MM70974

The Transactor

The Tech/News Journal For Commodore Computers Vol. 4.

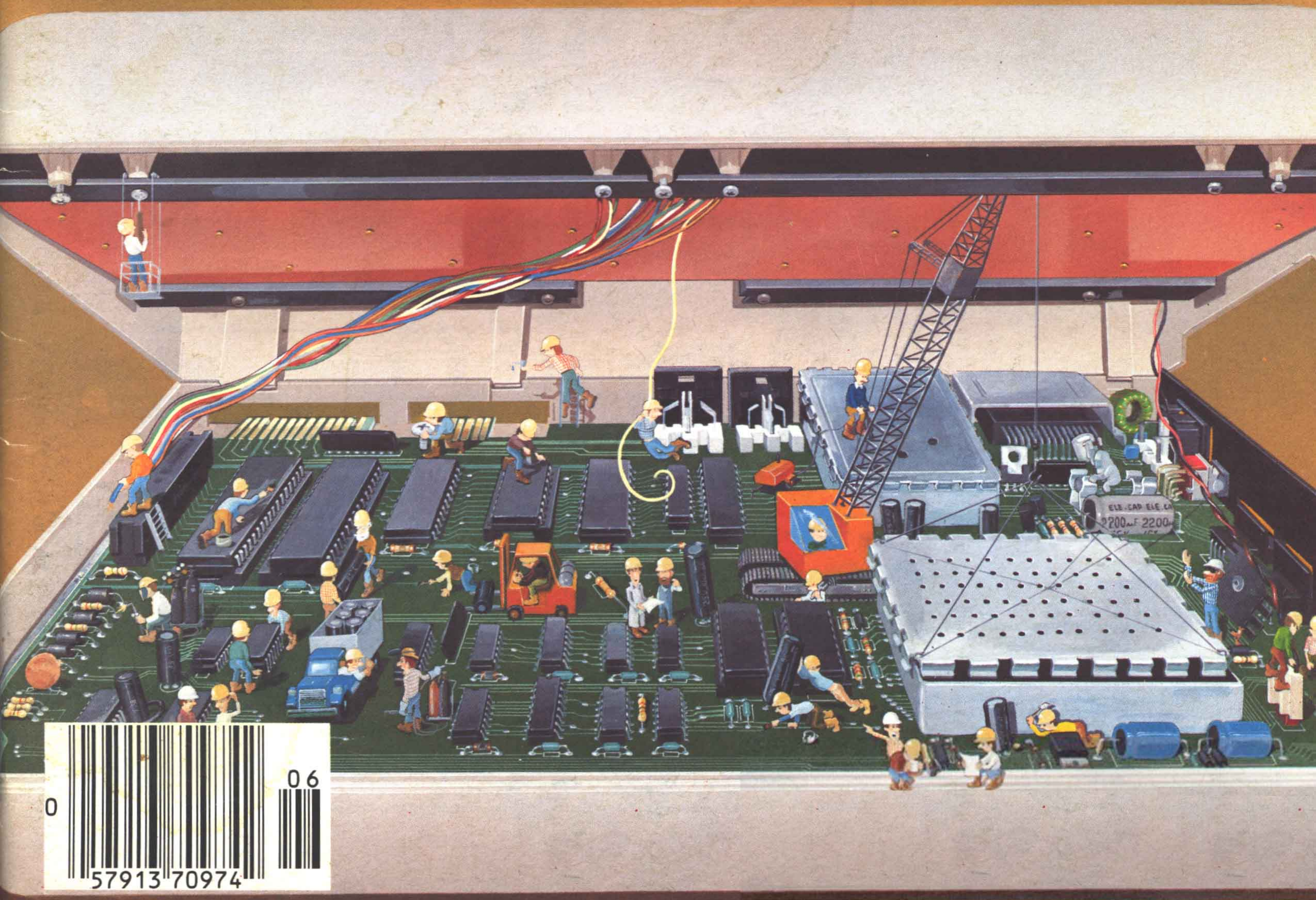
Issue 06
$2.95

Second Class Mail Permit Pending
Postage Paid in Milton, Ontario

BASIC Construction

Building Programs Should Be FUN, Not Work!

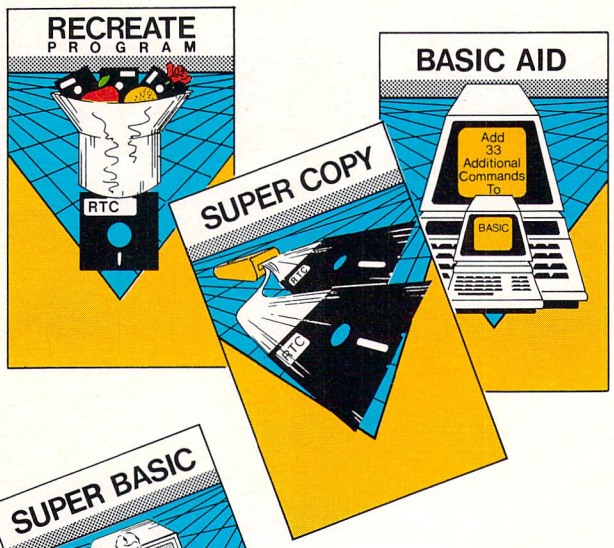
- Main Menu and Sub Menu Drivers
- Deciphering What You GET
- Subroutine Eliminators
- Disk Catalog Retrieval
- Disk Status Interrogation
- Updating Your Programs Neatly
- Put Those Function Keys To Work!
- Plus: Commodore 64 Meets Bach

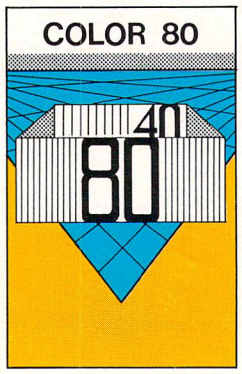


0 06
57913 70974

Developing a mind for the Future.

RTC





SUPER BASIC **$46.95** CDN.
- Gives you 3 different versions of Commodore Basic Programing Language Version 4 PLUS!
  A Built in Machine Language Monitor!
- Disk & File Maintenance Commands
  Data Handling Commands
  Graphics Plus Basic
  Compatible with Commmodore's "B" Series &
  Much Much More!

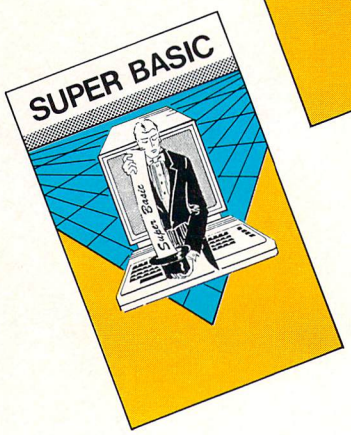
RECREATE PROGRAM **$39.95** CDN.
RTC's Answers to Program Recreation
Converts Printer's File to SCRIPT 64's Files

SUPER COPY **$39.95** CDN.
Super Fast Disk Copies on a 1541
Copy Entire Disk in 7 Minutes or Less
Copy Selected Files
Complete Pattern Matching
Full Prompts

BASIC AID **$49.95** CDN.
Your Aid to Writing Programs
Allows Scrolling Through Programs
Adds 33 more Commands to the Basic Language
Has Find, Change, Merge, Move Commands
Convert Hex, Binary and Decimal Numbers and More!

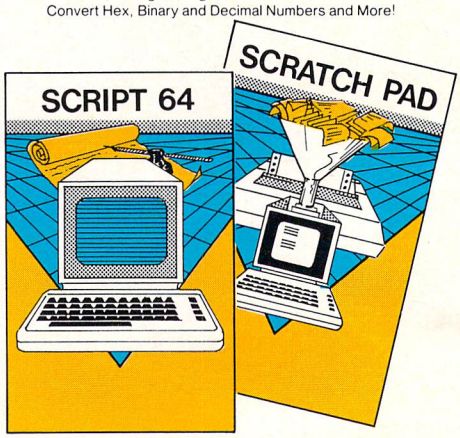
Color 80 **$59.95** CDN.

Produces 80 Columns With Color!
Needs No Extra Hardware
Use With Terminal Programs
Uses No Basic Memory in the 64
Develop Your Own Programs



NEW Combined PACKAGE
SCRIPT 64 & SCRATCH PAD 64 **$129.00** CDN.

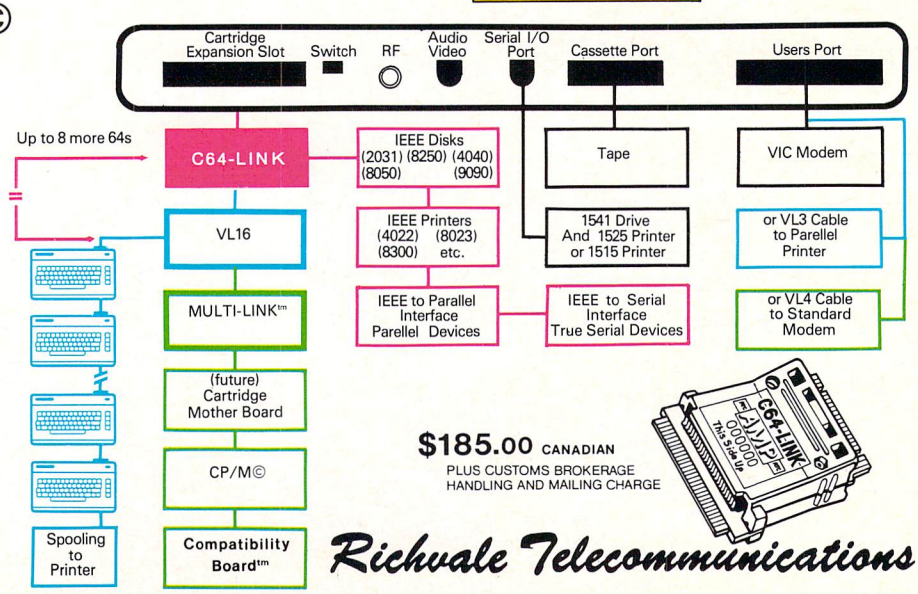
Script 64:
Word Processor in French and English
80 Columns
Global Search and Replace
User Created Dictionary
Spelling Check
Scratch Pad 64:
The Database/Mail List in One!
Merges with Script 64 Word Processor
Print out Labels, Envelopes, Mail List & More!
Suitable with both Single and Dual Disk Drives
Fully C64 Link Compatible



C64 LINK©
The Smart 64

Give These Expanded Capabilities to Your 64 and VIC 20

* The ability to transfer data from any type of device to another (IEEE, Serial, Parallel)
* BASIC 4.0 which allows you to run more PET BASIC programs and gives you extended disk and I/O commands.
* The ability to have several 64s on line together - sharing common IEEE devices such as disks or printers with Spooling Capability.
* Built-in machine language monitor.
* A built-in terminal or modem program which allows the system to communicate through a modem to many bulletin board systems and other computer mainframes.
* Compatability with CP/M.



$185.00 CANADIAN
PLUS CUSTOMS BROKERAGE HANDLING AND MAILING CHARGE

Contact your local Commodore dealer or RTC.
Payments by VISA, MASTERCARD or BANK TRANSFER.
Mail orders also by certified cheque, etc.

Richvale Telecommunications

10610 Bayview Avenue (Bayview Plaza) Richmond Hill, Ontario, Canada L4C 3N8 (416) 884-4165

nüFEKOP

P.O. Box 156, 21255 Hwy. 62,
Shady Cove, OR 97539

1-800-525-2529

Software for the VIC 20™ and Commodore 64™

Look at these Features

- Fully screen-oriented
- Horizontal and vertical scrolling
- Terminal mode — never seen before on a wordprocessor
- Supports Commodore disk and cassette handling
- Imbedded commands

Bliztext

Wordprocessor for Commodore 64

BLIZTEXT is a trademark of ELCOMP PUBLISHING, INC.

Commodore-64 and VIC-20 are trademarks of Commodore Business Machines.

Dealer and Distributor inquiries are invited.

BLIZTEXT — SUPER WORDPROCESSOR for the Commodore-64

— ON SALE NOW! —

- Fully screen-oriented, up/down, left and right scrolling — Upper and lower case
- More than 70 commands
- Full I/O compatibility with Commodore peripherals Upper and lower case
- Works with practically every printer on the market, user definable printer control commands
- INCLUDE command allows handling large files on up to 4 diskettes or on cassette.
- Build in terminal software for electronic mail and networking. Telecommunications mode, upload and download, save on disk or cassette.
- Dynamic formatting, Imbedded commands
- Single keystroke for disk directory and error channel
- Program comes on disk or cassette
- Double line spacing, left and right margin justification, centering, page numbering, and practically everything one expects from a good wordprocessor.

AVAILABLE NOW!

Order #4965 **$89.00**
Manual only (62 pages) **$29.95**

MACROFIRE - Editor/Assembler for the Commodore-64

ON SALE NOW
AVAILABLE IMMEDIATELY

One outstanding tool, consisting of 3 powerful elements combined into one efficient program!

1.) Fully screen-oriented Editor (more than 70 commands)
2.) Very fast assembler with macro capability
3.) Machine Language Monitor

Assembly can be started from the editor. Translates in 3 passes. More than 1,000 lables, screen oriented/no line numbers, scrolling, includes disk files.
Practically everything the serious machine language programmer needs everyday!

Manual only **$19.95**
Order #4963 **$89.00**

THE GREAT BOOK OF GAMES, VOL.I,

by Franz Ende
46 programs for the Commodore 64
Introduction to graphics and sound. How to program your own games. Walking pictures, animation, high resolution graphics, programming tips and tricks, hints and useful subroutines for the beginner and advanced programmer. This book is a MUST for every C-64 owner.
Come and get it — It's yours for only **$ 9.95**
Order #182 **128 pages** **$9.95**
Programs from the book on disk.
Order #4988 **$19.95**

MORE ON THE SIXTYFOUR, by H.-C. Wagner

How to get the most out of your powerful Commodore 64. Very important subroutines, tricks and hints in machine language for your C-64. How to modify DOS. How to connect a parallel and serial printer. How to design your own terminal program for communication and networking. Dig into I/O for cassette and disk.
Order #183 **$9.95**
Programs from the book on disk
Order #4989 **$19.95**

NEW PRODUCTS

Watch out for our new books, software and add-ons to come soon. ON SALE NOW! -- ORDER TODAY!

How to program in 6502 Machine Language on your C-64, by S. Roberts (Introduction)

Order-#184 **$12.95**

Commodore-64 Tune-up, Vol. I, by S. Roberts

How to expand and customize your C-64.
Order #185 **$12.95**

Small Business Programs for the Commodore-64

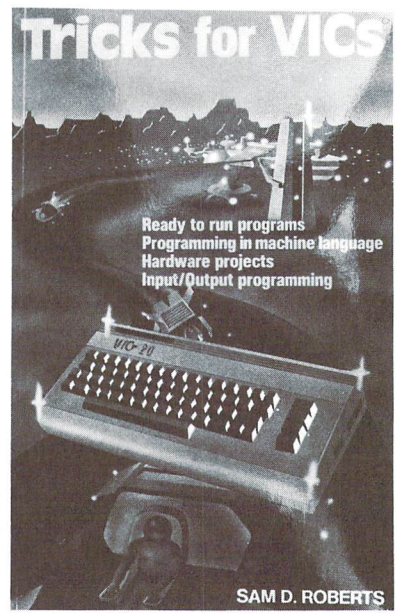
by S. Roberts
How to make money using your C-64. Mailing list, invoice writing, inventory, simple wordprocessing and much more.
Order # 186 **$12.95**

Hardware Add-Ons:

Parallel printer interface KIT	Order #4990	$ 19.95
Direct Connect Modem KIT	Order #4991	Ask f.price
Universal Experimenter Board	Order #4970	$ 9.95
Expansion Board, space for four experimenter boards(board only)	Order #4992	$ 29.95

For your VIC-20

Tricks for VICs **$ 9.95**
Universal Experimenter board



HOFACKER

PAYMENT: check, money order, VISA, MASTER CARD, Eurocheck, ACCESS, Interbank
Prepaid orders add $3.50 for shipping (USA)
$5.00 handling for C.O.D.
All orders outside USA: add 15 % shipping, California residents add 6.5 % sales tax.

ELCOMP PUBLISHING, INC
53 Redrock Lane
Pomona, CA 91766
Phone: (714) 623 8314
Telex: 29 81 91

The Transactor

Volume 4 Issue 06

The Transactor
The Tech/News Journal For Commodore Computers

Managing Editor
Karl J. H. Hildon

Editor
Richard Evers

Advertising Manager
Kelly M. George
416 826 1662

Art Director
John Mostacci

Contributing Writers
Eric Armson
Greg Beaumont
Don Bell
Dave Berezowski
Brad Bjorndahl
Jim Butterfield
Gord Campbell
Chuan Chee
Mike Donegan
Bob Drake
Alex Gaul
Jeff Goebel
Melissa Gibbins
Donna Green
Fred Hambrecht
Paul Higginbottom
Dave Hook
Eike Kaiser
Peter Lear
Bill MacLean
Mike Panning
Howy Parkins
Darren J. Spruyt
John Stoveken
Brad Templeton
Don White

Production
Attic Typesetting Ltd.

Printing
Printed in Canada by
MacLean Hunter Printing

Program Listings In The Transactor

All programs listed in The Transactor will appear as they would on your screen in Upper/Lower case mode. To clarify two potential character mix-ups, zeroes will appear as '0' and the letter "o" will of course be in lower case. Secondly, the lower case L ('l') has a flat top as opposed to the number 1 which has an angled top.

Many programs will contain reverse video characters that represent cursor movements, colours, or function keys. These will also be shown exactly as they would appear on your screen, but they're listed here for reference.

Occasionally programs will contain lines that show consecutive spaces. Often the number of spaces you insert will not be critical to correct operation of the program. When it is, the required number of spaces will be shown. For example:

print" flush right" – would be shown as – print" [space10]flush right"

Cursor Characters For PET / CBM / VIC / 64

Down – q	Insert – T
Up – Q	Delete – t
Right –]	Clear Scrn – S
Left – [Lft]	Home – s
RVS – r	STOP – c
RVS Off – R	

Colour Characters For VIC / 64

Black – P	Orange – A
White – e	Brown – U
Red – £	Lt. Red – V
Cyan – [Cyn]	Grey 1 – W
Purple – [Pur]	Grey 2 – X
Green – ↑	Lt. Green – Y
Blue – ←	Lt. Blue – Z
Yellow – [Yel]	Grey 3 – [Gr3]

Function Keys For VIC / 64

F1 – E	F5 – G
F2 – I	F6 – K
F3 – F	F7 – H
F4 – J	F8 – L

The Transactor is published quarterly by Transactor Publishing Inc. It is in no way connected with Commodore Business Machines Ltd. or Commodore Incorporated. Commodore and Commodore product names (PET, CBM, VIC, 64) are registered trademarks of Commodore Inc.

Volume 4 Subscriptions: Canada $15 Cdn
U.S.A. $15 US.
All other $18 US.
Second Class Mail
Permit Pending

Back issues are still available for Volume 3: $15 Cdn., U.S.A $17 US., all other $19 US.

Canadian Distributor:
Access Computer Services
630B Magnetic Drive
Downsview, Ontario
M3J 2C4
(416) 736 4402

or:

CompuLit
PO Box 352
Port Coquitlam, B.C.
V5C 4K6
604 464 3396

U.S. Distributor
Prairie News
2404 West Hirsch
Chicago, IL
60622
(312) 384 5350

Send all subscriptions to: The Transactor, Subscriptions Department, 500 Steeles Avenue, Milton, Ontario, Canada, L9T 3P7, 416 876 4741. From Toronto call 826 1662

Want to advertise a product or service? Call or write for more information.

Editorial contributions are always welcome and will appear in the issue immediately following receipt. Remuneration is $40 per printed page. Preferred media is 2031, 4040, 8050, or 8250 diskettes with WordPro, WordCraft, Superscript, or SEQ text files. Program listings over 25 lines should be provided on disk or tape. Manuscripts should be typewritten, double spaced, with special characters or formats clearly marked. Photos of authors or equipment, and illustrations will be included with articles depending on quality. Diskettes, tapes and/or photos will be returned on request.

From The Editor's Desk

As 1984 begins, Volume 4 of The Transactor ends with our sixth issue as an independent magazine. It will be time to renew for many subscribers - mostly those who have stayed with us since our print run was barely over 1000. I would like to extend thanks to all our subscribers – your loyalty and patience is surpassed only by our talent for publishing late.

With that thought in mind, I'm going to subject you to some shock treatment. Are you sitting down? Good. This issue marks the first one released on schedule. For now we'll be printing once every two months, paying close attention to deadlines. Our advertising, production and circulation departments have all responded to organization therapy. One of the most effective treatments has been our decision to publish "theme" issues. Page 63 has a list of projected topics. Advertisers can now coordinate ad copy with mag contents. And writers who want to submit material will find them useful as a guideline. Don't restrict yourself, however. Any topic given reasonably complete coverage will undoubtedly be accepted.

This issue also marks our largest print run ever - 20,000! Through our U.S. distributor, The Transactor will be appearing in over 750 B. Dalton book stores and also the Little Professor chain. Unfortunately this means our ad rates are going up too. Our current rates were designed for a circulation of 5000. Much as we regret price increases, it almost costs us money to accept an ad. Without the increase (our first by the way) there won't be a staff to make the increases.

With that out of the way, I'd like to address the subject of product reviews. Traditionally we haven't published reviews for several reasons. First, a review, favourable or otherwise, doesn't always represent a product accurately. Harsh criticism can ruin a product that might have been quite valuable to many. And on the other hand, a product that can "do no wrong" might turn out to be an Etzel.

On a more suspicious level, isn't it possible that the reviewer could have an axe to grind? When a review shows up in the mail that makes an item sound like a disease, I can't help thinking there might be two sides to the story. And the glittering, shining report might be just the neighbors' way of helping out with sales.

Magazine integrity is affected too. Advertisers have been known to withdraw from a publication after it prints an unfavorable critique. However, publishing nothing but good reviews falls somewhat short of responsible journalism.

All these factors add up to an extremely sensitive issue. But with an aftermarket that's so active, I feel we should be doing something to keep you informed. So after much consideration, we've come up with some positive compromises.

Products that are similar with a very distinct set of competitors will be pitted head to head in "comparison reviews". Comparable products will be chosen or qualified by us on the basis of purpose, necessary equipment, price range, etcetera, and no less than three items of the same nature will appear in each comparison. Supplementing each review will be a unique features section. For example, we all know that wordprocessors do search & replace, but what does one do that the others don't? This will not only make the buyer informed, but also the vendor. After all, products improve only when the designer knows why the competition is better.

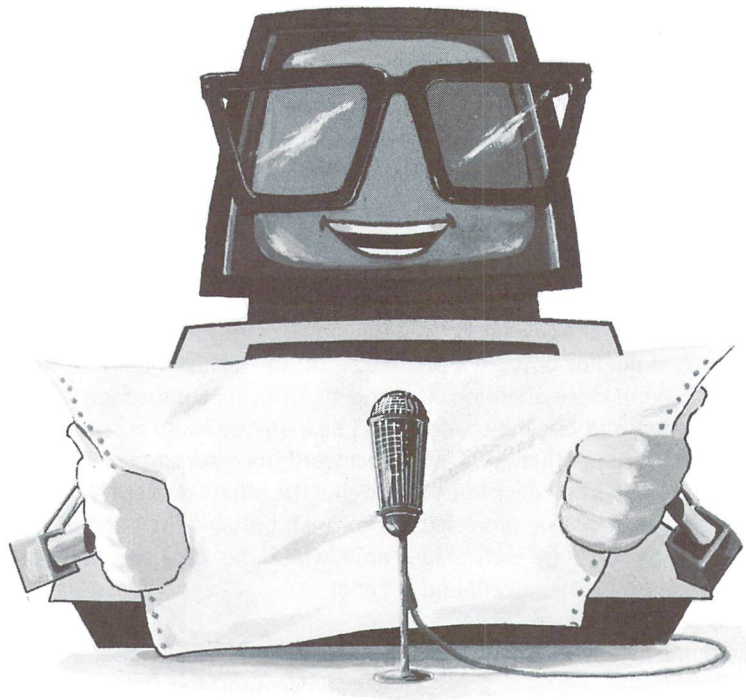
Naturally, not every product can be categorized like wordprocessors and spreadsheet calculators. Things like non-instructional books, certain games, and specialized accessories will be accepted for review and every one worthy of publication WILL be published. Since we'll be supplying the products to selected writers, we can be reasonably sure of an objective opinion. But to be fair, we'll send a copy to the manufacturer and/or distributor to allow for a response. We'll do the same with the comparisons to catch inaccuracies.

This may not be the answer but my technical half has convinced me to experiment. The approach may seem complicated, but the element of control will make this transition a comfortable one. Other details are still under advisement. Such as what happens to the items we've supplied once the review is complete. Vendors probably won't mind supplying complimentary copies of software, but would understandably be reluctant to give away expensive hardware. Perhaps someone experienced with this situation could offer some suggestion.

There's nothing as constant as change. . . until next issue, I remain,

Karl J.H. Hildon
Managing Editor, The Transactor

News BRK



Events / General

Annual May TPUG Conference

The third annual TPUG Conference is set to go for Saturday & Sunday, May 26/27 '84 at the Constellation Hotel. The two days will be packed with activity, including: speakers with presentations on a variety of topics from beginner to advanced (schedule next issue); availability of entire club software library; dealer displays; a traders' corner; an "answer room", and lots more.

The Constellation is right at the airport, 900 Dixon Road, Toronto. Lots of free parking; separate lobby for conference participants; and a special club banquet Saturday night will make this years conference the most professional yet.

You must be a TPUG member to register. Memberships are still $30 regular, $20 associate (out-of-towners). Pre-reg for the conference costs $20 and begins Feb 1. Spouses and children are $10 per person. Late registration, after April 15 or at the door is $25. Special hotel rates are being offered to registrants: $65 per night double occupancy for one night, $55 per night for 2 nights.

For more info contact the club at 416 782 9252.

NECC 6th Annual Educational Conference

The National Educational Computing Conference provides a broad and rich forum for discussion among individuals at all levels and from all institutions with interests in educational computing. It offers a unique opportunity for the cross-pollination of ideas and experiences which should result in a new higher quality of educational computing. Building on the success of the five previous conferences, the planned sessions and related activities of NECC '84 will benefit both experienced and new computer users and will stress the practical nature of computer use. The end product should be better instruction in the classroom.

With the cooperation of 13 scientific and professional organizations interested in educational computing, NECC '84 has the following major objectives: (1) to present in one forum major work regarding computers in instruction; (2) to promote interaction among individuals at all levels in the various aspects of computer uses in education; (3) to develop and coordinate the various professional groups involved with computer uses in instruction; and (4) to produce a proceedings documenting the status of computers in education.

The conference will be held at The University of Dayton, Dayton, Ohio, from June 13 to 15. In addition to seminars

and workshops, family activities and tours are offered. The conference committee is also offering travel subsidy for those who qualify. For more, contact:

Lawrence A. Jehn
Computer Science Department
University of Dayton
Dayton, Ohio 45469
513 229 3831

VDTs No Health Danger: Report

There is no reason for any person, male or female, young or old, pregnant or not, to be concerned about radiation health effects from video–display terminals (VDTs), according to a report released by Health Minister Monique Begin recently.

VDTs linked to computers are widely used in many businesses, including editorial departments of newspapers and airline ticket offices.

The report says, "radiation emissions from VDTs are either non–existant or are so low that no standard in the world would classify these emissions as hazardous." The report was based on tests the department conducted on 227 VDTs representing 122 different models.

None of the tests dealt with other health problems associated with VDTs, such as cataracts, eye strain, or neck, back and shoulder pain.

The tests found there were no microwaves, no X–ray emissions and either no or miniscule amounts of ultraviolet and infrared emissions. It did find some low–frequency radiation – the kind associated with radio transmissions – very close to the surface of some VDTs.

Education Software Donated by Micro–Ed

Micro–Ed Inc., a Minneapolis based publisher of educational software, will donate up to ten–thousand dollars worth to any elementary school library system that can meet certain standards for strengthening home/school cooperation in the area of computer–assisted instruction.

Micro–Ed's President, Thorwald Esbensen, had this to say about his company's proposed grant: "Based on my thirty–seven years of experience as a teacher and school administrator, one of the persistent problems for educators and parents has been the development of effective communication channels between home and school. Confronted now with the growth of the home computer market, it behooves boards of education and their administrators to respond vigorously to the challenge of helping families make informed decisions with respect to the proper use of educational software that can effectively supplement the academic goals of their local school systems."

"Micro–Ed's proposed software grant is designed to encourage the development of an orderly and comprehensive program for dealing with this problem. In part, it envisions the establishment of a free lending library of educational software for families. Important guidance components related to the regular school curriculum would need to be established as well."

Inquiries concerning specific qualifying standards for this grant should be directed to:

Thorwald Esbensen, President
Micro–Ed Inc.
PO Box 24156
Minneapolis, Minnesota
55424 1 800 MICRO ED

Publications

New TPUG Magazine

TPUG club members will be receiving a new magazine starting in February '84. Titled "TPUG Magazine", it is now the official club publication. The TORPET will continue to publish, but is now an independent magazine.

Computer Publications Directory

Data Courier of Louisville, Kentucky, has published a directory of magazines and newsletters that service the computer and software industries.

The 600 page book, "ABI/Selects": The Annotated Bibliography of Computer Periodicals is designed for anyone interested in industry publications. This reference describes 533 publications about personal computing, systems and software, data processing, office automation, trade and application and data communications and teletext.

Each entry includes: publication name, content description, cost, publisher and address, average number of pages, audience, advertising, as well as other information. It also contains a publisher index and a title index with cross reference to locate publications with title changes.

ABI/Selects is $50 U.S. (incl. P&H) Visa & MasterCard accepted. Order direct from:

Data Courier Inc
620 S. Fifth Street
Louisville, KY 40202
800 626 2823 or 502 582 4111

The Computer Newsletter

Rapidly expanding microcomputer hardware and software industries have created a steadily increasing need for information and instruction on the part of the computer users. Hundreds of magazines, newsletters and books containing thousands of articles are published every month. The few general publications provide introductory and background information, but most material worthy of special attention is printed in special purpose magazines designed to satisfy the needs of finite segments of the computer market.

Because of the vast amount of material, serious microcomputer users are finding that locating up–to–date information is not easy. To this end, The Computer Newsletter was conceived to provide a guide to articles. It is published 10 times per year in six different editions. There's one for Apple, Radio Shack TRS–80, RS Colour Computer, all Commodore computers, a combined edition for Atari, TI–99 and Timex/Sinclair, and a combined edition for IBM and compatible models.

A subscription costs $17.50 U.S. per year (10 issues) or ask for it at your local computer retailer. Be sure to include the name and model of your computer when subscribing.

MHN Services Inc.
Dept F4, PO Box 952
Cleveland, Ohio
44120

Multilingual Dictionary of Computer Terminology

This 1000 page, hard cover dictionary has been compiled as a source of terms used in a field marked by phenomenal worldwide growth and expansion. 11,000 computer science terms and expressions are covered in five languages – English, French, Italian, Spanish, and Portuguese.

The book costs $98.95, ISBN# 0–8442–9108–0.

Copp Clark Pitman Ltd.
495 Wellington Street West
Toronto, Ontario M5V 1E9
416 593 9911

What's For The 64

Richard V. Mucci has released this 125 page guide to products for use with the Commodore 64 exclusively. It is not a mail order company catalog.

Categories include Programs, Peripherals, Interfaces, Book Titles, Magazines, and User Groups catering to the 64. Also contains a directory of Commodore 64 support sources plus a magazine article bibliography for Commodore 64 material.

Send name and address with $15 U.S. + $2 P&H (+5% sales tax for Florida residents) to:

What's For The 64
3494 Chickasaw Circle
Lake Worth, FL 33463

Commodore News

Gortek and The Microchips

Commodore has just released a new teaching package for the 64. It's called "Gortek and The Microchips" – an exciting space adventure that teaches BASIC programming.

The planet Syntax is being invaded by Zitrons, and Gortek must work furiously to teach the Microchips how to program the great computer and ward off attacks. The full colour storybook of this adventure incorporates the Microchip Training Manual that teaches children the fundamentals of programming in BASIC.

The package includes two cassettes with educational games and other programs. Also, throughout the manual there are programs to be typed into the computer by the "trainee". At the end of the story, the great computer "Creativity" is saved by the programming the child and the Microchips have learned from Gortek. Available from all Commodore dealers.

Vanilla Pilot

Vanilla Pilot is a simple easy–to–learn programming language released by Commodore for the 64. The commands have meaningful names that are easy to remember making

it an ideal language for students or hobbyists to learn.

Originally, Pilot was developed for teachers to write programs for their students. It supports Turtle Graphics, has colour control commands, and also sound and joystick commands. The package comes with an extensive programming manual and is available now from Commodore dealers

New Public Domain Educational Software

Commodore has announced Version 2 of its public domain educational software. The new package contains 835 free programs, 258 of which are new.

"During the summer of 1983, a number of Ontario school boards and Commodore programmers created Version 2 of the Public Domain Educational Software," said Franke Winter, Manager of Commodore's educational department. "This involved analyzing, correcting and updating our previous release and adding new programs," he added.

In 4040 disk drive format there are 58 diskettes in two volumes. In 8050 format there are 27 disks. Both come with a catalogue containing titles and short descriptions.

Being in the public domain, the programs are free with a charge only for the cost to copy them. Information of pricing and availability are at your Commodore dealer.

Win VIC 20s from Kellogg's

Kellogg Salada Canada Ltd. has combined computers and education in an in-package cereal promotion in which consumers can win a VIC 20 in three ways.

