIC program redefines function key **f2** and saves the new key definitions in a disk file called **merge**. Enter and run this program first to create the **merge** file. To use it, follow these three easy steps:

1. Before you start an editing session, type in **bload"merge",b0** to retrieve the special function key defintions.

2. Load the first program or enter it from the keyboard.

3. When you want to merge a previously-saved file, type in **me$ = "filename2"**, *then press* **f2** instead of the **return** key. This step can be repeated as often as you wish without having to reload the **merge** definition file each time.

That's all there is to it. A series of BASIC commands will be printed on the screen and executed while the disk drive comes on for a moment. The method is based on a series of immediate mode commands that are stored as a function key definition. First, the *start of BASIC* pointer is reset to the end of the first program. The second program is then loaded in at the end of

the first. The *start of BASIC* pointer is then restored to its original value and the combined program is ready to run.

```
10 rem functionkey merge setup by m. gor
   amszeghy
20 cr$=chr$(13):a$=cr$+"poke250,peek(45)
   :poke251,peek(46):x=65278-fre(0):poke
   45,xand255:poke46,x/256"+cr$
30 b$="dload(me$)"+cr$+"poke45,peek(250)
   :poke46,peek(251)"+cr$:key2,(a$+b$):b
   save"merge",b0,p4096top4352
300 next:x$=x$+chr$(34):return
```

This method is perhaps the easiest way to implement a merge command on the C-128. In addition, this method gives you the fully merged program in RAM ready to run. There is one relatively minor restriction on the line numbering of the programs being merged. Since the merge works by appending all of the lines in one program module to the end of another, the line numbers of the second module must all be greater than the line numbers of the first. If you do not adhere to this restriction, some odd things may happen to the merged program. For example, when you list the program you may find that line 50 comes after line 100, or before line 20. This can be avoided by ensuring that the line numbers are in correct sequence before merging the programs.

It should also be noted that this method of merging does no error checking, so make sure that the file you specify as **me$** actually exists on your current disk. Not to worry though, a simple **run/stop-restore** key sequence will abort the function key commands if necessary. □

Watch That Notch!

by Karl Thurber

I rarely 'double-side' my disks, and generally use single-sided, double-density disks in my 1541 disk drive. However, I occasionally purchase double-sided, double-density (DSDD) disks when I want to put related programs and data back-to-back. This usually requires that a write protect notch be cut out on the opposite side of the disk, a task easily accomplished using a small hand punch.

However, I recently had the experience of being unable to write to the back side of a disk so modified, even though I was using a certified DSDD disk. After much consternation when the disk wouldn't format or otherwise respond , I finally realized that the hole I had punched was not *exactly* in the right spot, having 'eyeballed' its location rather than locating it precisely! This misalignment of the write protect notch caused the disk drive to think that the notch was covered, or that the disk was single-sided. The fix was simple enough: a slight enlargement of the hole was all that was required.

When punching out a new notch, it's best to take another disk and flip it over so as to provide a 'stencil' for the punch-out. Simply lay this disk over the one to be punched so that the notch is positioned properly over the intended hole, and lightly ink-in the notch using a red felt-tip pen. This will give you the exact spot to punch out. □

Products Received

Presented by Astrid Kumas

The following products have been received by TPUG Magazine *in recent weeks. Please note that these descriptions are based on the manufacturers' own announcements, and are not the result of evaluation by* TPUG Magazine.

World Geography

World Geography from Bobco,200 7th Ave., Suite 111, Santa Cruz, CA 95062. Price: 24.95 (US) plus $2.00 for shipping and handling.

World Geography is an educational, trivia-type game for the Commododore 64 which provides information about capital cities, location, language and currency of 175 countries.

Throughout the game, the right half of the screen displays the world globe, while the left window shows the blown-up shape and flag of a particular country. The player can choose the geographical area from which the computer will select five countries for each round of the game. When the country is found, the player is asked to specify the four particulars mentioned above. The correct information can be selected from the multiple choice window at the bottom of the screen. A correct response awards a player with the timer value multiplied by 10. The game is over when the player has no countries left or all of the countries have been played.

World Geography can be played by one or two players. It offers three difficulty levels: the selected level determines the speed of the timer. Players can also select the review mode in which the computer automatically displays each country with the correct answers.

Spell of Destruction

Spell of Destruction from Mindscape, Inc., 3444 Dundee Road, Northbrook, IL 600062. Suggested retail price: $29.95 (US).

Spell of Destruction is an adventure game for C-64/128 computers which is played with the combination of the joystick and the keyboard.

In the game the player is to identify himself with an apprentice wizard called Drinn. As Drinn, he or she is about to face the ultimate trial on the road to mastery — The Castle of Illusions. The magnitude of Drinn's task is best explained by the Loremaster himself:

"In a few moments you will be transported into The Castle of Illusions, a place where you will meet every kind of demon and your skill both in fighting and sorcery will be tested to the full. Your task is straighforward, for you shall simply have to seek out the Prime Elemental in this minor city of illusions, and destroy it with a single spell... You shall enter the Castle with a supply of spells in your possession and, of course, a goodly supply

of Fireballs and your trusty sword. ...there are other spells fo be found around the castle as well..."

Spell of Destruction features over 70 locations, "real-time" action combined with problem-solving and strategy and "motion picture" musical score that reflects the action as it happens.

Three Games on One Disk

Brian Bloodaxe,Revelation,Quo Vadis from Mindscape, Inc., 3444 Dundee Rd., Northbrook, IL 60062. Suggested retail price: $14.95 (US).

For those who like classic arcade/adventure type of game this new product for the Commodore 64/128 from Mindscape will probably be a treat. The disk contains not just one but three different games: **Brian Bloodaxe**, **Revelation** and **Quo Vadis**. The last two games require the use of joystick.

In **Brian Bloodaxe** the player will invade Britain and try to find the Crown Jewels. While on this mission, he/she will have to pass through 100 puzzle-filled screens.

Revelation sets up another challenge: the goal is to destroy the Monster of the Apocalypse hidden in one of the caverns. Before the final confrontation, the player must get through forty five caverns filled with evil creatures and deadly towers. The game offers different levels of difficulty: the higher the level, the further into the caverns will be the start of the quest.

The last game on the disk, **Quo Vadis**, takes the player on a quest to combat the Dark Lord and free humankind from his shadow power. To complete the task the player must find the Sceptre of Hope and learn the Words of Power. There are more than a thousand screens to go through. Demons and riddling clues block player's way at every path.

This game presents the additional challenge of taking part in the **Quo Vadis** contest and winning the Sceptre of Hope, worth approximately $10,000. The contest requirement is to discover all of the riddles in the game. Solving each riddle will take the player one word closer to the solution.

Windows

Windows from St. Mars Systems, Inc., 1400 Clay Street, Winter Park, FL 32789.

Windows, a screen processor utility for C-64, has been designed to give the user windowing capabilities in BASIC or assembly language programs.

A window is simply a rectangle on the screen. Various rectangles, each with a single menu option written on them, can be created using this utility. These windows may then be overlaid over each other to simulate a filing cabinet, distributed over the screen, scrolled etc.

The program features many options available to the user and some of them are as follows: complete position placement control; window colour control, border presence/absence control, independent window scrolling, internal error checking, no limit on number of windows, screen colour control, cursor positioning control, formatted data input routines, window move commands.

Floppy Stopper

Floppy Stopper from The Floppy Stopper Co., Box 53268, Baton Rouge, LA 70805. 1-800-222-7867.

Here is some good news for those who are tired of having to scrape, tug, pull and struggle with sloppy, sticky write protect adhesive tape tabs. The Floppy Stopper Co. is now producing floppy diskette with an on-off write protect switch. What's more: both diskette and switch have a lifetime guarantee. □

Library Additions

TPUG's library of public domain software grows month by month. Hundreds of disks containing thousands of programs are available to TPUG members at the nominal cost of ten dollars per disk. Considering that each disk is packed with good programs, at today's software prices, this is a fantastic value.

In order for the library to keep growing, our librarians need a constant supply of new programs. If you have written a program or a collection of programs that you think might be an asset to the library, please send it to: TPUG Program Library, 101 Duncan Mill Road, Suite G7, Don Mills, Ontario M3B 1Z3, Canada. If your contribution is accepted, you will be sent the library disk of your choice. If, for some reason, your contribution is not needed, your original disk will be returned to you.

C-64 disk (C)AB

Presented by Derick Campbell

The April disk is not quite as full as it should be. Program submissions are dropping to an all time low.

Luscher test.c gives you a personality description based on your colour preferences, with some added tips on how to behave in certain situations. **hrg** stands for High Risk Game — a simulation of Risk on the computer for 2 or 3 players.

```
0 "tpug apr86 (c)ab"
5    "list me.l"          p
10   "luscher test.c"     p
283  "luscher data.d"     r
37   "hr-boot"            p
66   "hr-instr"           p
100  "hrg"                p
53   "library index.c"    p
8    "amiga demo.c"       p
10   "fadein/out"         p
40   "amiga"              p
```

It involves strategy and luck; and it comes with full documentation! **Library index.c** is a program filer. The good thing about it is that it stores the programs in headings: games, utilities, and so on. This makes it possible to store the files on more that one disk, reducing the possibilty of running out of space on a disk.

Amiga demo.c is a graphic representation of the one and only Amiga, with it's famous bouncing ball, on the 64.

A couple of quick notes: submittors should give themselves credit for their program; put name and member number or address in the first few lines of the program. That way we can send you a disk! Also, one thing that prevents a program from entering the library is the lack of documentation. If your program is missing documentation, make some up now!

VIC 20 disk (V)AA

Prepared by Richard Best

The past year has been a good one for the VIC library, despite the VIC's disappearance from the marketplace. Orders for disks and tapes have been steady, and TPUG has every intention of supporting this terrific little machine. However, our reserve of contributions is getting dangerously low, and new submissions have been reduced to a mere trickle. And the majority of VIC programs are being sent in by just a handful of members. The VIC library needs your support in the form of donations. So get on that idea, write the program and send it in. Remember, we will send you the disk/tape of your choice for your submission.

This month VIC owners are getting something special: two disks and a tape. On top of the regular monthly release, we are adding our first 'freeware' disk and a special issue tape.

The March disk, (V)AA starts out with a unique progam called **bingo caller**. In addition to calling bingo numbers, it keeps track of called numbers on a scoreboard. Games can be started or stopped at any time. **Latin bowl** is an educational game similar to 'Reach for The Top'. The program acts as the timekeeper for two teams who must answer questions you provide. For less educational fun, try **bombardier**, a familiar depth-charge program that runs with a Superexpander.