Playing cards have been enclosed in approximately 4 million boxes of Sugar Frosted Flakes, Apple Jacks, and Sugar Pops. The first part of the card is a "Scratch 'n' Win" for one of 50 VIC 20 Edupacks, 1250 tote bags and 1250 binders. The second part is a "Collect-a-Win". Collect the letter and numbers to spell "VIC 20" and win one of 150 VIC 20s. The last section of the card can be filled out and sent to Kellogg's for a draw to win one of 10 Edupaks.

The contest is not all left to chance. Part of the card is a skill test in math and Canadian oriented social science. The questions have varying degrees of difficulty - players scratch out the answer they believe correct to reveal a true or false.

"Commodore is extremely pleased to be involved with Kellogg's in this promotion," said James Copland, Commodore Canada's General Sales Manager. "It's educational and it's fun which is exactly the spirit of Commodore products."

New Plant Scheduled

Commodore has announced that construction will begin on a new 125,000 square foot plant and office building on Bayly Road in Pickering, Ontario.

"The plant is expected to be in operation in the spring of 1984, weather permitting," said James Copland. "It will employ 200 people and will produce the popular VIC 20 and Commodore 64 personal computers."

World of Commodore Show a Smash Hit

One of the most successful shows of any kind at Toronto's International Center was The World of Commodore Show. It had the highest single day attendance of any show in the centers history. See our two page spread on the show following News BRK.

Software News

Easy Spell for Easy Script

Easy Spell is a spelling checker designed specifically for users of Easy Script on the Commodore 64. The master dictionary can be expanded to 20,000 words. On completing a spelling check, the file is automatically resaved. Easy Spell will also generate a word frequency report and has pattern match search capabilities.

The package comes with a User Guide, two system diskettes and a standard dictionary disk. Available from all Commodore dealers

GraphEase Computer Graphics System

GraphEase, a new computer graphics system developed by Limicon Inc. of Toronto for use on any Commodore with 32K, is a reasonably priced, very powerful package which is fully compatible with NAPLPS, the international computer graphics standard of the telecommunications computer and telephone industries.

GraphEase features: 32,000 user controllable colours; 256x256, 512x512 resolution and higher; and 1/10 second animation speed. GraphEase includes software on disk, a GraphEase replacement ROM, an RS232 interface with null modem, a NAPLPS (North American Presentation Level Protocol System) or Telidon decoder with D2 ROMs, and a complete user manual. The system will operate with normal or RGB monitors.

Telecalc II, another Limicon software product, automatically transforms Visicalc print files into GraphEase graphics, and has additional editing capabilities. Available for Commodore 4032 and 8032. Contact:

Limicon Inc.
144 Hampton Avenue
Toronto, Ontario
M4K 2Z2 416 465 4058

Computerized Lesson Authoring

Touch Technologies Inc. announces the release of an easy to use educational courseware authoring system: CLAS – Computerized Lesson Authoring System – enables educators and parents alike to produce "courseware" without any computer programming experience.

CLAS is not just a test writer It can be used to design lessons in anything from basic math and spelling, to physics and technical writing, at any level from elementary to college graduate.

Options include multiple choice, fill–in–the–blanks, and true or false questions, and the material can presented in random or prescribed order.

This version of CLAS does not include branching, record keeping, or other advanced CAI features. Instead it offers the teacher a straight forward way to develop lessons. A 20 minute lesson can be created quickly by anyone familiar with the simple Users Guide. The package includes the manual with an Author disk and Student disk. Price: $89.95 U.S. Contact:

Tom Piszkin
Touch Technologies
609 South Escondido Blvd., Suite 101
Escondido, CA 92025
619 743 0494 or 1 800 525 CLAS

Codewriter from Dynatech

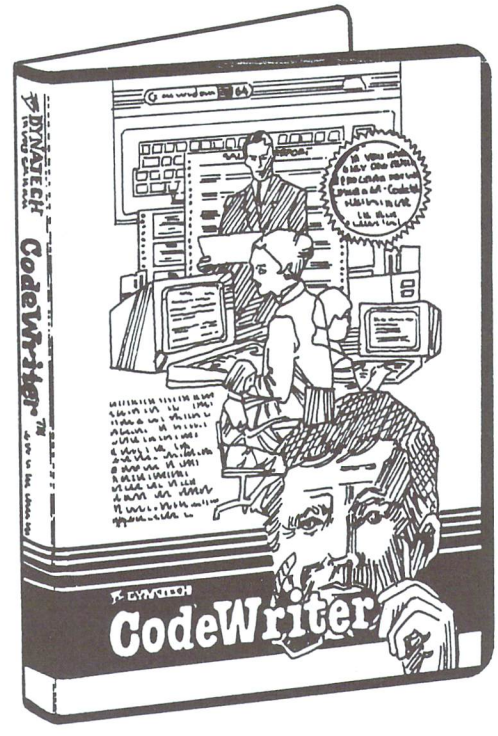
Codewriter, a new program from Dynatech Microsoftware Inc., allows users to design application software without any knowledge of computer language or programming.

Programs are designed simply by completing the screen layout form and entering the calculations onto the computer's screen in plain English. Codewriter translates that plain English into computer code and creates your own program that is specifically designed to your individual needs. Once produced, Codewriter is put aside and the new application program is used.

Codewriter will produce payables and receivables, sales analysis, customer and personnel files, mailing lists, invoicing, inventory, print reports, form letters and many other business and personal uses.

Codewriter for the Commodore 64 is $149.95 including manual. Contact your local dealer or:

Micro Marketing Canada
169 Inglewood Drive
Toronto, Ontario
416 484 7773



School Administration Software

Melcher Software has a number of programs for the PET, C64, and VIC 20 of interest to educators. Heading the list is Compugrade – a complete grading program. Compugrade will list (screen or printer) student names, averages, and letter grades (including + and – prefix) according to the scale provided by the teacher. Listings can be in alphabetical or rank order.

The Attendance Master can store complete attendance information for an entire semester. There is no limit to the number of classes. Alphabetized class lists, complete attendance records, number of absences, or tardies per student can be displayed to screen or printer.

Drill programs, clever games, statistics series, business, administration, English, science, and mathematics programs are also available. Contact:

Melcher Software
PO Box 213
Midland, Michigan
48640

Canadian Payroll Software

Robert N. L'Esperance and Associates Inc. has introduced the first of many applications compatible with Commodore equipment. The Canada Payroll system was specifically designed for use in Canada.

The system is bilingual, English and French. It will support as many as 152 employees, calculate wages hourly, weekly or bi–weekly, automatically print numbered cheques and update payroll records, and performs F.I.T., P.I.T., UIC and CPP calculations. It also offer password protection and space for an additional 5 revenues and 4 deductions to payroll records. The package is also supported by yearly tax tables and T4 output software at a nominal extra fee. Contact:

Robert N. L'Esperance and Associates
9780 Hamel
Montreal, Quebec
H2C 2W7 514 388 6962

TaxAid For C64 and VIC 20

TaxAid is an icome tax calculation program designed for home use and American tax returns for 1983. It comes in three versions, 2 for the VIC 20 and 1 for the C64. It was designed by experienced tax accountants for a line–by–line preparation of the IRS FORM 1040 and related schedules. Comes on disk or tape with a step–by–step manual.

TaxAid I for the unexpanded VIC is 24.95. TaxAid II is for VIC 20s with at least 16K expansion – $29.95. TaxAid III is for the Commodore 64 and costs $29.95. Prices are in U.S. dollars, subtract $5 for tape version of each package. Future updates will be released for nominal cost.

Northland Accounting Inc.
Software Department
606 Second Avenue
Two Harbors, MN
55616 218 834 5012

The Tax Advantage for C64

Continental Software has released their 1983 tax season version of "The Tax Advantage", a tax planning and assistance program for US IRS form 1040 and related tax schedules.

The Tax Advantage is one program you can run the first time without documentation. The program simulates the 1040 form of the computer screen and, on request, offers plain English descriptions of each item of the return.

The Commodore 64 version is $69.95 U.S. and is available at Commodore dealers and software outlets.

Continental Software
Product Information
11223 S. Hindry Avenue
Los Angeles, CA
90045 213 410 3977

Also from Continental: "The Home Accountant", world's best selling home and business financial management program (to be reviewed in The Transactor shortly); "F.C.M.", a filing, cataloging and mailing program; and "Property Management", a financial management program for owners and managers of residential, commercial, and industrial properties.

Instant Productivity Software for C64

The INSTA series of productivity software for the Commodore 64 personal computer is now available from Cimarron, a division of Microsci Corporation. The nine packages in the series are designed for immediate application with no

lengthy study and training period.

INSTA–Writer is a cartridge based wordprocessor featuring "instant–on" operation with a simplified approach to document generation. INSTA–Mail, a mailing list program fully compatible with INSTA–Writer, features merge, sorting and versatile label printing capabilities.

INSTA–Calc is a low cost spreadsheet program. INSTA–Vestor is a stock management program with buy and sell tracking, earnings and price averaging reports. Both can be interfaced with INSTA–Graph for plotting bar or line charts.

INSTA–File is the data base management package and is fully compatible with other INSTA programs.

Other misc: INSTA–Sched allows the user to manage daily appointments via a monthly calendar system. INSTA–Music helps write, store, and play music via the 64's built–in synthesizer chip. INSTA–Speed is a BASIC compiler that can increase program speed by 55 times and reduce size by 20 to 50%.

Package prices range from $19.95 to $99.95 U.S.

Microsci Corporation
2158 S. Hathaway Street
Santa Ana, CA
92705 714 662 2801

SuperTerm for VIC 20 and C64

SuperTerm, an intelligent terminal program, has been created by Midwest Micro Inc. It allows C64 and VIC 20 owners to upload and download to a wide range of computers including university mainframes, hobby bulletin boards, and Compuserve.

Features include: a text editor that can manipulate up to 18.4K of data with control approaching that of many wordprocessors; text display in 40, 80, or 132 columns with side–scrolling; transfer to and from disk; auto answer/auto dial using automodems from Commodore, HES or Microperipheral Corp; Xon/Xoff characters for transmission interrupt; 52 user–definable function keys; 26 display functions; and baud rate, parity, word size, stop bits and other communications parameters.

SuperTerm comes with a hardware module that plugs into the cartridge port, plus extensive documentation. Price: $149.95 U.S.

MidWest Micro Inc.
311 W. 72nd Street
Kansas City, MO.
64114 816 333 7200

Hardware News

Fire Command for C64 and VIC 20

Fire Command uses the finest controls like those found on real arcade game machines. An 8 way joystick is complemented by two durable fire buttons for left or right handed operation. Dimensions are 6" deep, 11" wide, and 5" high with a six foot cord for ample breathing space.

It comes in an unbreakable 5 pound die cast metal housing (no plastic) for the feel and stability of a real arcade machine. Moulded rubber feet allow no slip on smooth surfaces. The six foot cord is just right for breathing space.

[I've used Fire Command and it's by far the best one I've tried yet. The five pound base gives you that "planted down" feeling that's just unmatched by those light weight or hand held models. And the controls are (just like they say) the same as on real arcade machines. Fire Command will take a beating too! Ed.]

For more information contact your dealer or:

Access Computer Services
630B Magnetic Drive
Downsview, Ontario
M3J 2C4 416 736 4402

PET/CBM Joystick Interface

J Systems announces the immediate availability of the new PET Joystick Interface II. The interface makes the PET/CBM compatible with all popular VIC/64/Atari joysticks with connections for 2 controls. The unit plugs into the PET Parallel User Port and a fast machine language program which reads the joystick data is included along with a demo program.

Price: $19.95 U.S. plus $2.00 S&H. Includes circuit card, documentation and software. Add 5% for VISA and MasterCard orders.

J Systems Corporation
1 Edmund Place
Ann Arbor, Michigan

48103 313 662 4714

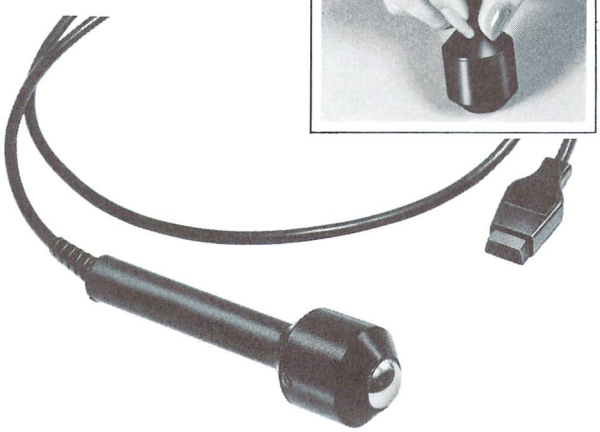
Light Pen and Track Pen for C64/VIC 20

Tech-Sketch Inc. announces two light pens for use with Commodore home computers: the LP-10S is ideal for games and educational programs. It's made of rugged high impact material, has a tooled tip and cord strain relief bushing, and a push barrel switch. Price is $39.95 U.S.

The LP-15 is a precision light pen designed for applications demanding high accuracy. It's super sensitive and can control activity from as far as six inches from the screen. The all-metal unit houses a miniature low noise amplifier and precision optics and also has a push barrel switch. Price: $119.95 U.S.

The TP-50 Track Pen is similar in function to a mouse popularized by Apple's Lisa. The pen can be used on any surface and actuates commands by simply pressing the pen down. No price information.

Tech-Sketch Inc.
26 Just Road
Fairfield, NJ
07006 201 227 7724



Koala Pad for the C64

Koala Pad is a graphics tablet input interface for the Commodore 64. The pad allows the user to draw high resolution displays on the computer screen without ever glancing at a single line of software. Attractive and meaningful colour presentations can be produced in minutes, then stored for future recall.

The package comes with the Koala Pad, carbon-less pencil, software, and user manual. Contact your Commodore dealer for details.

[Koala Pad promises to be one of the most popular accessories for the 64. Retailers would be foolish not to carry it. The software is superb – a truly fine piece of work. Ed.]

RB Robot Seeks Software Developers

Micro Marketing Canada, Canadian distributor of the RB5X Robot, is looking for Canadian software developers to write commercial quality programs for the RB. Selected programs will be produced, packaged, and distributed by the RB Robot Corporation through its nation-wide retail dealer network.

Software for the RB will be evaluated for creative use of sound, sensing, and movement capabilities, appeal, effectiveness, error handling, documentation, and ease of use.

Participants must purchase or otherwise independently obtain an RB5X and the appropriate hardware options to develop their programs. They may purchase robots at a special price directly from Micro Marketing Canada; if their programs are accepted for distribution, developers will receive a 25% rebate on their hardware purchase. For more information:

Micro Marketing Canada
169 Inglewood Drive
Toronto, Ontario

RB Robot To Robot Communications

The RB Robot Corporation, in cooperation with the Center for Science and Industry (COSI) in Columbus, Ohio, and the Museum of Natural History in Denver, Colorado, announced the first transcontinental phone conversation and interaction between two robots, initiated by the robots themselves.

The event featured two RB5X personal robots. The RB in Columbus initiated the call that was answered by the other in Denver. They conversed, and one instructed the other to perform a simple task. The task was completed and the second robot reported back to the first.

RB President Joe Bosworth said, "This event represents a step as dramatic for the field of home robots as Alexander Graham Bell's first demonstration of transcontinental telephone. Using communications technology, the ability for the robot to summon help in emergencies will be of the most useful features in the home."

The First Annual World Of Commodore Show

This show had to be the computer "event of the year". Boasting a long list of highlights, it was Canada's answer to "The PET Show" in England. Commodore will tell you they held it to celebrate their 25th anniversary. Either way, the show was a smash success, and Commodore already has a list of improvements planned for next year's occasion.

The most impressive point had to be the attendance figures - 38,000 total with 12,000 of those on the final day, the largest single day attendance of any show in the history of the International Center.

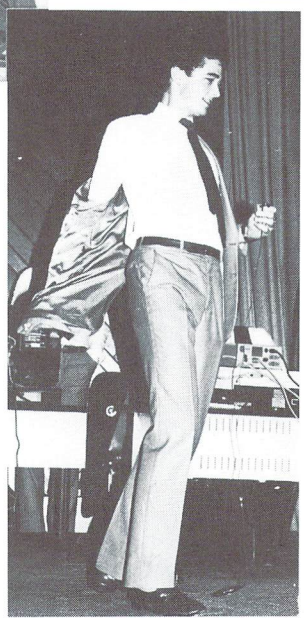
Any company needs quite some following to sponsor a show as specialized as this. Needless to say, every exhibitor was somehow related to Commodore. The concentration was a perfect atmosphere for business and pleasure. The Canadian Computer Show, held a month earlier at the same location, was dull by comparison. Too many product lines representing a cross section of the entire computer industry means a lot of walking in a place that size. The first World of Commodore show could easily be the start of a new trend in computer exhibitions.

Originally, there were to be several more exhibitors than actually showed up. Probably due to a lack of reputation as computer shows go. Undoubtedly those vendors won't be making the same mistake next year.

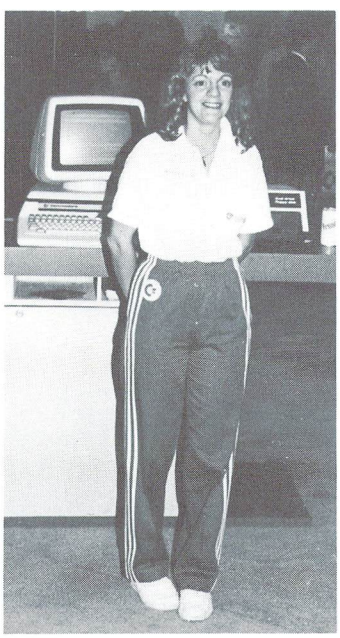
Donna Green, Dave Berezowski, Paul 'SID' Higginbottom, Karl Hildon, Joe Ferrari and Jim Butterfield

Jim Gracely, Dianne LeBold, and John O'Brien
of Commodore's magazine department

Kelly George, Transactor Ad Manager.



The Music Demo with Paul, Frank Covitz and...



...The after effects of a computer show, demonstrated by
Jack Livesly (TVOntario), Jim B., Paul, Steve Murri (CBM US), Frank and Clifford Ashcraft.

Bits and Pieces

Incrementation

Our screen dazzler this issue is "Incrementation", a concept originally from our new Editor, Richard Evers. Since the original version there have been several mods – portability, set–up procedures – and credit for them in line 10. Frankly, the program itself is virtually useless, but aren't all screen dazzlers? Perhaps someone looking for an eyecatcher can let us know how well it works.

Moreover, the program demonstrates once again the phenomenal speed of machine language. To start, a pattern of incrementing ASCII values is POKEd into screen memory. Hit 'S' and the machine code takes over. Quite simply, the code increments the contents of every byte in screen memory by 1, unless the byte contains a space. They are left untouched but this part of the program could be removed for some really brain twisting effects.

The program is also a lesson in extending the boundaries of the thought process to beyond the ultimate goal. When I first saw the program in action, I immediately thought of several more difficult ways to accomplish the same effect.

Operation is simple – use the cursor keys to direct the character stream. Alternate between the space bar and cursor keys to leave gaps (ie. for messages). 'S' starts it and Shift stops it. Once stopped the pattern can be continued or cleared. The program has been published so that parts can be removed to suit your machine. Use line 15 to store changes – just space over the line# and "REM" and hit return.

Of course you need not enter the parts that will get removed, but if you do, SAVE a copy first so that it can be passed to a friend that may have different equipment than yours. For the VIC 20, the first "4" would change to a 16 or 30 depending on the high order address of your screen memory. You will also need to ensure the machine code is deposited in "real" memory which means the FOR/NEXT loop (line 120) and the SYS address (line 60) will require changes.

```
 0 rem *** this version is totally relocatable !!! ***
10 rem a rico mariani, chris zamara, rick evers,
   and karl hildon production
15 rem save"@0:incrementation",8:verify
   "0:incrementation",8
20 print chr$(147);
25 rem ******************************
30 rem * 4.0 - key=151:q1=196:q2=197
   :q3=198:sp=32
35 rem * c64 - key=203:q1=209:q2=210
   :q3=211:sp=60:poke53281,493-peek(53281)
```

```
 40 rem ****************************
 45 a=0:gosub120:rem * stash the data at $6000 *
 50 get a$:if a$="" then 50
 55 aa=asc(a$):if aa = 19 or aa = 147 then print a$;
    :goto 50
 60 if a$ = "s" then sys24576:rem *** press the 's'
    key to start increment ***
 65 if peek(key) = sp then 90:rem * space bar to
    indicate desire to move *
 70 aa=asc(a$):if aa <>17 and aa <> 29
    and aa <> 145 and aa <> 157 then50
 75 printa$;:gosub115:pokee,a:a=a+1
    :ifa=256thena=0
 80 if a = 32 then a = 33:rem * it's a space !! *
 85 goto50
 90 gosub115:pokee,peek(e)or128
 95 geta$:ifa$="" then95
100 aa=asc(a$):if aa <>17 and aa <> 29 and
    aa <> 145 and aa <> 157 then95
105 printa$;:pokee,f:goto50
110 rem *** mark the screen location and remember
    what was there ***
115 e=peek(q1)+peek(q2)*256+peek(q3)
    :f=peek(e):return
120 for j=24576 to 24625 :readx:pokej,x:next:return
125 rem *** data for inc-80 ***
130 data 169, 128, 133,   1, 169,   0,
         133,   0, 168, 177,   0, 201,  32
135 data 240,  14, 133,   2, 201,  31,
         208,   2, 230,   2, 230,   2, 165
140 data   2, 145,   0, 165, 152, 208,
          15, 200, 192,   0, 208, 227, 230
145 data   1, 165,   1, 201, 136, 208,
         219, 240, 208,  96,   0
150 rem for 4000 series change 136 in line 145 to a 132
155 rem *** data for inc-commodore 64 ***
160 data 169,   4, 133, 106, 169,   0,
         133, 105, 168, 177, 105, 201,  32
165 data 240,  14, 133, 107, 201,  31,
         208,   2, 230, 107, 230, 107, 165
170 data 107, 145, 105, 173, 141,   2,
         208,  15, 200, 192,   0, 208, 226
175 data 230, 106, 165, 106, 201,   8,
         208, 218, 240, 207,  96
```

Moneyout

No this program doesn't deal with the Christmas season aftermath. It's a subroutine that will format dollar figures for output. Sure you've seen lots of them before, but this 7 liner is so compact and tidy, we felt it worthy of a re-press. Of course it was written by Jim Butterfield.

The routine formats to 2 decimal places and adds trailing zeroes. V1 specifies the maximum number of digits left of the decimal place – V2 is the precision after the decimal place. An overflow will be displayed as all asterisks. The demo routine (lines 100 & 110) will show you the possibilities when the control variables are changed.

Although this routine won't dissolve any financial muck, it will make the muck look prettier.

```
  100 v1=4:v2=2
  110 v=rnd(1)*12000:gosub9000:printv$:run
 9000 rem print format for money
 9010 v4=int(v*10↑v2+.5)
 9020 v$=right$("          "+str$(v4),v1+v2+1)
 9020 v$=rig :rem 8 spcs
 9030 if v2<1 goto 9070
 9040 forv5=v1+2 to v1+v2+1
       :ifasc(mid$(v$,v5))<48thennextv5
 9050 v6=v5-v1-1
 9060 v$=mid$(v$,v6,v1+1)+left$(" .00000 ",v6)
       +mid$(v$,v5)
 9070 ifasc(v$)>47then
       v$=left$(" ********** ",v1+v2+2+(v2=0))
 9080 return
```

Palindrome

You have just become one of the few people that have actually read the word "palindrome" – it's not exactly part of everyday conversation. Nor should it be. A palindrome is something that reads the same way backwards as it does forwards. Words like mom, dad, eve, and clumsmulc (clumsmulc?) are all palindromes. Same for numbers – 23632 is a palindrome.

As it turns out, all numbers can eventually be made into palindromes, except one (more later). The idea is: pick a number, reverse the order of the digits, and add it to itself. If you don't get a palindrome, repeat the above using the result of the previous summation. For example:

```
Number:    158
Reverse    851
=         1009
Reverse   9001
=        10010
Reverse  01001
=        11011 – a Palindrome!
```

Not all numbers take so many iterations (eg. 56). Others require several, like 98. Regardless, Jim Butterfield felt it

was a perfect candidate for a machine language program since a similar program in BASIC might take hours to eventually reach ?OVERFLOW ERROR.

The following program generates palindromes from some given value. Jim cuts it off at 255 digits – "I figure if it doesn't palindrome by 255 digits, it's not gonna. And it seems there's one relatively small number that doesn't" – What's that Jim? – "well it lies between 150 and 200." – I hate it when he does that.

```
100 data 162,  0, 142, 226,  3, 189, 219,  3
110 data  32, 210, 255, 201,  32, 240,  3, 232
120 data 208, 243,  32, 228, 255, 201,  13, 240
130 data  24, 201,  48, 144, 245, 201,  58, 176
140 data 241,  32, 210, 255,  41,  15, 174, 226
150 data  3, 157,  0,  24, 238, 226,  3, 208
160 data 225,  32, 210, 255, 234,  32, 225, 255
170 data 240, 100, 162,  0, 172, 226,  3, 169
180 data  0, 141, 227,  3, 141, 228,  3, 189
190 data  0,  24,  9,  48,  32, 210, 255,  41
200 data  15, 217, 255,  23, 240,  3, 238, 227
210 data  3,  24, 121, 255,  23, 109, 228,  3
220 data  78, 228,  3, 201,  10, 144,  5, 233
230 data  10, 238, 228,  3, 153, 255,  24, 232
240 data 136, 208, 212, 173, 228,  3, 141,  0
250 data  24, 174, 226,  3, 172, 226,  3, 173
260 data 228,  3, 240,  6, 200, 238, 226,  3
270 data 240,  20, 189, 255,  24, 153, 255,  23
280 data 136, 202, 208, 246, 169,  13,  32, 210
290 data 255, 173, 227,  3, 208, 153,  96,  86
300 data  65,  76,  85,  69,  63,  32
310 for j=828 to 993 : read x : t=t+x
320 poke j,x : next j
330 if t<>22051 then stop
400 sys 828
410 goto 400
```

Auto Liner

Most program listings in print have usually been "renumbered" – they start at some line like 100 or 1000 and proceed in nice neat increments of 10. Depending on the length of the listing, just entering the line numbers can take a considerable percentage of the total time to enter the entire program. This is why several of the programmers packages available have included an Auto Line Numbering feature.

Quite simply, Auto Liners print the next line number and leave the cursor just beyond for you to enter the code. That's exactly what these do, 'cept it won't cost you anything. The first is for BASIC 4.0/2.0 users, the second for VIC 20/C64.

```
60000 input "auto: start, increment";s,i
60010 print "Sqqq"; s;:poke167,0
60020 geta$ : if a$ = "" then 60020
60030 print a$; : if asc(a$)<>13 then 60020
60040 p = peek(33009 + len(str$(s))) : if p = 320
      or p = 160 then 60010
60050 print "s = "s+i ":i = "i ":goto60010s"
60060 poke158, 2 : poke 623,13 : poke624,13
60070 end
```

```
60000 input "auto: start, increment";s,i
60010 print "Sqqq"; s;:poke204,0
60020 geta$ : if a$ = "" then 60020
60030 print a$; : if asc(a$)<>13 then 60020
60040 p = peek(1265 + len(str$(s))) : if p = 320
      or p = 160 then 60010
60050 print "s = "s+i ":i = "i ":goto60010s"
60060 poke198, 2 : poke 631,13 : poke632,13
60070 end
```

DisClosed Files

How many times have you been in a file routine when something goes wrong. The program breaks and the disk file is left open with the LED left staring you down with a look of inadequecy. If you're just reading the file, a DCLOSE will tidy things up. That's if you have BASIC 4.0 or equivalent (ie. V/C–Link). With BASIC 2.0 you need to give the basic CLOSE command, followed by the logical file number. And if you can't remember which number you used, start looking.

If you're just reading files, cleaning up disk channels is not really critical. The disk unit can deal with this oversight and your files are not affected. But it is especially important that files open for writing are properly closed. It could mean the difference between a smooth operation and chronic teeth gnashing!

Once again, a DCLOSE command or a CLOSE followed by the correct file number will close your write–files under most conditions. But certain program errors, and especially program editing, will terminate communications with the disk, and your files are left in a state of potential doom.

Fear not! All Commodore disk units have a built–in procedure for closing any open files, read or write. Simply closing the Error Channel does it all! If the Error Channel isn't open, OPEN it – then CLOSE it. Quite simply:

```
OPEN 1, 8, 15 : CLOSE 1
```

The ", 15" specifies the Error Channel – common to all CBM drives. A look at the code in the disk ROMs shows a routine that examines all possible secondary addresses – 0 to 14 – and closes any with apparent activity. The routine is executed whenever the Error Channel is CLOSEd.

This might sound too good to be true. Well, sometimes it is. Other more nasty errors like Disk Full and machine crashes may force you to leave a file improperly closed. When this happens an asterisk is displayed beside the file type when a Directory is printed. Do NOT Scratch them. "Star files" can be hazardous to the good health of other files. Sector links at the end of the star file might lead to other sectors that are part of another file(s). These sectors will also be de–allocated which means the disk could use them later for new stuff and whammo. Instead, use the Collect command. It will wipe out any star files while protecting the integrity of the others.

"But I've got valuable data in that blasted star file that I don't want to lose!" If that's the case you can still get at it. One little known and virtually undocumented feature of all CBM drives is the file Mishaps option. That's right – you can OPEN for Reading, Writing, or Mishaps. It allows you to dig into a star file, the result of some unfortunate event, and extract as much data as you deem necessary. For example:

```
OPEN 8, 8, 8, "0:SOME STAR FILE, S, M"
```

Actually the M probably doesn't stand for Mishaps. Regardless, once the file is open you can start to GET# data and transfer it to a new file. Depending on how badly mangled the end of the file is, you can repair the end of the new file with direct PRINT#'s.

Once you feel the new file is complete AND properly Closed, do a Collect to purge the bad file(s).

Direct Error Reads

Reading the Error Channel (Secondary Address 15) isn't a problem for BASIC 4.0 users – you simply PRINT DS$, the Disk Status string. Those without BASIC 4.0 (ie. 2.0, C64, VIC 20) are probably familiar with this subroutine:

```
60000 open 1, 8, 15
60010 input#1, e1$, e2$, e3$, e4$
60020 print e1$;e2$,e3$;e4$
60030 return
```

This routine needs to be programmed in memory because the INPUT# command won't work in direct mode. Of course if this code is already part of your program you can RUN it or GOTO it, but that can cause other headaches. The following allows reading the Error Channel in direct mode:

```
open 1, 8, 15    :unless already open
for j = 1 to 40:sys51844 #1,e$:print e$;:if st = 0 then nextj
```

This is for BASIC 2.0. For the 64 the SYS address changes to 43906 and for the VIC 20 it's 52098. Although you won't need for this circumstance, the address for BASIC 4.0 is 48001. You can omit any spaces from the statement.

The SYS jumps into the ROM GET routine 7 bytes past the start – this is where it checks for direct mode and sends the "?illegal direct" error message. This works for GET# but not INPUT# – it checks for direct mode differently and can't be skipped short of writing your own machine language routine.

Understandably this can become rather tedious. You might consider one of several programmers aids that include a direct mode error reading command.

Hard Disk Formatting

If you have any plans to install a Commodore hard disk drive, chances are the first command you send to it will be a New or HEADER operation. Disk users will know that this formating procedure is necessary to prepare the unit for all future operations. But once you get it started, you might as well find something else to do for a while like learn to play piano or re–build the engine in your car. A Header operation on the hard disk can take as long as 1 hour 45 minutes because the ID you select is recorded on every sector header.

You need only do this once unless you wish to change the ID. A Header without the ID merely clears the BAM (Block Allocation Map) and the directory – the rest of the disk is left untouched. If you do decide a re–format is necessary, just remember it will take a while.

Two other hard disk notes: The unit should never be moved while the cylinder is spinning. It takes a minute or so for the cylinder to come to a complete stop after power–down. When moving it, keep the unit level – don't set it on end or its side. Hard disks should be kept on a good solid surface during operation. Even small vibrations can cause undue wear on the disk bearings. Avoid shelfs, tables with long legs or spots that may get bumped by passers–by.

Lastly, Commodore hard disks don't have a drive 0 and drive 1 – only drive 0. Some software packages assume you

have a dual floppy and will attempt to access drive 1. Even BASIC tries to read drive 1 when you give a Catalog or Directory command with no drive specified. If you're experiencing any trouble, just slip in a ",D0" or "0:".

Disk De-Activity Indicator

If you're disk unit gets into some long operation, like the one mentioned above, you might not notice that it's finished until the next time you browse by. If you have a bell built into your computer, here is one way to make it useful:

```
print ds$ : poke 231, 100 : print " ggg "
```

The POKE increases the chime time of the bell, and the 3 reverse G's invoke it 3 times. If you're within ear-shot, this should be enough to get your attention. Or you can put the bell in an endless loop that stops when you hit a key. Only one problem with this though – if you happen to step out or something, that insidious chiming is enough to drive someone bonkers if left exposed to it for too long. You might come home to find your new hard disk is now a chopping block in the kitchen!

Weirdities

Here's our latest collection of crashes and assorted phenos for Commodore machines. None of them will harm your computer (unless otherwise specified) but don't try them with anything important in memory – we get enough hate mail as it is.

DLOAD'N

On any BASIC 4.0 machine type:

```
dload " [ESC][RVS] n
```

Don't ask why. We don't know. Could be like eating a power cookie.

Five and Dime

Try this on an 8000 series machine. The screen should be in upper/lower case with "unsquashed" lines. It alters the 6545 video chip and although won't hurt anything, just the same, don't let it run too long.

```
poke 59520, 5 : poke 59521, 10
```

To get out of it, hit both Shift keys and "2" (the one over the Q, not the one on the keypad).

Pirate Peeves

Want to drive program pirates crazy. You must admit, if a burglar really wants in, he'll get in no matter how much protection you have. Program pirates are no different. The idea is to make them work for it. As they remove one lock, you check for it later in the program and throw them a couple of knuckle balls. Here's a couple of knuckle balls:

```
poke 175, 3
```

. . .switches the input device from the keyboard to the screen. For VIC 20/C64 use poke 153,3. Of course the pirate will remove this rather unsophisticated excuse for protection. So, you check for it. Then execute:

```
sys 57441
```

This turns off the keyboard completely except for carriage returns. It has no VIC20/C64 equivalent. The point is, if you make it appear as though the more they unprotect your software, the more foul it behaves, pretty soon they'll be replacing the protection they removed just so they can use it. The Transactor will be devoting an entire issue to software protection and piracy in August 1984. Watch for it.

RAM Expansion

Wanna show off to that overinflated 48K Apple user next door. Try this on 'im.

```
sys 54295   ;BASIC 4.0
```

Of course you know there can't possibly be that much, but how's he gonna know. Stay sharp though – he may know his Apple ROMs with equal impunity.

Marquis de Sade

Try this first. It won't be too impressive to begin with but give it a chance to reach the 5 digit numbers.

```
for j = 1 to 1e30 : print j " [CRSR UP] " ; : next
```

What an effective display for some 5 letter message – if I could only think of one.

Instant BASIC Monitor

You can execute this SYS directly, but it won't mean much. However, put on the end of some line in your program and it

will report what line that is. It's part of the error message routine – the part that reports the line number after a run is interrupted. For example: ?syntax error in 6010.

Immediately upon entering direct mode, the operating system deposits an FF into the high order byte of the Current Line Number word, thus rendering that information meaningless. During program execution, the current line being executed is copied here. Try this:

```
100 print "line 100"
110 print "some line " : sys 53112:?
120 sys 53112
130 goto 100
```

For C64 it's SYS 48585 and the VIC 20 is 56777. As you can see, if the program is renumbered, the message in line 100 will need updating. The ROM call, however, is universal. You can skip the "in" display by adding 7 to the address.

Taking this one baby step further, this routine could be used as a primitive hex to decimal converter. For example:

```
poke 55, 10 : poke 54, 7 : sys 53119
```

. . .would print the decimal equivalent of $0A07. (C64/VIC20: use 58 and 57 for the POKE addresses, with the above SYS address + 7) This is ok when you can convert the low order and high order to decimal in your head. When numbers get like CF7D it's a little tougher. Perhaps someone with BASIC 4.0 (built–in MLM) will investigate a more sophisticated approach to the same end. (Dave Hook, you there? You did that once long ago, did you not?)

Text In Drag

Back when there was only 40 Column PET/CBMs, The Transactor discussed a method of attaching the code from one line of BASIC onto the end of the line above it. For example, the following two lines need not be on two separate lines:

```
100 for j = 1 to 10
110 next
```

The two lines would be LISTed on the screen. Then using the space bar to wrap around the top line onto the bottom line, the two lines would become one double line. Naturally it only worked for a range of 80 characters. The top line had to be less than 40 characters long and only the first 40 characters of the line below could be transfered. With the two screen lines linked together, the DELete key would be used to "drag" the line below up onto the line above, squeezing out the line number plus those extra spaces in the process. Of course you'd have to insert a colon yourself and also delete line 110. Try it! (By the way, this doesn't work on the VIC 20/C64 – the screen editor has been changed and is now too smart for this trick to be pulled off)

When 80 Column machines came along there was no longer any need for a line wrap table in low RAM – all the lines were identical regardless of how much code was entered. So the situation above could still occur, but to add one line to another required re–typing.

(You must know what I'm leading up to. Yes, a POKE this time) By making the operating system "think" that each line is longer than 80 characters, this same trick can be played on the 80 column screen editor. Slight of hand? No. More accurately, "right of hand". Location 213 is the right hand window margin. Normally it contains 79 for 80 column lines (0–79):

```
poke 213, 159
```

. . .will make the editor think that each screen line is 160 characters long, however, you're still limited to 80 characters per program line.

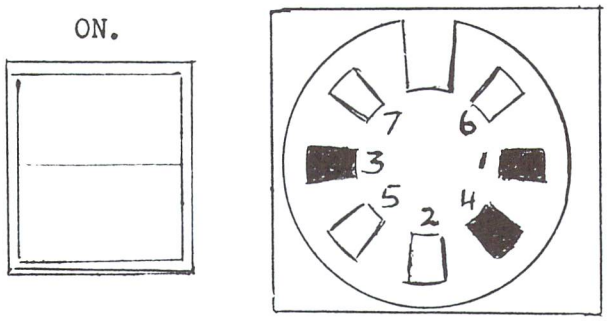
Now, about the only way I could make this work was to start from the beginning of the first line and " cursor right " all the way over to about column 75. Only then would DELete cooperate and drag the bottom line up. Using cursor left to "go around the other way" created some other problems.

Once you're done you'll want to restore 213 to a 79. Otherwise you'll get some strangeness occurring. Try POKE 213, 255 and cursor down off the bottom of the screen. Weird, eh? Again, I don't know why. And if you think that's weird, try listing a program that will cause screen scrolling. You won't crash the machine, but you may have to clear the screen before entering any new direct commands. Try experimenting. I'm not sure, but it might even work for 3 screen lines (ie. POKE 213, 239).

The Wooden Wedge

Ted Evers of Toronto has this valuable advice: It has been observed that if the power–supply plug for the C–64 is supplied with 4–pins instead of the standard 7–pin type, damage can, and will occur to the computer and power supply circuitry if this plug is inserted into its power socket the wrong way. The shallow key of the plug will allow this to happen.

To overcome this shortcoming, fill the three unused socket holes with wooden toothpicks. The offending sockets are pin #'s 1, 3 and 4, indicated by the blocked in areas of the diagram.



Power Plug
C64 Right Side View

Some More C64 Hardware Tips

In an earlier issue we published one procedure for a video allignment on your C64. Tony Lamartina of Chicago has another, but it's more suited to service technicians or those with access to necessary equipment.

1. Remove metal cover from video RF area.
2. Connect scope lead between ground and pin 1 of IC 31.
3. Adjust R27 for 1.5 volt DC level.
4. Connect scope lead between ground and pin 4 of DIN plug.
5. Adjust R25 for midposition
6. Connect decade box between pin 4 of DIN plug and right side of C78
7. Adjust resistance of decade box for 0.8 to 1 volt of signal level on scope
8. Connect resistor of value determined by decade box across pin 1 and pin 4 of DIN plug
9. Fine adjust R25 for best display of colour monitor.

Tony also suggests the following be checked on early releases of the 64:

1. Loose RF box covers. Tighten the metal tabs and re-install.
2. Check for a missing heat sink on VR2-7805 voltage regulator. Install suitable heat sink.
3. If the unit displays "sparkling" (interference across the CRT screen, random in nature) connect a 330 picoF capacitor from pin 20 (ground) to pin 30 (address 6) of the 6567 (VIC II) video chip. Make this connection outside the RF shielded box.

Octopus Syndrome

Tony also has this tip for those using a VIC20/C64 with multiple peripherals: If more than one 1541 disk drive is connected, bus lock-up will occur if all the units are cycled on at the same time (such as using a power bar main switch). To avoid this, turn on the 64 first, then one disk, then the other. Same with the serial printers – turn them on last.

DATAdjuster Update

In Volume 4, Issue 04 we presented an item called "DATAdjuster"; a routine using POKE that would position the Data Pointer for your next READ command. It seems a potential bug can invade that version of the routine and Elizabeth Deal of Malvern, PA, has sent us a new one.

There are two POKEs that do all the work. Quite simply they deposit a copy of the CHRGET pointer into the Data Pointer. However, if a page boundary of text memory is crossed in between the two POKEs, the high byte of the CHRGET Pointer is incremented by 1 and this value throws the Data Pointer forward by almost 256 bytes. Changing the order of the POKEs doesn't help because the same thing would happen, only in reverse – the pointer would be sent backwards through memory by 256 extra bytes. Therefore, this routine is what's called "position dependent". Of course you can't write programs to accommodate subroutines. . . quite the contrary.

Elizabeth has come up with a routine that is "position independent". It no longer uses the CHRGET Pointer but rather the BASIC CONT Pointer. This pointer is constantly updated by the operating system to point at the "link bytes" (stored at the beginning of each line of text) of the line currently being executed. The link pointer always points to the link bytes of the NEXT line of text. This value, then, is perfect for transfering to the Data Pointer. Liz subtracts 1 so that it ends up pointing at the zero byte at the end of the current line – this would be the line that contains the POKEs. The next scan for DATA would begin with the line immediately following.

Type in the demo program with no extra spaces in the first 6 lines.

```
100 data  0123456789  0123456789  0123456789
110 data  0123456789  0123456789  0123456789
120 data  0123456789  0123456789  0123456789
130 data  0123456789  0123456789  0123456789
140 data  0123456789  0123456789  0123456789
150 data  0123456789  0123456789  0123456789
160 poke 62, peek(119) : poke 63, peek(120)
170 read a$ : print a$
180 data next read should be of this line
190 data however
200 data this will never be found
210 data the colon in line 160
220 data crossed a page boundary
230 rem ----------------------
240 rem   random restore - liz deal
250 rem ----------------------
260 a1 = 58 : a2 = 62 : rem c64/vic20  =  61 & 65
270 def fn pp(q) = peek(q)  + 256 * peek(q + 1)
280 def fn hi(q) = int(q/256)
290 def fn lo(q) = q-256 * fn hi(q)
300 y = fn pp(fn pp(a1) + 1)-1
310 poke a2, fn lo(y) : poke a2 + 1, fn hi(y)
320 read a$ : print a
330 data this is position independent
```

For VIC 20 and Commodore 64 users, line 160 changes to:

```
160 poke 65, peek(122) : poke 66, peek(123)
```

Lines 260 to 290 would be placed near the start of text – they're only executed once. Lines 300 and 310 do the work. They need to be duplicated immediately preceding each READ for which positioning is desired.

As previously discussed, one might save the contents of the Data Pointer in two other variables before altering it. This way it could be restored to its previous position at some later time.

Reference Issue Update

On page 56 of The Transactor Reference Issue is a list of User Callable Subroutines. Number 21 prints a string from an address in the A and Y registers. Somewhere around there, if you can fit it, an additional note should be added: the end of the string is indicated by a zero (ie CHR$(0)). That's all for now.

This list is one we would very much like to expand for future re-release. Any suggestions are welcome. Of course if you find any errors, we would like to hear about them too.

Book Review :

Melissa Gibbins – Oakville, Ont.

Megabucks From Your Microcomputer

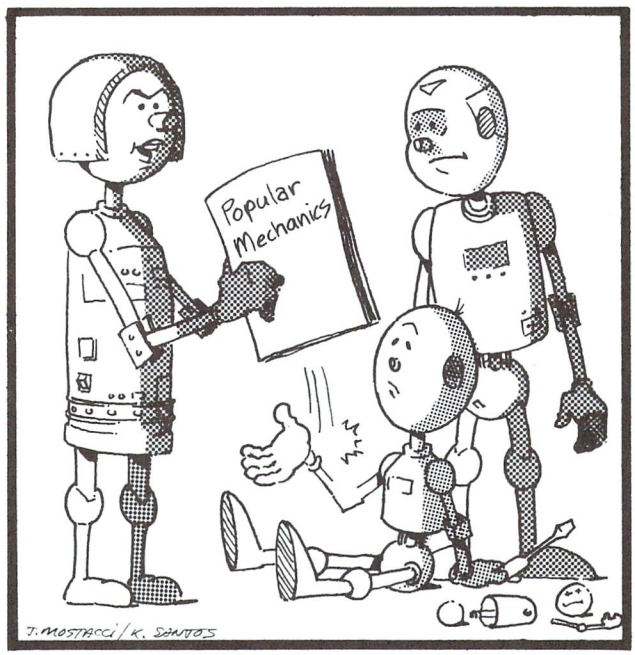
by Timothy Orr Knight

The rapid technological growth that we have experienced has paved the way for so called "Whiz Kids". Timothy Orr Knight, a 17 year old on his way to becoming a millionaire, is one such Whiz Kid.

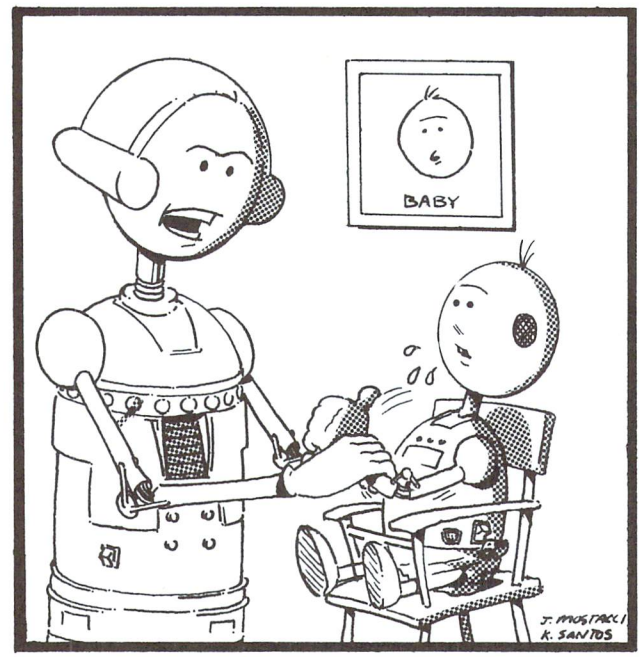
In his book, "Megabucks From Your Microcomputer", Orr Knight shows us how to get the best microcomputer for our money and then how to make money from that microcomputer. The book covers a range of topics, including guidelines for purchasing a micro; getting started in the microcomputer field by writing product reviews, magazine articles, and books; writing your own custom software and marketing it; coping with adults and their negative feelings for young "computer stars"; and an overall view of the different avenues that can be taken in the microcomputer field.

While the book is written by a 17 year old and is geared toward the young computer enthusiast, it makes interesting reading for us "older folks". The advent of the microcomputer has come so quickly that many people are confused. This book is written so that it is easy to understand the computer revolution and how our children are being affected. It made light, enjoyable reading material and should be favoured by people contemplating entering the micro field as opposed to those of us already in it. But who knows? You may just find an idea that catches your eye and makes you a fortune.

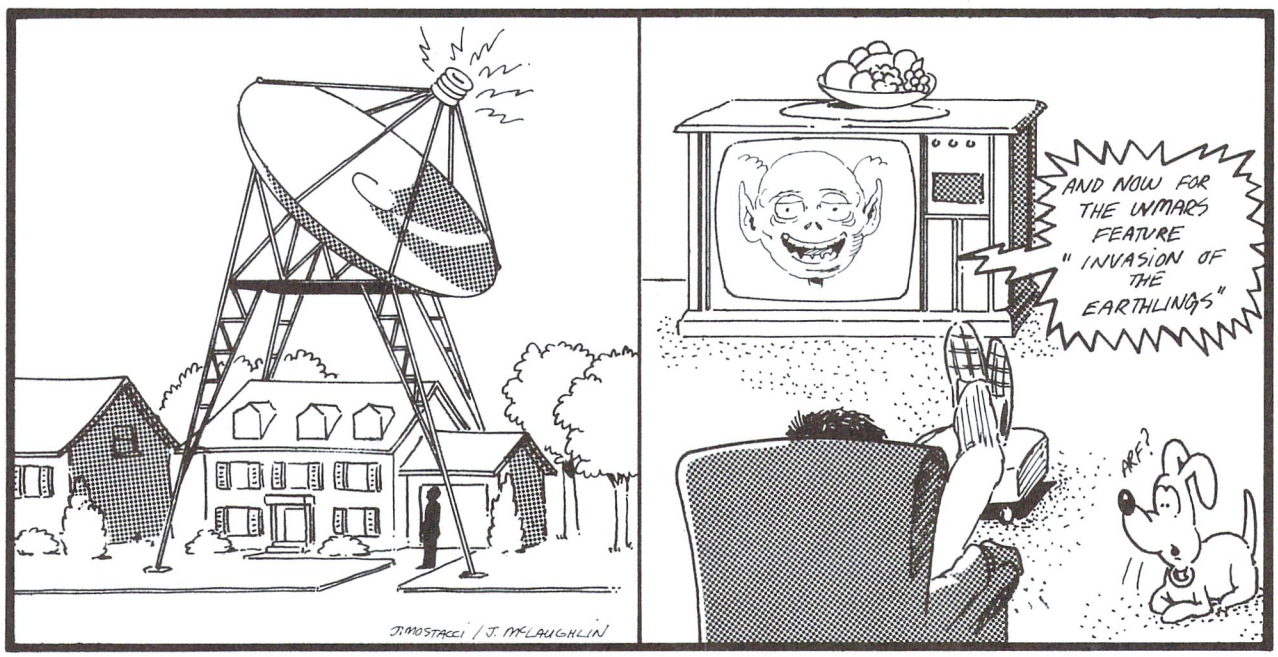
CompuKinks.



Would You Keep Your Pornographic Magazines OUT of HIS Reach!



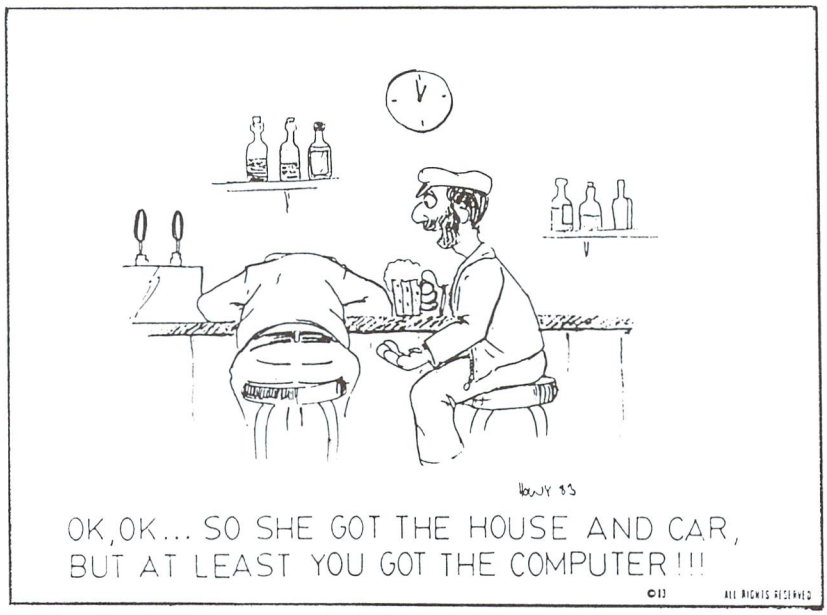
Keep Sucking Your Thumb and You'll Lose A Digit!



There. Maybe Now I'll Get Some Half Decent Stations!

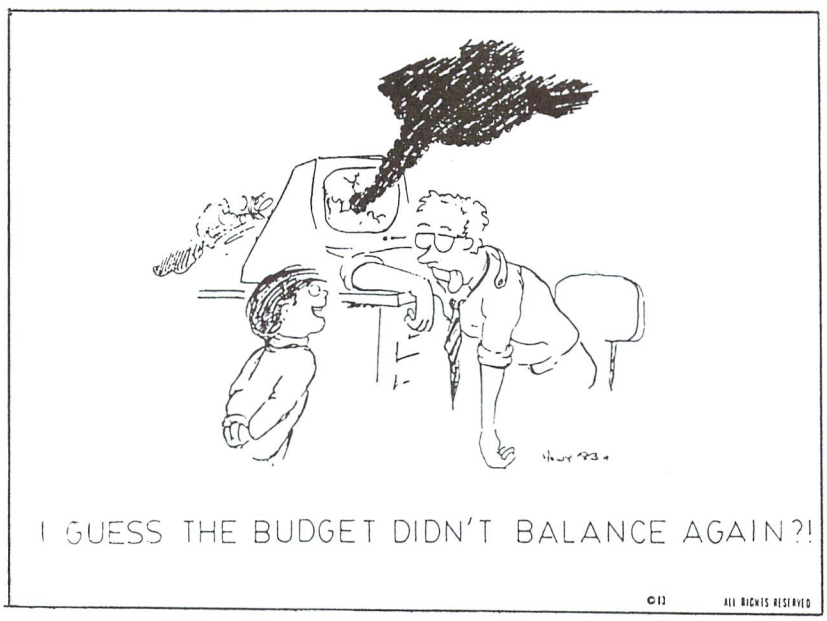
COMPUTE

Mike Panning & Howy Parkins



COMPUTE

Mike Panning & Howy Parkins



There is an old language called Cobol
Whose programs could really snowball.
So when he asked
To write it quite fast
The programmer was aghast at the gall.

The language of the machine
Makes a program very lean
When run, few waits,
But also creates
A programmer very mean.

There once was a chip from Silicon Valley
Which was so fast that it could tally
Some ten thousand digits
From a thousand widgets
And it often had time to dilly and dally.

Well this new chip made such a big hit
That everyone tried to make it fit
On any new board
But oh my Lord
Once in a while it would drop a bit.

– Brad Bjorndahl

The Column

Don Bell
Milton, Ontario

In this issue we present the second part of the Christmas card and gift list application. It should be noted that although it was designed with Christmas in mind, it is equally useful all year round. Part 2 deals with the sorting, searching and indexing aspects of a typical data base. Next issue we'll cover the hard copy output procedures.

The '64 Manager is only one of several data base packages available for the Commodore 64. The applications presented here should be equally applicable to other data bases. Only the procedure for implementing them will change.

In the second half of this column is Eric Armson's "Re-Field" - a modification for The Manager on 8000 series machines. Re-Field lets you change the left-to-right, top-to-bottom field order during Enter/Edit. Great for columnar input.

A '64 MANAGER Gift List – Part 2

THIS COULD BE THE DAY OF JUDGEMENT! No, it's not your soul I'm worried about right now, it's the fate of your computer! Ponder this sad technological fact – 1 out of every 4 home computers suffers a premature, ignominious death, not even given half a chance to prove its self worth. After the rush of Christmas and the thrill of playing video games has worn off, many home computers are abandoned by their owners, banished to the bowels of the basement or closet concentration camps to gather dust along with old hockey games, worn–out clothes, ravaged patch dolls and other unmentionables. The usual excuse is the owner doesn't know what else to do with it.

A lack of knowledge of useful computer applications combined with a few humiliating encounters with BASIC, your computer, software, instruction manual, or all 4 – may provoke the final death sentence. If you are about to pass such a rash judgement on your computer, I implore you, HOLD OFF! At least, give an application on the '64 MANAGER a fair trial run before retiring your Commodore '64 to an indefinite cryogenic slumber.

You may consider this particular application trivial or unnecessary, but look at it this way: it's a safe application to play with – you aren't risking your money or business with it; and secondly, using this application will help you learn how to design your own applications in the future – some of which may prove to be of inestimable value.

This column will focus on: (1) potential benefits to the user of this application; (2) my improved Enter/Edit screen layout – an explanation of the fields and how to make entries; (3) some tricks for entering records quickly; and (4) how search functions can be applied to this application. The next article will use the REPORT GENERATE option to produce a mailing list and other useful reports.

Potential Benefits To The User Of This Application

- fast lookups of names, addresses, birthdays, anniversaries, card, gift and party information using detailed search criteria
- a fast way of printing mailing labels
- a backup copy of an address or calendar book
- a quick way of generating party/gift lists
- a birthday/anniversary reminder list in order by date
- a calendar of next contact (or appointment) dates for family, clients, or customers
- group telephone lists
- use this application as a model to create other related applications such as a customer/client list.

Error In Loading Instruction

Before jumping into the details of this application, I would first of all like to apologize for an error in the previous MANAGER column. I gave the load instruction as LOAD "64.M*",8. The correct command is LOAD "*",8,1 as stated in the MANAGER manual.

MANAGER Diskettes

You will always be using 2 diskettes, the system diskette with the MANAGER program on it and your own data diskette, which contains your own personal screens and files you have created with the MANAGER program. The previous MANAGER column discussed formatting a data diskette.

Beware The RESTORE Key!

Hitting the RESTORE key instead of the RETURN key may force a crude jump from your current entry mode to somewhere else in the program. Watch out for this kangaroo!

Getting A Printout Of Field Lengths And Field Numbers

After you have created your file, The '64 MANAGER has included a neat feature whereby you can easily get printouts of field lengths and field numbers. LATER, YOU WILL NEED THESE PRINTOUTS TO REFER TO WHEN DOING SEARCHES OR DESIGNING REPORTS. To get the printouts: (1) Select Enter/Edit option; (2) To get field lengths – press 'up arrow', then 'p' to print screen; (3) To get field numbers – press Shift 'up arrow', then 'p' to print screen.

Printouts of field numbers and field lengths in my revised screen format are shown below. The dividing line on the screen separates personal information from Christmas information.

```
*** CHRISTMAS LIST '84 ***

LAST NAME:
MAIL NAME:
ADDRESS  :
CITY     :
STATE    :                    M/F:
COUNTRY  :                  S/M/D:
ZIP      :          NEXT CONTACT:
HOME TEL :          BUS:
BIRTHDAY :          ANNIVERSARY :
FAMILY   :
NOTES    :
INTEREST :

** RECEIVED/GAVE LAST YEAR **
REC'D: CARD    PARTY      VISIT/CALL
GIFT :                  $

GAVE : CARD    PARTY      VISIT/CALL
GIFT :                  $
** GIVING THIS YEAR **
CARD :    PARTY INVIT.    VISIT/CALL
GIFT :                  $
REC LEN= 404 CHARS, # OF RECORDS = 10
```

```
*** CHRISTMAS LIST '84 ***

LAST NAME:
MAIL NAME:
ADDRESS  :
CITY     :
STATE    :                    M/F:
COUNTRY  :                  S/M/D:
ZIP      :          NEXT CONTACT:
HOME TEL :          BUS:
BIRTHDAY :          ANNIVERSARY :
FAMILY   :
NOTES    :
INTEREST :

** RECEIVED/GAVE LAST YEAR **
REC'D: CARD    PARTY      VISIT/CALL
GIFT :                  $

GAVE : CARD    PARTY      VISIT/CALL
GIFT :                  $
** GIVING THIS YEAR **
CARD :    PARTY INVIT.    VISIT/CALL
GIFT :                  $
REC LEN= 404 CHARS, # OF RECORDS = 10
```

An Explanation Of Fields
In My Revised Screen Format

(A) indicates an alphanumeric field
(N) indicates a numeric field
(*) zip code is numeric for American, alphanumeric for Canadian

Personal Information

1. LAST NAME (A)
This is necessary for an alphabetical sorting of names. If several people have the same last name, add a space and their first name to the field.

2. MAIL NAME (A)
This is the actual name(s) you want to appear on a mailing list. For example, you may want to group a couple together as Mr. & Mrs. Jones.

3. ADDRESS (A)

4. CITY (A)

5. STATE (A)

6. M/F (A)
Enter 'M' for male, 'F' for female

7. COUNTRY (A)

8. S/M/D (A)
Enter 'S' for single, 'M' for married, 'D' for divorced.

9. ZIP (*)

10. NEXT CONTACT(N)
This must be entered as a 6 digit year-month-day number, e.g. Jan 9 1984 would be 840109

11-12 HOME TEL (N)
Both phone numbers have separate field

13-14 BUS (N)
For the area code. Be sure to leave a space as a separator (not a '-') between numbers e.g. 446 0001

15-16 BIRTHDAY (N)
In both dates the 4 digit month/day numbers are separated from the 2 digit year field. This aids in sorting

17-18 ANNIVERSARY (N)
Birthday or anniversary reminder lists in date order. e.g. Enter Feb 1, 1984 as 84 0201

19. FAMILY (A)
In this field enter names of family members with ages in brackets, e.g. John(4), Alice(6)

20. NOTES (A)
In this field enter notes about your last contact or upcoming contact. This field could also act as a catch-all for more family information or any descriptor you might want to group people under.

21. INTEREST (A)
In this field enter hobbies, clubs or your special affiliation with this person. "Family" might be used to indicate a family affiliation, "business" might be used to indicate a business affiliation.

Christmas Information

RECEIVED FROM THIS PERSON LAST YEAR
22. CARD (A)
A 'Y' entry indicates you received a

23. PARTY (A)
Card or party invitation. A blank entry indicates no card or party invitation received last year.

24. VISIT/CALL (A)
A 'V' means they visited you last year. A 'C' means they phoned you. A blank entry indicates no response.

25. GIFT (A)
A description of any gifts received

26. $ (N)
Approximate value of the gift(s)

Entries for fields 27-36 are similar to those in fields 22-26.

Tricks For Entering Records Quickly

Remember, you don't have to fill in every field!

F3 duplicates the field in the previous record. This is handy for entering similar information in a number of records (e.g. area codes, cities, state, country)

If you want to enter a record that is substantially the same as another record already entered, simply get that record on the screen and then press Shift 'E'. You may now modify the information on the screen and press backarrow to save the record. This trick is handy for entering records for people in

the same family, or at the same address.

Be careful that all dates are 6 digit year–month–day entries.

Searches

When using the search string editor (F5), use 'F' to indicate an alphanumeric field, and 'N' to indicate a numeric field. Here are some useful search functions in the "Enter/Edit" option.

1. To search for a record by a person's last name:
Press 'S', place the cursor in the last name field. Type in the name, press F3, press 'back arrow'. To get the next record of a person with that same name, press space bar.

2. To scan your file:
Press 'S', 'back arrow'. To display the next record press space bar.

3. To search for 1 or more beginning letters in a field:
(e.g. all names beginning with the letter 'B')
Press 'S', place the cursor at the beginning of the correct field. Type in the letter(s) to search for, press F3, press 'back arrow'. To display the next record satisfying the search criteria, press space bar.

4. To search for a first name in the 'mail name' field, perform a character (or position) independent search:
Press 'S', locate the cursor anywhere in the "mail name" field. Press F4(hunt), press 'back arrow'

5. To search for people who sent you a card last year:
Press 'S', place the cursor in the field indicating whether they sent you a card (or gift) last year, enter 'Y', press F3, press 'back arrow'. To view the next record satisfying the search criteria, press space bar.

6. To search for people who gave you a gift last year:
Press 'S', F5(search string editor). Enter F#> ' ' (# is the number of the field indicating you received a gift from them last year). To find the next person who sent you a gift, press space bar.

7. You can search for appointments (next contact dates), birthdays or anniversaries on a certain date, before a certain date or between 2 dates:

Press 'S', F5(to enter the search string editor)

For each of the following, # is the number of the field in which you store the relevant date (next contact, birthday or anniversary date). For this search to work, the field must have been designated as "numeric" when you created the file.

For a specific date ,e.g. May 8, 1984
Enter N# = 840508

Before a specific date, e.g. April 18, 1984.
Enter N#> ' ' and N#<840418
(N#> ' ' excludes records with blank entries for "Next Contact")

Between 2 dates, eg. April 18,1984 and May 8, 1984
Enter N#>840418 and N#<840508

After specifying a date criteria in the search string editor, press 'back arrow'. To view the next record satisfying the search criteria, press space bar.

DON'T PHONE – WRITE!

If you have questions regarding this application or you would like to suggest ideas for future columns, please write me a legible, coherent letter, including a self–addressed return envelope with stamp. Try to make sure the postman gets to it before the dog does. I will attempt to answer letters either directly or in this column. Write to: Don Bell, c/o The Transactor, 500 Steeles Ave., Milton, Ontario, Canada, L9T 3P7.

Re–Field — Eric Armson, Toronto, Ont.

This next program is a modification to The MANAGER on Commodore 8000 series machines. It allows you to re–order the sequence in which fields are edited in the Enter/Edit function. For example, say you have set up a screen with three columns of fields that are perhaps used for dollar figures. The Enter/Edit function starts the cursor at the top left field. The next field to get service will be the top field of the second column, then the third, and back to the first. So in order to deal with the first column only, one must use the cursor up/down key to skip over fields of the other two. After using Re–Field, one can now go from the top of one column to the bottom, then on to the next column. Actually, it will allow you to service fields in any order you wish.

This routine is best used when a new screen is created (ie. before any data is stored). However, you can use it to re–order the field sequence of an existing data base, but you will also need to re–arrange the data in your file. Take the above example. File data that appeared in the top field of each column, will now be sent to the first three fields of the first column. Using the Report Generate program, a new

data file can be produced that reflects the new order of your fields. The 1st, 4th, 7th and 10th fields, etc. would become the 1st, 2nd, 3rd and 4th fields, and so on. Probably the best way to attack this is with two copies of the screen layout on paper – one with your fields numbered the old way and the other numbered with the new order. Depending on the size of your file, you should maybe test it out with, say, the first ten records. Of course you should keep a copy of the previous file, and always make changes to The MANAGER on a new disk.