Sheriff and **super fight** are clever demos done in hi-res. They run on an unexpanded VIC and should provide the kids with a bit of fun. If you need even more amusement, try **dyslexia**, a pseudo-utility that is guaranteed to keep you awake. Another demo here is **Canada map** for the Superexpander. (The TPUG offices are located directly under the flashing dot.) If you're a fan of 'electric wallpaper', **kaleidoscope** will provide endless patterns, also using a Superexpander.

Chemistry tutorials are not very common, and this month we have two. The first, **ph titration**, will calculate and plot the titration of PH, and the other does the same for CL. **Algebra** will drill you on two-variable equations.

More educational amusement is provided by **synthi-20**, a music generator that has five different sounds and sets up the keyboard like a real piano. If you enjoy working in the kitchen, try **butter tarts** which is a recipe for... butter tarts.

```
0 "tpug(v)-aa 03/86"
9    "list-me (v)-aa/1" p
8    "list-me (v)-aa/2" p
6    "bingo caller.v5k" p
6    "sheriff.v5k"      p
7    "algebra.v5k"      p
2    "pixel map.v"      p
12   "latin bowl 2.v"   p
11   "super fight.v"    p
2    "dyslexia.v"       p
8    "synthi-20.v"      p
6    "ph titration.v"   p
10   "cl titration.v"   p
2    "kaleidoscope.vsx" p
9    "canada map.vsx"   p
12   "bombardier.vsx"   p
17   "butter tarts.v"   p
2    "constrictor.v12k" p
43   "finances.v12k"    p
9    "squeezer.v12k"    p
12   "backup 2.v12k"    p
4    "tiny dir print.v" p
6    "dir"              p
```

This month's utility department is quite full. Always useful are programs like **constrictor**, which hides 8K or 16K expanders, and **pixel map**, which will print out the value and bit pattern of any memory location.

Disk owners will have a great time with **finances**, a menu-driven package that will perform all of your routine business calculations, including amortization, future values, compound interest etc.

Also disk-based is **squeezer**. Translated from a PET program, this wonder removes idle space and concatenates program lines, and can reduce a program's memory requirement by over 10%.

Two of the handiest disk utilities I have seen are **backup 2** and **tiny dir print**. **Backup 2** is a pro-style copy program that copies a disk block-by-block. It includes a **format** routine and will copy as many blocks as possible in each pass. **Tiny dir print** produces a printed copy of a disk directory in tiny print in two columns. The entire directory will fit on the front of a disk envelope!

VIC tape (V)AB

Fat-40 appeared on (V)TJ last year, but it would not run from tape. Tape users may now order (V)AB which will boot **fat-40** and its demo programs from a datasette. This tape contains only the 40-column program, and requires a utility to make copies.

VIC disk (V)AC

Our first offering in the freeware category is a disk-based file manager called **infer-structure**. The program will run on any machine but this version was tailored for the VIC. It's very powerful, has a large capacity, and can search key fields amazingly fast. Included is an 18

```
1 "tpug (v)-ac free"
37   "inftpg/85a.txt"    s
12   "infbot/2.1.v"      p
2    "infstr/2.1"        p
143  "infusr/2.0.txt"    s
4    "list.me"           p
```

page user manual (required reading before using the program) and an index of *TPUG Magazine* articles. All freeware notices are contained in the opening screens.

C-128 disk (Y)AAA

Presented by James Kokkinen

Welcome to the third C-128 monthly disk. It differs in a few ways from the first two disks. For starters, it is the first C-128 disk to feature TPUG's new numbering conventions. Do not confuse this disk,

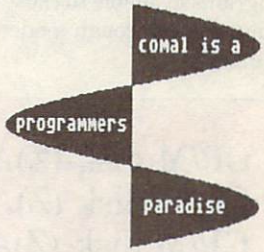
```
1  "may '86 (y)a     "
6    "list-me(y)aaa"     p
1    "....games....."    p
25   "grungy towers.y"   p
52   "adventure c7.y"    p
2    "graphic 1 dump"    p
5    "advf 1"            s
5    "advf 2"            s
4    "advf 3"            s
3    "advf 4"            s
4    "advf 5"            s
6    "advf 6"            s
3    "advf 7"            s
4    "advf 8"            s
4    "advf 9"            s
18   "advshor"           s
2    "advf 10"           s
2    "advf 11"           s
2    "advf 12"           s
2    "advf 13"           s
3    "advf 20"           s
7    "advf 21"           s
6    "advf 22"           s
5    "advf 23"           s
6    "advf 24"           s
7    "advf 25"           s
11   "advf 26"           s
6    "advf 27"           s
5    "advf 28"           s
1    "advf 29"           s
7    "advf 31"           s
8    "advkeys"           s
9    "advitm"            s
21   "advmap"            s
9    "advf 0"            s
1    ".....misc....."    p
85   "c-128 tutorial"    p
22   "trivia c-128.y"    p
26   "isoplot 128.y   "  p
20   "calculator 128.y"  p
1    "...utilities..."   p
13   "128 incl'dir.y"    p
8    "disk tidier c7.y"  p
69   "dd128.img.y"       p
8    "tri directory.y"   p
19   "disk guard c128"   p
54   "superdisk128.y"    p
9    "seq read(128/80"   p
14   "1571dd.bin.y"      p
1    "...terminal...."   p
9    "terminal.c1/128"   p
8    "term.c1/128"       p
31   "128 xmodem.cor"    p
```

(Y)AAA, with the first disk, (Y)AA. It's a small change, but it will save running into numbering problems for a long time. Next, this is the first C-128 disk that can be considered full, having only 5 blocks of available space free. Thanks to all who have submitted programs. Please keep them coming in.

Easy Curves

1) Insert your COMAL disk in drive*.
2) Type LOAD "C64 COMAL*",8
3) Type RUN (starts COMAL)
4) Type AUTO (turn on auto line#'s)
5) Enter the program lines shown below (COMAL indents lines for you)
6) Hit RETURN key twice when done
7) Type RUN

```
0010 setup
0020 curve
0030 paint'it
0040 add'words
0050 //
0060 proc setup
0070   black:=0; yellow:=7
0080   background black
0090   pencolor yellow
0100   setgraphic 0 //hi res screen
0110   hideturtle
0120 endproc setup
0130 //
0140 proc curve
0150   moveto 110,0
0160   drawto 110,199
0170   for row:=0 to 10 step .03 do
0180     drawto 110+99*sin(row),row*20
0190   endfor row
0200 endproc curve
0210 //
0220 proc paint'it
0230   fill 120,20
0240   fill 100,90
0250   fill 120,180
0260   fill 100,198
0270 endproc paint'it
0280 //
0290 proc add'words
0300   pencolor black
0310   background yellow
0320   plottext 120,155,"comal is a"
0330   plottext 16,90,"programmers"
0340   plottext 120,30,"paradise"
0350 endproc add'words
```



Notice how easy graphics are in COMAL. Lines 70-100 set up the screen colors. Lines 150-190 draw on the screen. Lines 230-260 fill (paint) whole parts. Even putting text on the graphic screen is easy. See lines 320-340. All this is standard and built in as part of COMAL. Plus a full turtle graphics system. Now you know why there are 100,000 users.

* If you don't have COMAL yet, order a Programmer's Paradise Package-$19.95. It includes the complete COMAL system plus over 400 pages of information. Add $5 more to get our 20 interactive lesson Tutorial Disk. Add $2 shipping. Visa/MC or US funds check accepted. Send to:

COMAL Users Group USA
6041 Monona Drive, Room 109
Madison, WI 53716
phone 608-222-4432

Here is this month's offering.... Following **list-me(y)aaa**, we have two adventure games converted for C-128 use from TPUG's extensive C64 library. These are **grungy towers.y**, and **adventure c7.y**. Both run in 80-column format and will provide hours of entertainment for adventure buffs. **Graphic 1 dump** and all of the **ADVF** sequential files belong to **adventure c7.y**. **C-128 tutorial.y** runs on a 40-column screen and provides excellent understanding of C-128 mode using the example programs from your operating manual for demonstration. **Trivia c-128.y** contains several screens of useful information. **Isoplot 128.y** draws isometric pictures on your 40-column screen. Add to your math collection with **calculator 128.y** which performs several mathematical functions by following the prompts. **128 incl'dir.y** provides information regarding start and ending locations of directory entries. **Disk tidier c7.y** is yet another program to scratch unwanted files from a diskette. **Dd128.img.y** is the C-128 version of Disk Doctor. It runs in 40 column only. **Tri directory.y** prints 3-column directories in tiny form if you have a Gemini printer. **Disk guard c128** helps protect your diskette from accidental erasure. Use **superdisk128.y** for multiple utility functions in 80-column mode and **Seq read(128/80)** to read sequential files utilizing your 80-column screen. **1571dd.bin.y** allows copying using 2 1571 drives. We have included **terminal.c1/128** as a new addition to the many programs available to those of you who communicate through modems and the phone system.

CP/M disk (Z)AD
CP/M disk (Z)AE
CP/M disk (Z)AF

Presented by Adam Herst

This has been a busy month for the CP/M library and a busy month for me, the CP/M librarian. Three disks were released this month and in keeping with the CP/M library, all disks are theme disks.

Many of the files on these disks are contained in library (**.lbr**) files and some are in a squeezed (**.xQx**) format. To turn these files into a recognizable form, they will first have to be extracted from the libraries with **lu.com** and then unsqueezed with **usq.com** or **nswp.com**. More detailed instructions are contained on each disk. **Lu.com** and **usq.com** are available on disk (Z)AB.

Before using any of the TPUG CP/M disks it is a good idea to copy them to a fresh disk. The reason for this is that the CP/M disks are copied on a 4040 drive and the resultant disk may not be write compatible with your drive. This is true for any TPUG disk and is a safe procedure to follow with all of them.

This disk is a utility disk and is crammed full of programs, some of them written expressly for the 128. **C1571/2.com** is version two of **c1571.com**. It speeds up the write speed of a 1571 in CP/M mode by disabling some redundant verifying. Version one did not support MFM formats. Type **C1571** for more info. **Fcopy.com** is a file copy program written for the C-128.

zad/type	me	bishow	com
c1571/2	com	cpm2plus	lbr
cpm3-cat	lbr	cpm3util	lbr
d	com	d	doc
delbr	cqm	eraq	com
erase	com	fcopy	com
index	lbr	printer	lbr
qs-cpm3	lbr	regions	sqg
scan	com	sd	lbr
sq/usq	lbr	unerapl	lbr
wipe	com	xtype	com

A variety of disk utilities are included. **Eraq.com** is an erase with query program. **Erase.com** is another of the same. **Scan.com** locates bad sectors on a disk. **Wipe.com** wipes **.bak** etc. files from a disk. **Cpm3-cat.lbr** is a CP/M Plus program to generate and maintain a catalogue of disks. **Sd.lbr** is a small library of the **sd.com** (Super Directory) files. The name says it all. Docs are included. **D.com** keeps a chronological record of file access. See **d.doc** for instructions. **Qs-cpm3.lbr** sets disk file attributes. It's shorter and easier to use than **set.com**. **Unerapl.lbr**, like a Commodore DOS unscratch, will recover erased files.

Cpm2plus.lbr is a library of files that will let you run CP/M 2.2 specific programs under CP/M Plus. **Cpm3util.lbr** is a library of CP/M Plus utility programs including a disk editor, directory lister, text editor and more. See the included documentation for more info.

To help you extract library files and unsqueeze them, variations of already released utilities are included. **Xtype.com** will type out squeezed (**.xQx**) files. **Sq/usq.lbr** is a library of yet another version of the squeeze/unsqueeze files. This one rejects files that will not benefit from squeezing. **Delbr.com** will remove files from a library file.

To view and manipulate text files, the following utilities are available. **Bishow.com** types a file to screen, letting you scroll backwards and forwards through the file. **Printer.lbr** contains files to set the options on some Epson and Okidata printers.

Index.lbr contains a program that will generate an index to a **WordStar**-style document. See the included documentation.

zae/type	me	e-basic	lbr
e-basic	doc	e-bgames	lbr
othello	lbr		

As promised a number of public domain languages have been added to the CP/M library. CP/M disk Z(AE) contains files for a version of BASIC. **E-basic.lbr** contains a number of versions of Gordon Eubanks public domain EBASIC, a BASIC interpreter. See **ebasic.doc** for elementary instructions. **Ebgames.lbr** contains a number of games that will run under EBASIC as supplied on this disk. Only the **.int** files are included, no source code. Follow the directions in **ebasic.doc** to run them. As you may expect, they are text based games. **Othello.lbr** is the game of othello designed to run under EBASIC as supplied on this disk. Source code (**.bas**), **.int** and **.doc** files are included.

zaf/type	me	kermit	lbr
mex	lbr		

CP/M disk Z(AF) is TPUG's second telecommunications disk. **Mex.lbr** contains the **mex.com, mex.hlp** files (a full featured, professional style, telecommunications package), as well as a number of documentation files. **Mex110.ws** is a WordStar-style manual. It can be printed using **vdo.com** on TPUG disk (Z)AC. The terminal contained in **Kermit.lbr** has fewer features than **mex.com** but it does support the KERMIT protocol for file transfers, widely used for micro to mainframe communications. Documentation is included.

Thats all for this month, I hope it keeps you busy until our next releases. Remember, if you collect programs from some of the other CP/M user groups, share them with your fellow TPUG members. We all depend on each others contributions and it entitles you to free disks in return. □

Software order form

NAME _____

STREET ADDRESS _____

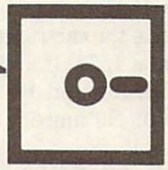
CITY/TOWN/P.O. _____

PROV/STATE _____ POSTAL/ZIP CODE _____

TELEPHONE _____ MEMBERSHIP NO. _____

TORONTO PET USERS GROUP, 101 Duncan Mill Road, Suite G7, Don Mills, Ontario M3B 1Z3 416-445-4524

disks



To order club disks by mail, send $10.00 for each 4040/2031/1540/1541 disk (4040 format), discount price 5-10 $9.00 each, 11 or more $8.00 each; and $12.00 for each 8050/8250 disk (8050 format). We do honour purchase orders from school boards.

These disks are for use with a _____ computer and a _____ disk drive.

Please send me the following:

3 Letter/No. Code	Description	4040 or 8050 Format	Price

Total $ _____ .00

tapes

To order VIC 20 or Commodore 64 library tapes, send $6.00 for each tape.
To order PET/CBM or Commodore Educational Software, send $10.00 for each tape.

These tapes are for use with a _____ computer and a datasette.

If for a PET computer, what model - _____ - BASIC - 1.0( ); 2.0( ); 4.0( )?

3 Letter/No. Code	Description	Price

Total $ _____ .00

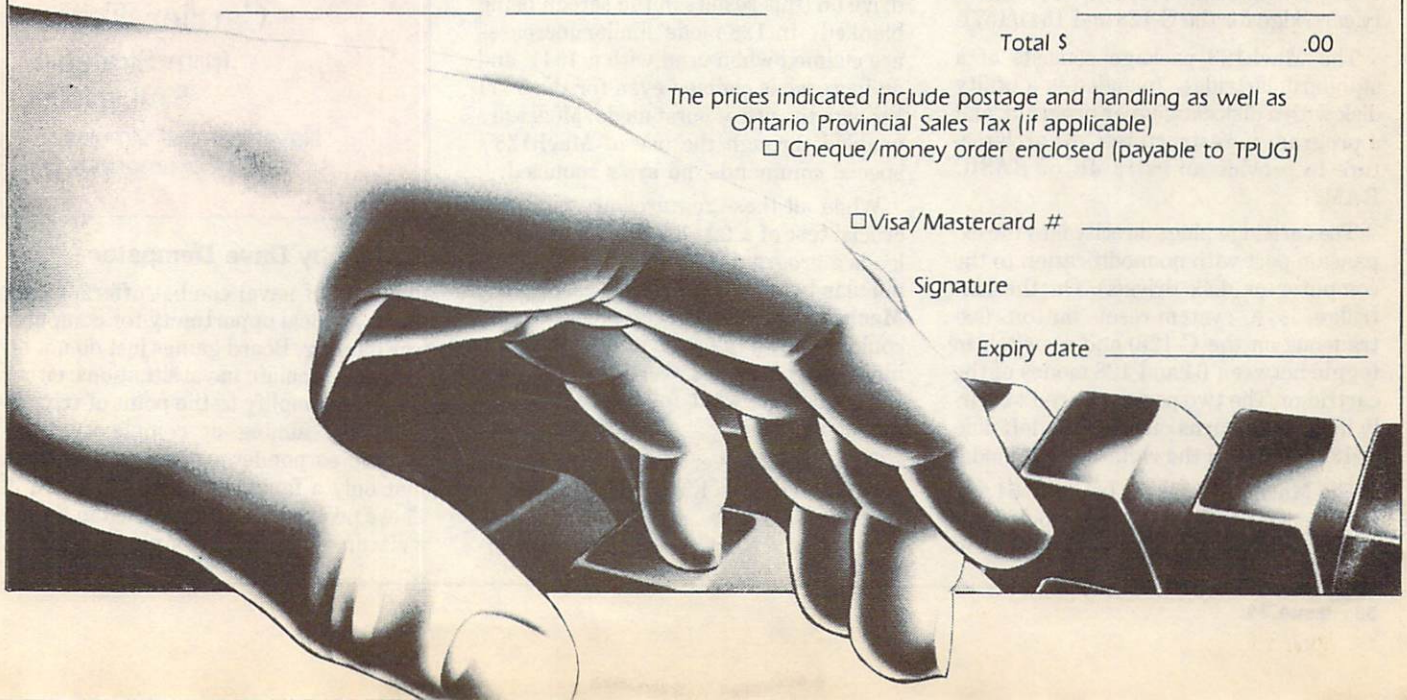
The prices indicated include postage and handling as well as Ontario Provincial Sales Tax (if applicable)

☐ Cheque/money order enclosed (payable to TPUG)

☐Visa/Mastercard # _____

Signature _____

Expiry date _____



Reviews

Mach128
from Access Software
Fast DOS cartridge for the C-128

Review by Adam Herst

One of the features most eagerly anticipated on the C-128 was the increase in disk access speeds available in 128 mode in conjunction with a 1571 disk drive. Most 128 users have not been disappointed, especially if they cut their teeth on the VIC 20 or the C-64 with the notoriously slow 1541 drive. One of the most popular firmware additions to these systems are fast DOS cartridges, designed to overcome slow 1541 access speeds. Since the C-128 with a 1571 provides these increases, a fast DOS cartridge for the 128 seems unnecessary.

'Unnecessary', that is, only if you don't own a 1541. The 1541 works in all three modes of the C-128 and makes an excellent utility drive. Unfortunately, 128 speed benefits do not accrue to the 1541. It runs at the same slow speed we've come to know how to complain about so well. While the C-64 fast DOS programs work well in 64 mode, they will not work in 128 mode. Also, C-64 cartridges will cause the 128 to boot up in 64 mode, and must be removed from the expansion port for the other modes to be accessed. It is with this in mind that I looked forward to reviewing **Mach128**, a fast-load utility cartridge for the C-128 and 1541/1571.

The **Mach128** package consists of a standard cartridge. Included is a utility disk with a disk cataloguer program, and a program to restructure C-64 architecture to provide an extra 4K of BASIC RAM.

The cartridge plugs direclty into the expansion port with no modification to the computer or disk drive(s). On the cartridge is a system-reset button (extraneous on the C-128) and a switch to toggle between 64 and 128 modes of the cartridge. The two positions aren't clearly marked; it turns out that the left side is 128 mode and the right side 64 mode.

The **Mach128** works in both the 64 and 128 modes of the C-128. It would probably work on a standard C-64 (if you need a fast loader but are thinking of upgrading to the C-128 in the near future) but I never got around to trying it out. The cartridge works with both the 1571 and 1541 disk drives. The manual claims that it will also work with compatible drives but does not list the models.

In 128 mode the cartridge functions with a 1571, a 1541 and a 1571 in 1541 mode. In 64 mode the cartridge functions with a 1541 and a 1571. If a 1571 is locked into 1571 mode from 64 mode, the system locks up. No more double-sided disks in 64 mode!

Along with the fast load, the **Mach128** supports wedge-type DOS commands. The standard directory, load, save and disk commands are included, as well as some unique commands that allow autobooting of the first program on the disk, a fast format for 1541 drives (one of many claimed enhancements that did not work) and a drive rattle eliminator, for example. Non-DOS commands include ones that will open printer channels, unnew a program, and dump a screen to the printer. A menu of these commands can be called up at any time, and a hardcopy of the screen made for a reference sheet that is better organized than the manual. The set of commands is the same in 64 and 128 modes, with slight variations.

Within any mode of the computer **Mach128** can operate in three modes: fast load off, 'mach drive' on and 'warp drive' on. This is true for both 64 and 128 modes. In 64 mode, mach drive promises an increase in load speed of 500 per cent, and an increase of 600 per cent with warp drive on (this results in the screen being blanked). In 128 mode similar increases are claimed when used with a 1541, and an increase is claimed even for the 1571 through the use of burst mode. Modes are enabled through the use of **Mach128**'s special commands, no **sys**'s required.