To set this up, the following changes must be made to the program "create" on the system diskette:

```
672 rem  \s,fs$,"6500","7dff"
675 rem  \x,"0:menu"
676 rem  lines 677 - 690 are extentions for refield option
677 sysba+60,24,00,"[v]       re-arrange field order (y/n) ? "
678 x=peek(198):y=peek(216)
680 x$="":sysba+57,y,x,1:gosub20000:ifx$<>"y"then690
682 z1=1000:pokez1,len(fa$)
683 fori=1topeek(z1):pokei+z1,asc(mid$(fa$,i,1)):next
684 pokei+z1,val(d$)
685 \s,fs$,"6500","7dff":\x,"0:refield"
690 \s,fs$,"6500","7dff":\x,"0:menu"
```

Then replace "create" on a new system disk. Now enter the following program and save it on the same disk using the title "refield". After using Create/Revise, you will be prompted for re-arranging the field order as shown in line 677 above.

```
2 rem        the manager , extentions written by
3 rem                eric armson
4 rem                august 1983
5 rem        g a m e  p l a y e r s  s o f t w a r e
6 :
9 clr:x=fre(0):goto100:  gosub10 for input with length and rf=0
10 ifrfthen60
20 y=peek(216):x=peek(198):rf=1:goto60
30 rem     gosub40 for (y/n) input with length and rf=0
40 gosub10:ifx$<>"y"andx$<>"n"then10
50 f=abs(x$="y"):return
60 x$="":sysin,y,x,l:x$=x$:ifx$="q"thenpoke224,0:\x,"0:menu"
70 ifx$="Q"thenpoke224,0:run
80 poke158,0:return
90 syspr,24,12,"[v]":return
100 print"[S]":poke59468,12:ifpeek(145)=228then:\r
110 ba=18432:sysba:in=ba+57:pr=ba+60:xy=57447:mv=ba+3:z1=1000
120 fori=1to8:tl$=tl$+"**********":next
130 print"[s]"tl$;tab(30)"reorganise fields":printtl$:print
140 goto180
150 syspr,24,12,"any changes on this page (y/n) ? ":l=1:rf=0:gosub40:return
160 lc=a(i,0,fd)*80+a(i,1,fd)+32768:cr=peek(lc):return
180 print"arithmetic programs must be rerun after any reordering,":print
190 print"all fields will be set to a default order.":print
200 fori=1topeek(z1):fa$=fa$+chr$(peek(i+z1)):next:d$=mid$(str$(peek(i+z1)),2)
210 print"do you want to continue (y/n)? ";:poke158,0:l=1:rf=0:gosub40
220 iff=0then:\x,"0:menu"
240 print:print"[qq]loading screen . . ."
250 open1,8,15,"i"+d$:close1:fs$=d$+":"+fa$+".scr":\l;fs$
```

```
280 nf=peek(27633):np=peek(27638):rem t#flds,#pgs
290 poke248,106:poke247,0:poke32767,0:ch=0
300 dim a(np,2,nf),o(nf),f(nf):poke231,3:cm=0:gosub800:print" s ";
310 fori=0tonp:mn(i)=peek(27632+i*2560):mx(i)=peek(27640+i*2560)
320 poke248,106+i*10:sysmv:syspr,24,16," field #          r field order R "
330 fork=mn(i)tomx(i):syspr,24,23,"  ":printk+1;:forj=0to2
340 a(i,j,k)=peek(27136+i*2560+(k-mn(i))*3+j):nextj
350 fd=k:gosub160:pokelc,32:pokelc,160
360 pokelc,32:pokelc,160:print" g ";:pokelc,cr
370 if((k/2)=int(k/2))thensyspr,24,36," * ":goto390
380 syspr,24,36," r * R "
390 nextk:gosub90:poke158,0:gosub150:iff=0thengosub90:gosub720:nexti:goto740
400 k$=" Qq [Lft] ] "+chr$(13)+" qQ s h ":rem possible keys [Lft]=Crsr Left
410 gosub90:ch=ch+1:qx=1:print" sgggggggggg ";:cm=cm+mx(i)-mn(i)+i:fc=mn(i)
420 fd=mn(i):forc=mn(i)tomx(i):o(c)=0:f(c)=0:nextc
425 syspr,24,12," command (cu,cd,cl,cr,return, r q R uit, r h R elp,home) "
430 x=a(i,1,fd):y=a(i,0,fd):gosub160
440 qx=-qx:pokelc,160+((qx>0)*128):geta$:ifa$=" "then440
450 cc=0:fors=1tolen(k$):ifa$=mid$(k$,s,1)then470
460 nexts:goto440
470 pokelc,cr:onsgoto500,520,500,520,540,480,490,760,780
475 rem      onsgoto up,dwn,lft,rgt,ret,mnu,run,rst,hlp
480 \x," 0:menu "
490 poke158,0:run
500 fd=fd-1:iffd<mn(i)thenfd=mx(i)
510 goto570:left  & up
520 fd=fd+1:iffd>mx(i)thenfd=mn(i)
530 goto570:right & down
540 rem * carriage return,(store field's order)
550 iff(fd)<>0then520
560 f(fd)=1:o(fc)=fd:fc=fc+1:goto520
570 iffc>cmthennexti:goto740
580 iff(fd)<>0then470
590 goto430
600 h=24320:bb=32256:e=0:pokeh,0
610 fori=0tonp:poke248,106+i*10:sysmv
620 fork=mn(i)tomx(i):g=bb+(k-mn(i))*3
630 pokeg,a(i,0,o(k)):b=a(i,1,o(k))
640 c=a(i,2,o(k)):pokeg+1,b:pokeg+2,c
650 h=h+1:e=e+1+c-b:pokeh,e
660 nextk:sysba+36:sysba+39:nexti
665 fori=0tonp:poke248,106+i*10:sysmv:sysba+36:sysba+39:nexti
670 d=val(d$):scratch(fa$+" .scr "),d(d)
680 \s,fs$," 6500 "," 7dff "
690 \x," 0:menu "
699 stop
700 rem              **(patch jobs)**
710 rem save normal order
720 fork=mn(i)tomx(i):o(k)=k:nextk:ch=ch+1:return
730 rem check for any changes
740 ifch=0then:\x," 0:menu "
750 goto600
```

```
760 ifithencm = mx(0)-mn(0):ch = 1:goto410
770 cm = 0:ch = 0:goto410
780 \|; " 0:help3 " :syspr,24,12, " r press 'r' to return R "
790 geta$:ifa$ = " " ora$<> " r " then790
795 sysmv:goto425
800 rem *** reset both screen's fields then return
810 sys21408:rem* fill 6700-69ff(ff),5f00-5ff0(ff),6a00-($f7,f8)
820 poke248,106 : sysmv:rem* set screen #1 & move in
830 sysba + 33:rem* reset field info. ,7e00-7eff,5f00-5ff0
840 sysba + 36:rem* move 5f00-5ff0 to 7f00 ,stop if ($ff)
850 sysba + 39:rem* move screen back in place
860 poke248,116 : sysmv:rem* set screen #2 & move in
870 sysba + 33:rem* reset field info. ,7e00-7eff,5f00-5ff0
880 sysba + 36:rem* move 5f00-5ff0 to 7f00 ,stop if ($ff)
890 sysba + 39:return:move screen back in place
```

'64 Manager Updates

Commodore has released 5,000 copies of version 1.04 of The '64 Manager. Be advised that this is not the correct version. For information on how to receive updates, contact Commodore head office:

Commodore Business Machines
1200 Wilson Drive
West Chester, PA
19380
(215) 431 9100

Commodore Business Machines
3370 Pharmacy Avenue
Agincourt, Ontario
M1W 2K4
(416) 499 4292

Updating Programs

Bob Drake
Brantford, Ont.

I find when I'm working on a program that I do one or two things which I shouldn't do. First, I don't save or list the program regularly enough, something goes wrong, the system crashes and I lose all my work. Second, I keep saving new versions of the program under new names and I can't remember which is which. Here is a simple technique to make updating a program simpler.

My programs always start at line 100. Lines 0 to 99 are left for routines I use constantly. Line zero is always a title. Line one is a GOTO 100. Lines 2 through 9 are my updating routines. Let's suppose the program currently under way is called "FISH". The first lines will look like this:

```
0 rem "fish – r.drake (c)1983"
1 goto 100
2 close1:open1,8,15,"s0:fish":save"0:fish",8:end
3 open 1,4:cmd1:print"■S■":list
4 end
```

Line 2 does a replacement save. Commodore equipment has had problems with the SAVE "@0:fish" statement. This is overcome by first scratching the file and then doing a new save. Line 3 lists the program on a printer. To use these routines just type RUN 2 or RUN 3.

These simple routines leave room for trouble. The replacement save doesn't check for disk errors. The printer listing doesn't close the file. You can also become frustrated by accidently running the wrong line and doing something you really didn't want to do. This second routine is longer but covers those problems.

```
0 rem "fish – r.drake (c)1983"
1 goto 100
2 fi$="fish":q$=chr$(34):print"■Sr■update – ";fi$
3 input"replace/list";a$:a$=left$(a$,1):if
a$<>"r"then6
4 open1,8,15,"s0:"+fi$:input#1,a$,b$,c$,d$
  :print c$;" ";b$
5 if a$="01" then save"0:"+fi$,8:input#1,a$,b$
  :print b$:end
6 if a$<>"l" then end
7 print"■Sqqq■open1,4:cmd1:print#1," q$ chr$(147)
  q$ ":list"
8 print"print#1," q$ chr$(19) q$ ":close 1"
9 poke631,13:poke632,13:poke198,2:print
```

To use the routine enter RUN 2. Line 2 assumes you have previously entered your program name as FI$. Line 2 also tells you that you are in the updating routine. Q$ is set up as the quote character in preparation for listing the program. Line 3 asks your choice and takes the first letter of your reply. If you enter 'R' then lines 5 and 6 scratch the old file, check and print the disk status, save the new file and recheck the disk status.

If you entered 'L' then lines 7 and 8 print the proper commands on the screen. Line 9 pokes two carriage returns into the keyboard buffer (locations 631, 632) and tells the computer that there are 2 keystrokes stored there (location 198). The cursor is homed and the program is ended causing the 2 keystrokes to be executed. Your program is listed and the file closed when done.

To use this program on a PET (BASIC 2.0 or 4.0) requires only changing the three pokes. Locations 631, 632 become 623 and 624. Location 198 becomes 158. The routine works as is on a VIC.

Don't type in UPDATE more than once. When you begin to work on a program, load it into your computer, enter the correct file name in lines one and two, and start programming at line 100.

For those feeling lost because they are using tape, not disk, have heart. Use these instead.

The short version becomes:

```
0 rem "fish – r.drake (c)1983"
1 goto 100
2 for i=1 to 3: save"fish":next i:end
3 open 1,4:cmd1:print"■S■":list
```

The long version replaces lines 4 and 5 with:

```
4 gosub 5:for i=1 to 3:save fi$:next:gosub 5:for i=1
  to 3:verify fi$:next:end
5 print"■S■rewind your tape. press a key to
  continue.":poke 198,1:end
```

For the PET, the wait statements become WAIT 158,1

Editor's Note

Bob has written these handy little utilities on a BASIC 2.0 machine. So while they will probably work on a VIC 20, they will need changes for the 64 and BASIC 4.0 machines. Since Bob has explained the function of each POKE and WAIT statement so beautifully, I'll let you look them up in the memory maps published in our Reference Issue.

A Mid Stringers Night Dream

Richard Evers

One of the more forgotten about and useful commands in the Commodore BASIC language is MID$. Though very often not the high point of any conversation, it does have some admirable points often overlooked. I am writing this article for the sole purpose of enlightening everyone who is not completely familiar with the power of MID$.

The actual working concept of MID$ is very similar to that of LEFT$ and RIGHT$, but is more versatile in what it can accomplish. With LEFT$ and RIGHT$, you can produce (length(string)+1) different results, which represents six different results from a string of length five. With MID$, you can produce a total of ((length(string)–1)squared) different results, or 16 different combinations from a string of length five. This is the reason why I referred to MID$ as the more versatile command.

MID$ can take one of two forms. They are as follows:

a$ = "abcde" – variable a$ has a length of five
mid$(a$,a) – where a = 1 through 255
mid$(a$,a,b) – where a = 1 through 255 and b = 0 through 255

In the first case, if variable A was given a value of 2 the result of MID$ would be "bcde". When the second variable B is left out, it is assumed to be the balance of the string in question. The result "bcde" is arrived at by spacing over to the second position within, and returning with a result of the remainder of the string.

In the second case, we could get any portion of the string that could possibly be desired with variations on variables A and B. These variables are useful up to the length of the string in question. Beyond that is redundant.

Below is a chart that I have prepared to help show you all of the possible combinations for a string of length five. As explained earlier, 16 different combinations can be achieved from a string of length five. This chart will show you how this has been arrived at.

mid$("abcde",a,b)

a=1	b=5	result =	"abcde"
	b=4		"abcd"
	b=3		"abc"
	b=2		"ab"
	b=1		"a"
	b=0		(null)
a=2	b=5	result =	"bcde"
	b=4		"bcde"
	b=3		"bcd"
	b=2		"bc"
	b=1		"b"
	b=0		(null)
a=3	b=5	result =	"cde"
	b=4		"cde"
	b=3		"cde"
	b=2		"cd"
	b=1		"c"
	b=0		(null)
a=4	b=5	result =	"de"
	b=4		"de"
	b=3		"de"
	b=2		"de"
	b=1		"d"
	b=0		(null)
a=5	b=5	result =	"e"
	b=4		"e"
	b=3		"e"
	b=2		"e"
	b=1		"e"
	b=0		(null)

From the chart I am sure you have been able to formulate in your mind how powerful MID$ can be. With some imagination, and a set course to follow, any number of useful or entertaining results can be obtained with variations of MID$. Each variation can be achieved by combining com-

mands from the Commodore BASIC instruction set, and MID$. These combinations are limited only by your imagination.

To follow are five routines that I have written for the occasion. When run, this program will take you through four useful subroutines, and one rather useless but interesting one. The useless one was included to break up the monotony, and also show you how fascinating screen effects can be achieved with a little imagination.

Please enjoy my program, and write to us if ever you find more thrilling things that can be accomplished with any of the commands. We would like to continue supplying you with the latest and the greatest in programming conceptions, a feat that can only be accomplished with your continued support. Keep us in mind the next time a violent programming brain storm assaults you, one in which you have the sadist urge to pass on to others. Thousands of eager programming masochists are waiting throughout the world for a chance to become enlightened. You could be the key to their enlightenment.

```
10 rem * show various uses for mid$ *
20 forloop = 1to5:onloopgosub50,110,180,240,320:next:end

40 rem * mid$ used to condense the day given by user to a three digit string *
50 ad$ = " montuewedthufrisatsun " :printchr$(147)
60 input " please specify the day " ;d$
70 fora = 1tolen(ad$)step3:td$ = mid$(ad$,a,3):ifmid$(d$,1,3) = tdthen90:rem * ok *
80 next:printchr$(147) " the day given is not accurate " :goto60
90 print " the day chosen  " d$ "  is listed as  " td$:fordelay = 1to750:next:return

100 rem * mid$ used to take month given in range 1-12 and match to actual month
110 printchr$(147)
120 mt$ = " janfebmaraprmayjunjulaugsepoctnovdec  "
140 input " please enter the month in range 1-12 / mm " ;mm
150 ifmm>12thenprintchr$(147) " please enter within range " :goto140
160 am$ = mid$(mt$,3*mm-2,3):printam$:fordelay = 1to750:next:return

170 rem * and what is a totally useless but interesting use *
180 printchr$(147):u$ = chr$(145):rem * cursor up *
190 print " enter an interesting string " :inputs$:s = len(s$)
200 fora = 1tos:printtab(a)mid$(s$,1,a)tab(s-a)mid$(s$,1,s-a):printu$,:next
210 fora = sto1step-1:printtab(a)mid$(s$,1,a)tab(s-a)mid$(s$,1,s-a):printu$,:next
220 fordelay = 1to750:next:return

230 rem * a method to get rid of the leading space supplied by str$ *
240 printchr$(147) " please enter a number " :inputn:n$ = str$(n)
250 print " the string value is best shown in reverse "
260 print " that value is  " chr$(18)n$:n = len(n$):f$ = mid$(n$,2)
270 print " this string is non-useable for most file applications "
280 print " to get rid of the leading space, use fs$ = mid$(n$,2) "
290 print " f$ will hold the value of  " chr$(18)fs$
300 print " this amounts to a usable commodity !! " :fordelay = 1to750:next:return

310 rem * an interesting concept for a password system *
320 printchr$(147):gosub380:rem * get a string of all the passwords *
330 input " enter password " ;pw$:iflen(pw$)>4thenprintchr$(147):goto330
340 fora = 1tolen(p$)step4:ifpw$ = mid$(p$,a,4)then360:rem * ok *
350 next:printchr$(147):goto330
360 printchr$(147) " your password is ok  " pw$:fordelay = 1to750:next:return
370 rem * set up the password variable p$ sub-routine *
380 p$ = " johnkarlrichbillbarbkriskely " :return
390 rem * each password is 4 in length (ie.john karl rich bill barb etc.) *
```

The INPUT Glitch

Bob Drake
Brantford, Ont.

Recently while writing a program I ran into a problem using INPUTs on the VIC and C64. Turn on your 64 or VIC and I'll show you what I found.

Enter a line like my line 10. Make sure the input prompt (the part in quotes) leaves the question mark on the very end of the line or on the next line.

```
10 input "please enter a number between one and one thousand";a
```

Now enter a value and press return. Did you get caught in a never ending ?REDO FROM START error loop? I did. Press RUN/STOP and RESTORE to regain control over the computer.

Here's another one. Change line 10 to accept a string variable. Add line 20.

```
10 input "please enter a number between one and one thousand";a$
20 print "a$=";a$
```

Now RUN the program and enter a word. If you typed "test" you should get

```
a$=please enter a number between one and
one thousand test
```

And you thought you had just entered "test".

These two problems are both caused and cured the same way. Both are more of a pain on the VIC because the screen line is so much shorter (22 columns versus 40). If the input prompt is so long that the question mark is at the end of the line or on the next line, then you have the problem. The cure? Keep the input line shorter than one line long. How? Use the technique that is used with some older dialects of BASIC. PRINT your prompt and then use the INPUT.

Change the programs like this:

```
10 print "please enter a number between one and"
20 input "one thousand";a
```

The prompt now leaves the question mark well before the end of the line. No more problem with either strings or numbers. One word of warning, don't use a semicolon or comma with the print. Your computer will again think you have a prompt which is too long.

Here's another one to try. Enter this (use a shorter tab for the VIC):

```
10 input "enter a word";a$
20 print tab(30) a$
30 input "enter a word";a$
40 print tab(30) a$
```

RUN the program. Type something for the first word. For the second word just hit return. If you typed "test" you should get:

```
ENTER A WORD? TEST
TEST
ENTER A WORD?
TEST
```

Although you made a null input, the computer remembers the last value you had for the string. This can be good if you want to make multiple entries of the same value. It can be a pain, if you assumed as I did that A$ would be empty on the next input. Exactly the same situation holds for numbers, previous values are used when you have a null input. How do you beat this one? Do the same thing that is done on mainframes. Don't assume a variable has a value unless you have assigned it. Changing the program like this will do the job.

```
10 input "enter a word";a$
20 print tab(30) a$
25 a$=""
30 input "enter a word";a$
40 print tab(30) a$
```

Now A$ is empty going into the second input.

Next time weird things are happening with your programs, remember that your INPUTs may be causing the problem.

Subroutine Eliminators

Jeff Goebel
Georgetown, Ont.

Hello and welcome to a new concept in computer articles. This is the first in a series dedicated entirely to eliminating entire subroutines. It delves into the many existing but vastly un-known features of the Commodore PET series. Although these articles have been written for the PET 4032 computer specifically, many of them are applicable to other Commodore machines. In the future, I will be releasing similar articles for the Commodore 64, and VIC20.

If this article sounds interesting to you, and you like the idea of typing in one line instead of a twenty line subroutine, then you will be even happier when you discover that this month, in my premier article for the TRANSACTOR, I have included two deadly POKES, both which replace the most commonly written about problem with Commodore machines. I have the feeling I may lose a few readers with this, but if you bear with me, I assure you, it's well worth it. This month's article is another substitute for the infamous Commodore input statement.

Yes, I'm sure that you have read at least 20 or 30 articles about alternatives to the dreaded input. Each longer and more useless than the last. Each, stating "THIS ONE IS THE BEST." Well I don't state that. I think it is, but I'll let you decide for yourself. Poke 16,1. There. That's it. Did you miss it? Poke 16,1. There it is again. Done. Now your Commodore input will react as you have dreamed. Nothing significant has changed, but try hitting a return on a null entry. You can't. Short and to the point. It eliminates lines of text. After INPUTing, do a PRINT and POKE16,0 to return to normal. TA DA!

For those of you who still prefer the GET subroutines because you need commas or colons in your input, I have another equally short but just as powerful POKE. It too destroys the need for long drawn out subroutines which need to create a cursor (something which the GET statement requires to prompt the user to type on the keyboard). POKE 167,0 solves it. Simply put, it turns on the computer's cursor when normally it would be off thus creating a needed prompt for the GET statement. It reacts exactly the same as a regular cursor because it is a real cursor. When the GET routine is complete, turn it off with POKE 167,1. If you forget, it'll stay on and you'll have a cursor popping up everywhere as you print to the screen.

How much would you expect to pay? But wait. . . You also get, at no extra charge, a powerful memory saving method of keeping your screen "awake" while your computer is off doing a function. For example, you are sorting 1000 names with a three line bubble sort. After waiting ten minutes, you start to get suspicious that maybe your computer crashed. After fifteen minutes you are starting to sweat because your friend is watching you and you don't want to look like a fool sitting in front of your computer if you aren't sure it's working. Suddenly, it a fit of madness, you hit the break key and are greeted with BREAK IN 60. It wasn't crashed, but now you have to start again. Wouldn't it have been easier to have the computer screen flashing that "YES, I'm sorting." Obviously, this sort of a subroutine would slow down the procedure even more if it had to run through that loop everytime right? WRONG! Enter POKE 167,0. Bingo! A flashing box that takes no loop routines and no extra memory.

But you say, YEECH! I don't want a cursor on the screen. People will think it's an input prompt! I say, disguise the cursor behind a graphic symbol. Remember, it is a regular cursor, if you place it on top of a symbol, that symbol will flash ON and OFF between reverse field and regular field. With some symbols, this is quite attractive. Experiment, but be sure to shut if off when your sort is finished. (I also use it when outputting to the printer, just to make my screen pretty,)

Well, that sums up this months blitz. With two POKEs, I have managed to create hundreds of obsolete magazine articles across the nation. I would love to hear from anyone about famous, or not so famous "one liners" on any of the Commodore machines. As it happens, I have aquired most of mine from back issues of the original TRANSACTORS or COMPUTES, and a few from friends. But many of the best things are never published widely, and some, that are never published at all, get used within a program with no explanations given. I felt it was a shame, so I did something about it. This article is that something.

Three GET Subroutines

Richard Evers

The GET statement is possibly one of the most useful statements in the entire Commodore lingo. With GET you can pick and choose what comes in from the keyboard. If you don't want the user to do rude things with assorted keys, you can block it with usually one line of code. If you want to limit the length of the string that you receive, that is also very easy by checking the total length of the string that you have been concatenating. And unlike the INPUT statement, GET will not crash with a single carriage return.

Below are three different GET subroutines for your evaluation. Each subroutine is fully capable of doing what is expected of it, but each differs in the manner in which it is accomplished.

The first type is a very simple, three line GET subroutine. From within your normal program flow, you GOSUB to this address to get a character from the keyboard returned in string variable A$. From that point you should check if the string is within your needs, and if so print it or whatever you feel like doing with it.

The second type is a little bit more complex. When a key is pressed by the user, the choice is immediately shown on the screen, with the cursor flashing on top of it. This is accomplished by first printing the variable, then printing a (cursor left) after it. Until the user hits the return key, this routine will keep on going. When returned, the final answer given is held in string variable B$.

The third and final type is my personal favourite. There is no flashing cursor on this one at all. The contents below where the cursor would be is immediately OR'd with 128 to achieve the on cursor look. At that point, like all other GET routines, it just sits there waiting for a reply. When the reply is given, the position that was OR'd with 128 earlier, has that 128 subtracted from it to get rid of the reverse block.

The reply from the keyboard is held in string variable A$ at that point. It is immediately converted over to an ASCII value to help with the checking procedure for acceptable entries, and also the check to see if anything special has to be done at that point. If the key pressed is a carriage return, and the string B$ has a length to it, then the subroutine returns with B$ holding the total entry with a maximum length of ML, a variable that you set up before entering the subroutine.

If the key pressed was a (delete), a sub-section is branched to that will delete the entry before last, and re-display the entire string. From there it branches back up to the get a character routine again.

Please notice line number 5006 in the listing. It is there to stop any rotten keys from being accepted. The rotten keys in mention are all of the (cursor) keys, (home), (clear) and every other lethal key that could possibly be used for evil things like making a mess out of the screen. If A$ has passed that test, it is added to string B$, printed to the screen and checked to see if it is within the maximum length. If so it branches back up to get another character.

The 'OR with 128, subtract 128' technique can be used to help with any keyboard entry system that you conceive, one in which you don't want a flashing cursor dashing around leaving behind tracers.

For your own programs, the INPUT statement is probably all you need. Since you're familiar with their operation, it's unlikely you'll need to protect against yourself. Even INPUT can be foolproofed to some extent (see Subroutine Eliminators page 37). But if you're expecting trouble, a GET routine is what you need. Most programs will require their own individual versions – there is no such thing as a universal get routine. Hopefully one of these will GET you started with a subroutine that you can modify until it fits just right.

```
0 rem * normal get subroutine *
1 rem **********************
2 rem * 4.0 basic - cr = 167
3 rem * vic + c64 - cr = 204
4 rem **********************
5 rem
5000 poke cr,0
5001 geta$:ifa$ = " " then5001
5002 poke cr,1:return
```

```
0 rem * expanded get subroutine *
1 rem ***********************
2 rem * 4.0 basic - cr = 167
3 rem * vic + c64 - cr = 204
4 rem ***********************
5 rem
6 rem * this subroutine will return with single character reply in b$
5000 b$ = " "
5001 poke cr,0
5002 geta$:ifa$ = " " then5002
5003 poke cr,1:if a$ = chr$(13) and len(b$) then return
5004 printa$;chr$(157);:b$ = a$:goto5001
```

```
0 rem * special get subroutine *
1 rem ***********************
2 rem * 80 col  - mp = 80
3 rem * 4.0 basic - ss = 32768:ya = 216:xa = 198
4 rem * 40 col  - mp = 40
5 rem * vic + c64 - ss = 1024 :ya = 214:xa = 211
6 rem * 22 col  - mp = 22
7 rem * set ml = maximum length of string before entering this subroutine *
8 rem ***********************
9 rem
5000 b$ = " "
5001 a = ss + peek(xa) + peek(ya)*mp:poke a,peek(a) or 128
5002 geta$:ifa$ = " " then5002
5003 poke a,peek(a) - 128:b = asc(a$)
5004 ifb = 13 and len(b$) then return
5005 ifb = 20 then5009:rem * delete *
5006 ifb<32 or b>128 and b<160 then 5001
5007 b$ = b$ + a$:printa$;:iflen(b$)< ml then5001
5008 return
5009 b = len(b$):ifbthenb$ = mid$(b$,1,b-1)
5010 print:printchr$(145)chr$(22);b$;:goto5001
```

Master Menu Driver Type 1

Richard Evers

In this issue you will find two master menu programs to choose from. Each use completely different methods to achieve the same result. What is to follow is an explanation of the operating characteristics of the first type, a version that achieves its goal without any special effects, but as such proceeds in a very business-like manner.

While writing this menu, I have tried to keep future modifications in mind as much as possible. Due to this fact, five separate subroutines and one final sub-section have been incorporated together in such a way that will help stop major surgery on the program when changes are to be made. You will notice that the flow is also very understandable, something that you will appreciate later when it comes to adding new subroutines. When everything has been considered, you will find this to be a very friendly and efficient routine to use.

Quick Explanation Of Variables And Subroutines:

EP = pointer to the end of basic program and the start of variables
AE = pointer to the actual end of basic program in memory
TT = tab position for first display line
TS = tab position for second display line
TL = left hand tab for choices on menu
TR = right hand tab for choices on menu
CR = cursor location within memory
NK = counter in zero page of number of characters in keyboard buffer

print chr$(14) or poke53272,23 – set the display into text mode

GOSUB 55 = set up the variables sub-section
GOSUB 145 = display the screen for the audience
GOSUB 185 = get a choice from the keyboard
GOSUB 210 = check if choice is accurate, if so return with re-evaluated string
GOSUB 230 = check if program exists, and load in if found
GOTO 245 = error routine if program not found

Please take note of variables EP and AE. They are very pertinent variables and have to be the first ones set up in this routine. Substitute actual values for these variables on the first executed line of the program, and every program that you intend to boot in. The reason for this line is to stop what I call the 'Microsoft Munch', a rude condition brought on by large loaded programs and small older variables. A very bloody clash occurs in which the little guy always wins, destroying all chances for your recently loaded in program to survive. What you will find is that your new, larger program will be amputated at about the same spot where your old program ended. Strange line numbers and odd strings of characters will mark that spot, a condition so frightening that many programmers minds have literally self-destructed upon occurence, never to regain total sanity. Please avoid all troubles and do as requested. 'Microsoft Munch' is a disease, but one that can be cured through knowledge.

A very strange technique has been used in setting up the arrays for display, with the programs to be loaded being the first in the array. When a program has been chosen by the user, the choice is first broken down to an ASCII value, with 65 subtracted from it thereafter, then the actual string is checked to see if it is within range. If it is, the subroutine returns to where it came from to continue on. If the choice was wrong, a loop occurs in which the only method out is by answering correctly.

After the choice has been verified, the program goes to the checking and loading in routine. When 65 was subtracted from the ASCII value of the choice given, its value dropped down to something in the range of 0 through 6. When the arrays of program names were originally set up, their corresponding numbers were also in the range of 0 through 6. This number, used in conjunction with the string B$, is used to first check if the program exists, then load it in if it does. By using this technique, quite a bit of room has been saved, with future updates being possible with no modifications necessary to this area.

If the program does not exist, a special sub-section is called to let the user know. The only time that the program will not be found is if they are purposely trying to foul your program up, without succeeding I might add.

With my explanation finally complete, the program is before you for your evaluation. Please don't hesitate to modify it until you have precisely what you require. What I have written is actually just a skeleton, one that only you can expand on with your thoughts.

```
0  rem universal master menu type one
5  poke ep,peek(ae):poke ep + 1,peek(ae + 1):clr:rem * first evaluated line *
10 rem ****************************
15 rem * 80 col        - tt = 28:ts = 22:tl = 17:tr = 46
20 rem * 4.0 basic     - ep = 42:ae = 201:cr = 167:nk = 158:printchr$(14)
25 rem * 40 col        - tt = 7:ts = 0:tl = 0:tr = 20
30 rem * c64           - ep = 45:ae = 174:poke53272,23:cr = 204:nk = 198
35 rem ****************************
40 clr : rem * a good line to place the variables *
45 gosub55:gosub145:gosub185:gosub210:gosub230:goto245
50 rem *** variable set-up subroutine ***
55 dimb$(21)
60 b$(0) = " prg1 " :b$(7)  = " (a) Program One "
65 b$(1) = " prg2 " :b$(8)  = " (b) Program Two "
70 b$(2) = " prg3 " :b$(9)  = " (c) Program Three "
75 b$(3) = " prg4 " :b$(10) = " (d) Program Four "
80 b$(4) = " prg5 " :b$(11) = " (e) Program Five "
85 b$(5) = " prg6 " :b$(12) = " (f) Program Six "
90 b$(6) = " prg7 " :b$(13) = " (g) Program Seven "
95 b$(14) = " (q) Quit Program "
100 b$(15) = " Master Menu Version 1.0 "
105 b$(16) = " Please Choose The Program Required "
110 b$(17) = " Your  Choice  (*) " + chr$(157) + chr$(157):rem (cursor left)*2
115 b$(18) = chr$(19) + chr$(19) + chr$(147):rem (home)x2 + (clear)
120 b$(19) = " The Program Chosen Has Not Been Located "
125 b$(20) = " Please Ensure That Your Program Disk Is "
130 b$(21) = " In Drive Zero. Press Any Key When Ready "
135 return
140 rem *** screen display subroutine ***
145 printb$(18):print:printtab(tt)b$(15)
150 print:printtab(ts)b$(16):print
155 print:printtab(tl)b$(7)tab(tr)b$(8)
160 print:printtab(tl)b$(9)tab(tr)b$(10)
165 print:printtab(tl)b$(11)tab(tr)b$(12)
170 print:printtab(tl)b$(13)tab(tr)b$(14)
175 print:printtab(tl)b$(17);:return
180 rem *** get the choice subroutine ***
185 poke cr, 0
190 getc$:ifc$ = " " then190
195 poke cr, 1:return
200 rem *** check choice and assign variable subroutine ***
205 gosub185
210 ifc$ = " q " then end
215 c = asc(c$)-65:ifc$< " a "  or c$> " g "  then205:rem * wrong choice *
220 return
225 rem *** check if program exists and load in program ***
230 open5,8,5, " 0: "  + b$(c) + " " :get#5,c$:ifst<>0thenclose5:return
235 close5:load " 0: "  + b$(c) + " ",8
240 rem *** program does not exist sub-section ***
245 printb$(18);b$(19):print:printb$(20):printb$(21)
250 poke nk,0:wait nk,1:poke nk,0:goto40
```

Master Menu Driver Type 2

Richard Evers

Of the two master menus that you will find in this issue, this one is by far the most interesting to try. The technique used in deciphering the chosen program and the routine used to load that choice will make this a favourite of many people.

To start with, the deciphering process to get a choice from the keyboard, and then figure out what to do with it minimizes IF statements. The variable B$ is given the value of every choice that can be made from the keyboard, ie. B$ = "ABCDEGHIJQ". From there, the program branches to the sub-section that gets the choice and figures out if the choice is legitimate. If the choice is good, the variable B will hold the number associated with the array of the program chosen. With everything complete in this section, the program returns to finally execute the actual loading procedure.

The loading procedure is really the best part. The choice is checked out to see if the user wants to quit. If not, the balance of the program is executed. The screen is cleared, a (cursor down) is printed, followed by the phrase LOAD, a quotation mark, the program name to be loaded in (arrived at by the variable B), another quotation mark and the phrase ",8". Four consecutive PRINT statements are printed to the screen, with the phrase RUN being printed thereafter. To finish the display routine, a cursor home is printed, to position the cursor at the top of the screen.

From this point on the concept gets a little sticky, but with perseverance and luck on our side, we will be able to pull through.

The variable NK is the location in memory that keeps track of how many characters are currently held in the keyboard buffer, with the location of the keyboard buffer itself going under the variable name KB. The value of 4 is POKEd into variable NK to signify four characters in the keyboard buffer, after which the process of putting those four characters into the buffer begins.

First, a cursor home or CHR$(19) is placed in the start of the buffer. This will ensure that the cursor is positioned at the top of the screen. Second, third and final, the next three positions of the buffer are filled with carriage returns. When that is complete the program ends, and the characters we have POKEd into the buffer will be treated by the computer as if they were typed on the keyboard.