While all these features are nice, the crucial test of a fast loader is how fast it loads a program. This calls for the ever-popular benchmarks. In the case of the **Mach128** cartridge, complete results could occupy a full page if all of the combinations of modes were documented. Consequently, what follows are merely the highlights.

The first discovery of note was that, with all respect to Kirk, Spock and Scotty, warp drive is a figment of some copywriter's imagination. No load differences were documented in comparisons of warp drive engaged and warp drive disabled. The second fiction discovered was the effect of the **Mach128** on 1571 load speeds. 30K loaded in 9 seconds with or without the cartridge engaged. The cartridge did speed up load speeds with a 1541, from 82 seconds without the cartridge to 32 seconds with it, in both 64 and 128 modes. While this is an improvement, it is not the 500 per cent increase claimed.

Another major concern with fast loaders is their compatability with software. With the C-128, compatability with the CP/M operating system software is a paramount concern. The **Mach128** achieves this admirably. CP/M boots up without any alterations to the **Mach128** system. It does not achieve compatability with other software so readily. I was unable to load many games in 64 mode, although the applications programs I tried worked fine, with increases in load speeds. In 128 mode, I experienced problems with programs that had to be autobooted. Others worked well, however.

It is difficult to recommend this cartridge. It doesn't live up to many of its claims, although it does have many unique and useful features. Most notably absent is a fast save routine, useful for both 1541 and 1571 saves. Nonetheless, I use this cartridge every day, avoiding software that isn't compatible, and until a better one comes around, I will probably continue to do so. □

Carrier Force
from Strategic Simulations
Naval combat simulation for Commodore 64

Review by Dave Dempster

The area of naval combat offers, to my mind, an ideal opportunity for computer war gaming. Board games just do not effectively simulate naval situations: either they oversimplify to the point of triviality or, by aiming at complexity, they become so ponderous and cumbersome that only a fanatic would get to turn 3. These problems can be avoided in a well-written computer implementation.

Strategic Simulations' **Carrier Force** covers the war in the Pacific in 1942-1944 — a very promising period for gaming. Unfortunately, it does not go all the way in taking advantage of the computer's potential.

Carrier Force offers four scenarios covering such campaigns as Coral Sea, Midway and the Solomons. The game can be played solitaire or against another player (though in the latter case it does require that the players not watch the screen when the opponent is plotting a move — hmmm). The game advertises that each ship, each plane and so on are represented in the game and, by George, they all appear to be there.

Carrier Force operates in one-hour turns. For each turn you are successively presented with radar reports (if any) and weather reports, and are then cycled through menus that permit you to make decisions regarding fleet courses and speed, re-allocating fleet components, strike launch and aircraft preparation. The last phase requires you to steer your reconnaissance or strike missions to their targets hex by hex. The computer then carries out the attack if an attack force is in the area of a spotted enemy unit. An aircraft may fly unwittingly into an enemy area and be shot down by a nearby, yet presently invisible fleet. If you 'ready' aircraft, they will be prepared for launch next turn. The 'Are you sure?' question is asked just before you exit each menu — you cannot return after exiting. *You must step through each menu, each turn.*

The graphics are only adequate, and the sound is limited — this obviously is not a shoot-'em-up. The game-aids provided are necessary and useful (a grease pencil is also necessary for your plot), yet I wondered why (considering my trusty 64 in front of me) I had to manually plot sightings — yes, I know they did it in real life. I was permitted to do dumb things like launch aircraft to a target they didn't have the range to return from — it's a little disconcerting to see your last squadron make little splashes in the big Pacific. The rule book, as expected from SSI, is clear, complete and well written. The map display shows only a small segment of the map at any one time, and scrolls as you move the cursor.

Carrier Force is detailed and involved, and takes a long time to play. I got annoyed at having to continually ready, turn to wind, launch and land CAP (combat air patrol). I was not amused at being promoted to Admiral of the Fleet to set Task Force make-up, disposition and course, then being demoted to Lieutenant and plunked into the front seat of a PBY to go look for the enemy. **Carrier Force** tied me up so much in details, such as tracing the path for each of those necessary recce aircraft each turn, that I began to lose sight of the overall strategy. The game is difficult to finish, though it does offer the option of saving the game each turn. The action, if one gets to it, can be intense, but is short-lived. There are those long periods of waiting, and waiting, and...

If you've guessed that this game plays very much like a board game on a computer, you're not too far from the truth. So if you like a game with a lot of input, where you're personally involved in detail, you may well enjoy **Carrier Force** very much. I did not. □

The Super-G Printer Interface

from CardCo

Printer interface for VIC 20, C-64 and C-128

Review by Greg Payne

There is a wide variety of dot matrix printers on the market these days. Many offer excellent features: near letter quality mode, boldface print, subscripts, superscripts, expanded print, condensed print, and many other little goodies are standard to a lot of these machines.

Unfortunately, most use a communications format that is not compatible with Commodore computers. This problem can be solved by using a printer interface. The interface translates the printer codes coming from your computer and changes it into a code the printer understands. It's like an interpreter that translates English into French.

Cardco's Super-G is a new high-speed parallel printer interface for use with the VIC 20, C-64 and the C-128, with all the features of their G-Wiz interface and more. The Super-G is compatible with many popular printers, including Epson, Epson compatibles, Star Micronics, Panasonic, Okidata, NEC and the Smith-Corona Fastext-80.

The Super-G is enclosed in a sturdy plastic case, but has open access to its eight recessed DIP switches (through a small window in the case). A long cable hooks up to the serial port on your computer or disk drive, and a 36-conductor 18-inch ribbon runs to the printer. I like the way the ribbon plugs in almost flush to the back of my printer, keeping it out of the way of the paper feeder.

Cardco states that with the Super-G you get true 100 per cent Commodore 1525 printer emulation, high-speed operation (up to two times faster than the G-Wiz on some graphics tasks), a 300 per cent larger graphics buffer and the full Commodore character set, including all reversed text and graphic characters.

When I tested these claims on my Epson-compatible printer, I found that Cardco wasn't exaggerating a bit. I used the sample programs that are included in the Super-G manual and got exactly the results that I was supposed to get. My printer did everything a 1525 can do, but 3 to 5 times faster.

Hi-res screen dump programs designed for the 1525 that were impossible to use before worked exceptionally well, and my Compute Mate CP80 whizzed along as though the programs were made for it. My printer also worked well with all my other text software that was designed for use with a Commodore printer.

If you are writing your own programs that use a printer, you will like the way the Super-G can easily switch between different modes: normal print (line feed on/off), upper case only (line feed on/off), hex, transparent and lock mode can be accessed by using special secondary addresses in **open** commands to your printer. By switching from normal 1525 emulation to transparent mode briefly in a program, special printer functions unique to your printer can be turned on or off.

The DIP switches are for different interfacing and printer functions. Four are for printer configurations; the others specify transparent mode, hex mode, auto line feed and device number (4 or 5). All of the DIP switches, with the exception of the printer device number, can be turned on or off by using special commands from within a program. You can also tell what settings are currently being used by using a special Command Mode that lets you examine and change the current DIP switch settings.

The Super-G has a special listing mode called Quote Mode. When a program is listed to the printer, instead of Commodore's strange code symbols for things such as colour changes and cursor movements, you get special listing characters on your printout that are much easier to understand. Programs can also be listed to the printer in hex. This is valuable when debugging hi-res graphic programs and could be useful to see

whether or not special function control codes are being sent to the printer.

Another interesting feature is the Lock Mode. This lets you lock the interface into the mode you want. This feature comes in handy with a program — such as a word processor — that uses a printer file designed for your particular printer. Lock Mode can also be used to lock out any unwanted secondary address commands that are sent to the interface by programs that were designed to be used only by Commodore printers.

Super-G's manual is well written and easy to understand. Programmers will like it because it contains many example programs and tips for getting the most from their interface/printer combination. Most users will probably read it only once, to find the DIP switch settings for their printer. Once these are set, the interface pretty well runs on its own.

I am very pleased with this product. The Super-G passed all of my tests with flying colours, and is an excellent printer interface with many useful features. □

Broadsides
from Strategic Simulations, Inc.
Strategy game for Commodore 64

Review by Dave Dempster

'If only . . .' are the two saddest words in the English language. I bought this game about a year ago with considerable anticipation. 'The Age of Fighting Sail', complete with sailing, gunnery and boarding — it sounded great!

Make no mistake, I've probably played this game more times . . . Mind you, one game doesn't take too long. I then get annoyed and leave it for a while. Then, with hope rekindled . . .

The game permits you to select your scenario and set specific variables. As the game starts, you may input commands by scrolling the 14 available commands past the command display line, pressing the fire button to indicate that a command has been issued, and watching for the 'Aye, Aye, Sir' which indicates its input. The scrolling is slow, and a little frustrating in the heat of battle — keyboard entry is a little less so. One good feature: the game does not permit you to change an order (such as a turn) until the present order is completed. A bad feature is the necessity to order fire for each salvo. You're permitted to turn, to shoot at sails or hull, and to choose the type of shot and either battle or full sail. The game determines wind speed and direction, changing it randomly during the game — a not-so realistic feature. One strange item, manoeuverability (speed of turn), seemed to be unaffected by hull speed.

The manual is fair: mine included data for other computers as well as my C-64. In paragraph 8.5, HMS Victory seemed to be missing a full deck of guns.

The graphics in the game are very good. You can see, for instance, the opponents being shot away during combat. As you get closer, the game shifts to a close-up mode. Movement is jerky, not smooth.

There are eleven scenarios provided, and data and help are provided so that you can make and save four more of your own — a good thing, as some of those provided are grotesque. One particular scenario matches the Victory and the Constitution. C'mon guys — Victory's first broadside at Trafalgar killed 400 French sailors, dismounted 20 guns, and virtually crippled a well-built French 74 (battleship) — against a frigate! Several of the more interesting single ship actions such as the Cheasapeake/Shannon or the Essex/Phoebe (a battle between long gun and carronade) were not included. They're now on my disk, by the way.

What don't I like? Too much! The most interesting and main manoeuvering (for the weather guage) took place before fire was joined. At that point, tactical manoeuvering was pretty staightforward, because the largest number of my sailors were pretty busy on the guns. The ship scenarios only vaguely include those vital features of morale and state of training. The scale seemed wrong, although I didn't crunch the numbers. The guess-the-range option (the computer always guesses right!) can, thankfully, be removed. The boarding aspect is horrible, as it consists of two stick figures thrusting and hacking while you play a 'rock/scissors/paper' guessing game with your opponent to see who looses. I've been known to dump the program at this point.

Dear SSI, you have produced some superb simulations. Why not consider an upgrade? Provide an approach option, perhaps with an enemy to be identified. Permit the use of all hands, or only one of either port or starboard watch for tasks, with requisite delays if only part of the crew is utilized. Perhaps you could add commands like those found in 'Combat Leader': prepare to repel boarders or shake out sail or beat to quarters or fire as you bear . . . Once ships were grappled together, the captain has little input (apart from personal example) on the outcome. Perhaps he could leave gun crews still firing at the enemy ship, now impossible to miss, and call up only one watch to repel the enemy — but chancing loss of his deck and ship.

Because of a lack of stern guns (mounted facing behind), this game often degenerates into a tail chase, with the computer following. That would be a disastrous tactic in reality: it would present the structurally weakest part of your ship to shots that would also pass the length of your ship, causing all sorts of distress and unrest.

The concept of the game is good; its super, in fact, but the implementation . . .! I cannot and will not recommend this game. I still, however, trot the darned thing out — what can I say?

***Broadsides**, from Strategic Simulations, Inc., 883 Stierlin Road, Bldg. A-200, Mountain View, California 94043* □

VIC 20 Starter Book
by J. Titus, C. Titus and D. Larsen
from Howard W. Sams
$15.95 (US), $22.50 (Cdn.)

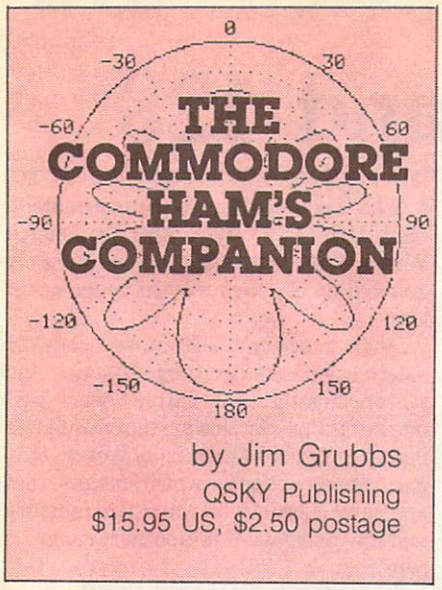
Review by Roger Burge

Although it may be a bit pricy, this book could be very helpful to the newer computer user. While some sections deal with matters specific to the VIC (such as joystick registers), much of the information also applies to the Commodore 64.

Illustrations are used frequently to augment the well-written text as you are familarized with everything from how to connect your VIC to a TV, to what the keys on the keyboard do. From there the authors take you on a tour of BASIC that is not only easily understood, but will help you to appreciate the power of your VIC 20.

Tape files, debugging, sound, accessories and computer care are examples of the many other topics covered. Mind you, such very popular areas as printer control and custom graphics are either ignored or mentioned only briefly.

Although this is not a substitute for Commodore's *VIC 20 Programmer's Reference Guide*, it is a very useful companion to the *VIC User's Guide* that came with your computer. □



by Jim Grubbs
QSKY Publishing
$15.95 US, $2.50 postage

Review by A. Vic Forde

The Commodore Ham's Companion by Jim Grubbs is written for ham radio operators interested in the concept of interfacing a Commodore computer with their radio.

This paperback edition contains 160 pages consisting of 14 chapters and 6 appendices. The topics discussed range from the choice of computers and the selection of software to the actual interfaces themselves.

The author's experience has been with the VIC 20 — he uses three of them in his 'radio shack'. One is used for a 'Message Storage Operation', another for 'Slow Scan Television' and a third as a terminal for 'Packet Radio'. In the first chapter he discusses, in a cursory way, the possible uses of the various Commodore computers including the new C-128.

Chapters two, three and four are the key sections of the book. In chapter two the author attempts to discuss the transmission of data through the use of BAUDOT, ASCII and AMTOR. Unfortunately, too much of the subject matter is covered in too little space, resulting in a lack of in-depth discussion. A glossary of terms in one of the appendices is intended to replace any deep coverage. This means that the reader who is not already well versed in ham radio terminology must continually refer to the explanations in the appendix. The chapter is fine for those hams who have considerable knowledge of the subject before reading this book.

The author must have realized this shortcoming, because an eight-line explanation occurs between chapters three and four directing readers to the glossary if they find the terminology confusing. Unfortunately, many technical writers from the US use acronyms, short forms or individual letters in place of proper nouns. This technique can be confusing for those readers who are only vaguely familiar with the subject matter.

Chapters three and four could save you considerable literature research: the author briefly describes many of the common interfaces available, along with their advantages and disadvantages. A considerable quantity of information is crammed into these two small chapters.

The chapter on Packet Radio is very descriptive and, for those hams who are not familiar with the subject, provides a good understanding of how it works.

A valuable list of suppliers and distributors of software and interfacing equipment appears in appendix B. Of the 83 suppliers listed, only two are in Canada and one of those is TPUG. This may be the reason why Canadian hams are not as involved with interfacing their computers to their radios as our American counterparts.

The Commodore Ham's Companion is not a 'how to' book, but a compendium of information about the hardware and software available for interfacing a Commodore computer with a ham radio transceiver. Despite some shortcomings, this book would be a worthwhile addition to a ham's reference library.

***The Commodore Ham's Companion**, from QSKY Publishing. P.O. Box 3042, Springfield, Illinois 62708.* □

Koronis Rift
from Epyx
Action game
for Commodore 64

Review by Thomas Jones

The plot of **Koronis Rift** is that you are a treasure hunter in the distant future, attempting to recover artifacts for profit from a planet once inhabited by an advanced race, now vanished. They have left robot sentries to guard their property — deadly saucers that will make suicidal attacks on any intruder. You are equipped with a land crawler that has laser-type weapons to defend against the attacks, and advanced robots to help you recover treasures.

The view of the hilly landscape through your window tosses in a more or less convincing way for a land crawler, and if an obstructing hill is too steep, you will stop. Nevertheless, navigation is not much of a problem and you can find the artifacts relatively easily, thanks to a radar-like instrument that leads you right to the nearest alien structure (invariably some kind of crawler). Once you have reached your target, and have shot down all the pesky little saucers that show up, you send out your robot helper to loot the hulk, a scenario presented in a quite entertaining fashion. You must store the loot in an empty bin, and continue the search.

While they can be challenging, keeping you on the edge of your seat gripping the joystick, the saucers are not inordinately difficult to shoot down.

Any time you are not under attack by the saucers, you can call the mother ship to 'beam you up'. Here another robot will ask you to select an artifact bin and move it to the conveyor belt. The robot will then examine the artifact, and with the help of a computer and instruments, announce the value of the item. You can either sell it or keep it, and move on to the next. If you keep it, your ship may acquire different characteristics that will help you survive. If you dismantle and sell it, you gain points. The ship's robot is exceptionally well done, and fascinating to watch. The animation of the robot using a computer is usually a hit with kibitzers.

The game controls are icon driven: you pull straight back on the joystick and press the fire button to go into command mode from the drive/shoot mode, and use the joystick to highlight the command in red.

You do not have to start from scratch each session: the Save command will save your position and score to the game disk. It will only save the last position, however, and only for one player. It does give you a way to keep most of your points if you get killed by going back to the last saved position.

The game plot is not intricate; nor is the best strategy too difficult to figure out. I found the scenery and the repetitive nature of the artifacts grew a little tiresome after a few long sessions. It would be especially nice if the artifacts were more varied and interesting.

All in all, I would say **Koronis Rift** falls somewhere between an arcade shoot-'em-up with exceptional graphics, and the newer generation of simulation/strategy games. For some people it might serve as a transition towards more sophisticated games.

***Koronis Rift**, $32.00 US (price approximate), from Epyx, Inc., 1043 Kiel Court, Sunnyvale, California 94089.* □

Paperback Writer 64 and 128

Paperback Writer 64
from Digital Solutions
Word processor
for Commodore 64

Review by Ian Wright

Paperback Writer 64 is the first attempt to implement a WYSIWYG word processor on Commodore machines. The term WYSIWYG is an acronym for "what you see is what you get", and means that the text on your screen is preformatted to look like the final printout.

Paperback Writer 64 uses word wrap. If you set the margins at 10 and 70 spaces from the left edge of the paper, and you type 'supercallifragilistic' so that it extends past the right margin, it will not be broken but will appear on the next line. If you type in boldface, the screen displays the text in a brighter shade. Italics letters are slanted, and underlined text really is underlined.

WYSIWYG does not mean that screen formatting commands are absent. It means that you cannot *see* them on-screen. No longer do indented quotes start with: **lm+10:rm-10:sp1:fp10**. Those commands (or their equivalents) are embedded in the text but are not printed to the screen. On the screen your quotation looks just like it will on the final paper printout.

How can you show 80-column margins on a 40-column screen? Two methods. After loading the boot program, **Paperback Writer 64** lets you choose from three separate programs: a 40-column version, an 80-column version, and a separate spelling checker. To move from one version to the other means exiting the current program and reloading another choice. There is no "Save this file?" prompt, and your current work will be lost if you have not saved it before exiting.

Writing 80-column files in the 40-column version of **Paperback Writer 64** is done through the use of a 40-column screen window, which jumps sideways across the screen rather than using a smooth scrolling movement. I found this jumping movement nauseating after a short time and quickly learned that it was best to use the 80-column version of **Paperback Writer 64** whenever possible. The 40-column characters, however, are easier to read, and the program's commands work more quickly because the screen does not have to be redrawn in hi-res.

The 80-column version of **Paperback Writer 64** uses a high resolution bit-mapped screen to display all 80 columns of text on the screen at one time. The characters are necessarily smaller and less distinct than the 40-column version. If you are not using a good monitor (either colour or monochrome), the 80-column characters may be illegible. This problem is a hardware restriction inherent in the Commodore 64, which was designed to be attached to a standard colour TV and display 40 by 25 text lines. **Paperback Writer 64** has taken the hardware to its limits, because the 80-column video output *is* legible on a 1700-series colour monitor or a good monochrome monitor.

If WYSIWYG were the only difference between **Paperback Writer 64** and the other Commodore word processing programs, **Paperback Writer 64** would be a significant improvement for many users — but there are lots more features.

Text entry and editing

Paperback Writer 64 allows you to move the cursor around the text file in 14 different ways — forward by word or back by screen for example. Screen width can be set from 1 to anything, and the screen scrolls its 40 or 80-column window across this width. I found that moving the cursor to the right or using the insert mode did not wrap around the right margin as I expected. If I inserted text into a file with a right margin set at 75, the insert would carry on into columns beyond the margin until I reformatted the screen using the **f6** function key: an annoying quirk, since moving backwards doesn't do this at all.