——— top of screen ———

load "prg# ",8 + chr$(13)	; 'load etc' by program, chr$(13) by buffer
searching for prg#	; printed by operating system while trying to load
loading	; again, printed when program found
ready.	; printed by system after program has been loaded
(cursor) + chr$(13)	; the cursor returns, and a chr$(13) printed by buffer
run + chr$(13)	; 'run' by earlier program, and chr$(13) by buffer

As I have attempted to show above, the loading procedure goes pretty much the same as manual mode. The keyboard buffer now starts to do its job, with the cursor being placed on the second line from the top of the screen, or the line with "LOAD "prg# ",8" printed on it. It has landed on this line due to the lack of a semi-colon after (home) was printed. To activate the loading procedure, the keyboard buffer now prints out another character, a CHR$(13) this time. This carriage return, due to the fact that the computer is really in direct mode, will activate the LOAD statement and the program will be loaded into memory.

When the loading in procedure has been completed, the buffer continues on its merry way again, this time with two more carriage returns. The first will drop the cursor to the line with RUN printed on it, and the second will activate it. With that complete, the program loaded in has been put into play, happily running away under its own power.

One point in favour of using this technique is to help avoid 'Microsoft Munch'; the problem that occurs when a small program loads in a larger one. Normally, this would cause

huge headaches due to the fact that the new program and the old variables bump into each other. This problem is totally alleviated due to the fact that the computer is essentially in direct mode while everything takes place, and is therefore exempt from these problems.

With all that said, my verbose rundown has come to its merciful end. Please modify the routine to your hearts content. That is the purpose of this issue and I hope that you take advantage of it.

```
10 rem universal master menu type two
15 a$(1) = " prg1 " :a$(2) = " prg2 " :a$(3) = " prg3 " :a$(4) = " prg4 " :a$(5) = " prg5 "
20 a$(6) = " prg6 " :a$(7) = " prg7 " :a$(8) = " prg8 " :a$(9) = " prg9 " :a$(10) = " prg10 "
25 rem ********************************
30 rem * 80 col    - tl = 23:tf = 11:ts = 40
35 rem * 4.0 basic - printchr$(14):nk = 158:kb = 623:cr = 167
40 rem * 40 col    - tl = 6:tf = 0:ts = 20
45 rem * c64 -       poke53272,23:nk = 198:kb = 631:cr = 204
50 rem ********************************
55 rem
60 printchr$(147):printtab(tl) " Master Menu Version 2.0 " :print:print
65 printtab(tf) " (a) Program One   " tab(ts) " (b) Program Two "   :print
70 printtab(tf) " (c) Program Three " tab(ts) " (d) Program Four "  :print
75 printtab(tf) " (e) Program Five  " tab(ts) " (f) Program Six "   :print
80 printtab(tf) " (g) Program Seven " tab(ts) " (h) Program Eight " :print
85 printtab(tf) " (i) Program Nine  " tab(ts) " (j) Program Ten "   :print
90 printtab(tf) " (q) Quit          " tab(ts) " (*) " chr$(157)chr$(157);
95 b$ = " abcdefghijq " :gosub115:ifc$ = " q " thenend
100 printchr$(147)chr$(17) " load " chr$(34)a$(b)chr$(34) " ,8 "
105 print:print:print:print:print " run " chr$(19)
110 poke nk,4:pokekb,19:forb = 1to3:pokekb + b,13:next:end
115 gosub130:forb = 1tolen(b$):ifc$<>mid$(b$,b,1)thennextb:goto115
120 return
125 rem *** get a character routine ***
130 poke cr,0
135 get c$ : if c$ = " " then135
140 poke cr,1 : return
```

Sub Section Menu Driver

Richard Evers

In this issue I have shown you two ways in which to produce a master menu for your programs. Each one of these have been written to boot in other programs of the users choice. What is now to follow is the skeletal framework of the program that the others were booting. The concept is very simple to grasp, and has been written with modifications for the C64, 4000, 8000 and 9000 series.

Something that you will notice about the program is that there is no massive 'IF' table located anywhere to determine what to do with the users choice. What normally would take a table of 7 lines, or more, has been reduced to 2 lines in total, including the check for inaccurate choice. Once a choice has been deciphered, an ON.GOTO statement transfers execution to some other section of your program. The variable here is simply the ASCII value of the key typed by the user, minus 64. The result is a number from 1 to N, where N is the number of sub-sections in your program.

There are only seven variables used in total in this program, of which the purpose of each will be found in the following list.

TT = The tab position of the top display line
TN = The tab position of the 2nd display line
TF = The tab position of the left most of the choice display
TS = The tab position of the right most of the choice display
CR = Cursor Location

```
10 rem a menu for all occasions
15 clr:cl$ = " ":fora = 1to14:cl$ = cl$ + chr$(157):next:rem cursor lefts
20 rem ************************************
25 rem * 80 col - tt = 28:tn = 21:tf = 12:ts = 42
30 rem * 4.0 basic - cr = 167:printchr$(14)
35 rem * 40 col - tt = 9:tn = 1:tf = 0:ts = 20
40 rem * c64      - cr = 204:poke53272,23
45 rem ************************************
50 rem
55 printchr$(147):poke ca, vc:print
60 printtab(tt) " My Menu - Version One " :print
65 printtab(tn) " Please choose the section required " :print:print
70 printtab(tf) " (a) Section One " tab(ts) " (b) Section Two " :print
75 printtab(tf) " (c) Section Three " tab(ts) " (d) Section Four " :print
80 printtab(tf) " (e) Section Five " tab(ts) " (f) Section Six " :print
85 printtab(tf) " (x) End Session " tab(ts) " (*) Your Choice " cl$;
90 rem *** get choice and act on it section ***
95 gosub185:ifa$ = " x " then115:rem * end session routine *
100 a = asc(a$)-64:if a<1 or a>6 then95
105 on a goto 150,155,160,165,170,175
110 rem *** end session sub-section ***
115 printchr$(147):print:printtab(3) " End Session "
120 printtab(2) " Are You Sure ? "
125 print " (y) Yes or (n) No "
130 gosub185:ifa$ = " y " thenend
135 ifa$ = " n " then15
140 goto130
145 rem *** start of the main routines ***
150 printchr$(147) " Section One "  :stop
155 printchr$(147) " Section Two "  :stop
160 printchr$(147) " Section Three " :stop
165 printchr$(147) " Section Four "  :stop
170 printchr$(147) " Section Five "  :stop
175 printchr$(147) " Section Six "  :stop
180 rem *** get a character sub-routine ***
185 poke cr ,0
190 geta$:ifa$ = " " then190
195 poke cr ,1:return
```

Directory Subroutines

Richard Evers

Though never a problem for users of 4.0 equipment, it is a totally different matter for the users of the VIC and C64 computers. Directories can be achieved, but they have to be loaded into memory, thus destroying everything in their path. What is to follow is two methods of achieving this goal while in program mode. You will find each method with redeeming qualities, best suited for applications only you can determine.

The first routine prints only the filenames in columnar format on the screen. The VIC 20 is exempt from this printing format due to lack of room. Quotation marks, size of files and all that other interesting programming information is purposely left out. What is left is actually quite pleasing to see, a totally uncluttered compilation of files held on disk.

The second routine, written by Paul Higginbottom, gives an exact duplicate of what would be achieved by a 4.0 directory of the disk. Everything of interest to the programmer is printed on the screen, including the file sizes, file types and blocks left free.

As specified earlier, the choice of routine is best determined by the type of application that you are writing for. For business and general applications, I feel that the first type is the most useful. Non computer oriented people are best not exposed to information that would only confuse them while performing their job functions. The second type is more suited to a programmers atmosphere.

Both routines are display oriented. That is, the information held in the disk directory file is merely displayed. For those of you writing programs that deal with disk on a more intimate level (eg. monitors or file handlers), you may need more details from the directory file than either of these routines is set up to deliver. If that's the case, a program called "Catstrapolator" appears in Volume 4, Issue 03 of The Transactor that breaks down the directory data into more accessible information for these types of programs.

```
100 rem *** directory - type one ***
105 rem * 80 col - dt=4
110 rem * 40 col - dt=2
115 rem * vic20  - dt=1
120 input" Please Specify Drive Number For Directory ";a$:a=0:b=0
125 ifa$<"0" or a$>"1"then120
130 open1,8,0,"$"+a$+"":get#1,a$:get#1,a$
135 ifa=(dt) then a=0:print
140 printtab(20*a);:forc=1to4:get#1,a$:next:ifst<>0thenclose5:return
145 get#1,a$:ifa$=""thena=a+1:goto135
150 ifa$=chr$(34)thenb=notb:goto145
155 ifbthenprinta$;
160 goto145
```

```
100 rem *** director type two ***
105 rem * 4000/8000/9000 - kb=158
110 rem * vic20 + c64   - kb=198
115 input" Please Specify Drive Number For Directory ";dr
120 ifdr<0 or dr>1 then115
125 n$=chr$(0):h=256:poke(kb),0:open1,8,0,"$"+mid$(str$(dr),2)
130 get#1,a$,a$
135 get#1,a$,a$,a$,a1$:ifstthen160
140 printasc(a$+n$)+asc(a1$+n$)*h;
145 ifpeek(kb)then160:rem * if key pressed / abort *
150 get#1,a$:ifa$<>""thenprinta$;:goto150
155 print:goto135
160 close1:poke(kb),0:return
```

Two Universal Date Input Subroutines

Richard Evers

For a vast majority of programs, some method to get a date from the user, and thereafter verify the validity of that date is required. To follow are two complete subroutines to help cure that problem. Both subroutines are have been written as efficiently as possible, and as such are worthy of your close inspection before choosing the one for your particular application. Please remember that these are subroutines. A gosub to the start address is required to put them into action, with your results being held in string variable TD$.

Both routines ask for the date in numerical format. Using this information, you can convert TD$ into a day - date - month - year format. See the article on MID$ (page 34) for procedures to make these conversions.

```
5000 rem * first date subroutine
5001 rem ********************************
5002 rem * 4.0 basic - cr=167:poke 59468,14
5003 rem * c64     - cr=204:poke 53272,23
5004 rem ********************************
5005 td$="":print:c=0
5010 printchr$(147)"Please Enter Todays Date (mm/dd/yy):";
5020 gosub5150
5030 ifd$=chr$(27)then5005:rem (escape) key cancels entry
5040 ifd$<"0"ord$>"9"then5020
5050 td$=td$+d$:printd$;
5060 iflen(td$)=2thentd$(c)=td$:d=val(td$(c)):td$="":goto5080
5070 goto5020
5080 ifc=0thenprint"/";:ifd<1ord>12then5005
5090 ifc=1thenprint"/";:ifd<1ord>31then5005
5100 ifc=2thenifd<83then5005:rem the years 1983 to 1999
5110 c=c+1:ifc<3then5020
5120 td$=td$(0)+td$(1)+td$(2)
5130 return:rem *** this subroutine has returned with the date in td$
5140 rem *** get a character from the keyboard ***
5150 pokecr,0
5160 getd$:ifd$=""then5160
5170 pokecr,1:return

6000 rem * another date subroutine
6001 rem ********************************
6002 rem * 4.0 basic - poke 59468,14:cr=167
6003 rem * c64     - poke 53272,23:cr=204
6004 rem ********************************
6005 printchr$(147)"Please Enter Today's Date (mmddyy):";:gosub6035
6010 yr=val(right$(td$,2)):ifyr<83then6005
6015 dy=val(mid$(td$,3,2)):ifdy<1ornd>31then6005
6020 mt=val(left$(td$,2)):ifmt<1ornm>12then6005
6025 return:rem *** this subroutine has returned with the date in td$
6030 rem *** get a string of six from the keyboard ***
6035 td$="":pokecr,0:fori=1to6
6040 getd$:ifd$=""then6040
6045 td$=td$+d$:printd$;:next:pokecr,1:return
```

Status Interrogator

Richard Evers
Karl J. Hildon

After performing a disk or tape operation, a good program will always check the status. The status is the state of the Input/Output system that is built in to every Commodore machine. It tells us if everything went OK, or if there is some type of error that should be dealt with. In user oriented programs, it is essential that errors not go undetected. Telling the user about problems that occur as they occur, can help them avoid further complications.

Not all errors are true errors. Some are system messages that are used by the program. For example, after a disk Scratch operation, the number of files scratched will be returned as part of the "error" message. Another is the End-of-File detect that is flagged in ST. Reporting these conditions to the user keeps him informed and confident about proceeding.

This subroutine was designed as a base for your error routine. Error routines can be as individual as the program they live in. For example, a File Not Found error could mean trouble for one program, whereas another program would be expecting this to happen. It depends on the type of task you are trying to accomplish.

The last part of the program interrogates DS$ – the Disk Status. If you have BASIC 2.0 (or C64) the operating system will treat DS$ like any other variable and the report will be invalid. In the Bits and Pieces section of this issue is an alternative. It's a substitute routine for reading the Error Channel from the disk when you don't have BASIC 4.0. (There's also an alternative to that for reading it in direct mode) A GOSUB to this routine would be inserted into line 63015 instead of D$ = DS$.

```
62997 rem save " @0:st routine " ,8:verify " 0:st routine " ,8
62998 rem ** richard evers / transactor magazine **
62999 gosub63001:gosub63012:end
63000 rem *** set up the variables for st ***
63001 s$(0) = " OK – Everything Appears To Be Just Right "
63002 s$(1) = " Time Out On IEEE Write ! "
63003 s$(2) = " Time Out On IEEE Read ! "
63004 s$(3) = " Short Block (cassette read,load/verify) "
63005 s$(4) = " Long Block  (cassette read,load/verify) "
63006 s$(5) = " Unrecoverable Read Error (cass read) / Mismatch (tape load/ver) "
63007 s$(6) = " Checksum Error (cassette read,load/verify) "
63008 s$(7) = " End Of File (cassette read) or EOI (IEEE) "
63009 s$(8) = " End Of Tape (cass read, load/ver) or Device Not Present (IEEE) "
63010 return
63011 rem *** show st status ***
63012 d$ = " 00, ok,00,00,0 "
63013 a = abs(st):b = 0:ifa = 0then63016
63014 ifa>2↑bthenb = b + 1:goto63014
63015 d$ = ds$:b = b + 1
63016 prints$(b):print " Disk Status " d$:return
```

Universal Disk Routines

Richard Evers

When first writing these subroutines, I felt it best to write a separate article on each to explain the virtues to be found within. As time progressed and the subroutines were completed, a strange feeling came over me, a feeling best described as 'fatherly'. Each routine compliments each other so well that I felt it to be almost a sacrilege to consider breaking them up. In my slightly obtuse mind I had assimilated this with splitting up a very close knit family. This all adds up to a very strange analogy, and an obvious mental disorder.

Well, they have been reunited and I am sure that you will soon agree with me that the split was best avoided. This joint effort has now been comprised of three major subroutines, each being fully capable of achieving independence at any time at all. But, with all three working together, you will find that you have a very powerful trio at your access. The choice is yours.

The first subroutine is responsible for reading files from disk. Program, sequential or USR files make no difference to this one, with reason being the method incorporated to handle characters read from disk. Normally, when accessing files from disk, trouble occurs with characters that pretend to be control characters. They usually can be identified by the fact that they are in reverse case and mess up the screen quite a bit. To get around this problem, I have OR'd the ASCII value of the character brought from disk with 64, thus getting rid of any rude temptations it might have had.

As with all disk operations, the program continues bringing in information until the disk status variable ST tells it otherwise. When this happens, the file is closed up and you are asked to press any key to continue. After complying with this request, you will be promptly brought back to the doorstep of the mini-menu.

The second subroutine is responsible for reading specific sectors from disk. When you choose this option, you will be asked for the drive, track and sector numbers. Error checking has been incorporated into this section, so fouling it up will be slightly more difficult. I stress now that I am not saying that it cannot be accomplished. As the old axiom goes, 'it is impossible to make anything fool proof, for fools are so ingenious'. Now, who am I to go against an old axiom.

The program has been set up to loop through all 256 characters located in the block. As in the file reading routine, I have OR'd the ASCII value of the character from disk with 64 to keep it non-lethal. This is something you will appreciate if ever you have read in a block yourself and ended up with weird screen gyrations due to the pseudo control characters.

As with the file reading routine, when finished you will be asked to press any key to continue. When you have granted their request, you will find yourself once more propelled back through time to the mini-menu section.

The third and final member of this cartel is a block write routine, a routine that can be pretty deadly in the wrong hands. If ever you have wanted to leave a message on disk in a location that few would find, this will perform the trick for you. Remember to update the BAM later to allocate that location, or you may find your message missing in action with the advent of new files.

The get routine used in this section has been resurrected from elsewhere in this issue. All rude key strokes are eliminated from within, thus letting only the less deadly prevail. The delete key has been left operative, giving you slight editing capabilities. When a string of 254 characters has been produced, which is one less than maximum, the bell will ring once to let you know the end is near.

When the string is at its maximum of 255 characters, you will be asked whether or not to write it to disk. Any character but 'y' will be taken as no. Due to the fact that there is only one string being written to disk, the write procedure takes very little time, and will have you back to the mini-menu before you expect.

```
 0 rem * universal disk routines *
 1 rem * 1541/2031/4040 drives - mt=35:ms=21
 2 rem * 8050 drive           - mt=77:ms=29
 3 rem * 8250 drive           - mt=154:ms=29
 4 rem * 80 col       - mp=80
 5 rem * 4.0 basic    - ss=32768:ya=216:xa=198:cr=167:nc=158
 6 rem * 40 col       - mp=40
 7 rem * vic + c64    - ss=1024 :ya=214:xa=211:cr=204:nc=198
 8 rem * 22 col       - mp=22

100 print chr$(147);
105 print " (a) file reader (b) block reader (c) block writer (q) quit  ";
110 pokecr,0
115 geta$:ifa$ = " " then115
120 pokecr,1:a = asc(a$)-64:ifa$ = " q " thenend
125 ifa<1 or a>3 then 100
130 ona gosub 1000, 2000, 3000
135 goto100

1000 printchr$(147);
1005 input " Filename, Drive Number " ;f$,d$
1015 ifd$< " 0 " ord$> " 1 "  or len(f$)>16 then 1000
1020 printchr$(147);:open5,8,5, " " + d$ + " : " + f$ + " "
1025 get#5,a$:ifa$ = " " thena$ = chr$(0)
1030 ifa$ = chr$(13)thenprinta$;:goto1040
1035 printchr$(asc(a$)or64);
1040 ifst = 0then1025
1045 close5:pokenc,0
1050 print:print " press any key to return " :wait nc,1:pokenc,0:return

2000 printchr$(147);
2005 input " Specify Drive, Track and Sector :  " ;d,t,s
2010 ifd>1 or t> mt or s>ms then 2000
2015 open5,8,5, " # " :open15,8,15:print#15, " u1 " ;5;d;t;s
2020 fora = 0to255:get#5,a$:ifa$ = " " thena$ = chr$(0)
2025 printchr$(asc(a$)or64);:next:close15:pokenc,0
2030 print:print " press any key to return " :wait nc,1:pokenc,0:return

3000 printchr$(147);
3005 input " Specify Drive, Track and Sector :  " ;d,t,s
3010 ifd>1 or t>mt or s>ms then 3000
3015 printchr$(147);:gosub3045:print
3020 input " Are You Sure (y) Yes or (n) No :  " ;c$:ifc$ = " y " then3030
3025 goto3000
3030 open5,8,5, " # " :open15,8,15:print#15, " b-p " ;5;d;
3035 print#5,b$;:print#15, " u2 " ;5;d;t;s;:close5:close15:pokenc,0
3040 rem *** start of get a string of 255 characters routine ***
3045 b$ = " "
3050 a = ss + peek(xa) + peek(ya)*mp:poke a,peek(a) or 128
3055 geta$:ifa$ = " " then3055
3060 poke a,peek(a) - 128:b = asc(a$)
3065 ifb = 20 then3090:rem * delete *
3070 ifb<32 or b>128 and b<160 then 3050
3075 b$ = b$ + a$:printa$;:c = len(b$):ifc = 255 thenreturn
3080 ifc = 254thenprintchr$(7);
3085 goto3050
3090 b = len(b$):ifbthenb$ = mid$(b$,1,b-1)
3095 print:printchr$(147)b$;:goto3050
```

Making Friends With SID Part 4

Paul Higginbottom
Dallas, Texas

Have you recovered from the VERY wordy article last time? I hope so, because this time, we're moving onto NEW ground. As mentioned at the end of the last article, we will now explore the capabilities of SID's filter, which has been mentioned a couple of times, but not explained fully.

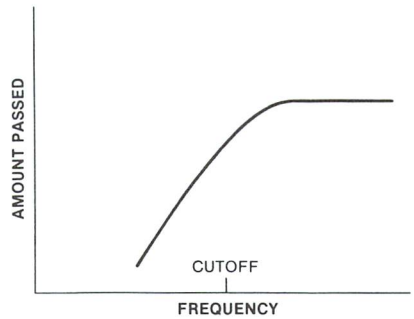
So, what is a FILTER? Let's make a comparison. You've probably heard of, and maybe used, filter paper. They're used in almost all coffee percolators. The FILTERing action of the paper in a coffee percolator, allows very fine particles of coffee to pass through, with the hot water, to form what we know as the drink; coffee, but does not allow the larger particles through, so that all the water may pass through the coffee itself, and into the container below. Essentially then, the filtering process is one where only PART of the 'somethings' involved are allowed to pass through, and the rest is not. In the case of the coffee filter, the water and fine coffee particles can pass through, while the larger ones cannot. A SOUND filter, is a process where only a certain portion (or portions) of the sound INPUT can pass through to the OUTPUT (your speaker, and then, your ears).

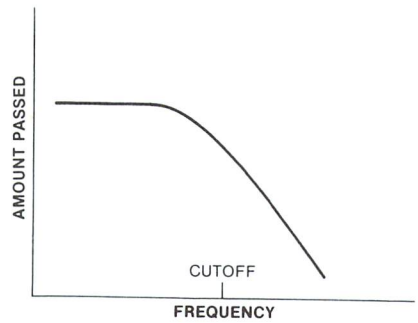
How do we control a filter? Think of the potato grading machine. Such a machine can be 'programmed' to give us a certain selection of potatoes. If we want just the small ones, we allow the potatoes to roll over a tray which has holes in it with the diameter of the MAXIMUM size we want. The bigger we make the holes, the more we let through, until eventually, we would have holes so big that EVERYTHING could 'PASS' through. How about if we wanted only medium sized potatoes? The holes would need to be the diameter of the MAXIMUM sized medium potatoe. But this would allow all the small ones through too, so we would let the potatoes falling through the first tray, roll onto a NEW tray with holes slightly smaller than the MINIMUM size. The smaller than our requirements would fall through the holes, and we would take WHAT'S ON THE TRAY (i.e., not too big, and not too small). Similarly, the ones left on the top tray would be the 'large' ones.

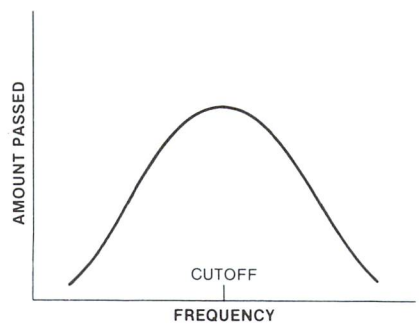
This is quite similar to the SID's sound filtering process. For example, if we only want the frequencies BELOW a given frequency, we use the LOW PASS MODE in SID. This only ALLOWS frequencies LOWer than a specified frequency to PASS through to the output. Like the potatoe sorter, we have other modes to select different portions, also. We have BAND PASS MODE, which will allow a given amount above, and below, a specified frequency to pass through. We also have HIGH PASS MODE (I know, you guessed it!) which will only allow frequencies HIGHer than a specified frequency to pass through.

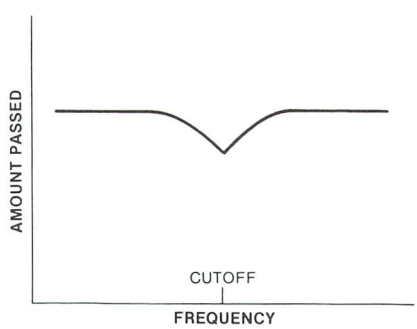
The 'specified frequency' I keep referring to is known as the CUT-OFF FREQUENCY. Also, I have made it seem like the filter cuts out completely any sound immediately below, slightly below and above, or above (depending on whether it's LOW, BAND, or HIGH pass). This isn't so; let's take LOW pass mode for example:

As shown in the diagram, the filter suppresses frequencies above the cutoff frequency in proportion to how far above it they are. If a frequency is only slightly above the cutoff, it will be suppressed slightly. If a long way above it (relatively), it will be cut out altogether practically.









That's the theory; now the practice. After loading the final program from the last article into your Commodore–64, enter the following lines:

Change line 540 which reads:

```
540 poke sid+24,15
to:
540 poke sid+24,15+16
```

The addition of 16 to this POKE sets the filter to LOW PASS mode, thus suppressing frequencies higher than the cutoff.

For BAND PASS, we would add 32 instead of 16 (suppress frequencies proportionate to how far above AND below the cutoff they are), and for HIGH pass, we add 64 (suppress frequencies below the cutoff). Modes can also be added, so we could have low AND high pass, so as to suppress frequencies AROUND the cutoff (this is called NOTCH PASS, or BAND REJECT).

Change line 550 which reads:

```
550 read a,d,s,r,pw
to:
550 read a,d,s,r,pw,c,re
```

The extra parameter C in the READ statement is our cutoff value which we will put in the DATA statement in line 600.

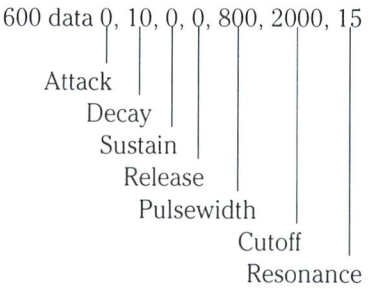
The extra parameter RE in the READ statement is a parameter not mentioned before, called RESONANCE. The filter emphasizes frequencies (makes a sharper sound) AT the cutoff, and the degree to which it does this is known as "resonance". We can select a value of 0–15 for this. I wouldn't worry if this doesn't make a lot of sense; it's use is only truly efffective in machine language programs, where the resonance can be changed rapidly to produce more complex harmonics.

Enter:

```
610 poke sid+23,7+re*16
```

SID's filter doesn't HAVE to filter ALL the voices. We can specify which of the three we wish to filter. Of course, in our "piano" program, we want all the sounds to be the same so we tell SID to filter ALL the voices. This is controlled in register 23. The value 7 equals 4+2+1, and each of the three numbers added are the ones to filter the voices 1, 2, and 3 respectively. The addition of RE*16 is adding the previously explained parameter: resonance.

Change line 600's DATA to:



Make sure line 510 reads:

```
510 voice=0:oct=3:wave=32
```

Enter:

```
620 ch=int(c/8):cl=c-ch*8
630 poke sid+21,cl:poke sid+22,ch
```

These two lines set the cutoff frequency: C. The cutoff is specified in registers 21 and 22 and can range from 0–2047. Unlike the way we have specified values larger than 255 before (i.e. 'lots' of 256 + some other value between 0 and 255), this one is specified in 'lots' of 8 and some other value (remainder) between 0 and 7, so we can calculate 'how many' 8's by finding the integer value of the cutoff divided by 8 (ch=int(c/8)) and find the remainder by subtracting how many 8's multiplied by 8 from the cutoff (c–ch*8).

Now To Experiment

If you run the program with the changes mentioned it won't sound particularly different from what you heard before you made the changes. That's because our filter mode is LOW PASS, and the cutoff is almost at MAXIMUM, therefore letting almost ALL the sound through since almost all the sound WILL have frequencies lower than the cutoff. If we change the value of the cutoff in line 600 from 2000 to say, 1200 and run the program again, you should DEFINITELY be able to hear a difference. The sound seems "muffled". This is because the filter has removed a lot of the high frequency parts (harmonics) of the sound. (Note: in between the two values used for cutoff here, you may have noticed that a GOOD piano sound could be made. Experiment!)

If we now change then +16 to +64 in line 540 so that we select HIGH PASS filter mode instead, and run the program again, we will hear that the "lows" have gone from the sound, because only those frequencies higher than the cutoff are getting through without being suppressed. Now change the cutoff in line 600 to 1800 and run the program again and you'll hear that we've REALLY chopped out any low harmonics, and it sounds more like a harpsichord.

Try changing the OCT (octave) variable (between 1 and 6) and the WAVE (waveform) variable (one of 16, 32, 64, 128) in line 510 and you should begin to see that the filter is probably THE feature which makes the SID surpass ANY sound chip around.

If you really want to get adventurous, you might be able to work out how to change the cutoff dynamically (while the notes are being played) to REALLY produce some interesting effects, (like guitar "twangs" for example).

Next time we'll be going from this "playing" program to one where we can put a tune into the program in DATA statements and thus have the computer play FOR us!

The Commodore 64 Programmer's Reference Guide gives a LOT of information on the SID and programming it (as well as another 400 pages on LOADS of other interesting aspects), so I advise you to get a copy.

Good luck!

```
100 fr = 3520 : rem a in top octave
110 co = 2↑(1/12) : rem constant multiplyer for next semitone
120 for i = 1 to 9 : fr = fr/co : next : rem start fr at c by going back 9 semitones
130 ss = 16777216 : rem sid clock
140 cs = 1022730 : rem cpu clock
150 fc = ss/cs : rem frequency multiplying constant
200 dim f(7,11) : rem frequency array (octave, semitone)
300 for i = 0 to 11 : rem cycle through 12 semitones
310 s = fr*fc : rem calculate sid value of semitone in top octave
400 for j = 7 to 0 step-1 : f(j,i) = s : s = s/2
410 next : rem calc value for all 8 octaves
420 fr = fr*co : rem go onto next semitone
430 next : rem continue through all 12 semitones
500 sid = 54272
510 voice = 0 : oct = 3 : wave = 32
520 vm = 7 : hi = 256
530 for i = 0 to 23 : poke sid + i,0 : next
540 poke sid + 24,15 + 16
550 read a,d,s,r,pw,c,re
560 for i = 0 to 2 : index = sid + i*vm
570 poke index + 5,a*16 + d
580 poke index + 6,s*16 + r
590 next
600 data 3,10,0,0,800,2000,15
610 poke sid + 23,7 + re*16
620 ch = int(c/8) : cl = c-ch*8
630 poke sid + 21,cl : poke sid + 22,ch
700 k$ = " q2w3er5t6y7ui9o0p@-*\↑
710 dim k(255)
720 for i = 1 to len(k$)
730 k(asc(mid$(k$,i,1))) = i
740 next
750 print : print "  2 3  5 6 7  9 0  - \
760 print : print " q w e r t y u i o p @ * ↑
800 get a$ : if a$ = "" goto 800
810 key = k(asc(a$))-1 : if key<0 goto 800
820 if key>11 then frq = f(oct + 1,key-12) : goto 835
830 frq = f(oct,key)
835 fh = int(frq/hi) : fl = frq-fh*hi
840 index = sid + voice*vm
850 poke index,fl : poke index + 1,fh
860 poke index + 4,wave : poke index + 4,wave + 1
870 voice = voice + 1 : if voice>2 then voice = 0
880 goto 800
```

Making Friends With SID Part 5

Paul Higginbottom

As I said at the end of the last article, this, the last in this series of articles, will add lines to our program to let us put a tune into the computer, in DATA statements, which the computer will READ and play when we RUN the program, instead of us playing a tune on the keyboard as our program stands at present.

To do this, all we have to do is to change the program to read DATA statements instead of the keyboard, and play the notes accordingly.

A logical format for the DATA is 3 numbers per note: octave, semitone, and duration of the note. The modifications to this program will not allow 3 part tunes to be played, it will simply be a modification to read the notes from DATA instead of our "typed" input, playing the notes in voices one, two, and three, and then back to one, as explained in Part 4.

LOAD the final program from Part 4, and delete lines 700 to 760 (simply enter the line numbers, and press RETURN; the numbers being 700, 710, 720, 730, 740, 750, and 760). These lines were used to create an array of numbers corresponding to the ASCII codes of the keys in our keyboard layout, as well as printing the layout on the screen.

At the start of our tune DATA we'll have a "tempo" (the speed the notes will be played and read). So enter:

```
770 read tempo
```

Our playing loop will replace some of the lines in the 'playing' section (lines 800 onward).

Change line 800 from:

```
800 get a$ : if a$ = "" goto 800
```

To:

```
800 read oct,sem,dur:if oct<0 goto 900:rem read
    note and exit if end
```

This line changes the note input from keyboard to DATA statements, and checks to see if it's reached the end of the tune, by seeing if the OCTave variable is negative (that's how we'll mark the end of a tune in our DATA). It will branch to line 900 if the end is reached.

Delete lines 810 and 820 which used to work out the key position from the A$ input in the GET statement, and which we obviously no longer need. Change 830 from:

```
830 frq=f(oct,key)
```

To:

```
830 frq=f(oct,sem)
```

This simply changes the variable for the semitone subscript from KEY (which we no longer use) to SEM, the value of which was READ from the DATA in our new line 800.

(Lines 835, 840, 850, 860, and 870 will work just as they did before.) Enter:

```
875 for i=1 to dur*tempo:next
```

This is our timekeeping line, which waits for the duration of the note (DUR) multiplied by our overall TEMPO (our first DATA value, read in 770).

```
900 for i=1 to 1000:next
910 poke sid+24,0:end
```

Line 900 is branched to from line 800 when the end of the DATA is reached, and it simply waits for approximately one second (an 'empty' loop) to allow the last note to end. 910 turns off the volume because the tune has finished and the program ENDs.