Some writers like to delete characters to the left of the cursor (called destructive backspacing), while others like to suck up their errors from the right of the cursor. In **Paperback Writer 64** you can choose the method you prefer using **ctrl-e**, which is a nice touch. Unfortunately, there is no undo feature in **Paperback Writer 64**, so anything mistakenly gobbled up is gone away to never-never land.

Block functions like move, copy or delete use a memory range function and screen highlighting rather than text markers. I prefer this system for clarity and ease of use. There is, however, only one memory bank, which means that anything currently in the range memory will be destroyed if another range is selected.

Search and replace is neatly implemented in **Paperback Writer 64**, so that the command (**logo-f**) calls up both searching and replacing rather than having two separate commands as in most other word processors. The upper case (**logo-F**) version of this command means continue the search to find the next occurrence. Generally speaking **Paperback Writer 64**'s commands are more easily learned than those of earlier Commodore word processors.

Formatting features

Formatting in **Paperback Writer 64** is done by pressing the **f5** key to call up the menu of format options. You can select one or more of the 18 options by moving the cursor and pressing **return**. You do not have to memorize **OF3** for offsetting the text 3 spaces, for example: just pick 'printer offset' from the menu and type in 3. It's easy.

Paging is usually a complicated function in a sophisticated word processor like **Paperback Writer 64**, because you can have blank lines, blanks at the top of the page, titles, headers and footers, as well as your text. **Paperback Writer 64**'s manual uses a neat diagram to display and explain how each of these features relates to and affects the others, helping you to visualize the page format. The page length can be up to 250 lines, the lines per page up to 249, and the lines can be printed at 6 or 8 lines per inch. Other lpi settings can be arranged by rewriting a printer file if you need them.

Paperback Writer 64 gives you complete control over margins (fixed, relative, indented, or released), justification, left or right alignment and centering, and displays these paragraph formats on-screen just as they will appear

on paper. This is a tremendous advantage for people using complicated formats. Similarly, it's easy to decide where to put hyphens when using right justified text because the 80-column screen display can be edited directly (there's no automatic hyphenation). Paragraph spacing can be any integer — but you cannot use 1.5 line spacing, which I prefer for formal letters. Tabs default to 10 spaces apart, and can easily be reset to suit your needs. **Paperback Writer 64** has all the commonly used paragraph formats expected of a high-quality word processing program.

Column manipulation in **Paperback Writer 64** is restricted to charts of numbers and/or text (you can't do newspaper-like columns of text output as with **WP64**), so the column commands are easier to use than those in most other Commodore word processors. You can align numbers, add numbers in columns or rows, add negative numbers (to subtract), sort lists of numbers or text in ascending or descending order — but you cannot move or shift a column using a block range. This is an unfortunate oversight since I have used this feature in **PaperClip** more times than I care to remember when working with complicated charts of statistics.

Foreign or special characters can be designed from scratch, or you can use the French set and the six additional characters that come with **Paperback Writer 64**. In either case, your defined character is displayed on the screen. It's nice to see *garçon* written correctly on the screen with the cedilla, but my Epson printer produced the wrong character despite double-checking I had loaded the correct printer file. I know how to fix the output, but the Epson printer file that comes with **Paperback Writer 64** should work as is. This is another example of the problems associated with incompatibility among peripherals.

If you want to display and print other characters (like the English pound symbol) you can redefine up to ten characters using their ASCII value. You can even combine two characters into one (for example, **1 = 27 + 69** for emphasized print on the Epson). Unlike the optional underline and italics characters, which appear correctly on-screen, superscripts and subscripts are shown only by coloured ranges in the C-64 version of **Paperback Writer**. I do like the ability to choose from one of four different cursors — block or underline shapes with each shape flashing or solid. The error bell can also be toggled on and off for those quiet sessions at 2:00 a.m. These are just some of many well-thought out features.

Headers and footers can be more than one line long in **Paperback Writer 64** — a useful feature for many administrative reports — and making one of these multi-line formats is really easy. The headers and footers can be set with different pitches and different margins from the rest of the text, and automatic page numbering is also part of the header and footer options.

Files and printers

The text files in **Paperback Writer 64** are held in memory so there's no wait while pages of text are drawn in from and written back to disk. But file size is still the weakest area of the program. It's not the fault of the designers or writers of **Paperback Writer 64**, because the problem stems from the limited size of the C-64. Who would have thought that 64K was small! My first 32K PET was considered to have an immense amount of RAM in 1980 when it was new. Today, however, even 64K is below the minimum requirements for productivity software, even when used in the home. By comparison, 256K is a minimum RAM for most of today's business programs, and soon even 512K will not be enough to run the latest business software. **Paperback Writer 64** in 80-column mode can manipulate a file of only 6895 bytes (that's less than 7K), less than my 32K PET could manipulate with **PaperClip** in 1983. The 7K translates to about 139 lines of 80-column text, or just under three pages of single-spaced output. For a student writing high-school essays, or a small business owner wanting to prepare a contract proposal, that's simply not enough without using a lot of linked files. The program has so many features and such extensive help available that there's little space left for text! **Paperback Writer 64** does make using linked files easier by using a global mode to carry the established formatting and page location between linked files, but it's still not something that I'd like to have to do all the time.

Disk operations are easy and varied in **Paperback Writer 64**. You can load and save files quickly using on-screen menus and without retyping file names: just cursor to the name of your file in the on-screen (non-destructive) directory and press **return**. You can save your text as PRG, SEQ, ASCII files or even as files printed to disk. This covers all conceivable needs, whether you want to send files over a modem to other users of **Paperback Writer 64**, to other Commodore users, or even to non-Commodore machines.

Paperback Writer 64 will *not* load from a 4040 drive because of the protection scheme used on the disk. The manual and help screens in the program have examples of multiple-drive use, and they work as advertised — after you load the program from a 1541 drive!

Directories can be loaded from drive 0 or 1, or from various single disk-drive device numbers. You can do complete pattern-matching of directories, and they will scroll up and down to help you quickly search for that elusive letter to the boss.

The version of **Paperback Writer 64** that I have comes with 18 printer files, and my so-so experience with the Epson file is not necessarily true for the other printers listed. Rewriting a printer file is quite easy: just follow the step-by-step instructions in the reference guide. (It must be easy — I made a special file for my daisywheel printer without problems.)

Special features

One of the three options available from the boot program is to load the spelling checker. If you are writing a file you must first exit **Paperback Writer 64**, then load the spelling checker and follow the prompts. However, the spelling checker won't work initially because there's no dictionary! The spelling program will let you make a new dictionary disk (any idea how long it takes to add 30,000 words?), or you can buy the $19.95 dictionary disk with 32,000 words (and room for another 8,000) that's available from Digital Solutions. You cannot use another manufacturer's dictionary disk — I know, I tried. So if you want to check your spelling, it's best to put aside another $20.00 for Digital Solution's disk.

Paperback Writer 64 has no built-in macro function to allow you to define a statement or a procedure, like defining the phrase *Paperback Writer 64* as **ctrl-a** for example. Definitions are limited to one character. Footnoting is not supported, nor are indexing or tables of contents, so the program may not be as useful for senior students or administrators. However, few programs that for the C-64 have these kinds of features built-in.

You can alter the colour of any of the eight screen displays (characters, backgrounds, and so on) using any combination of 16 colours. I prefer light grey on dark grey for 80-column work, but the extensive choice of colour combinations is a nice touch. The mail-merge function for generating form letters is well explained and easily implemented in **Paper-**

back Writer 64, making simple what could be a quite complicated process.

The on-screen help function is very extensive. You can call up help at any time by pressing **f7** to get a 5-option menu. Select a number from 1 to 5 to see the sub-menus. If these don't cover your difficulty, you can press **f7** again and get context-related information from disk help files. To many people, this feature alone will make **Paperback Writer 64** the word processor of choice for use on the C-64. Not all the functions of **Paperback Writer 64** are in these files, however, so don't throw away the manual.

Speaking of the manual, some parts are very well explained, but others are rather vague:

> Footers are usually set in the first paragraph.
>
> However, if you CHANGE A FOOTER in the first paragraph of a NEW page, the PREVIOUS page's footer also changes when it is printed. To ensure that the old footer prints on the previous page and that the new footer starts at the bottom of the new page, enter a forced page (see above) in the paragraph BEFORE the footer is changed. Do this by inserting a return arrow (blank paragraph) with a forced page immediately before the paragraph in which the footer changes — either in the new page or at the bottom of the previous page.

Huh? After rereading a few times, the message becomes clear. But this kind of information might be better presented through a tutorial. The reference guide does not have any tutorial function, but does have constant references to the 128 version of **Paperback Writer 64** because the same manual is used for both versions of the program. I realize that this helps to keep the cost down, but it is perplexing enough even to those experienced with word processing. It will be very confusing to the neophyte user.

Conclusion

Paperback Writer 64 may be too complicated a word processor for an absolute beginner, but its help screens and menu-driven functions make it the simplest to operate of all the available sophisticated word processors for the home user. **Paperback Writer 64** is not suited to the professional writer because of the limitations of the machine it runs on. Nonetheless, the program itself has a wide selection of functions that are well implemented. Is this the word processor for you? At $49.95, including reference guide, **Paperback Writer 64** is definitely worth a close look. □

Paperback Writer 128

from Digital Solutions

Word processor for Commodore 128

Review by Greg Payne

Paperback Writer 128 makes full use of the new C-128 computer (in 80 column mode). It has all of the features of **Paperback Writer 64** plus many new ones of its own. The program automatically boots if it is in your disk drive (either a 1541 or 1571) when the computer is turned on. If the 40/80 display key is up, you will get a menu on the monitor giving you two choices: a forty column **Paperback Writer**, or the Spelling Checker dictionary. If the key is down, you go right into the 80 column **Paperback Writer 128**. In both modes all the new C-128 keys can be used, including the numeric keyboard.

The forty column **Paperback Writer 128** is really a slightly souped-up version of **Paperback Writer 64**. It even puts your C-128 computer into 64 mode. The forty column program has approximately 15K free for text. This isn't bad on a C-64, but pretty skimpy on a 128K machine.

The Spelling Checker in **Paperback Writer 128** is the same one as on the **Paperback Writer 64** program disk. It's empty, and has to have the words added (a very time consuming chore). Since it's a C-64 program, it can only check small documents.

I consider **Paperback Writer 128** in 80 columns one of the best word processing programs I've seen. As I stated earlier, it has all of the same functions as the C-64 program, and more. If you've used the 64 program, you will have no problem using **Paperback Writer 128** because all of the commands are the same with the exception of a few added features.

The 80 column mode of the program uses RGBI or monochrome output to your monitor, and if you are using a 1902 or monochrome set, the characters are really quite easy to read — I would say 100 per cent better than the bit-mapped version on the C-64. A 1700-series Commodore monitor can also be used with good results, by purchasing a special cable that allows you to hook up to the RGBI port on the computer. This cable is also needed for the monochrome monitor.

Paperback Writer 128 in 80 column mode has 64K of free text memory. This can be split into two 32K areas that can contain completely separate files. Cut-and-paste operations can easily be made between the two file areas. The files can be printed, saved, loaded and formatted completely separately. This feature has many possibilities and should prove useful to many users.

Superscripts and subscripts actually look like they're supposed to in 80 column mode. Superscripts are raised about a third of a line, and subscripts are lowered by the same amount. The other text enhancements (boldface, italics and underlining) are represented in the same way as on the 64 version of the program.

The extra keys on the C-128 are all utilized. The **esc** key switches back and forth between the edit mode and the text formatting areas. The **tab** key moves the cursor from tab stop to tab stop. Tabs can be placed wherever you like. The **caps lock** key shifts all alphabetical keys into upper case. The **help** key is used to bring up help screens. The **line feed** key can turn line feed on or off if your printer file does not support auto line feed. If it does have auto line feed, double spacing results. All of the other keys have the same functions as they do in **Paperback Writer 64**.

The ability of **Paperback Writer 128** to auto-boot is a nice feature. Users that have non-Commodore printers may think that this a problem because it doesn't allow them to lock their printer interface into transparent mode. The programmers of **Paperback Writer 128** have come up with a really good solution. The lock command can be included in your printer file, and when the program loads, it automatically puts the interface into transparent mode. **Paperback Writer** can also be manually booted by leaving the disk out of the drive until after the computer is turned on, and typing **boot**.

I have been using **Paperback Writer 128** for about three months now, and I think it's one of the top word processing programs available for the Commodore 128 computer. It's very easy to use (no strange formatting symbols to memorize) and very powerful. I especially like the way the text formats itself right in front of you on the screen. No more spending half an hour trying to right-align an address in a letter. A couple of things I would like to see in the near future: a 40 column C-128 version, and an 80 column C-128 Spelling Checker. Otherwise a nice product. □

Calendar of TPUG Events

Meeting Places

Amiga Chapter: The Amiga Chapter meetings are held in conjunction with the Westside Chapter meetings.

Brampton Chapter: Brampton Public Library, Four Corners Branch, 65 Queen St., on the second Thursday of the month, at 7:30 pm.

Business Chapter: The June meeting of the Business chapter has been cancelled.

Central Chapter: The Central Chapter will no longer be meeting.

COMAL Chapter: York Public Library, 1745 Eglinton Ave. W. (just east of Dufferin) on the fourth Thursday of the month, at 7:30 pm in the Story Hour Room (adjacent to the auditorium).

Commodore 128 Chapter: York Public Library, 1745 Eglinton Ave. W. (just east of Dufferin), on the first Wednesday of the month, at 7:30 pm in the Story Hour Room.

Commodore 64 Chapter: York Mills CI, 490 York Mills Rd. (east of Bayview) on the second Monday of June, at 7:30 pm in the cafetorium.

Communications Chapter: TPUG Office, 101 Duncan Mill Rd., Suite G-7, Don Mills, on the fourth Wednesday of the month, at 7:30 pm.

Eastside Chapter: Dunbarton High School (go north on Whites Rd. from the traffic lights at Highway 2 and Whites Rd. to next traffic lights; turn left to parking lots) on the first Monday of the month, at 7:30 pm.

Hardware Chapter: TPUG Office, 101 Duncan Mill Rd., Suite G-7, Don Mills, on the second Tuesday of the month, at 7:30 pm.

New Users Chapter: The June meeting of the New Users meeting has been cancelled.

SuperPET Chapter: York University, Petrie Science Building (check in room 340). Use north door of Petrie to access building. On the third Wednesday of the month, at 7:30 pm.

VIC 20 Chapter: York Public Library, 1745 Eglinton Ave. W. (just east of Dufferin), on the first Tuesday of the month, at 7:30 pm in the auditorium.

Westside Chapter: Clarkson Secondary School, Bromsgrove just east of Winston Churchill Blvd., on the third Thursday of the month, at 7:30 pm.

TPUG makes every effort to ensure that meetings take place when and where scheduled. However, unforeseen problems may occasionally arise that lead to a particular meeting being changed or cancelled. The TPUG meetings line (445-9040) is the best source of fully up-to-date information on meeting times, and should be consulted.

Are you interested in organizing some other interest group in the Greater Toronto area? Please let the club office know, by mail, phone or TPUG bulletin board. □

JUNE

MON	TUES	WED	THURS
2 Eastside	3 VIC 20	4 C-128	5
9 Commodore 64	10	11 Business (cancelled this month)	12 Brampton
16 New Users (cancelled this month)	17	18 SuperPET	19 Westside-Amiga
23	24	25 Communications	26 COMAL
30			

Classifieds

This space is for the ads of TPUG members. Wanted or for sale items only. Cost is 25 cents per word. No dealer ads accepted.

Wanted: Cymbal accounting software with instruction manual. 463-4736.

For sale: VIC 20 manuals, software, 16K expander, Super Expander, 3-slot expansion board, and VIC modem. Phone Stuart at (204) 687-7146 after 7 pm.

For sale: B-128 computer w/256K upgrade, 8050 disk drive, 2031 disk drive, 4023 printer, USI amber monitor, 300 baud modem, Calc Result, Superscript (3 copies), Superbase (3 copies), General ledger, A/R, A/P, Payroll, Inventory, B-Term, 10 disks of utility programs, 2 VIC 20 computers, dataset, vicmodem, 5 cartridges, 6 books of commodore programming, all cables and documentation. All for only $1500.00 (U.S.) Call Chuck (303)247-2260 after 6 p.m.

Bulletin Board

The Music Studio for Amiga

Activision, Inc. has announced the release of an enhanced and expanded version of **The Music Studio** for the Amiga. **The Music Studio**, designed by Audio Light, is a music composition tool which allows the user to create compositions, from an elementary tune to an elaborate fifteen-channel, three-verse score. The new version features pull-down windows for composition and editing, and advanced editing functions, such as commands to move, copy, repeat and transpose. Complete musical notation includes time signatures, ties, rests, measure bars, sharps and flats.

The Music Studio package contains a complete library of original musical compositions so that the user can hear and see the work of professional musicians.

The Music Studio, released under the Activision Creativity Software label, was scheduled for release in the first quarter of 1986 with the suggested retail price of $59.95 (US). The program is also available for the Commodore 64/128 on a two-sided disk disk at a suggested retail price of $34.95 (US).

For more information contact: Activision, Inc., 2350 Bayshore Frontage Road, Mountain View, CA 94043.

BobsTerm Pro-128

BobsTerm Pro-128 is a powerful, new communications package from Progressive Peripherals & Software. It is menu-driven and supports VT-100 and VT-52 80 ADM-31 (CP/M type) terminal emulation. Offering a full screen text editor and on-screen status display of the entire contents of the 60K buffer, **BobsTerm Pro-128** allows the user to directly edit the files while it simultaneously reads, writes, uploads and downloads to any disk type (including CP/M).

BobsTerm Pro-128 can be used in a remote mode transforming the computer into a mini-bulletin board system. The macro and answer back string functions can be used together.

BobsTerm Pro-128 retails for $79.95. For more information contact Progressive Peripherals & Software, 464 Kalamath Street, Denver, CO 80204, (303)825-4144.

MOD Keyboard System for Handicapped

MOD Keyboard System is a plug-in cartridge from Tash Inc. which enables physically disabled people to create numerous screen displays of characters, words and phrases, and computer commands. At present there are two versions of **MOD Keyboard** : elementary and advanced. The **MOD Keyboard System** requires following items: a **MOD Keyboard**, either elementary or advanced; an input device chosen to best suit the user; a VIC-20; a Commodore 14" colour monitor; and an interconnection kit for the host computer and Commodore datasette tape recorder.

Elementary MOD Keyboard has been designed primarily for educational applications where an instructor, rather than the disabled user, would create or edit the screen pages. It is intended for severly physically handicapped students who require large letters and clear presentation of items.

Elementary MOD Keyboard supports Apple II Plus or Apple IIe as host computers. Versions supporting IBM PC and Commodore 64 will soon follow.

Advanced MOD Keyboard allows the disabled user, in schools and businesses, to operate the *same* computers as their able-bodied collegues. With the appropriate interconnection kit, any of the following "host" computers, and *all standard software* normally operated from the keyboard of these computers, is accessible: Apple II Plus and IIe, IBM PC, C-64, Nelma Persona and any RS-232C.

You can read about some of Tash Inc. adaptations for disabled in William Bennett's article *Computer Aids for the Disabled* in July 1984 issue of *TPUG Magazine*.

To get more information, contact Tash Inc. at 70 Gibson Drive, Unit 1, Markham, Ontario, L3R 2Z3, (416) 4475-2212.