We'll make line 1000 contain the tempo, and lines 2000 onward the tune DATA, and add a line 9000 with the DATA containing a negative value for OCT to indicate the end of the tune (that gives us TONS of space for a tune!!). Thus, enter:

```
1000 data .5 : rem tempo
9000 data -1,0,0 : rem end of tune
```

Don't faint when you see the next TON of lines, you don't have to enter them all. I went 'mad' (how could I GO mad, I am, right?!) one night and worked out all the DATA for Bach's first Prelude. As the sun rose, I ran the program and it was (literally) music to my ears. You can enter as much of this DATA as you want, but I'm sure that you'll want to hear the whole tune, as I did!

As I said at the start of this article, this is to be the last in my mini-series on the SID. If you've followed them from the first, hopefully you're now a little more aware of the capabilities of the our friend SID.

Happy programming!

```
Data for Bach's
Prelude in C (#1):
1000 rem tempo
1000 data .25
2000 rem measure 1
2010 data 4,0,120
2020 data 4,4,120
2030 data 4,7,120
2040 data 5,0,120
2050 data 5,4,120
2060 data 4,7,120
2070 data 5,0,120
2080 data 5,4,120
2090 data 4,0,120
2100 data 4,4,120
2110 data 4,7,120
2120 data 5,0,120
2130 data 5,4,120
2140 data 4,7,120
2150 data 5,0,120
2160 data 5,4,120
2170 rem measure 2
2180 data 4,0,120
2190 data 4,2,120
2200 data 4,9,120
2210 data 5,2,120
2220 data 5,5,120
2230 data 4,9,120
2240 data 5,2,120
2250 data 5,5,120
2260 data 4,0,120
2270 data 4,2,120
2280 data 4,9,120
2290 data 5,2,120
2300 data 5,5,120
2310 data 4,9,120
2320 data 5,2,120
2330 data 5,5,120
2340 rem measure 3
2350 data 3,11,120
2360 data 4,2,120
2370 data 4,7,120
2380 data 5,2,120
2390 data 5,5,120
2400 data 4,7,120
2410 data 5,2,120
2420 data 5,5,120
2430 data 3,11,120
2440 data 4,2,120
2450 data 4,7,120
2460 data 5,2,120
2470 data 5,5,120
2480 data 4,7,120
2490 data 5,2,120
2500 data 5,5,120
2510 rem measure 4
     (same as 1)
2520 data 4,0,120
2530 data 4,4,120
2540 data 4,7,120
2550 data 5,0,120
2560 data 5,4,120
2570 data 4,7,120
2580 data 5,0,120
2590 data 5,4,120
2600 data 4,0,120
2610 data 4,4,120
2620 data 4,7,120
2630 data 5,0,120
2640 data 5,4,120
2650 data 4,7,120
2660 data 5,0,120
2670 data 5,4,120
2680 rem measure 5
2690 data 4,0,120
2700 data 4,4,120
2710 data 4,9,120
2720 data 5,4,120
2730 data 5,9,120
2740 data 4,9,120
2750 data 5,4,120
2760 data 5,9,120
2770 data 4,0,120
2780 data 4,4,120
2790 data 4,9,120
2800 data 5,4,120
2810 data 5,9,120
2820 data 4,9,120
2830 data 5,4,120
2840 data 5,9,120
2850 rem measure 6
2860 data 4,0,120
2870 data 4,2,120
2880 data 4,6,120
2890 data 4,9,120
2900 data 5,2,120
2910 data 4,6,120
2920 data 4,9,120
2930 data 5,2,120
2940 data 4,0,120
2950 data 4,2,120
2960 data 4,6,120
2970 data 4,9,120
2980 data 5,2,120
2990 data 4,6,120
3000 data 4,9,120
3010 data 5,2,120
3020 rem measure 7
3030 data 3,11,120
3040 data 4,2,120
3050 data 4,7,120
3060 data 5,2,120
3070 data 5,7,120
3080 data 4,7,120
3090 data 5,2,120
3100 data 5,7,120
3110 data 3,11,120
3120 data 4,2,120
3130 data 4,7,120
3140 data 5,2,120
3150 data 5,7,120
3160 data 4,7,120
3170 data 5,2,120
3180 data 5,7,120
3190 rem measure 8
3200 data 3,11,120
3210 data 4,0,120
3220 data 4,4,120
3230 data 4,7,120
3240 data 5,0,120
3250 data 4,4,120
3260 data 4,7,120
3270 data 5,0,120
3280 data 3,11,120
3290 data 4,0,120
3300 data 4,4,120
3310 data 4,7,120
3320 data 5,0,120
3330 data 4,4,120
3340 data 4,7,120
3350 data 5,0,120
3360 rem measure 9
3370 data 3,9,120
3380 data 4,0,120
3390 data 4,4,120
3400 data 4,7,120
3410 data 5,0,120
3420 data 4,4,120
3430 data 4,7,120
3440 data 5,0,120
3450 data 3,9,120
3460 data 4,0,120
3470 data 4,4,120
3480 data 4,7,120
3490 data 5,0,120
3500 data 4,4,120
3510 data 4,7,120
3520 data 5,0,120
3530 rem measure 10
3540 data 3,2,120
3550 data 3,9,120
3560 data 4,2,120
3570 data 4,6,120
3580 data 5,0,120
3590 data 4,2,120
3600 data 4,6,120
3610 data 5,0,120
3620 data 3,2,120
3630 data 3,9,120
3640 data 4,2,120
3650 data 4,6,120
3660 data 5,0,120
3670 data 4,2,120
3680 data 4,6,120
3690 data 5,0,120
3700 rem measure 11
3710 data 3,7,120
3720 data 3,11,120
3730 data 4,2,120
3740 data 4,7,120
3750 data 4,11,120
3760 data 4,2,120
3770 data 4,7,120
3780 data 4,11,120
3790 data 3,7,120
3800 data 3,11,120
3810 data 4,2,120
3820 data 4,7,120
3830 data 4,11,120
3840 data 4,2,120
3850 data 4,7,120
3860 data 4,11,120
3870 rem measure 12
3880 data 3,7,120
3890 data 3,10,120
3900 data 4,4,120
3910 data 4,7,120
3920 data 5,1,120
3930 data 4,4,120
3940 data 4,7,120
3950 data 5,1,120
3960 data 3,7,120
3970 data 3,10,120
3980 data 4,4,120
3990 data 4,7,120
4000 data 5,1,120
4010 data 4,4,120
4020 data 4,7,120
4030 data 5,1,120
4040 rem measure 13
4050 data 3,5,120
4060 data 3,9,120
4070 data 4,2,120
4080 data 4,9,120
4090 data 5,2,120
4100 data 4,2,120
4110 data 4,9,120
4120 data 5,2,120
4130 data 3,5,120
4140 data 3,9,120
4150 data 4,2,120
4160 data 4,9,120
4170 data 5,2,120
4180 data 4,2,120
4190 data 4,9,120
4200 data 5,2,120
4210 rem measure 14
4220 data 3,5,120
4230 data 3,8,120
4240 data 4,2,120
4250 data 4,5,120
4260 data 4,11,120
4270 data 4,2,120
4280 data 4,5,120
4290 data 4,11,120
4300 data 3,5,120
4310 data 3,8,120
4320 data 4,2,120
4330 data 4,5,120
4340 data 4,11,120
4350 data 4,2,120
4360 data 4,5,120
4370 data 4,11,120
4380 rem measure 15
4390 data 3,4,120
4400 data 3,7,120
4410 data 4,0,120
4420 data 4,7,120
4430 data 5,0,120
4440 data 4,0,120
4450 data 4,7,120
4460 data 5,0,120
4470 data 3,4,120
4480 data 3,7,120
4490 data 4,0,120
4500 data 4,7,120
4510 data 5,0,120
4520 data 4,0,120
4530 data 4,7,120
4540 data 5,0,120
4550 rem measure 16
4560 data 3,4,120
4570 data 3,5,120
4580 data 3,9,120
4590 data 4,0,120
4600 data 4,5,120
4610 data 3,9,120
4620 data 4,0,120
4630 data 4,5,120
4640 data 3,4,120
4650 data 3,5,120
4660 data 3,9,120
4670 data 4,0,120
4680 data 4,5,120
4690 data 3,9,120
4700 data 4,0,120
4710 data 4,5,120
4720 rem measure 17
4730 data 3,2,120
4740 data 3,5,120
4750 data 3,9,120
4760 data 4,0,120
4770 data 4,5,120
4780 data 3,9,120
4790 data 4,0,120
4800 data 4,5,120
4810 data 3,2,120
4820 data 3,5,120
4830 data 3,9,120
4840 data 4,0,120
4850 data 4,5,120
4860 data 3,9,120
4870 data 4,0,120
4880 data 4,5,120
4890 rem measure 18
```

```
4900 data 2,7,120
4910 data 3,2,120
4920 data 3,7,120
4930 data 3,11,120
4940 data 4,5,120
4950 data 3,7,120
4960 data 3,11,120
4970 data 4,5,120
4980 data 2,7,120
4990 data 3,2,120
5000 data 3,7,120
5010 data 3,11,120
5020 data 4,5,120
5030 data 3,7,120
5040 data 3,11,120
5050 data 4,5,120
5060 rem measure 19
5070 data 3,0,120
5080 data 3,4,120
5090 data 3,7,120
5100 data 4,0,120
5110 data 4,4,120
5120 data 3,7,120
5130 data 4,0,120
5140 data 4,4,120
5150 data 3,0,120
5160 data 3,4,120
5170 data 3,7,120
5180 data 4,0,120
5190 data 4,4,120
5200 data 3,7,120
5210 data 4,0,120
5220 data 4,4,120
5230 rem measure 20
5240 data 3,0,120
5250 data 3,7,120
5260 data 3,10,120
5270 data 4,0,120
5280 data 4,4,120
5290 data 3,10,120
5300 data 4,0,120
5310 data 4,4,120
5320 data 3,0,120
5330 data 3,7,120
5340 data 3,10,120
5350 data 4,0,120
5360 data 4,4,120
5370 data 3,10,120
5380 data 4,0,120
5390 data 4,4,120
5400 rem measure 21
5410 data 2,5,120
5420 data 3,5,120
5430 data 3,9,120
5440 data 4,0,120
5450 data 4,4,120
5460 data 3,9,120
5470 data 4,0,120
5480 data 4,4,120
5490 data 2,5,120
5500 data 3,5,120
5510 data 3,9,120
5520 data 4,0,120
5530 data 4,4,120
5540 data 3,9,120
5550 data 4,0,120
5560 data 4,4,120
5570 rem measure 22
5580 data 2,6,120
5590 data 3,0,120
5600 data 3,9,120
5610 data 4,0,120
5620 data 4,4,120
5630 data 3,9,120
5640 data 4,0,120
5650 data 4,4,120
5660 data 2,6,120
5670 data 3,0,120
5680 data 3,9,120
5690 data 4,0,120
5700 data 4,4,120
5710 data 3,9,120
5720 data 4,0,120
5730 data 4,4,120
5740 rem measure 23
5750 data 2,8,120
5760 data 3,5,120
5770 data 3,11,120
5780 data 4,0,120
5790 data 4,2,120
5800 data 3,11,120
5810 data 4,0,120
5820 data 4,2,120
5830 data 2,8,120
5840 data 3,5,120
5850 data 3,11,120
5860 data 4,0,120
5870 data 4,2,120
5880 data 3,11,120
5890 data 4,0,120
5900 data 4,2,120
5910 rem measure 24
5920 data 2,7,120
5930 data 3,5,120
5940 data 3,7,120
5950 data 3,11,120
5960 data 4,2,120
5970 data 3,7,120
5980 data 3,11,120
5990 data 4,2,120
6000 data 2,7,120
6010 data 3,5,120
6020 data 3,7,120
6030 data 3,11,120
6040 data 4,2,120
6050 data 3,7,120
6060 data 3,11,120
6070 data 4,2,120
6080 rem measure 25
6090 data 2,7,120
6100 data 3,4,120
6110 data 3,7,120
6120 data 4,0,120
6130 data 4,4,120
6140 data 3,7,120
6150 data 4,0,120
6160 data 4,4,120
6170 data 2,7,120
6180 data 3,4,120
6190 data 3,7,120
6200 data 4,0,120
6210 data 4,4,120
6220 data 3,7,120
6230 data 4,0,120
6240 data 4,4,120
6250 rem measure 26
6260 data 2,7,120
6270 data 3,2,120
6280 data 3,7,120
6290 data 4,0,120
6300 data 4,5,120
6310 data 3,7,120
6320 data 4,0,120
6330 data 4,5,120
6340 data 2,7,120
6350 data 3,2,120
6360 data 3,7,120
6370 data 4,0,120
6380 data 4,5,120
6390 data 3,7,120
6400 data 4,0,120
6410 data 4,5,120
6420 rem measure 27
6430 data 2,7,120
6440 data 3,2,120
6450 data 3,7,120
6460 data 3,11,120
6470 data 4,5,120
6480 data 3,7,120
6490 data 3,11,120
6500 data 4,5,120
6510 data 2,7,120
6520 data 3,2,120
6530 data 3,7,120
6540 data 3,11,120
6550 data 4,5,120
6560 data 3,7,120
6570 data 3,11,120
6580 data 4,5,120
6590 rem measure 28
6600 data 2,7,120
6610 data 3,3,120
6620 data 3,9,120
6630 data 4,0,120
6640 data 4,6,120
6650 data 3,9,120
6660 data 4,0,120
6670 data 4,6,120
6680 data 2,7,120
6690 data 3,3,120
6700 data 3,9,120
6710 data 4,0,120
6720 data 4,6,120
6730 data 3,9,120
6740 data 4,0,120
6750 data 4,6,120
6760 rem measure 29
6770 data 2,7,120
6780 data 3,5,120
6790 data 3,7,120
6800 data 4,0,120
6810 data 4,7,120
6820 data 3,7,120
6830 data 4,0,120
6840 data 4,7,120
6850 data 2,7,120
6860 data 3,5,120
6870 data 3,7,120
6880 data 4,0,120
6890 data 4,7,120
6900 data 3,7,120
6910 data 4,0,120
6920 data 4,7,120
6930 rem measure 30
6940 data 2,7,120
6950 data 3,2,120
6960 data 3,7,120
6970 data 4,0,120
6980 data 4,5,120
6990 data 3,7,120
7000 data 4,0,120
7010 data 4,5,120
7020 data 2,7,120
7030 data 3,2,120
7040 data 3,7,120
7050 data 4,0,120
7060 data 4,5,120
7070 data 3,7,120
7080 data 4,0,120
7090 data 4,5,120
7100 rem measure 31
7110 data 2,7,120
7120 data 3,2,120
7130 data 3,7,120
7140 data 3,11,120
7150 data 4,5,120
7160 data 3,7,120
7170 data 3,11,120
7180 data 4,5,120
7190 data 2,7,120
7200 data 3,2,120
7210 data 3,7,120
7220 data 3,11,120
7230 data 4,5,120
7240 data 3,7,120
7250 data 3,11,120
7260 data 4,5,120
7270 rem measure 32
7280 data 2,0,120
7290 data 3,0,120
7300 data 3,7,120
7310 data 3,10,120
7320 data 4,4,120
7330 data 3,7,120
7340 data 3,10,120
7350 data 4,4,120
7360 data 2,0,120
7370 data 3,0,120
7380 data 3,7,120
7390 data 3,10,120
7400 data 4,4,120
7410 data 3,7,120
7420 data 3,10,120
7430 data 4,4,120
7440 rem measure 33
7450 data 2,0,120
7460 data 3,0,120
7470 data 3,5,120
7480 data 3,9,120
7490 data 4,0,120
7500 data 4,5,120
7510 data 4,0,120
7520 data 3,9,120
7530 data 4,0,120
7540 data 3,9,120
7550 data 3,5,120
7560 data 3,9,120
7570 data 3,5,120
7580 data 3,2,120
7590 data 3,5,120
7600 data 3,2,120
7610 rem measure 34
7620 data 2,0,120
7630 data 2,11,120
7640 data 4,7,120
7650 data 4,11,120
7660 data 5,2,120
7670 data 5,5,120
7680 data 5,2,120
7690 data 4,11,120
7700 data 5,2,120
7710 data 4,11,120
7720 data 4,7,120
7730 data 4,11,120
7740 data 4,2,120
7750 data 4,5,120
7760 data 4,4,120
7770 data 4,2,120
7780 rem measure 35
7790 data 4,4,0
7800 data 4,7,0
7810 data 5,0,0
7820 data 0,0,-9999
9000 data -1,0,0
9010 rem end of tune
```

Put Those Function Keys To Work!

Darren J. Spruyt
Gravenhurst, Ont.

Here's a handy little utility for the Commodore 64. This utility lets you define any of the eight function keys as a line of BASIC code, a command such as LIST or RUN, or whatever you want.

The program is written in machine language and it operates off the internal interrupts. These interrupts occur every sixtieth of a second and they allow the keyboard to be scanned so that if one of the function keys is pressed, it can be detected.

Listing 1 is the BASIC loader for the machine code. Type it in and RUN it. If you have made a mistake, the program will display "Checksum error". Once it is fixed, SAVE it and VERIFY it.

Lisitng 2 is the disassembly of the program. Along with the comments, it should be very easy to follow.

The instructions displayed by the program are a little rough, so let's go over them.

Type: SYS 49152 and press Return

A display reading "Function 1" should appear. Below it there should be a prompt ("?") with the flashing cursor beside it. You can now enter the line of BASIC code or whatever you want to be assigned to function key 1. When finished, press Return.

"Function 2" should now be displayed, along with the prompt and the cursor. You can now type what you want this key to become and press Return. Continue this until all eight keys have been defined. After the eighth key, "READY." will be displayed and you are back in direct mode.

The function keys will now execute whatever is stored in them. If you have not defined a certain function key, it will simply give you the "READY." message. With a program running, the function keys will behave normally.

To re-define a key, enter SYS 49152 and hit return. Like before, you will be prompted for the definition. Hit Return and the old defintion will not be changed, or, enter the new definition.

One last thing: the interrupt needs to be set back to its pristine condition before any tape I/O will work. RUN-STOP/RESTORE will do the trick. To get the utility back to work, SYS 49152 and press Return (8 times) until the "READY." is displayed.

This is all you need to know to use this program. So to save yourself some work, just put this program to work. I wrote it for myself, and for just that reason. I hope you will enjoy it.

Listing 1:

```
10 print " Sq     c-64 function key definer
15 print "      by darren spruyt of
20 print "      gravenhurst, ontario, canada
25 print " qq]] setting up. . ... "
30 for j=0to227:readb:x=x+b:poke 49152+j,b:poke1024,b:nextj
40 if x <> 30606 then print " checksum error " :stop
45 for j=0to7:readb:pokeb,128:pokeb+1,0:pokeb+3,0:next
50 print " ready to use. "
60 print " q sys 49152 to define the function keys. "
70 print " q  to use, just press the function key. "
80 print " q  to re-define, sys 49152. to skip over
90 print "  re-defining a key, just press return
95 print "  at its number. "
99 end
100 data 162,   0, 134, 151,  32, 131
110 data 192, 173,   0,   2, 240,   3
120 data  32, 171, 192, 166, 151, 232
130 data 134, 151, 224,   8, 208, 236
140 data  32,  87, 192,  76, 116, 192
150 data  36, 157,  48,   3,  76,  49
160 data 234, 165, 215, 201, 133, 144
170 data 247, 201, 141, 176, 243, 162
180 data 250, 154, 165, 215,  56, 233
190 data 133, 170, 189, 203, 192, 170
200 data 189, 211, 192, 133, 122, 189
210 data 219, 192, 133, 123, 169, 255
220 data 133,  58, 164, 211, 165, 206
230 data 145, 209, 169,   1, 133, 204
240 data  76, 225, 167, 162,   0, 189
250 data 211, 192, 133,  34, 189, 219
260 data 192, 133,  35, 160,   1, 169
270 data 153, 145,  34, 200, 169,  58
280 data 145,  34, 232, 224,   8, 208
290 data 230,  96, 120, 169,  30, 141
300 data  20,   3, 169, 192, 141,  21
310 data   3,  88,  76, 116, 164, 169
320 data 160, 160, 192,  32,  30, 171
330 data 169,   0, 166, 151, 232,  32
340 data 205, 189, 169,  13,  32, 210
350 data 255,  32, 249, 171, 162,   0
360 data  32, 126, 165,  96,  13,  70
370 data  85,  78,  67,  84,  73,  79
380 data  78,  32,   0, 166, 151, 189
390 data 211, 192, 133,  34, 189, 219
400 data 192, 133,  35, 160,   2, 200
410 data 185, 253,   1, 145,  34, 200
420 data 200, 145,  34, 136, 136, 185
430 data 253,   1, 208, 239,  96,   0
440 data   2,   4,   6,   1,   3,   5
450 data   7, 255,  79, 159, 255,  79
460 data 159, 255,  79, 193, 194, 194
470 data 194, 195, 195, 195, 196, 234
500 data 49666, 49746, 49826, 49922
510 data 50002, 50082, 50178, 50258
```

Listing 2

```
c000  ldx   #$00     ;this routine
c002  stx   $97      ;is for the
c004  jsr   $c083    ;inputing of the
c007  lda   $0200    ;line for defin-
c00a  beq   $c00f    ;ing. it gets
c00c  jsr   $c0ab    ;input, then
c00f  ldx   $97      ;checks for no
c011  inx            ;input, then
c012  stx   $97      ;transfer (if
c014  cpx   #$08     ;req.) to the
c016  bne   $c004    ;memory section
c018  jsr   $c057    ;for the chosen
c01b  jmp   $c074    ;function key
c01e  bit   $9d      ;test direct
c020  bmi   $c025    ;go if direct
c022  jmp   $ea31    ;cont with int.
c025  lda   $d7      ;current char
c027  cmp   #$85     ;check low
c029  bcc   $c022    ;go if lower
c02b  cmp   #$8d     ;check high
c02d  bcs   $c022    ;go if higher
c02f  ldx   #$fa     ;set stack to
c031  txs            ;clear
c032  lda   $d7      ;get char
c034  sec            ;subtract to get
c035  sbc   #$85     ;range 0-7
c037  tax            ;
c038  lda   $c0cb,x  ;put through
c03b  tax            ;conver table
c03c  lda   $c0d3,x  ;get lo and high
c03f  sta   $7a      ;bytes for
c041  lda   $c0db,x  ;chrget indirect
c044  sta   $7b      ;address
c046  lda   #$ff     ;set to simulate
c048  sta   $3a      ;direct mode
c04a  ldy   $d3      ;restore char
c04c  lda   $ce      ;to where
c04e  sta   ($d1),y  ;the cursor
c050  lda   #$01     ;was and
c052  sta   $cc      ;turn off
c054  jmp   $a7e1    ;execute
c057  ldx   #$00     ;this routine
c059  lda   $c0d3,x  ;sets up parts
c05c  sta   $22      ;of the memory
c05e  lda   $c0db,x  ;with the code
c061  sta   $23      ;for a) "print"
c063  ldy   #$01     ;and b) ":"
c065  lda   #$99     ;so execution
c067  sta   ($22),y  ;starts with a
c069  iny            ;line being
c06a  lda   #$3a     ;printed before
c06c  sta   ($22),y  ;anything else
c06e  inx            ;
c06f  cpx   #$08     ;
c071  bne   $c059    ;
c073  rts            ;return from sub
c074  sei            ;this routine
c075  lda   #$1e     ;sets up the
c077  sta   $0314    ;interrrupt
c07a  lda   #$c0     ;vector to go to
c07c  sta   $0315    ;the correct
c07f  cli            ;memory
c080  jmp   $a474    ;location
c083  lda   #$a0     ;set up to print
c085  ldy   #$c0     ;the message
c087  jsr   $ab1e    ;"function "
c08a  lda   #$00     ;print the
c08c  ldx   $97      ;fixed-point
c08e  inx            ;number
c08f  jsr   $bdcd    ;next
c092  lda   #$0d     ;print a return
c094  jsr   $ffd2    ;character
c097  jsr   $abf9    ;input a line
c09a  ldx   #$00     ;and crunch
c09c  jsr   $a57e    ;the tokens
c09f  rts            ;end of routine
c0a0  0d 46 55       ;"<return>fu"
c0a3  4e 43 54       ;"nct"
c0a6  49 4f          ;"io"
c0a8  4e 20 00       ;"n "0-end
c0ab  ldx   $97      ;this routine
c0ad  lda   $c0d3,x  ;transfers
c0b0  sta   $22      ;the contents
c0b2  lda   $c0db,x  ;of the input
c0b5  sta   $23      ;buffer to the
c0b7  ldy   #$02     ;selected
c0b9  iny            ;memory part,
c0ba  lda   $01fd,y  ;for the
c0bd  sta   ($22),y  ;individual
c0bf  iny            ;function keys
c0c0  iny            ;
c0c1  sta   ($22),y  ;
c0c3  dey            ;
c0c4  dey            ;
c0c5  lda   $01fd,y  ;
c0c8  bne   $c0b9    ;
c0ca  rts            ;return from sub
c0cb  00 02 04 06    ;conver table
c0cc  01 03 05 07    ;
c0d3  ff 4f 9f ff    ;lo-byte table
c0d7  4f 9f ff 4f
c0db  c1 c2 c2 c2    ;hi-byte table
c0df  c3 c3 c3 c4
```

A Simple Disk Copier For The Commodore 64

Jim Butterfield
Toronto, Ont.

If you have a disk on your Commodore 64, program copying may not seem to be much of a task. A simple LOAD followed by a SAVE would seem to do all that is necessary.

There are some programs that don't cooperate, however. For example, there's a class of programs called "boot" (or bootstrap) programs that stitch chunks of memory together to make a system of programs which work together. These programs that make up the system don't look like ordinary Basic; so LOAD and SAVE, which were designed for Basic, won't work right. There are other programs which are cantankerous copiers, too: programs containing machine language, for example.

Sequential files can't be LOAD–ed, so you can't copy them with LOAD and SAVE.

You can often copy a disk by using a backup program. This copies everything over to a new disk. It moves this information over, disk sector by disk sector, so you must copy the whole disk. That's useful, but sometimes you don't want the whole thing. . . just a program or file or two.

Here's a program that will copy a file for you. If you want to copy two files, run the program twice. You must know the name of the program you want to copy: COPYFILE doesn't read the directory for you. It's just a simple minimum program to do the job.

COPY FILE has a bonus that goes with its simplicity, however. Since this small program doesn't take up much space, it can use lots of memory to do the copying; so it can copy big files. If your Commodore 64 doesn't have any other systems loaded into it – no DOS wedge program, no IEEE interface – you can copy files up to 50K in size. If you are using the "wedge" or an interface, you'll have to restrict yourself to smaller programs – probably not over 35K or so. But that's still plenty big.

Running The Program

Before running COPY FILE, make sure you have two disks ready: the one you want to copy from, and the one to which you want to write the copy. Be sure you know the names of the files: make a note if necessary. The "destination" disk must be formatted; it may already contain programs and files, since the new material will be added.

Load and run COPY FILE. You'll be asked for the file type. You may answer: S for Sequential, U for User (a rare file type), or P for Program. Program COPY FILE will not copy relative files.

Next, you'll be asked for the name of the program or file you wish to copy. Type in the name, and be sure it's correct. Press RETURN and COPY FILE will look for the file you have named.

If it can't find the file, or if it sees other problems, it will reply NO GO. Otherwise, it will ask OTHER DISK READY? Take the old disk out of the drive, put the new disk in, and answer anything: a letter Y followed by a RETURN will do the trick. The file will now be written to the new disk.

Watch for signs of disk errors. The destination disk might have problems: perhaps it hasn't been formatted, or has a write protect tab in place, or already has a file of that name, or has't enough room for the program you wish to copy. In any of these cases, the disk error light will flash.

But normally, the file will be copied. To copy another, say RUN and COPY FILE will do it again.

Where's The Program?

I'd like to list program COPY FILE for you and let you type it in. . . but there's a problem.

The program contains a machine language part. That's tricky to type in, even if you have a machine language monitor. The smallest mistake, and you're out of business.

So I've decided to go a different route. Instead of having you type in the program, I've set things up so that the program will be written for you. The listing contains – not COPY FILE – but a **program generator** that will produce COPY FILE for you.

What's a program generator? It's a program that writes a program. Why bother doing that? Because: the program generator will check things very closely for errors. There's little chance that you'll make a mistake in the DATA statements that eventually create COPY FILE for you.

Type in lines 210 onward very carefully. Be especially careful that you don't forget the semicolon at the end of line 300.

Now start working on the data lines. Don't worry too much about accuracy: if you make a mistake, the generator program will almost certainly pick it up and tell you about it. And it won't write program COPY FILE 64 to disk until you have gotten all the DATA statements right.

In fact, you can type in some of the DATA lines and try a RUN before you're finished. COPY FILE will check what you've done and tell you what's missing.

It's nice to feel that you can't make a mistake in typing in this program. It gives you a sense of security.

But the point of program GENERATOR is to give you program COPY FILE 64. And if you need to copy a file, COPY FILE 64 will be very useful.

There are other, bigger, copying programs that read the directory and check for errors and help you with other good things. But if you don't happen to have on of those, COPY FILE 64 will come in handy.

Program: GENERATOR for COPY FILE 64

```
 1 data   1,   8,  18,   8,  80,   0, 151,  53, -53
 2 data  49,  44,  56,  52,  58, 151,  53,  50, -10
 3 data  44,  57,   0,  55,   8,  90,   0, 133, -19
 4 data  34,  70,  73,  76,  69,  32,  84,  89, -42
 5 data  80,  69,  32,  32,  83, 157, 157, 157, -37
 6 data  34,  59,  84,  36,  58, 133,  34,  70,  -3
 7 data  73,  76,  69,  34,  59,  88,  36,   0, -29
 8 data  75,   8, 100,   0, 159,  49,  44,  56, -14
 9 data  44,  50,  44,  88,  36,  58, 158,  50,  -8
10 data  50,  50,  51,   0,  98,   8, 105,   0, -61
11 data 139,  83,  84, 179, 177,  54,  52, 167, -23
12 data 153,  34,  78,  79,  32,  71,  79,  34, -57
13 data  58, 144,   0, 128,   8, 120,   0, 160,  -2
14 data  49,  58, 133,  34,  79,  84,  72,  69, -47
15 data  82,  32,  68,  73,  83,  75,  32,  82, -49
16 data  69,  65,  68,  89,  34,  59,  65,  36, -48
17 data   0, 171,   8, 130,   0, 139, 195,  40, -23
18 data  65,  36,  41, 167, 159,  32,  49,  44, -22
19 data  56,  44,  50,  44,  88,  36, 170,  34, -32
20 data  44,  34, 170,  84,  36, 170,  34,  44, -61
21 data  87,  34,  58, 158,  50,  50,  54,  55, -20
22 data  58, 160,  49,   0,   0,   0,   0,   0, -38
23 data  32,   7,   9, 162,   1,  32, 198, 255, -30
24 data  32, 228, 255, 160,   0, 145, 251, 230, -12
25 data 251, 208,   8, 230, 252, 165, 252, 201, -43
26 data 208, 240,   4, 165, 144, 240, 233, 165, -50
27 data 251, 141,  64,   3, 165, 252, 141,  65, -54
28 data   3,  76, 204, 255,  32,   3,   9, 162, -29
29 data   1,  32, 201, 255, 160,   0, 177, 251, -49
30 data  32, 210, 255, 230, 251, 208,   2, 230, -25
31 data 252, 165, 251, 205,  64,   3, 165, 252, -61
32 data 237,  65,   3, 144, 231, 169,  55, 133, -28
33 data   1,  76, 204, 255, 169,  54, 133,   1, -61
34 data 169,   9, 133, 252, 169,  84, 133, 251, -55
35 data  96, -28
200 data  36
210 m=63
220 read x : l=peek(m) : h=l=200 : if h then l=x
230 v=r<>l : s=(t<>63 and r>0 and v)
240 if v then t=l : if not s then r=r+1 : s=r<>l
250 t=(t*3+x)and63
260 if s then print " error line ";r : e=-1
270 r=l : if not h goto 220
280 if e then stop
290 x=-1 : restore : open 1, 8, 3, "0 : copy file 64,p,w"
300 if x>=0 then print#1,chr$(x);
310 read x : l=peek(m) : if l<200 goto 300
320 close 1 : end
```

NEW

MPI INTRODUCES SUPER ACTION MEMORY EXPANDER BOARD FOR VIC 20*

Adds 24K and 3 Expansion Slots With Switches and Fuse $109.95

(Expandable To 35K By Simply Adding Memory Chips and Switches)

INTRODUCTORY PRICE

$109.95

Transfer Cartridges To Tape

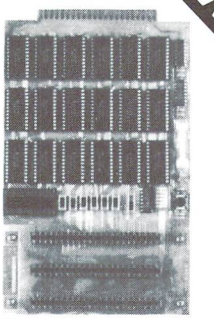
Evade Self Destruct Code With Write Protect Switch

All Boards Factory Tested

Complete Instructions And Illustrations Included.

Fuse Protected

Reset Switch



Pictured Above –
Action Board with all options

24K BOARD FEATURES:

- *Adds 24K Memory (29K with VIC* 5K).
- *Upgrade Board to 35K by adding IC's and switches.
- *Memory switchable in 8K sections. (No need to remove memory board to run your other programs).
- *3 expansion slots with switches (for game or extra utility cartridges).
- *Reset button allows restarting computer without turning power off.
- *.5 amp fuse protected.
- *Switch relocates expansion cartridges in memory so that it can be saved on tape as a backup for your valuable programs. (The unexpanded VIC will not allow cartridges to be saved on tape).
- *Write protect switches allow programs stored in RAM at ROM location or entire Memory to be protected against accidental write.
- *Switch allows memory to be moved between RAM and ROM location. (Useful for developing your own games and saving on tape).
- *Gold plated card edge connector.
- *No other memory expansion needed.
- *Easily plugs into your VIC, no modifications necessary. Saves wear on your VIC 20 since board never needs to be removed or power turned off and on to run other tapes or cartridges.
- **Optional 35K memory (40K with VIC* 5K).