Miami 2064 CUG

Miami 2064 Commodore Users Group informs its members and other interested parties that Group's new address and telephone number are: 11531 S.W. 84 St., Miami, FL 33173, (305)595-8612. A bulletin board system was expected to begin operation in May. It's phone number is (305)279-8605. **Miami 2064 CUG** meets on the third Thursday of each month at the Sunset Congregational Church, 9025 Sunset Dr., Miami, FL. Meetings are from seven to approximately nine at night. □

Commodore Canada appoints president

Bruce Hampson, General Manager of TPUG, presents Rich McIntyre with a framed cover of the March 1986 issue of ***TPUG Magazine*** *to commemorate Mr. McIntyre's appointment as President and General Manager of Commodore Business Machines (Canada) Ltd., which took effect on March 31, 1986. Mr. McIntyre was previously Vice-President of Sales for the Canadian operation.*

advertisers' index

TPUG Magazine Distributors

***Dealers:** If you would like to carry **TPUG Magazine** in your store, you may order from any one of the following distributors:*

CANADA

Master Media, Oakville, Ont.	416-842-1555
Ingram Software, Concord, Ont.	416-665-0222
Compulit Distributors, Port Coquitlam, BC	604-464-1221

USA

Prairie News, Chicago, IL	312-384-5350
Levity Distributors, North Hollywood, CA	818-506-7958
Whole Life Distributors, Englewood, CO	303-761-2435
M-6 Distribution, Houston, TX	713-778-3002
The Homing Pigeon, Elgin, TX	512-276-7962
Fred Bay News Co., Portland, OR	503-228-0251
Alonso Book & Periodical, Alexandria, VA	703-765-1211
Cornucopia Distribution, Seattle, WA	206-323-6247
Guild News, Atlanta, GA	404-252-4166
Micro-PACE, Champaign, IL	800-362-9653
Nelson News 4651 F Street, Omaha, NE	68127
Michianna News, Ft. Wayne, IN	219/484-0571
Total Circulation, South Hackensack, NJ	201/342-6334

TPUG Contacts

TPUG OFFICE 416/445-4524
TPUG BBS 416/273-6300
TPUG MEETINGS INFO 416/445-9040

Board of Directors

President	Chris Bennett	c/o 416/445-4524
Vice-President	Gerry Gold	416/225-8760
Vice-President	Carl Epstein	416/492-0222
Recording Sec.		
	David Bradley	c/o 416/445-4524
	Richard Bradley	c/o 416/445-4524
	Gary Croft	416/727-8795
	Mike Donegan	416/639-0329
	John Easton	416/251-1511
	Keith Falkner	416/481-0678
	Anne Gudz	c/o416/445-4524
General Manager	Bruce Hampson	416/445-4524

TPUG Magazine

Publisher	Bruce Hampson	416/445-4524
Editor	Nick Sullivan	416/445-4524
Assistant Editors	Tim Grantham	416/445-4524
	Adam Herst	416/445-4524
Production Manager	Astrid Kumas	416/445-4524
Ad Sales	John Matheson	416/445-4524

Meeting Co-ordinators

Brampton Chapter	Jackie Bingley	c/o 416/445-4524
C-64 Chapter	Keith Faulkner	416/481-0678
COMAL Chapter	Donald Dalley	416/742-3790
	Victor Gough	416/677-8840
Communications	Darrell Grainger	c/o 416/445-4524
Eastside Chapter	Nina Nanan	c/o 416/445-4524
Hardware Chapter	Frank Hutchings	c/o 416/445-4524
SuperPET Chapter	Gerry Gold	416/225-8760
VIC 20 Chapter	Anne Gudz	c/o 416/445-4524
Westside Chapter	John Easton	416/251-1511
	Al Farquharson	519/442-7000
Business Chapter		
New Users Chapter		
C-128 Chapter	Adam Herst	c/o 416/445-4524
Amiga Chapter	Mike Donegan	416/639-0329

Librarians

COMAL	Victor Gough	416/677-8840
PET	Mike Donegan	416/639-0329
SuperPET	Bill Dutfield	416/224-0642
VIC 20	Richard Best	c/o 416/445-4524
Commodore 64	Derick Campbell	416/492-9518
B-128	Paul Aitchison	c/o 416/445-4524
Amiga	Mike Donegan	416/639-0329
C-128	Adam Herst (CP/M)	c/o 416/445-4524
	James Kokkinen (C-128)	c/o 416/445-4524
MS/DOS	Colin Justason	c/o 416/445-4524

TPUG Bulletin Board

Sysop (voice, weekdays)	Sylvia Gallus	c/o 416/896-1446
Assistant Sysop	Steve Punter	c/o 416/896-1446

The Guide

A Monthly Publication For Commodore™ Owners

Commodore support with a twist . . . Personable and even humorous . . . Timely news . . . Helpful tutorials . . . On-going support for several languages: BASIC (including BASIC 7.0 as featured in the 128 PC), Machine Language, COMAL, and Pascal . . . Program Listings . . . *Honest* software reviews.

The Guide features some of the best computer humorists to be found.

- Introduce your "widow" to the Computer Widow's Compendium.
- Tutorials and feature articles by the famous Mindy Skelton.
- Featuring Shelly Roberts' "I'm Sorry . . . But I Don't Speak Hexidecimal." Discover why Shelly just may be the Andy Rooney of the computer world!

We feel we have assembled one of the most talented staff of writers in the Commodore world. Receive each month the most friendly and helpful Commodore publication available. Written by Commodore users who are writing to you, not down at you.

Limited offer — FREE With Each Subscription !

Subscribe *NOW* to capitalize on free software offer!

Receive up to $99.85 in *high quality* software!

With each year's subscription (or renewal) ordered, receive your choice of the software packages listed below, including the award winning educational games from Disney, or Omiterm Terminal written by our own Bob Richardson.

Subscription	Rate	Canadian *	Receive FREE
One Year	$18.00	$24.00 (U.S.)	Any software title listed below
Two Years	$35.00	$47.00 (U.S.)	Any two of the titles below
Three Years	$48.00	$64.00 (U.S.)	All three of the titles below

* Note: Canadian (and other foreign) subscribers must add $2 (U.S.) per program title to cover shipping and handling. No handling charge for U.S. subscribers. Canadian rates listed in U.S. dollars — send U.S. funds only, please.

Donald Duck's Playground

CES Software Showcase Award Winner! — Disney animation at its best! Children play four games to "earn" money to buy playground equipment. Builds money handling skills. Superb graphics. A bestseller! $39.95 retail value.

Winnie The Pooh In The Hundred Acre Wood

— Players explore the Hundred Acre Wood to find lost articles like Ow'ls books, Pooh's honey pot and Eeyore's tail, and return them to their rightful owners. Cheery music from the Disney movie caps off this computer rendition of the beloved classic. $39.95 retail value.

Omiterm Terminal Program

— Written by *The Guide's* own Bob Richardson. Fully supports the new 1660 Modem 300! (The *ONLY* commercial terminal package that currently fully supports the 1660!!!

- **Modem controls accessible from the keyboard**
- **Punter protocol — upload & download — 300/1200 baud**
- **Ten programmable function keys**
- **15 number phone directory**
- **20k receive buffer**
- **Tone or pulse dialing**
- **Auto dial/re-dial**
- **Half/full duplex**

A $19.95 retail value.

Don't delay — mail today! Supplies are limited.

The Guide
3808 S.E. Licyntra Ct.
Portland, OR 97222

Name_____

Address_____

City, State & Zip_____

Enter My Subscription for:

☐ **1 Year** ☐ **2 Years** ☐ **3 Years**

Please Check One: ☐ **New** ☐ **Renewal**

You are entitled to one FREE software package, as described above, for each year's subscription purchased. Please list your second choice, as supplies are limited on the Disney titles! Please allow six to eight weeks for delivery.

My first choice is:_____

My second choice is:_____

Software designed for AMIGA.

Lattice C Compiler — *$149.95*

With more than 30,000 users worldwide, Lattice C Compilers set the industry standard for MS-DOS software development. Lattice C gives you all you need for development of programs on the AMIGA. Lattice C is a full implementation of Kernighan and Ritchie with the ANSI C extensions and many additional features.

AMIGA C Cross Compiler — *$250.00*

Allows AMIGA development on your MS-DOS system. Price includes the above product.

Lattice Screen Editor (LSE™) — *$100.00*

Designed as a programmer's editor, *Lattice Screen Editor (LSE)* is fast, flexible and easy to learn. *LSE's* multi-window environment provides all the editor functions you need including block moves, pattern searches and "cut and paste." In addition, *LSE* offers special features for programmers such as an error tracking mode and three Assembly Language input modes. You can also create macros or customize keystrokes, menus, and prompts to your style and preferences.

Lattice dBC III Library™ — *$150.00*

The *dBC III library* lets you create, access and update files that are compatible with Ashton-Tate's dBASE system. *dBC III's* C functions let you extend existing dBASE applications or allow your users to process their data using *dBC III* or dBASE III.

Lattice Make Utility (LMK™) — *$125.00*

An automated product generation utility compatible with UNIX Make, *Lattice Make Utility (LMK)* lets you rebuild complex programs with a single command. Once you specify the relationships of the various pieces of your system in a dependency file, *LMK* automatically rebuilds your system the same way every time, and only compiles program files that have changed. But *LMK* is not limited to updating programs. You can use *LMK* to update documentation or perform **any** executable command!

Lattice Text Utilities™ — *$75.00*

Lattice Text Utilities (LTU) consists of eight software tools to help you manage your text files. GREP searches files for the specified pattern. DIFF compares two files and lists their differences. EXTRACT creates a list of file names to be extracted from the current directory. BUILD creates batch files from a previously generated file name list. WC displays the number of characters and optionally the checksum of a specified file. ED is a line editor which can utilize output from other *LTU* software in an automated batch mode. SPLAT searches files for a specified character string and replaces every occurrence with a specified string. And FILES lists, copies, erases or removes files or entire directory structures which meet the specified conditions.

Lattice Unicalc® Spreadsheet — *$79.95*

Unicalc is a simple-to-operate program that turns your AMIGA computer into an electronic spreadsheet. Using *Unicalc* you can easily create sales reports, expense accounts, balance sheets, or any other reports you had to do manually.

Unicalc offers the versatility you've come to expect from business software, plus the speed and processing power of the AMIGA.

• 8192 row by 256 column processing area • Comprehensive context-sensitive help screens • Cells can contain numeric, algebraic formulas and titles • Foreign language customization for all prompts and messages • Complete library of algebraic and conditional functions • Dual window capabilities • Floating point and scientific notation available • Complete load, save and print capabilities • Unique customization capability for your every application • Full compatibility with other leading spreadsheets.

Lattice MacLibrary™ — *$100.00*

The *Lattice MacLibrary™* is a collection of more than sixty C functions which allow you to quickly and efficiently take advantage of the powerful capabilities of the AMIGA.

Even if your knowledge of the AMIGA is limited, *MacLibrary* can ease your job of implementing screens, windows and gadgets by utilizing the functions, examples and sample programs included with the package.

Other *MacLibrary* routines are functionally compatible with the most widely used Apple® Macintosh™ Quickdraw Routines™, Standard File Package and Toolbox Utility Routines enabling you to rapidly convert your Macintosh programs to run on the AMIGA.

Panel™ — *$195.00*

Panel will help you write your screen programs and layer your screen designs with up to ten overlapping images. *Panel's* screen layouts can be assigned to individual windows and may be dynamically loaded from files or compiled into a program. *Panel* will output C source for including in your applications. A monitor and keyboard utility is also included to allow you to customize your applications for other systems.

With Lattice products you get *Lattice Service* including telephone support, notice of new products and enhancements and a 30-day money-back guarantee. Corporate license agreements available.

Lattice

Lattice, Incorporated
Post Office Box 3072
Glen Ellyn, Illinois 60138
(312) 858-7950 TWX 910-291-2190

INTERNATIONAL SALES OFFICES:
Benelux: Ines Datacom (32) 27205161 England: Roundhill. (0672)54675
Japan: Lifeboat Inc. (03) 293-4711 France: SFL (1) 46-66-11-55
Germany: (49) 7841/4500