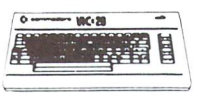
24K memory, 3 expansion slots, switches assembled and tested **$109.95**

Same as above with sockets that allow you to later add your own memory chips to bring memory from 24K to 35K **$124.95**

Full 35K memory, 3 expansion slots, 3K expander mode, eprom socket (switch selectable between BLK 3 & BLK 5) and all switches assembled and tested (eprom not included) **$149.95**

Bare 35K board with complete instruction and parts list **$39.95**

Prices are in US dollars.



VIC 20* is a registered trademark of Commodore.

COD Orders: 816–444–4651

MIDWEST PERIPHERAL INDUSTRIES

MPI

Add $5 for shipping and handling
Mo. residents add 5½% sales tax
Send check or money order to

MPI
Box 8123-B
Kansas City, MO 64112

Personal checks – Allow 3 weeks to clear.

NEW

NEW

NEW

Reduction of an actual sign

The Banner Machine™

For the Commodore 64 (with 5 fonts) or the VIC-20 with 24K memory. • Use on any Epson MX with Graftrax or the FX and RX printers. • Menu-driven program operates like a word processor. • Makes signs up to 10" tall by any length. • Makes borders of variable width up to ¾". • 8 sizes of letters from ¾" to 6½" high. • Proportional spacing; Automatic centering; Right and left justifying. $49.95 Tape or Disk (Specify computer equipment)

For the Commodore 64:

Space Raider An amazing arcade simulation. Your mission is to destroy the enemy ships. $19.95

Super Roller Challenging dice game. Sprite graphics and sound. Yahtzee-style rules of play. $14.95

Formulator A formula scientific calculator designed for tasks which require repetitive arithmetic computations. You can save formulas and numeric expressions. $39.95

Preschool Educational Programs ABC Fun; 123 Fun; and Ginger the Cat with: Addition and Subtraction, Number Hunt, and Letter Hunt. All programs have bright color, music, and action. Each $14.95

Microbroker Exciting, realistic and educational stock market simulation based on plausible financial events. $34.95 Tape or Disk

Sprite Editor The easy way to create, copy, alter, and save up to 224 sprite shapes. $24.95

Cross Reference Generator for BASIC programs Displays line numbers in which any word of BASIC vocabulary appears. Allows you to change variable name and ask for lines where it appears, and more. $19.95

For the VIC-20:

Caves of Windsor A cave adventure game. The object is to restore wealth and happiness to the small village of Windsor. $14.95

Burger & Fries Fast action joystick game. Eat the burgers and fries but avoid the shakes for a top score. $14.95

Catalog available Dealer inquiries invited

PHONE ORDERS: (703) 491-6502
HOURS: 10 a.m. to 4 p.m. Mon.—Sat.

Cardinal Software

Distributed by
Virginia Micro Systems
13646 Jeff Davis Hwy
Woodbridge, VA 22191

Commodore 64 and VIC-20 are registered trademarks of Commodore Electronics Ltd.

NEW

Assembler for the Commodore 64

PAL64

- **easy to learn**
- **easy to use**
- **fast**
- **comprehensive manual**

Personal assembly language
by Brad Templeton
also available for the Commodore
4,000 – 8,000 – 9,000 series

$99.95 from your local Commodore dealer.
For your nearest dealer call:

(416) 273-6350

PRO·LINE SOFTWARE

755 THE QUEENSWAY EAST, UNIT 8
MISSISSAUGA, ONTARIO L4Y 4C5

The Transactor
The Tech/News Journal For Commodore Computers

PAYS $40

per page for articles

We're also looking for professionally drawn cartoons!

Send all material to:

The Editor
The Transactor
500 Steeles Avenue
Milton, Ontario
L9T 3P7

Volume 5 Editorial Schedule

Issue#	Theme	Copy Due	Printed	Release Date
1	Graphics and Sound	Feb 1	Mar 19	April 1
2	The Transition to Machine Code	Apr 1	May 21	June 1
3	Software Protection & Piracy	Jun 1	Jul 23	August 1
4	Business and Education	Aug 1	Sep 17	October 1
5	Hardware and Peripherals	Oct 1	Nov 19	December 1

Advertisers and Authors should have material submitted no later than the 'Copy Due' date to be included with the respective issue.

Driving by Milton?
Come Visit Us!
Take Highway 401 to Highway 25
Go South to Steeles Avenue
(first lights past railway bridge)
Turn left (East) to first lights
100 Yards past there on your right
Is Us!

MAILPRO
STEVE PUNTER'S DATA ORGANIZER AND MAILING LIST PROGRAM FOR COMMODORE 64™

COMPARE THESE FEATURES:

- **fast file definition**
- **easy updating**
- **rapid printing with total format and record selection control**
- **WORDPRO compatible**
- **up to 4000 records on 1541**

MAILPRO 64.....$129.95

Also available for COMMODORE 8032 . . . $179.95

Call for the name of your local dealer:

PRO·LINE SOFTWARE
PRO-LINE SOFTWARE LTD.
(416) 273-6350
755 THE QUEENSWAY EAST, UNIT 8
MISSISSAUGA, ONTARIO CANADA, L4Y 4C5

CURSOR

For your Commodore 64

For only $12.95 each, our **CURSOR 64** tapes are your best buy for the Commodore 64. They take advantage of the color, sound, and sprites that make the 64 such a delight to use. Most of our packages include three excellent Basic programs on one cassette tape. The programs are not copy protected, so you can look at the source code, and learn how to make the 64 do its tricks.

We don't have room to describe all 25 of our **CURSOR 64** programs here. As a sample, you may want to order tape 64-5 with the exciting **Godzilla** program. You'll be challenged as you try to save Tokyo from from the rampaging Godzilla. Or try tape 64-3 with the popular **Miser** text adventure that will take you hours to solve (even if you cheat and read the program source).

We have super programs for the VIC 20, such as **Dungeon** ($12.95), a visual adventure for 16K VICs. Our **VIXEL** programs are also popular with VIC owners. And, we still sell all 30 of the original **CURSOR** cassettes for the original PET and CBM.

Call or write for a catalog today. Be sure and tell us whether you have a 64, a VIC, or a PET. We welcome credit cards, and ship most orders the same day they are received. Dealer inquiries invited.

CURSOR 64, Box 6905
Santa Barbara, CA 93110
805-683-1585

COMMODORE COMPUTER PRINTER ADAPTERS

—addressable-switch selectable upper/lower, lower/upper case.
—works with BASIC, WORDPRO, VISICALC and other software.
—IEEE card edge connector for connecting disks and other peripherals to the PET.
—power from printer unless otherwise noted.

RS-232 SERIAL ADAPTER — baud rates to 9600 — power supply included.
MODEL ADA 1450a $149.00

CENTRONICS/NEC PARALLEL ADAPTER — Centronics 36 pin ribbon connector — handles graphics.
MODEL ADA 1800 . . . . . $129.00

COMMUNICATIONS ADAPTER serial & parallel ports — true ASCII conversion — baud rates to 9600 — half or full duplex — X-ON, X-OFF - selectable carriage return delay — 32 character buffer — centronics compatible.
MODEL SADI $295.00

COMMODORE 64 to RS-232 CABLE ADAPTER
MODEL ADA 6410 . . . . . $79.00

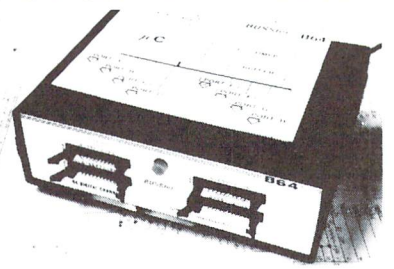
Prices are in US dollars.

BUSSter™

COMPUTER INTERFACES

IEEE — 488
RS — 232

ANALOG AND DIGITAL INPUT/OUTPUT MODULES



The BUSSter line of analog and digital products was designed to collect data and to output signals to laboratory and industrial equipment in conjunction with a microcomputer system. These powerful self-contained modules reduce a computer's workload by providing read or write operations to external devices. They are controlled as slave interfaces to real-world physical applications. Control is over an IEEE-488 (GPIB) bus or RS-232 port. BUSSter modules are available in several digital and analog configurations. The internal buffer and timer provide flexibility by allowing the BUSSter to collect data while the host computer is busy with other tasks.

BUSSter A64—64 channel digital input module to read 64 digital signals. Built-in buffer **$495.00**

BUSSter B64—64 channel digital output module to send 64 digital signals **$495.00**

BUSSter C64—64 channel digital input/output module to read 32 and write 32 digital signals. Built-in buffer **$495.00**

BUSSter D16—16 channel analog input module to read up to 16 analog signals with 8 bit resolution (¼%) Built-in buffer **$495.00**

BUSSter D32—32 channel version of the D16 **$595.00**

BUSSter E4—4 channel analog output module to send 4 analog signals with 12 bit resolution (.06%) **$495.00**

BUSSter E8—8 channel version of the E4 **$595.00**

BUSSter E16—16 channel version of the E4 **$695.00**

Add the suffix -G for IEEE-488 (GPIB) or -R for RS-232.

All prices are USA only. Prices and specifications subject to change without notice.

30 DAY TRIAL—
Purchase a BUSSter product, use it, and if you are not completely satisfied, return it within 30 days and receive a full refund.

US Dollars Quoted
$10.00 Shipping & Handling
MASTERCARD/VISA

CmC

Connecticut microComputer, Inc.
INSTRUMENT DIVISION
36 Del Mar Drive
Brookfield, Ct. 06804
(203) 775-4595 TWX: 710-456-0052

COMMODORE 64

ADD all this:

- RUN-TIME COMPILER
- 40 Graphics Statements
- 11 Sprite Statements
- "LOGO" TURTLE GRAPHICS
- FAST program execution
- auto line numbering
- line renumbering
- program structures
- pretty printing
- merging programs segments
- long variable names
- named procedures
- parameter passing
- local and global variables
- random access disk files
- stop key disable
- End Of File detection

What does this and more? **COMAL**
What is the cost? **Only $19.95**

Disk loaded with many sample programs.
Also available: COMAL HANDBOOK, $18.95
Send SASE for more information or
Send Check or Money Order in US Dollars
plus $2.00 shipping/handling to:
COMAL Users Group, 5501 Groveland Ter.
Madison WI 53716 (608) 222-4432
(not affiliated with COMAL Interest Group)

C64-FORTH
for the Commodore 64

FORTH SOFTWARE FOR THE COMMODORE 64

C64-FORTH (TM) for the Commodore 64 - $99.95
- Fig Forth-79 implementation with extensions
- Full feature screen editor and macro assembler
- Trace feature for easy debugging
- 320x200, 2 color bit mapped graphics
- 16 color sprite and character graphics
- Compatible with VIC peripherals including disks, data set, modem, printer and cartridges
- Extensive 144 page manual with examples and application screens
- "SAVETURNKEY" normally allows application program distribution without licensing or royalties

C64-XTEND (TM) FORTH Extension for **C64-FORTH** - $59.95
(Requires original **C64-FORTH** copy)
- Fully compatible floating point package including arithmetic, relational, logical and transcendental functions
- Floating point range of 1E+38 to 2E-39
- String extensions including LEFT$, RIGHT$, and MID$
- BCD functions for 10 digit numbers including multiply, divide, and percentage. BCD numbers may by used for DOLLAR.CENTS calculations without the round-off error inherent in BASIC real numbers.
- Special words are provided for inputting and outputting DOLLAR.CENTS values
- Detailed manual with examples and applications screens

(Commodore 64 is a trademark of Commodore)

TO ORDER - Specify disk or cassette version
- Check, money order, bank card, COD's add $1.50
- Add $4.00 postage and handling in USA and Canada
- Mass. orders add 5% sales tax
- Foreign orders add 20% shipping and handling
- Dealer inquiries welcome

PERFORMANCE MICRO PRODUCTS

770 Dedham Street. S-2
Canton, MA 02021
(617) 828-1209

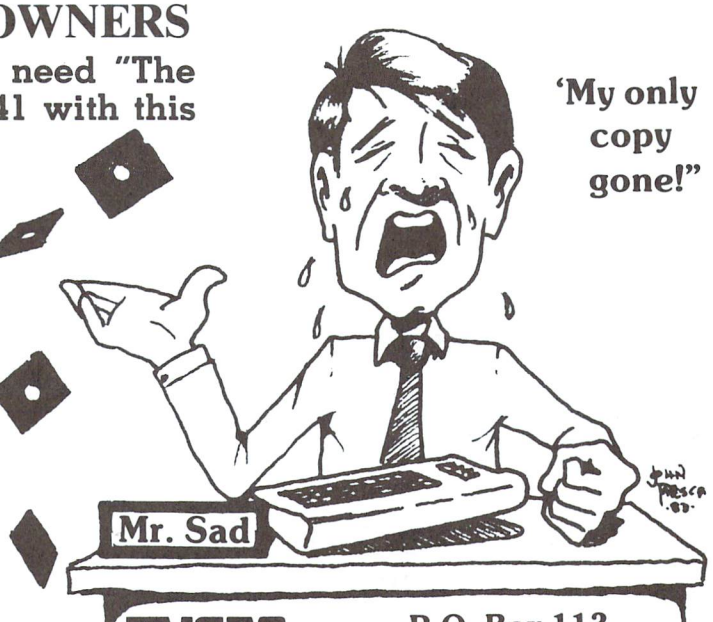
We'll back you up!

MasterCard VISA

ATTENTION COMMODORE 64 OWNERS

If you own a disk drive then you'll need "The Clone Machine". Take control of your 1541 with this package that includes:

1.) Complete and thorough users manual
2.) Copy with one or two drives
3.) Investigate and back-up many "PROTECTED" disks
4.) Copy all file types including relative types
5.) Edit and view track/block in Hex or ASCII
6.) Display full contents of directory and print
7.) Change program names, add, delete files with single keystroke
8.) Easy disk initialization
9.) Supports up to four drives



~~List $49.95~~ **Special intro $39.95**

Dealers & Distributors Inquiries Invited

CALL (201) 838-9027

MICRO WARE

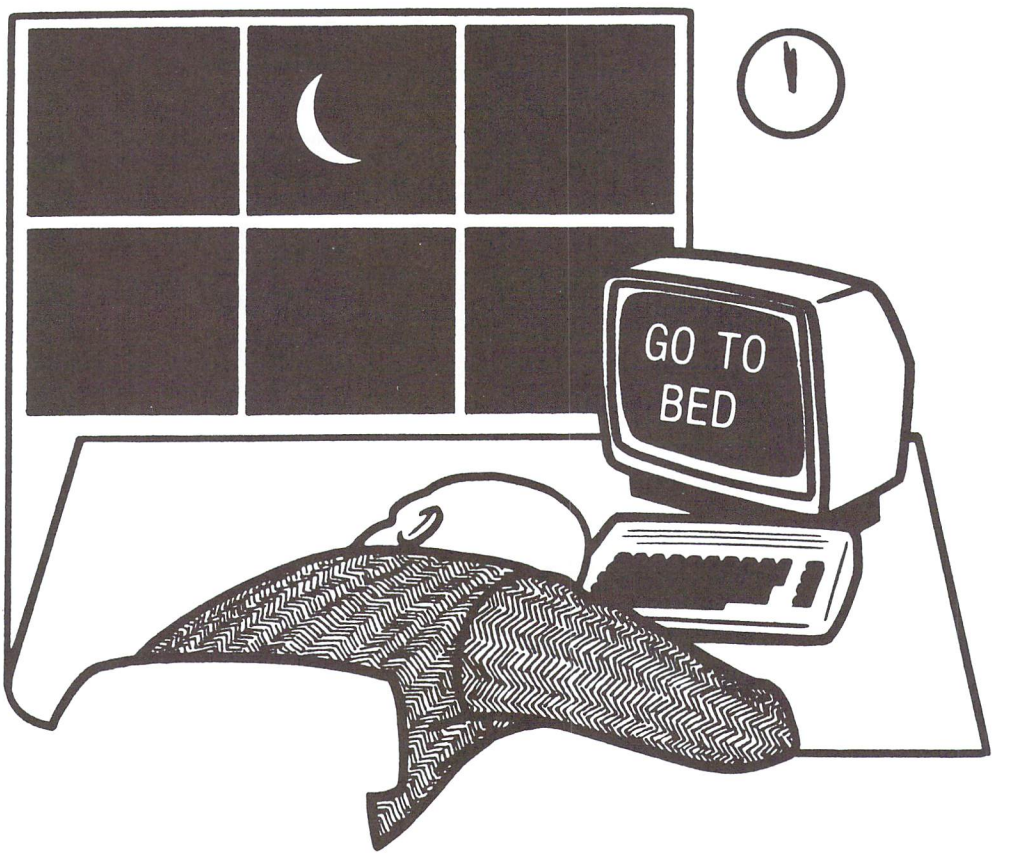
**P.O. Box 113
Pompton Plains, N.J.
07444**

Prices are in US dollars.

The MIDNITE SOFTWARE GAZETTE

The PAPER

Five years of service to the PET community.



The Independent U.S. magazine for users of Commodore brand computers.

EDITORS: Jim and Ellen Strasma
Sample issue free on request, from:

635 MAPLE □ MT. ZION, IL 62549 USA

Finally! An Affordable Full-Size, Full-Feature PRINTER

For your VIC-20®, C-64® ATARI®

Centronics Parallel Types
And RS-232 Serial Types

SUG. LIST $299

$229.95!!

BASIC PRINTER
(Requires one
Option Below)

FEATURES:

- Full graphics capability.
- In the graphic mode, a column of graphic data can be repeated as many times as you want with a single command.
- Double width character output under software control (5 char. per inch).
- Print position addressable by character or dot (positioning control).
- Graphic character and double width character modes can be intermixed on a single line.
- Automatic printing. When the text exceeds the maximum line length no data is lost due to overflow.
- Self-test printing mode.
- Paper width is adjustable up to 10 inches. Standard plain paper.
- 50 cps print speed.
- 80 characters per line.
- 5 × 7 dot matrix.
- Full 2 yr. Warranty.
- Foreign character sets For U.S., U.K., Sweden, and Germany.

Any of these **Options** allow you to connect and print - cables included.

APROPRINT-2064™ (pictured) . . . .Add: $35.95
For Commodore **VIC-20 & C-64** - Cable included.

APROPRINT-4080™ . . . . . .Add: $45.95
For **all Atari Computers** - Cable included.

APROPRINT-1000™ . . . . . .Add: $29.95
RS-232-Serial - Name your computer

APROPRINT-8000™ . . . . . .Add: $29.95
Centronics type Parallel - Name your computer

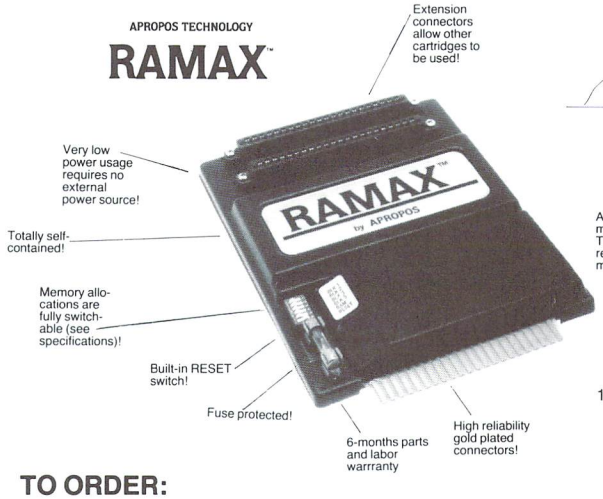
ADD: $8.00 shipping (cont. USA), $25.00 (Canada, HI, AK)

(All other foreign orders Add $55.00 (shipped by Air)

The ONE VIC-20® Memory Expansion Board that DOES IT ALL!

Maximum Memory allows you to use more powerful programs for:
- EDUCATION • ENTERTAINMENT • MAIL LISTS
- BUSINESS APPLICATIONS • FINANCIAL RECORDS

APROPOS TECHNOLOGY
RAMAX™



A perfect investment to give your family and yourself more enjoyment and use from your home computer! The ease of operation, the neat appearance, and the real POWER it adds to your VIC at this low price makes it a MUST for every VIC home!

SPECIAL LOW PRICE!

Only $149.00

Price includes shipping and handling within Continental USA. Foreign orders please add $15.00. Calif. Residents add 6% sales tax.

10 DAY MONEY-BACK GUARANTEE
If not satisfied, simply return in original condition for your money back.

RAMAX Jr.™

Already own an 8k Expander? Get the NEW **RAMAX Jr.™**! Identical to the RAMAX™ except with 19k instead of 27k. Our instructions will show you how to use your 8k as BLK 3 with Jr. to get the full complement of memory!

Special Only $129.00
Shipping included

To equal the total memory of RAMAX™ you would have to buy a 16k Memory Expansion, PLUS an 8k Expansion, PLUS 3k Expansion. THEN you would need a "mother board". With RAMAX™ you buy just ONE piece . . .at ABOUT HALF THE PRICE!

RAMAX™ Features and Specifications:

- Adds up to a full 27k bytes of additional RAM to the standard VIC-20's internal RAM of 5k.
- Built-in switch allows User selection of any combination of 5 areas or RAM memory*:

  BLK1 (8k: Adr. 8192-16383
  BLK 2 (8k: Adr. 16384-24575)
  BLK 3 (8k: Adr. 24576-32767)
  BLK 5 (Adr. 40960-49151, allows/disallows 8k ROM games)
  RAM (3k: Adr. 1024-4095)
  RESET (Resets computer without power off/on)
- Built-in electrical Fuse to protect equipment.
- Totally self-contained. No external power supply needed.
- Two (2) extension connectors allow ANY additional cartridges and/or devices designed for the VIC expansion port.
- Very low power consumption (.175 amp usual).
- High reliability gold-plated connectors are designed for long life.
- Complete Operating Manual.
- 6 month parts and labor warranty to original purchaser.
- Factory service.

*Many VIC-20 cartridges and programs require certain configurations of the memory (i.e certain games will only run on the unexpanded VIC while others require the upper portion of the expanded memory). With RAMAX™ you have switches that turn-on and turn-off portions of the memory to provide the right area of memory - all without plugging or unplugging. It's so easy!

TO ORDER:

Send Check or Money Order For the Total
Calif. residents add 6% tax.
Or Contact your Local Dealer
Phone orders Call **(805) 482-3604**

VISA All Prices U.S. Dollars MasterCard

DEALER INQUIRIES WELCOME

WE SERVICE WHAT WE SELL

VIC-20 & Commodore-64 are registered trademarks of Commodore International. Atari is a trademark of Atari Inc.

APROPOS TECHNOLOGY
1071-A Avenida Acaso
Camarillo, CA 93010

New Product!

APROSPAND-64™ Gives your Commodore 64 full expandability. This superbly designed expansion module plugs into the 64 & gives you 4 switchable (singly or in any combination) expansion connectors - plus fuse protection - plus a reset button! only **$54.95**
Shipping included

APROPOS TECHNOLOGY

PRO GOLF

Here is your chance to play golf on a championship course without all the headaches of getting a tee time, waiting for that slow foursome ahead of you, losing balls, getting rained out or spoiling a good handicap. This game may be played in the privacy of your home or in a clubhouse lounge for the enjoyment of many members. A challenge to even the best players, this game requires a high degree of practice, expertise and accuracy to attain a good score.

PRO GOLF Features :

- A full range of golf clubs (driveway, fairway wood, wedge and irons 2–9)
- Realistic shot distances depending on club and swing
- The ability to hook or slice a shot
- Up to 4 players in one game
- Detailed, colourful screen layouts of 18 different holes (tee, trees, sandtraps, rough, water, out of bounds)
- Simulated ball reaction to course hazards (e.g. ball bounces off trees)
- Hole distances, par, yards to green, strokes taken on hole, total strokes per round and player totals displayed
- A full screen enlargement of greens for putting
- Accurate putting simulation for angle and distance
- Practice of real golf skills – club selection, type of shot (normal, hook, slice), length of swing, special shot strategy (e.g. chipping, getting around or over trees, water, sandtraps)

PRO GOLF
For The Commodore 64
$34.95

written by George Adams

distributed by BMB Compuscience Canada Ltd.
500 Steeles Avenue,
Milton, Ontario
L9T 3P7
416–876–4741

Dealer Inquiries Invited

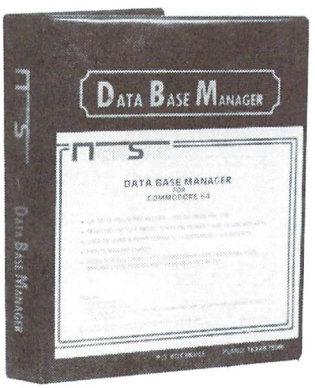
Name _____
Address _____
Prov/State _____ Postal/Zip Code _____

☐Money Order ☐VISA ☐MasterCard ☐Cheque
Acc# _____ Expiry _____
Please include numbers above name

Add $2.00 for shipping & handling
Ontario residents add 7% sales tax.

MICROSPEC

SOFTWARE MEANS BUSINESS FOR THE COMMODORE 64



When it's time to get serious, it's time to boot up MicroSpec business software. Our complete line of business software is made to give you some real applications for your Commodore 64. From data base management to full accounting software, we have the package for you.

It's attention to detail that makes our packages so beautiful and makes them stand out from the rest. We realize that most people are first time users, so we designed all our packages to be completely menu driven and user prompted for each input. We also know that most people use only one disk drive, so we designed all our packages to virtually eliminate disk swapping. Other features like non destructive input routines really make our software easy to use. But all this doesn't restrict you. Pure random access file structure maximizes your disk capacity and allows you to bring up any record for viewing in less than a second.

In our efforts to put together the best packages available, we worked on more than the software. We took the same approach with the documentation as the software. We made it complete and easily understood for the first time user. We even provide sample reports in many cases.

The Demonstration Package, which shows how each program runs, is available for $19.95. So, if you're serious about your 64, call or write for a complete brochure or **go right down to your nearest computer retailer for a demonstration.**

WHEN YOU AND YOUR 64 ARE READY TO GET DOWN TO BUSINESS GIVE US A CALL

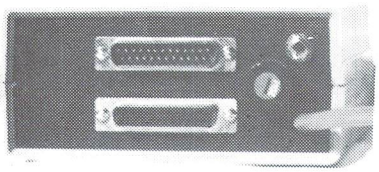
MICROSPEC

P.O. BOX 863085 • PLANO, TX 75086
(214) 867-1333

P.O. Box 4364
Flint, Michigan 48504
(313) 233-5731
(313) 233-3125

ZANIM SYSTEMS

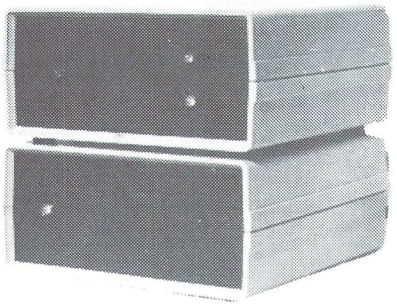
HOME CONTROL FOR YOUR VIC UNDER $200



The ZCM-1 is the Master Control module that provides the interface between your computer and our line of Zanim Application Modules. Up to 15 Application Modules can be piggy-backed to the ZCM-1 Master Control module. The ZCM-1 is compatable with any standard RS-232 (serial) interface. A special Master Control module, the ZCM-1V is available for the VIC-20 and Commodore-64 computers.

*The ZCM-1V is available for VIC-20 and C-64 users.

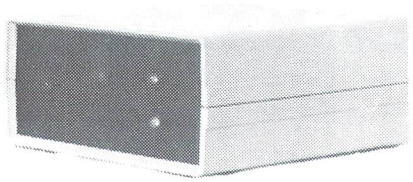
VIC/C64 $129.95
RS-232 $149.95



ZAM-1 is the home control interface module that provides a fully versatile computer controlled environment. ZAM-1 can control up to 256 different lamps and appliances in your home or business providing you with an effective and easy to implement energy management and electric control system. No special modifications are necessary to your building as all control signals are sent over your existing wiring. ZAM-1 can be programmed in BASIC or optional home control software is available. ZAM-1 requires one ZCM-1 Master Control module.

*The ZCM-1/ZCM-1V Master Control module is required to use the ZAM-1 Home Control module.

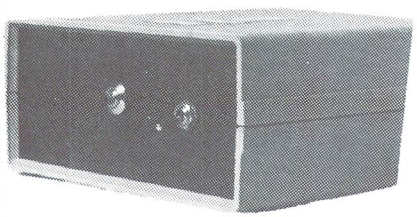
$169.95



The ZAM-2 allows your computer to continuously monitor up to 15 different doors or windows around your home or business. ZAM-2 is a basic building block in a complete computer controlled home security system. With our ZAM-1 Home Control module, you can have a fully integrated security and environment control system. Upon an intrusion, your computer can take the action most appropriate, whether that is to ring an alarm bell, flash all the lights around your home, or dial the police.

*The ZCM-1/ZCM-1V Master Control module is required to use the ZAM-2 Security module.

$179.95



The ZAM-3 is a complete telephone answering and dialing system. It is capable of taking the phone off-hook and dialing a number under computer control or of answering the phone when it rings. With the ZAM-1 Home Control module and the ZAM-2 Security module, the ZAM-3 Phone Dialer module can be integrated into a complete home or business security/monitoring system. Applications include security, auto phone dialing, and computer-answering systems.

*Pulse dialing option is available as ZAM-3P. $199.95

*The ZCM-1/ZCM-1V Master Control module is required to use the ZAM-3/ZAM-3P Phone Dialer module.

Prices are in US dollars.

SERIAL OR PARALLEL (CENTRONICS) PORT SWITCHER

DOES YOUR COMPUTER LOOK LIKE THIS?

A PORT SWITCHER NOW AVAILABLE FOR YOUR COMPUTER (ZSW1)

P.O. BOX 4364
Flint, Michigan 48504
(313) 233-5731
(313) 233-3125

☐ Please send me more information or catalogue!

Name _____

Company _____

Address _____

City _____ State _____ Zip _____

What Make/Model Computer do you own?

_____

TELSTAR 64

Sophisticated Terminal Communications Cartridge for the 64.

PFO 10D 00D CP D1 D2 BELL 12:30:00 10:14:36
(TELSTAR's Status Line)

Don't settle for less than the best!

- Upload/Download to/from disk or tape.
- Automatic File Translation.
- Communicates in Industry Standard ASCII.
- Real-Time Clock plus Alarm Clock.
- Line editing capability allows correcting and resending long command lines.
- 9 Quick Read functions.
- Menu-driven.
- Similar to our famous STCP Terminal package.
- Works with Commodore Modems and supports auto-dialing.

The best feature is the price — **only $49.95** (Cartridge and Manual)

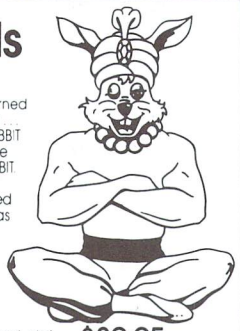
Machine Language Monitor Cartridge
for the CBM 64

More than 20 commands allow you to access the CBM 64's Microprocessors Registers and Memory Contents. Commands include assemble, disassemble, registers, memory, transfer, compare, plus many more.

Someday every CBM 64 owner will need a monitor such as this.

Cartridge and Manual — **$24.95**

8K in 30 Seconds
for your VIC 20 or CBM 64



If you own a VIC 20 or a CBM 64 and have been concerned about the high cost of a disk to store your programs on, worry yourself no longer. Now there's the RABBIT. The RABBIT comes in a cartridge, and at a much, much lower price than the average disk. And speed . . . this is one fast RABBIT. With the RABBIT you can load and store on your CBM datasette an 8K program in almost 30 seconds, compared to the current 3 minutes of a VIC 20 or CBM 64, almost as fast as the 1541 disk drive.

The RABBIT is easy to install, allows one to Append Basic Programs, works with or without Expansion Memory, and provides two data file modes. The RABBIT is not only fast but reliable.

(The Rabbit for the VIC 20 contains an expansion connector so you can simultaneously use your memory board, etc.)

$39.95

MAE NOW THE BEST FOR LESS!

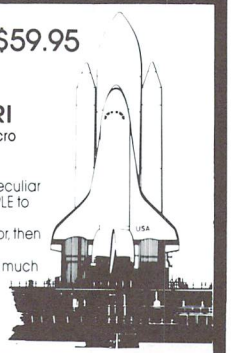
$59.95

For CBM 64, PET, APPLE, and ATARI

Now, you can have the same professionally designed Macro Assembler/Editor as used on Space Shuttle projects.

- Designed to improve Programmer Productivity.
- Similar syntax and commands - No need to relearn peculiar syntaxes and commands when you go from PET to APPLE to ATARI.
- Coresident Assembler/Editor - No need to load the Editor, then the Assembler, then the Editor, etc.
- Also includes Word Processor, Relocating Loader, and much more.
- Powerful Editor, Macros, Conditional and Interactive Assembly, and Auto - zero page addressing.

Still not convinced; send for our free spec sheet!



Eastern House

3239 Linda Dr.
Winston-Salem, N.C. 27106
(919) 924-2889 (919) 748-8446
Send for free catalog!

VISA MasterCard

Prices are in US dollars.

C 64 PROVINCIAL PAYROLL

A complete Canadian Payroll System for Small Business.

• 50 Employees per disk (1541) • Calculate and Print Journals • Print Cheques • Calculate submissions summary for Revenue Canada • Accumulates data and prints T-4s • Also available for 4032 and 8032 Commodore Computers.

Available from your Commodore Dealer.

Distributed by:

MICROCOMPUTER SOLUTIONS
1262 DON MILLS RD. STE. 4
DON MILLS, ONTARIO M3B 2W7
TEL: (416) 447-4811

TYPRO
DATA MANAGER & WORD PROCESSOR
For COMMODORE 8032 Computer — 8050/4040 Dr.

NOW AVAILABLE FOR THE COMMODORE 64®

DATA MANAGER

Number of records is only limited by your disk capacity. Up to 50 fields per record. Maximum of 75 characters per field. User formatted. Screen editing. Sort and search feature. Pattern match search. Selective field printing and formatting. Form letter addressing. Mailing list and mailing label printing. Format for fanfold Rolodex and index card printing.

WORD PROCESSOR

Screen editing. Automatic line length set. Add, move or delete text. Global edit. Page numbering and titling. Form letter addressing. File append for printing.

BOTH PROGRAMS ABOVE, ONLY $89.00

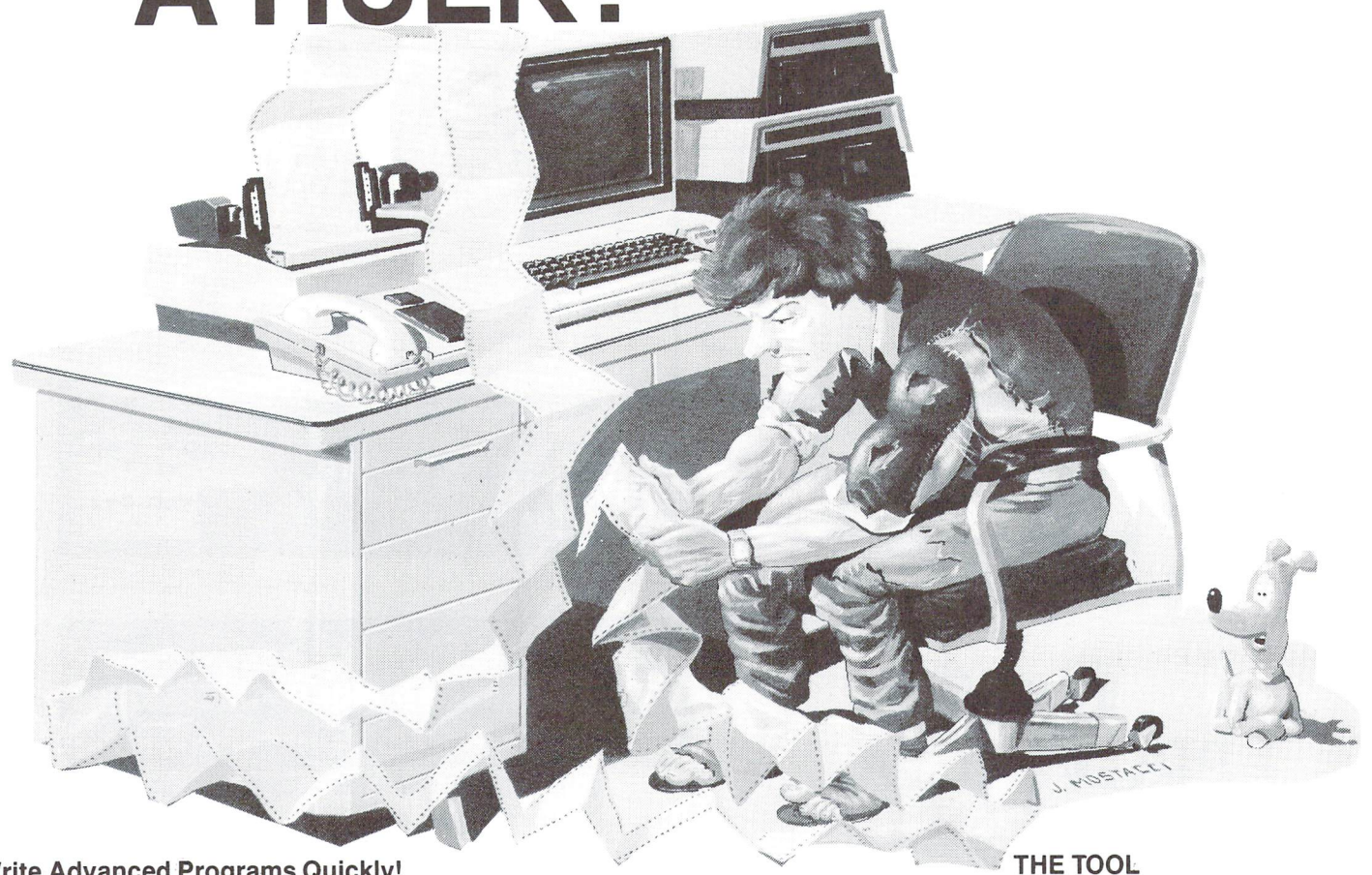
* Commodore 64 is a trademark of Commodore Electronics, Ltd.

Also for Commodore 64 and 8032
AMORTIZATION SCHEDULE — $30.00 (Disk)
INVENTORY MANAGEMENT — $55.00 (Disk)

All software is fully supported for updates and revisions for up to six months after purchase.
Specify Computer model number and Disk model number.

INPUT SYSTEMS, INC.
25101 S.W. 194 Ave. Homestead, FL 33031 (305) 245-3141
DEALER INQUIRIES INVITED.

IS PROGRAMMING TURNING YOU INTO A HULK?



Write Advanced Programs Quickly!

Tired of writing reams of code? Take a quantum jump into the future! Tommorrow's programmers are using software development tools such as THE TOOL. THE TOOL lets you make use of powerful machine language subroutines. Your programs will execute fast using less code. Input/output routines and professional looking screens are easily created.

Features of **THE TOOL** include :

- Screen Design functions which allow controlled input and output
- High Resolution Graphics with alpha/numeric display
- Screen Save and Load functions (for hi-res and text screens)
- Structured BASIC instructions , e.g. IF THEN ELSE
- Programming Aids (e.g. auto, renumber, delete, find, trace, hardcopy)
- 2 keystroke disk commands (DOS support extensions)
- Game Design Instructions (joy, scroll, screen, colour)
- A 50 page user manual

THE TOOL
For The Commodore 64 ™
$65.00

developed by Micro Application

distributed by BMB Compuscience Canada Ltd.
500 Steeles Avenue,
Milton, Ontario
L9T 3P7
416–876–4741

Dealer Inquiries Invited

Name _____
Address _____
Prov/State _____ Postal/Zip Code _____

Order ☐VISA ☐MasterCard ☐Cheque
Acc# _____ Expiry _____
Please include numbers above name

Add $2.00 for shipping & handling
Ontario residents add 7% sales tax.

(M)agreeable™

software

STOCK HELPER™

Commodore 64 and VIC-20

Stock HELPER is a tool to maintain a history of stock prices and market indicators on diskette, to display charts, and to calculate moving averages. Stock HELPER was designed and written by a "weekend investor" for other weekend investors.

Stock HELPER is available on diskette for:

Commodore 64	$30.00	($37.00 Canadian)
VIC-20 (16K)	$27.00	($33.25 Canadian)

plus $1.25 shipping ($1.55 Canadian)

The VIC-20 version only charts 26 bi-weekly periods rather than 52 weekly periods.

(M)agreeable software, inc.

5925 Magnolia Lane • Plymouth, MN 55442

(612) 559-1108

(M)agreeable and HELPER are trademarks of (M)agreeable software, inc.
Commodore 64 and VIC-20 are trademarks of Commodore Electronics Ltd.

Intelligent Software for the Commodore computers

Catalog 5/1/83

My line of programs (such as it is) consists of the following products. All are written for Commodore computers; any of my programs will load and run without modification in the entire line (including older PET's).

1. Word Processor; $25. It includes the following features: VERY fast file routines, including a disk file catalog; automatic form handling on tractor- or friction-feed printers; fully imbedded margin, justification, spacing, formatting, and paging controls; block commands and error-trapping in editing mode; and a spool routine (formatted output to disk for later mass printing). I believe W/P is the most thoroughly tested, user-oriented word processor available at this time at anywhere near the price. for any machine. Requires a minimum of 10k of memory (8k expansion on VIC), and a printer.

2. Copycalc; $20 ($15 if ordered with another program). Copycalc is a simplified version of the "electronic spreadsheets" that are becoming extremely popular for use on personal computers. It allows the user to set up a visible grid of numbers on the screen, and use the screen-editor to make changes in the grid, with the totals reflecting the changes. Requires 6k RAM (3k expansion on VIC); smaller version available for unexpanded VIC.

3. Baseball Manager; $30. This program maintains complete batting statistics for a baseball or softball league of up to 250 players. It generates reports on a player, team, or the entire league (including standings). It requires a minimum 10k of RAM; a printer is suggested but not required.

4. Inventory; $30. A general-purpose perpetual inventory control program. It produces a variety of reports, including order forms; multiple vendors are supported. Requires 10k of RAM; a printer is suggested.

All programs: support cassette and disk files and the CBM printers (easily modifiable to other printers), come on cassette, and include documentation. Prices include shipping; Calif. residents add 6%. All programs are copyrighted by the author; those rights will be enforced. Programs available from:

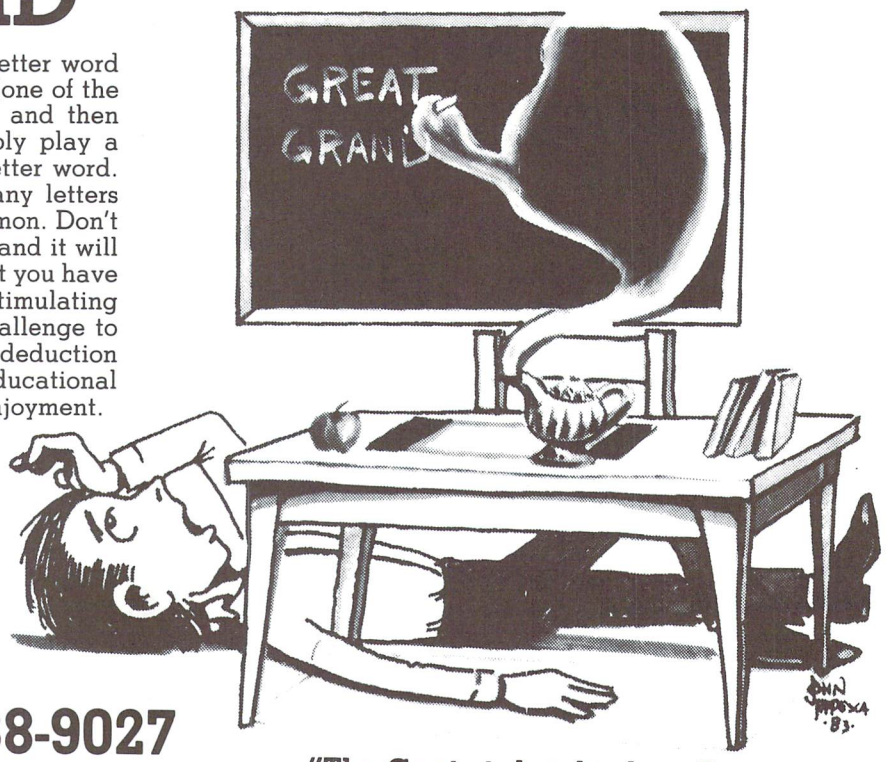
William Robbins, Box 3745, San Rafael, CA 94912

Disk Software for the Commodore 64™

JOT-A-WORD™

A computerized version of the old five letter word game. Simply pick a secret five letter word (one of the almost 5000 words contained on the disk) and then play against the Jot-A-Word Genie or simply play a solitaire version. Start by typing in a five letter word. The Genie responds by telling you how many letters your guess and the secret work have in common. Don't try to cheat, because the Genie is too smart and it will not accept non-words or continue a game that you have given it wrong scores. This is a simple but stimulating game for ages 9 to senior citizen. A real challenge to your intellect, reasoning powers, logic and deduction skills. It's simply hard to beat; as a fun and educational experience! Graphics and music add to the enjoyment.

MasterCard VISA



"The Genie is hard to beat!"

ONLY $29.95

MICRO WARE D 1342B RT. 23 BUTLER, N.J. 07405

Dealers & Distributors Inquiries Invited 201-838-9027

Prices are in US dollars.

Strengthen your hand with Superbase 64

The complete information control system for the Commodore 64

The World Famous Commodore 64.

No matter what your business or interest, with Superbase 64 you have a totally flexible 'record' system, as big as you want it, as fast as you need it.

Create your own formats, enter your records, change layouts and datafields.

Superbase gives you unrivalled control in home or office, business or professional practice, with a range of features including:

TOTAL CONTROL

- Links to other programs and EASY SCRIPT for mailshots, high-quality letters, quotes, tables, etc.
- Effective management of invoices, addresses, stock, membership, appointments – any and every kind of record
- English like commands for easy conversational programming, plus built-in BASIC

FAST ACCESS

- Keyed records for high speed retrieval – 3 secs for any file size
- Select and search for any name, description, number, date, etc. in multiple combinations
- Sort records into any order
- Display selections or generate printed reports
- Browse through records matching on any criteria

DATABASE MANAGEMENT

- Easy to understand menus
- Add or amend fields, or alter length – no file rebuilding needed
- Update files with automatic batch processing option
- Calendar arithmetic for effective time management
- Display quantities, values, totals, as you enter them. Formulas for on-screen result calculation

YOUR OWN RECORDS

- Design **your** layout using text, numeric, calculated result, date, linking and key fields
- Record size up to 1100 characters spread over up to 127 items
- Number of records limited **only by your equipment** – can be thousands
- As many databases as you want – each with up to 15 files
- Learn fast through built-in HELP screens – then add your own notes

Precision Software

commodore

SILICOM INTERNATIONAL SOFTWARE INC.
990 HILLSIDE AVENUE
VICTORIA, B.C. V8T 2A1
TELEX 049-7102

OFFICIALLY APPROVED BY COMMODORE

SOFTWARE FOR
VIC ★ COMMODORE 64 ★ PET
FROM KING MICROWARE

- S D COPY FAST EFFICIENT SINGLE DISC COPIER FOR THE 1541 — $19.95
- WORDS & CALCS SPREAD SHEET FOR THE C-64 ALLOWS TEXT — $42.95
- CHART PAC 64 FINEST CHART MAKER AROUND — $42.95
- SMARTEES ACTION PACKED MAZE GAME — $22.95

NEW! THE BANKER THE FINEST CHECK BOOK RECONCILIATION PROGRAM ON THE MARKET — $38.95

- DAISY — DATA ADAPTABLE INFORMATION SYSTEM
  — THE DATA BASE WITH A DIFFERENCE
  — ALLOWS YOU TO CALCULATE BETWEEN FIELDS — $39.95
- ASTRO POSITIONS FIND THE STARS AND CAST YOUR HOROSCOPE — $43.95

LOOK AT THE LANGUAGES WE HAVE

YES! WE HAVE PASCAL — $52.95

ULTRABASIC WITH TURTLE GRAPHICS AND SOUND — $42.95

TINY BASIC COMPILER — $22.95

TINY FORTH FIG FORTH IMPLEMENTATION — $22.95

64-BUDGETEER
64-CRIBBAGE
SKIER-64
64 QUICK-CHART
SYNTHY-64

VIC TINY PILOT
VIC BUDGETEER
VIC VIGIL
VIC CRIBBAGE
GRAPHVICS

SCREEN DUMP
SPRITE-AID
VIC HIRES
VIC JOYSTICK PAINTER
VIC I-CHING

We are actively seeking SOFTWARE AUTHORS.
WHY NOT SEND US YOUR PROGRAM FOR EVALUATION.

Dealer Inquiries Invited
Write for our FREE Catalogue
for VIC and C-64

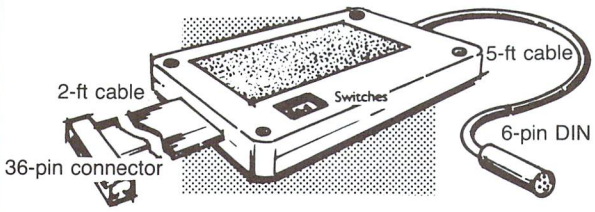
KING MICROWARE Ltd.

Suite 210,
5950 Côte des Neiges
Montreal, Quebec H3S 1Z6

Canadian manufacturer and distributor for ABACUS Software Products

MW-302: VIC-20/64

Parallel Printer Interface.



Works with all centronics type parallel matrix & letter printers and plotters—Epson, C.Itoh, Okidata, Nec, Gemini 10, TP-I Smith Corona, and most others. Hardware driven; works off the serial port. Quality construction: Steel DIN connectors & Shielded cables. Has these switch selectable options: Device 4, 5, 6 or 7; ASCII or PET ASCII; 7-bit or 8-bit output; upper & lower case or upper only. Recommended by PROFESSIONAL SOFTWARE for WordPro 3 Plus for the 64, and by City Software for PaperClip.

MW-302 . . . . Canadian $189.95

Micro World Electronix, Inc.
3333 S. Wadsworth Blvd. #C105, Lakewood, CO 80227
(303) 987-2671

CANADIAN DEALERS

ALBERTA

Computer Shop of Calgary
3515 18th St. S.W.
Calgary, T2T 4T9
(403) 243-4356

Hindson Computer Systems, Ltd.
7144 Fisher St. S.E.
Calgary, T2H 0W5
(403) 252-9576

TJB Micro Systems, Ltd.
10991 124th St.
Edmonton, T5M 0H9
(403) 433-3161

BRITISH COLUMBIA

Conti Electronics
7204 Main Street
Vancouver, V5X 3Y4
(604) 324-0505

ONTARIO

MGI Computer Corp.
1501 Carling Ave.
Ottawa, T1Z 7M1
(613) 722-1000

Richvale Telecommunications
10610 Bayview (Bayview Plaza)
Richmond Hill, L4C 3N8
(416)884-4165

SASKATCHEWAN

Micro Shack of West Canada
607 45th St. West
Saskatoon, S7L 5W5
(306) 244-6909

COMMODORE OWNERS

Join the world's largest, active Commodore Owners Association.

- **Access to thousands of public domain programs on tape and disk for your Commodore 64, VIC 20 and PET/CBM.**
- **Monthly Club Magazine**
- **Annual Convention**
- **Member Bulletin Board**
- **Local Chapter Meetings**

Send $1.00 for Program Information Catalogue.
(Free with membership).

Membership Fees for 12 Months	**Canada**	**—**	**$20 Can.**
	U.S.A.	**—**	**$20 U.S.**
	Overseas	**—**	**$30 U.S.**

T.P.U.G. Inc.
Department "M"
1912A Avenue Road, Suite 1
Toronto, Ontario, Canada M5M 4A1

*** LET US KNOW WHICH MACHINE YOU USE ***

NEW

Basic Utility for the Commodore 64

POWER 64

- **easy to learn**
- **easy to use**
- **program faster and more efficiently with better results**
- **MOREPOWER included FREE**

Powerful Programmer's Utility
by Brad Templeton
Manual by Jim Butterfield

$99.95 from your local Commodore dealer.
For your nearest dealer call:
(416) 273-6350
PRO·LINE SOFTWARE
755 THE QUEENSWAY EAST, UNIT 8
MISSISSAUGA, ONTARIO L4Y 4C5

Let The SMART 64 Terminal

COMMODORE 64* Do The DRIVING

No matter which direction you wish to travel in, experience the advantage of computer communications with The SMART 64 Terminal. Discover the program that puts you on the Right Road to: Public-Access Networks, University Systems, Private Company Computers and Financial Services.

The SMART 64 Terminal designed with Quality-Bred features, Affordable Pricing . . . And Service.

So why not travel the communications highways the SMART way!

Suggested $39.95 Retail

Accessories included:

- ☐ *Selective Storage of Received Data.*
- ☐ *Alarm Timer.*
- ☐ *40 or 80 Col. Operation*.*
- ☐ *Auto-Dial.*
- ☐ *User-Defined Function Keys, Screen Colors, Printer and Modem Setting.*
- ☐ *Screen Print.*
- ☐ *Disk Wedge Built-In!*
- ☐ *Formatted Lines.*
- ☐ *Review, Rearrange, Print Files.*
- ☐ *Sends/Receives Programs and Files of ANY SIZE.*
- ☐ *Adjustable transmit/receive tables allow custom requirements. These and other features make The SMART 64 Terminal the best choice for grand touring telecommunications.*

*Commodore 64 registered trademark of Commodore Business Machines Inc.
*Supports 80-column cartridge by Data 20 Corporation.

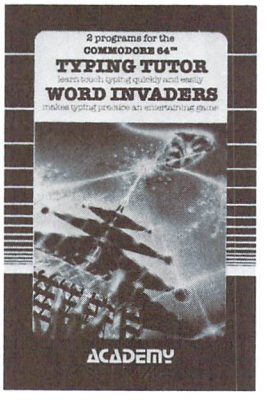
Dealer Availability
Call (203) 389-8383

MICROTECHNIC® SOLUTIONS
P.O. BOX 2940, New Haven, Ct. 06515

Prices are in US dollars.

VIC 20™ COMMODORE 64™

JOIN THE COMPUTER REVOLUTION WITH A MASTERY OF THE KEYBOARD!



In the age of the computer, *everyone* from the school child to the Chairman of the Board should be at home at the computer keyboard. Soon there will be a computer terminal on every desk and in every home. Learn how to use it right ...and have some fun at the same time!

Rated THE BEST educational program for the VIC 20™ by Creative Computing Magazine

TYPING TUTOR PLUS WORD INVADERS

The proven way to learn touch typing.

COMMODORE 64 Tape $21.95 COMMODORE 64 Disk $24.95
VIC 20 (unexpanded) Tape $21.95

Typing Tutor plus Word Invaders makes learning the keyboard easy and fun! Typing Tutor teaches the keyboard in easy steps. Word Invaders makes typing practice an entertaining game. Highly praised by customers:

"Typing Tutor is great!", "Fantastic", "Excellent", High quality", "Our children (ages 7-15) literally wait in line to use it.", "Even my little sister likes it", "Word Invaders is sensational!"

Customer comment says it all . . .

". . . it was everything you advertised it would be. In three weeks, my 13 year old son, who had never typed before, was typing 35 w.p.m. I had improved my typing speed 15 w.p.m. and my husband was able to keep up with his college typing class by practicing at home."

NEW!

IFR (FLIGHT SIMULATOR) CARTRIDGE FOR THE VIC 20

$39.95

COMMODORE 64 TAPE OR DISC $29.95

JOYSTICK REQUIRED

Put yourself in the pilot's seat! A very challenging realistic simulation of instrument flying in a light plane. Take off, navigate over difficult terrain, and land at one of the 4 airports. Artificial horizon, ILS, and other working instruments on screen. Full aircraft features. Realistic aircraft performance — stalls/spins, etc. Transport yourself to a real-time adventure in the sky. Flight tested by professional pilots and judged "terrific"!

VISA Shipping and handling $1.00 per order. CA residents add 6% tax. MasterCard

ACADEMY SOFTWARE
P.O. Box 9403, San Rafael, CA 94912 (415) 499-0850

Programmers: Write to our New Program Manager concerning any exceptional VIC 20™ or Commodore 64™ game or other program you have developed.

MICROCOMPUTER SUPPLIES

VISA

100% Guaranteed

DISKETTES FROM
10/$25.00
S.S. S.D

Shop Without Going Shopping

MEMOREX		DYSAN	
3481 SS/DD	$33.00 per 10	104/1D SS/DD Soft	$54.00 per 10
3491 DS/DD	$45.00 per 10	104/2D DS/DD Soft	$76.00 per 10
3481-2 SS/DD 2 Pack Mailer		$8.00 plus 75¢ postage	

All other sizes and format available and volume discount.

Flip 'N' File, hold 50, 5¼" diskettes	$38.50 each
Flip Pac, holds 10 5¼" diskettes	$ 7.00 each
Disk Banks, hold 10 5¼" diskettes	$ 7.00
Plastic Library Cases, choice of colors;	$4.50 each
EPSON RX80 Printer	$525.00
EPSON FX80 Printer	$850.00
EPSON FX100	$CALL
QUADRAM Interfaces	$CALL
COMREX CRII Daisy Wheel Printer;	$869.00
COMREX Monitors, 8 models	from $198.00
MX'80 Ribbons	$14.50

WRITE OR CALL FOR FREE CATALOG

To order: Send money order, certified cheque, personal cheques must clear our bank, VISA or MASTERCARD. (Include card # and expiry date & signature) Add 5% for shipping and handling. Minimum $3.00 per order. Quebec residents add 9% P.S.T.

INTERNATIONAL MARKETING SERVICES

P.O. Box 522, Boucherville, Quebec, J4B 6Y2
(514) 655-9232

CANADIAN SOFTWARE SOURCE

FOR **ALL** YOUR COMMODORE 64 NEEDS

• GREAT SAVINGS • OVER 400 PROGRAMS • FAST FRIENDLY SERVICE •*

	Canadian $ Retail	C.S.S.
HOME ACCOUNTANT (Continental) (D)	$ 99.95	$ 74.95
MASTER TYPE (Lightning) (D)	49.95	44.95
PARALLEL PRINTER INTERFACE (Cardco)	139.95	105.00
KINDERCOMP (Spinnaker) (Car)	44.95	37.95
TYPING TUDOR/WORD INV. (Academy) (T)	27.95	22.95
WITNESS (Infocom) (D)	69.95	46.95
SPRITEMASTER 64 (Access) (D)	62.95	46.95
NEUTRAL ZONE (Access) (T/D)	59.95	44.95
BLADE OF BLACKPOOLE (Sirius) (D)	51.95	45.95
REPTON (Sirius) (D)	51.95	45.95
KOALA PAD & MICRO ILLUSTRATOR	SPECIAL	125.00
DISKS (Control Data) SSDD (12 not 10/box)	SPECIAL	29.95
PRINTER PAPER - 30M-9-1/2"x11"-(3300 Shts./box)	SPECIAL	39.95

To order or for **FREE CATALOGUE** write or phone:

CANADIAN SOFTWARE SOURCE VISA MasterCard

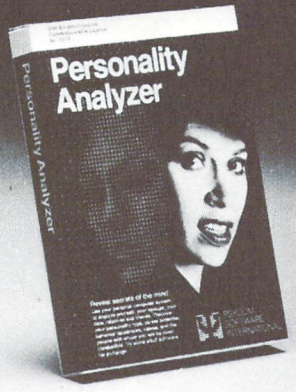
BOX 340 STATION "W" • TORONTO • ONT. M6M 5B9 • (416) 491-2942

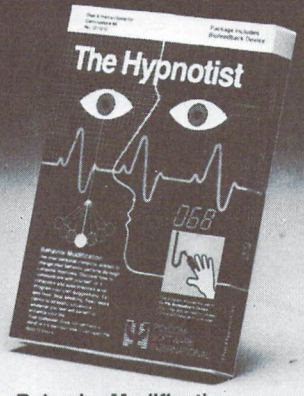
Ontario residents add 7% sales tax. Send certified cheque or money order. VISA & Master Card please include card number, expiry date and signature. Add $2.50 for shipping and handling. All items subject to availability. Prices subject to change without notice.

VISIT US AT **THE CANADIAN COMPUTER SHOW** AND **THE WORLD OF COMMODORE SHOW** IN TORONTO

*Delivery by U.P.S. within 3 days of order date if stocked by local suppliers.

Open your mind





© 1983 PATRICK J. WILLIAMS

Reveal secrets of the mind.
Use your Commodore 64 system to analyze yourself, your spouse, your date, relatives and friends. Discover your personality type, career potential, behavior tendencies, values, and the people with whom you will be most compatible. This program requires the use of a "joystick".
Price $32.95 Disk ($27.95 Cassette).

Behavior Modification.
Use your Commodore 64 system to change your behavior patterns through computer hypnosis. Discover how to communicate with *yourself*, on a conscious and subconscious level. Program your own post-hypnotic suggestions. The PSI Biofeedback Device is included with this program.
Price $87.95 Disk ($79.95 Cassette).

Get this software at your local dealer or order direct from:

PSYCOM SOFTWARE INTERNATIONAL

2118 Forest Lake Drive
Cincinnati, Ohio 45244 USA
Telephone: 513 474-2188

This Space Could Be Transacting For You!

Kelly M. George
Advertising Manager
416 876 4741

Advertising Index

Software

Advertiser	Issue# / Page 1	2	3	4	5	6	Product Name (Description)	Manufacturer
Academy Software				73	77	**78**	VIC20/C64 Software	
Beacon Software			74		85		Business Packages	
BMB Compuscience					73	**72**	The TOOL (programming aid)	Micro Application
					76	**68**	Pro Golf	
Boston Educational Computing				64			Educational Software	
Canadian Software Source						**79**	C64 Software	
Cardinal Software					72	**62**	VIC20/C64 Games, Utilities, Edu.	
COMAL Users Group						**65**	C64 COMAL	
Computer Marketing\Canadian Micro		IBC	IBC	IBC			Calc Result (spreadsheet prog.)	Handic Software ab
Data 20 Corp.				67			Word Manager	
Eastern House				68	81	**71**	MAE Assembler	
Execom Corp.				68			40/80 Screen Select	
Hofacker					87		C64 wordprocessor	
Input Systems Inc.				68	81	**71**	Typro (wordprocessor)	
Isis Hathor						**IBC**	Laser Strike	
King Microware						**76**	VIC/64/PET software	
Magreeable Software				66	75	**74**	Stock Helper	
Microcomputer Solutions					81	**71**	C64 Provincial Payroll	
Micro Ware					75	**74**	C64 JOT-A-WORD	
					82	**65**	C64 Disk Utility	
MicroSpec					70	**69**	C64 Business Software	
Microtechnic Solutions					77	**78**	C64 Terminal software	
Midwest Micro Inc.				69			VIC20/C64 Graphics Util	
					83	**73**	VIC20/C64 SuperTerm	
Nüfekop					1		VIC 20/C64 Games	
Pacific Coast Software		62					C64: Utility, Business, Games	
Performance Micro Products				64		**65**	C64 Forth	
P.F. Communications					86		J Butterfield video tutor	
Precision Software	61						Superscript (wordprocessor)	
Pro-Line Software			79	73	72	**62**	PAL 64 (assembler)	
			76	69	74	**77**	POWER 64 (programming aid)	
				74	78	**64**	MailPro	
			76	66	83	**73**	general	
Psycom Software Int'l					80	**79**	C64 software	
RAK Electronics					80		VIC20/C64 Games, Utilities	
Silicom International						**75**	SuperBase 64 (data base)	Precision Software
Software International				65			VIC20/C64 Software	
Southern Solutions			78	71			Business packages	
William Robbins Software				74	75	**74**	VIC/64/PET Software	
Wycor Business Systems		64					Provincial Payroll	

Hardware

Advertiser	Issue# / Page 1	2	3	4	5	6	Product Name (Description)	Manufacturer
Apropos Technology					79	**67**	VIC20/C64 Printer, Exp board	
Computer Workshops \Computer Marketing		63					Z-RAM (CP/M board)	Madison
Connecticut microComputer		62	79				PET/CBM Interface adapters	
				64	78	**64**	Analog/Digital I/O	
Eastern House				68			Trap 65	
				68			VIC Rabbit	
				68			Eprom Programmer	
				68			Communications Bd	
George M. Drake & Associates				BC	BC	**BC**	Colour Monitors	Amdek
Gosub International				75	82	**63**	Flexikey	
Micro Ware				70			Tape Interface	
				70			VIC20 RAM Expand	
Micro World Electronix				65	74	**77**	VIC20/C64 Printer Interface	
Midwest Micro Inc.				69			Smart ASCII Plus	
Midwest Peripherals				74	72	**62**	VIC20 Expander	
Precision Technology		61	79	73			VIC20/C64 Expander Boards	
Richvale Telecommunications		IFC	IFC	IFC	IFC	**IFC**	C64 Link (IEEE adapter)	
Zanim Systems					84	**70**	Home control hardware	

Accessories

Advertiser	Issue# / Page 1	2	3	4	5	6	Product Name (Description)	Manufacturer
Computer Workshops			76				Apr83 products list	
Consultors Int'l	64						Stock Market With Your PC (book)	
"	64						A To Z Book Of Games	
"	64						Dynamics Of Money Mgmt (book)	
"	64						Invment. Analysis w/Your Micro (book)	
The Code Works				69	78	**64**	'CURSOR', C64 Tape Magazine	
Int'l Marketing Services					80	**79**	Disk, printers, misc.	
Midnight Software Gazette				72	71	**66**	Subscriber Info	
Toronto PET Users Group		61		75	74	**77**	Membership info	

Laser Strike

for the commodore 64

**challenge the asteroid field,
maneuver the caves of ice,
experience the thrill,
play laser strike.**

Laser strike, written in full machine language for the Commodore 64.

**Commodore 64 is a registered trademark
of Commodore Business Machines Inc.**

Visa/MC/Check/Money Order accepted

In U.S.
Cassette $24.95
Disk $29.95
Isis Hathor Digital Productions
6184 Verdura Ave.
Goleta, CA 93117
(805) 964-6335
Add $2.00 postage and handling
California residents add 6% sales tax

*** Ask about Laser strike posters**

Call Toll Free - 1-800-558-8803

ISIS HATHOR
DIGITAL PRODUCTIONS

In U.K.
Cassette £ 9.00 VAT included
Disk £19.95 VAT included
Isis Hathor U.K.
Andrew Barrow
Royden, Perkslane
Prestwood, Gt. Missenden
Bucks, England HP16 OJD
02406-3224
**You will be billed
for postage and handling**

COMPATIBLE COLOR-I . . .

NEW 2 YEAR WARRANTY!
On all monitor electronics . . . 3 yrs. on all CRT's
(See details at dealer)

The popular choice for popular computers . . . at a popular price.

The Color-I Monitor is designed to perform superbly with your Apple II, Atari or VIC Commodore personal computer and others. Highly styled cabinet. It accepts a composite video signal to produce vivid, richly colored graphic and sharp text displays. Very reasonably priced, the Color-I is a giant step above home TV sets and other monitors.

Just write, or call to receive complete specifications on the Amdek Color-I Monitor.

- Quality 260(H) x 300(V) line resolution.
- Built-in speaker and audio amplifier.
- Front mounted controls for easy adjustment.
- Interface cables available for Atari and VIC Commodore computers.
- FCC/UL approved.

2201 Lively Blvd. • Elk Grove Village, IL 60007
(312) 364-1180 TLX: 25-4786
REGIONAL OFFICES: Calif. (714) 662-3949 • Texas (817) 498-2334

AMDEK CORP.

Amdek . . . your guide to innovative computing!